Makinen, 1:10-19

> According to a first aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) with a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), the method comprising reading a set of compressed RISC operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions.

Makinen, 1:63-2:4

> According to a second aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) and a read only memory (ROM), the method comprising reading a set of compressed operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions, the method further comprising generating one or more replacement or additional compressed instructions in the CPU and writing the compressed instruction(s) to the ROM.
>
> The above second aspect of the present invention makes it possible to amend the stored compressed instructions in a dynamic manner. This may, for example, allow a user to configure the computer according to his specific needs.
>
> Preferably, the method comprises the step of reading a set of operating instructions from the ROM into the CPU, which instructions define a program for compressing said replacement or additional instruction(s). More preferably, the instructions defining the compression program form part of said set of compressed operating instructions.

Makinen, 2:18-39

> Preferably, the method of the above first or second aspect of the invention comprises writing the decompressed instruction set to a random access memory (RAM). Thereafter, the decompressed instructions are read from the RAM by the CPU. It is noted that RAM typically offers high access speeds compared to slow (e.g. flash) ROM memory, giving a significant increase in system performance. In this case, increased speed also offers reduced power consumption compared

Makinen Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

to systems which use slow ROM memory and in which power is consumed even when the system is waiting to access the ROM.

Makinen, 2:40-50

According to a third aspect of the present invention there is provided apparatus having a central processing unit (CPU) a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), there being stored in the ROM a set of compressed RISC operating instructions, the CPU being arranged in use to read the compressed instructions from the ROM, to decompress these instructions, and subsequently to operate the apparatus in accordance with the decompressed instructions.

Makinen, 2:51-59

According to a fourth aspect of the present invention there is provided apparatus comprising a central processing unit (CPU), a read only memory (ROM), and a set of compressed operating instructions stored in the ROM, the CPU being arranged in use to read the compressed instructions from the ROM, decompress the compressed instructions, and thereafter operate the apparatus in accordance with the decompressed instructions, the apparatus being further arranged in use to compress replacement or additional operating instructions and to write these compressed instructions to the ROM.

Makinen, 2:60-3:3:2

The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program

Makinen, 3:45-50

In order to enable the microprocessor 1 to decompress the stored compressed instructions, the flash ROM 4 additionally stores a set of instructions, in uncompressed form, which define the Pkzip decompression program. FIG. 2 shows a memory map of the ROM 4 prior to booting the computer, where a part of the memory space is occupied by the compressed instructions 6 and a part is occupied by the uncompressed Pkzip instructions 7. In FIG. 2, the ROM 4 is shown

> coupled to the microprocessor 1, as is the RAM 3 which at this stage remains empty.

Makinen, 3:56-65

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:9

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:18-32

See also:

Makinen — Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:9-31

9.2 storing the additional portion of the operating system in the first memory, and	Makinen, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 8.1 above.

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:9-31

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Makinen, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 8.5 above.

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6, 8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′, 8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:9-31

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Makinen, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 9.1 above.

> A computer having a reduced instruction computer (RISC) architecture has a RISC central processing unit (CPU)(1) coupled to a RAM memory (3) and to a flash ROM memory (4). A set of compressed operating instructions (6,8), including a subset defining a compression method (8), are stored in the flash ROM (4) together with a set of uncompressed instructions (7) defining a compression algorithm. Upon booting of the computer, the uncompressed instructions (7) are read from the ROM (4) by the CPU (1) which then also reads the compressed instructions (6,8), decompresses them according to the decompression process (7), and writes the decompressed instructions (6′,8′) to the RAM (3). The compressed instructions (6,8) can be dynamically altered by the CPU (1), by generating an altered set of uncompressed instructions, compressing these in accordance with the now decompressed compression method (8′), and writing these to the flash ROM (4).

Makinen, Abstract

> At the heart of most modern electronic devices is a microprocessor or central processing unit which operates in accordance with a set of software operating instructions which together form an executable program. The instructions are stored in a digital memory which may be internal to the microprocessor or, as is more usually the case, externally connected to the microprocessor. The set of operating instructions generally define the basic input/output system (BIOS) of the microprocessor together with device drivers, libraries, and user applications.

Makinen, 1:10-19

> According to a first aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) with a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), the method comprising reading a set of compressed RISC operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions.

Makinen, 1:63-2:4

> According to a second aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) and a read only memory (ROM), the method comprising reading a set of compressed operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions, the method further comprising generating one or more replacement or additional compressed instructions in the CPU and writing the compressed instruction(s) to the ROM.
>
> The above second aspect of the present invention makes it possible to amend the stored compressed instructions in a dynamic manner. This may, for example, allow a user to configure the computer according to his specific needs.
>
> Preferably, the method comprises the step of reading a set of operating instructions from the ROM into the CPU, which instructions define a program for compressing said replacement or additional instruction(s). More preferably, the instructions defining the compression program form part of said set of compressed operating instructions.

Makinen, 2:18-39

> Preferably, the method of the above first or second aspect of the invention comprises writing the decompressed instruction set to a random access memory (RAM). Thereafter, the decompressed instructions are read from the RAM by the CPU. It is noted that RAM typically offers high access speeds compared to slow (e.g. flash) ROM memory, giving a significant increase in system performance. In this case, increased speed also offers reduced power consumption compared

Makinen Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

> to systems which use slow ROM memory and in which power is consumed even when the system is waiting to access the ROM.

Makinen, 2:40-50

> According to a third aspect of the present invention there is provided apparatus having a central processing unit (CPU) a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), there being stored in the ROM a set of compressed RISC operating instructions, the CPU being arranged in use to read the compressed instructions from the ROM, to decompress these instructions, and subsequently to operate the apparatus in accordance with the decompressed instructions.

Makinen, 2:51-59

> According to a fourth aspect of the present invention there is provided apparatus comprising a central processing unit (CPU), a read only memory (ROM), and a set of compressed operating instructions stored in the ROM, the CPU being arranged in use to read the compressed instructions from the ROM, decompress the compressed instructions, and thereafter operate the apparatus in accordance with the decompressed instructions, the apparatus being further arranged in use to compress replacement or additional operating instructions and to write these compressed instructions to the ROM.

Makinen, 2:60-3:3:2

> The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program

Makinen, 3:45-50

> In order to enable the microprocessor 1 to decompress the stored compressed instructions, the flash ROM 4 additionally stores a set of instructions, in uncompressed form, which define the Pkzip decompression program. FIG. 2 shows a memory map of the ROM 4 prior to booting the computer, where a part of the memory space is occupied by the compressed instructions 6 and a part is occupied by the uncompressed Pkzip instructions 7. In FIG. 2, the ROM 4 is shown

> coupled to the microprocessor 1, as is the RAM 3 which at this stage remains empty.

Makinen, 3:56-65

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:9

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:18-32

Makinen Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Appendix C13
Invalidity of U.S. Patent 8,880,862 based on Makinen

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Makinen, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Makinen, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 1.1 and 6(Preamble) above.

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:9

11.2 accessing the loaded boot data in compressed form from the memory;	Makinen, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Makinen, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Makinen, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

> A computer having a reduced instruction computer (RISC) architecture has a RISC central processing unit (CPU)(1) coupled to a RAM memory (3) and to a flash ROM memory (4). A set of compressed operating instructions (6,8), including a subset defining a compression method (8), are stored in the flash ROM (4) together with a set of uncompressed instructions (7) defining a compression algorithm. Upon booting of the computer, the uncompressed instructions (7) are read from the ROM (4) by the CPU (1) which then also reads the compressed instructions (6,8), decompresses them according to the decompression process (7), and writes the decompressed instructions (6′,8′) to the RAM (3). The compressed instructions (6,8) can be dynamically altered by the CPU (1), by generating an altered set of uncompressed instructions, compressing these in accordance with the now decompressed compression method (8′), and writing these to the flash ROM (4).

Makinen, Abstract

> At the heart of most modern electronic devices is a microprocessor or central processing unit which operates in accordance with a set of software operating instructions which together form an executable program. The instructions are stored in a digital memory which may be internal to the microprocessor or, as is more usually the case, externally connected to the microprocessor. The set of operating instructions generally define the basic input/output system (BIOS) of the microprocessor together with device drivers, libraries, and user applications.

Makinen, 1:10-19

According to a first aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) with a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), the method comprising reading a set of compressed RISC operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions.

Makinen, 1:63-2:4

According to a second aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) and a read only memory (ROM), the method comprising reading a set of compressed operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions, the method further comprising generating one or more replacement or additional compressed instructions in the CPU and writing the compressed instruction(s) to the ROM.

The above second aspect of the present invention makes it possible to amend the stored compressed instructions in a dynamic manner. This may, for example, allow a user to configure the computer according to his specific needs.

Preferably, the method comprises the step of reading a set of operating instructions from the ROM into the CPU, which instructions define a program for compressing said replacement or additional instruction(s). More preferably, the instructions defining the compression program form part of said set of compressed operating instructions.

Makinen, 2:18-39

Preferably, the method of the above first or second aspect of the invention comprises writing the decompressed instruction set to a random access memory (RAM). Thereafter, the decompressed instructions are read from the RAM by the CPU. It is noted that RAM typically offers high access speeds compared to slow (e.g. flash) ROM memory, giving a significant increase in system performance. In this case, increased speed also offers reduced power consumption compared to systems which use slow ROM memory and in which power is consumed even when the system is waiting to access the ROM.

Makinen, 2:40-50

Makinen Claim 11.4

"utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

> According to a third aspect of the present invention there is provided apparatus having a central processing unit (CPU) a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), there being stored in the ROM a set of compressed RISC operating instructions, the CPU being arranged in use to read the compressed instructions from the ROM, to decompress these instructions, and subsequently to operate the apparatus in accordance with the decompressed instructions.

Makinen, 2:51-59

> According to a fourth aspect of the present invention there is provided apparatus comprising a central processing unit (CPU), a read only memory (ROM), and a set of compressed operating instructions stored in the ROM, the CPU being arranged in use to read the compressed instructions from the ROM, decompress the compressed instructions, and thereafter operate the apparatus in accordance with the decompressed instructions, the apparatus being further arranged in use to compress replacement or additional operating instructions and to write these compressed instructions to the ROM.

Makinen, 2:60-3:3:2

> The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program

Makinen, 3:45-50

> In order to enable the microprocessor 1 to decompress the stored compressed instructions, the flash ROM 4 additionally stores a set of instructions, in uncompressed form, which define the Pkzip decompression program. FIG. 2 shows a memory map of the ROM 4 prior to booting the computer, where a part of the memory space is occupied by the compressed instructions 6 and a part is occupied by the uncompressed Pkzip instructions 7. In FIG. 2, the ROM 4 is shown coupled to the microprocessor 1, as is the RAM 3 which at this stage remains empty.

Makinen, 3:56-65

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:9

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:18-32

See also Makinen, Figs. 2-4

11.5 updating the boot data list.	Makinen, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C13
Invalidity of U.S. Patent 8,880,862 based on Makinen

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Makinen, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Makinen, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Makinen, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Makinen, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Makinen, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C13
Invalidity of U.S. Patent 8,880,862 based on Makinen

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Makinen, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Makinen, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1 and 1.2 above.

> A computer having a reduced instruction computer (RISC) architecture has a RISC central processing unit (CPU)(1) coupled to a RAM memory (3) and to a flash ROM memory (4). A set of compressed operating instructions (6,8), including a subset defining a compression method (8), are stored in the flash ROM (4) together with a set of uncompressed instructions (7) defining a compression algorithm. Upon booting of the computer, the uncompressed instructions (7) are read from the ROM (4) by the CPU (1) which then also reads the compressed instructions (6,8), decompresses them according to the decompression process (7), and writes the decompressed instructions (6′,8′) to the RAM (3). The compressed instructions (6,8) can be dynamically altered by the CPU (1), by generating an altered set of uncompressed instructions, compressing these in accordance with the now decompressed compression method (8′), and writing these to the flash ROM (4).

Makinen, Abstract

> According to a first aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) with a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), the method comprising reading a set of compressed RISC operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions.

Makinen, 1:63-2:4

According to a second aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) and a read only memory (ROM), the method comprising reading a set of compressed operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions, the method further comprising generating one or more replacement or additional compressed instructions in the CPU and writing the compressed instruction(s) to the ROM.

The above second aspect of the present invention makes it possible to amend the stored compressed instructions in a dynamic manner. This may, for example, allow a user to configure the computer according to his specific needs.

Preferably, the method comprises the step of reading a set of operating instructions from the ROM into the CPU, which instructions define a program for compressing said replacement or additional instruction(s). More preferably, the instructions defining the compression program form part of said set of compressed operating instructions.

Makinen, 2:18-39

Preferably, the method of the above first or second aspect of the invention comprises writing the decompressed instruction set to a random access memory (RAM). Thereafter, the decompressed instructions are read from the RAM by the CPU. It is noted that RAM typically offers high access speeds compared to slow (e.g. flash) ROM memory, giving a significant increase in system performance. In this case, increased speed also offers reduced power consumption compared to systems which use slow ROM memory and in which power is consumed even when the system is waiting to access the ROM.

Makinen, 2:40-50

According to a third aspect of the present invention there is provided apparatus having a central processing unit (CPU) a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), there being stored in the ROM a set of compressed RISC operating instructions, the CPU being arranged in use to read the compressed instructions from the ROM, to decompress these instructions, and subsequently to operate the apparatus in accordance with the decompressed instructions.

Makinen, 2:51-59

> According to a fourth aspect of the present invention there is provided apparatus comprising a central processing unit (CPU), a read only memory (ROM), and a set of compressed operating instructions stored in the ROM, the CPU being arranged in use to read the compressed instructions from the ROM, decompress the compressed instructions, and thereafter operate the apparatus in accordance with the decompressed instructions, the apparatus being further arranged in use to compress replacement or additional operating instructions and to write these compressed instructions to the ROM.

Makinen, 2:60-3:3:2

> The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program

Makinen, 3:45-50

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

14.2 loading the boot data into a memory; and	Makinen, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Makinen, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.2 and 1.3 above.

> A computer having a reduced instruction computer (RISC) architecture has a RISC central processing unit (CPU)(1) coupled to a RAM memory (3) and to a flash ROM memory (4). A set of compressed operating instructions (6,8), including a subset defining a compression method (8), are stored in the flash ROM (4) together with a set of uncompressed instructions (7) defining a compression algorithm. Upon booting of the computer, the uncompressed instructions (7) are read from the ROM (4) by the CPU (1) which then also reads the compressed instructions (6,8), decompresses them according to the decompression process (7), and writes the decompressed instructions (6′,8′) to the RAM (3). The compressed instructions (6,8) can be dynamically altered by the CPU (1), by generating an altered set of uncompressed instructions, compressing these in accordance with the now decompressed compression method (8′), and writing these to the flash ROM (4).

Makinen, Abstract

> At the heart of most modern electronic devices is a microprocessor or central processing unit which operates in accordance with a set of software operating instructions which together form an executable

> program. The instructions are stored in a digital memory which may be internal to the microprocessor or, as is more usually the case, externally connected to the microprocessor. The set of operating instructions generally define the basic input/output system (BIOS) of the microprocessor together with device drivers, libraries, and user applications.

Makinen, 1:10-19

> According to a first aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) with a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), the method comprising reading a set of compressed RISC operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions.

Makinen, 1:63-2:4

> According to a second aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) and a read only memory (ROM), the method comprising reading a set of compressed operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions, the method further comprising generating one or more replacement or additional compressed instructions in the CPU and writing the compressed instruction(s) to the ROM.
>
> The above second aspect of the present invention makes it possible to amend the stored compressed instructions in a dynamic manner. This may, for example, allow a user to configure the computer according to his specific needs.
>
> Preferably, the method comprises the step of reading a set of operating instructions from the ROM into the CPU, which instructions define a program for compressing said replacement or additional instruction(s). More preferably, the instructions defining the compression program form part of said set of compressed operating instructions.

Makinen, 2:18-39

> Preferably, the method of the above first or second aspect of the invention comprises writing the decompressed instruction set to a random access memory (RAM). Thereafter, the decompressed instructions are read from the RAM by the CPU. It is noted that RAM typically offers high access speeds compared to slow (e.g. flash) ROM memory, giving a significant increase in system performance. In this case, increased speed also offers reduced power consumption compared to systems which use slow ROM memory and in which power is consumed even when the system is waiting to access the ROM.

Makinen, 2:40-50

> According to a third aspect of the present invention there is provided apparatus having a central processing unit (CPU) a reduced instruction set computer (RISC) architecture, and a read only memory (ROM), there being stored in the ROM a set of compressed RISC operating instructions, the CPU being arranged in use to read the compressed instructions from the ROM, to decompress these instructions, and subsequently to operate the apparatus in accordance with the decompressed instructions.

Makinen, 2:51-59

> According to a fourth aspect of the present invention there is provided apparatus comprising a central processing unit (CPU), a read only memory (ROM), and a set of compressed operating instructions stored in the ROM, the CPU being arranged in use to read the compressed instructions from the ROM, decompress the compressed instructions, and thereafter operate the apparatus in accordance with the decompressed instructions, the apparatus being further arranged in use to compress replacement or additional operating instructions and to write these compressed instructions to the ROM.

Makinen, 2:60-3:3:2

> The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program

Makinen, 3:45-50

> In order to enable the microprocessor 1 to decompress the stored compressed instructions, the flash ROM 4 additionally stores a set of instructions, in uncompressed form, which define the Pkzip decompression program. FIG. 2 shows a memory map of the ROM 4 prior to booting the computer, where a part of the memory space is occupied by the compressed instructions 6 and a part is occupied by the uncompressed Pkzip instructions 7. In FIG. 2, the ROM 4 is shown coupled to the microprocessor 1, as is the RAM 3 which at this stage remains empty.

Makinen, 3:56-65

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:9

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:18-32

See also Makinen, Figs. 2-4

14.4 updating the boot data list.	Makinen, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Makinen, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1 (Preamble) and 1.1 above.

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:9

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-4:17

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

> It is noted that the compression ratio is maximised when the entire set of operating instructions 6,8 are compressed as a single 'unit'. However, it is also possible to compress the instructions in smaller sub-units, in which case only those units which have been amended need be compressed and written to the flash ROM 4.

Makinen, 32-37

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Makinen, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 15 above.

> At the heart of most modern electronic devices is a microprocessor or central processing unit which operates in accordance with a set of software operating instructions which together form an executable program. The instructions are stored in a digital memory which may be internal to the microprocessor or, as is more usually the case, externally connected to the microprocessor. The set of operating instructions generally define the basic input/output system (BIOS) of the microprocessor together with device drivers, libraries, and user applications.

Makinen, 1:10-19

> The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program. The compressed instructions occupy considerably less flash ROM space than would the corresponding uncompressed instructions, e.g. ½ to ⅕ of the memory space. Such a high compression ratio results from the particular structure of RISC instructions.

Makinen, 3:46-55

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Makinen, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1 and 1.2 above

> Compared to conventional ROM, flash ROM remains expensive. There therefore exists a desire to reduce the amount of flash ROM used in individual products. Furthermore, accessing flash ROM is relatively slow, significantly reducing the performance of electronic devices. Conventional computer architectures rely upon a complex instruction set computer (CISC) architecture. This utilises a large number (e.g. 1000) of instructions which define very specific tasks. The instructions are of variable length and are decoded by the computer's CPU before execution. There is described in Japanese non-examined patent publication no. 55-131848 a data processing unit which comprises a central processing unit, a main memory unit, and an external memory unit. A compressed program is stored in the external memory unit and, before commencing processing operations, the compressed program is read from the external memory unit in blocks and decompressed. The decompressed program is then written to the main memory unit. However, the compression ratio which can be achieved with CISC code is relatively low and it is not believed that the disclosure of JP-131848 has been widely used.

Makinen, 1:34-55

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last

> operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code. Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.

Makinen, 3:66-4:28

> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.

Makinen, 4:18-4:31

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Makinen, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 2 above.

> Compared to conventional ROM, flash ROM remains expensive. There therefore exists a desire to reduce the amount of flash ROM used in individual products. Furthermore, accessing flash ROM is relatively slow, significantly reducing the performance of electronic devices. Conventional computer architectures rely upon a complex instruction set computer (CISC) architecture. This utilises a large number (e.g. 1000) of instructions which define very specific tasks. The instructions are of variable length and are decoded by the computer's CPU before execution. There is described in Japanese non-examined patent publication no. 55-131848 a data processing unit which comprises a central processing unit, a main memory unit, and an external memory unit. A compressed program is stored in the external memory unit and, before commencing processing operations, the compressed program is read from the external memory unit in blocks and decompressed. The decompressed program is then written to the main memory unit. However, the compression ratio which can be achieved with CISC code is relatively low and it is not believed that the disclosure of JP-131848 has been widely used.

Makinen, 1:34-55

> Upon booting of the computer, the microprocessor 1 is directed by hardwired logic to read the first instruction of the decompressed instruction set 7, from the ROM 4. Thereafter the microprocessor is directed, by the decompressed set, to read and decompress the compressed instruction set 6. The expanded instruction set 6′ is then stored in the RAM 3 as shown in FIG. 3. The last operation that the set of decompression instructions perform is to cause the microprocessor 1 to jump to the first instruction in the decompressed code.

> Thereafter, the computer is operated in accordance with the decompressed operating instructions 6′.
>
> Makinen, 3:66-4:28
>
> In order to reduce memory requirements, the corresponding Pkzip instructions may be stored in compressed form 8 as illustrated in FIG. 4. During booting, the compressed Pkzip instructions 8 are read from the flash ROM 4 by the microprocessor 1, decompressed, and stored in the RAM 3 as decompressed instructions 8′ together with the decompressed operating instructions 6′ (FIG. 4). Alternatively, the compressed Pkzip instructions 8 may only be read from the flash ROM 4, and subsequently decompressed, when a specific request is made to alter the compressed instructions 6,8. In either case, the microprocessor 1 employs the decompressed Pkzip method to compress an amended version of the operating instructions 6′,8′ and writes the compressed instructions to the corresponding areas of the flash ROM.
>
> Makinen, 4:18-4:31

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1 and 1.2 above

> A computer having a reduced instruction computer (RISC) architecture has a RISC central processing unit (CPU)(1) coupled to a RAM memory (3) and to a flash ROM memory (4). A set of compressed operating instructions (6,8), including a subset defining a compression method (8), are stored in the flash ROM (4) together with a set of uncompressed instructions (7) defining a compression algorithm. Upon booting of the computer, the uncompressed instructions (7) are read from the ROM (4) by the CPU (1) which then also reads the compressed instructions (6,8), decompresses them according to the decompression process (7), and writes the decompressed instructions (6′,8′) to the RAM (3). The compressed instructions (6,8) can be dynamically altered by the CPU (1), by generating an altered set of uncompressed instructions, compressing these in accordance with the now decompressed compression method (8′), and writing these to the flash ROM (4).

Makinen, Abstract

> Software instructions defining the basic operation of a microprocessor are usually stored in non-volatile read only memory (ROM). Until recently, the preferred choice for storing operating instructions was UV light erasable programmable read only memory. More recently however, the preferred choice for storing operating instructions, especially in embedded devices (e.g. mobile phones, personal digital assistants, etc) has become flash ROM. Flash ROM is both non-volatile and electrically erasable and is used, to a large extent, because it can be programmed following assembly of the PCB containing the flash ROM oar of the completed device. A further advantage is that it is possible to upgrade operating instruction stored in the flash ROM at some future date.

Makinen, 1:20-33

Compared to conventional ROM, flash ROM remains expensive. There therefore exists a desire to reduce the amount of flash ROM used in individual products. Furthermore, accessing flash ROM is relatively slow, significantly reducing the performance of electronic devices. Conventional computer architectures rely upon a complex instruction set computer (CISC) architecture. This utilises a large number (e.g. 1000) of instructions which define very specific tasks. The instructions are of variable length and are decoded by the computer's CPU before execution. There is described in Japanese non-examined patent publication no. 55-131848 a data processing unit which comprises a central processing unit, a main memory unit, and an external memory unit. A compressed program is stored in the external memory unit and, before commencing processing operations, the compressed program is read from the external memory unit in blocks and decompressed. The decompressed program is then written to the main memory unit. However, the compression ratio which can be achieved with CISC code is relatively low and it is not believed that the disclosure of JP-131848 has been widely used.

Makinen, 1:34-55

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1 and 1.2 above

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:9-17

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 1.1 above	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Makinen, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1 and 32 above.

> The flash ROM 4 is used to store a set of RISC operating instructions which define the basic input/output system (BIOS) of the microprocessor as well as the device drivers, libraries, and user applications. The instructions are in compressed form, having previously been compressed using the Pkzip compression program. The compressed instructions occupy considerably less flash ROM space than would the corresponding uncompressed instructions, e.g. ½ to ⅕ of the memory space. Such a high compression ratio results from the particular structure of RISC instructions.

Makinen, 3:46-55

> It will be appreciated by the person of skill in the art that other modifications may be made to the embodiments described above without departing from the scope of the present invention. For example, compression algorithms other than Pkzip may be used, including Gzip-9, Zoo-a, and Arj-a. Compression methods may also be optimised for ARM (Trade Mark) processors.

Makinen, 4:38-45

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1 and 1.2 above

> Compared to conventional ROM, flash ROM remains expensive. There therefore exists a desire to reduce the amount of flash ROM used in individual products. Furthermore, accessing flash ROM is relatively slow, significantly reducing the performance of electronic devices. Conventional computer architectures rely upon a complex instruction set computer (CISC) architecture. This utilises a large number (e.g. 1000) of instructions which define very specific tasks. The instructions are of variable length and are decoded by the computer's CPU before execution. There is described in Japanese non-examined patent publication no. 55-131848 a data processing unit which comprises a central processing unit, a main memory unit, and an external memory unit. A compressed program is stored in the external memory unit and, before commencing processing operations, the compressed program is read from the external memory unit in blocks and decompressed. The decompressed program is then written to the main memory unit. However, the compression ratio which can be achieved with CISC code is relatively low and it is not believed that the disclosure of JP-131848 has been widely used.

Makinen 1:34-55

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claim 10 above.

> Compared to conventional ROM, flash ROM remains expensive. There therefore exists a desire to reduce the amount of flash ROM used in individual products. Furthermore, accessing flash ROM is relatively slow, significantly reducing the performance of electronic devices. Conventional computer architectures rely upon a complex instruction set computer (CISC) architecture. This utilises a large number (e.g. 1000) of instructions which define very specific tasks. The instructions are of variable length and are decoded by the computer's CPU before execution. There is described in Japanese non-examined patent publication no. 55-131848 a data processing unit which comprises a central processing unit, a main memory unit, and an external memory unit. A compressed program is stored in the external memory unit and, before commencing processing operations, the compressed program is read from the external memory unit in blocks and decompressed. The decompressed program is then written to the main memory unit. However, the compression ratio which can be achieved with CISC code is relatively low and it is not believed that the disclosure of JP-131848 has been widely used.

Makinen, 1:34-55

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

Compared to conventional ROM, flash ROM remains expensive. There therefore exists a desire to reduce the amount of flash ROM used in individual products. Furthermore, accessing flash ROM is relatively slow, significantly reducing the performance of electronic devices. Conventional computer architectures rely upon a complex instruction set computer (CISC) architecture. This utilises a large number (e.g. 1000) of instructions which define very specific tasks. The instructions are of variable length and are decoded by the computer's CPU before execution. There is described in Japanese non-examined patent publication no. 55-131848 a data processing unit which comprises a central processing unit, a main memory unit, and an external memory unit. A compressed program is stored in the external memory unit and, before commencing processing operations, the compressed program is read from the external memory unit in blocks and decompressed. The decompressed program is then written to the main memory unit. However, the compression ratio which can be achieved with CISC code is relatively low and it is not believed that the disclosure of JP-131848 has been widely used.

Makinen, 1:34-55

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Makinen, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Makinen, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Makinen, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Makinen, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Makinen, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Makinen, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 32, 33 and 49 above.	

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Makinen, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claim 32, 33, and 49 above.	

Makinen **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Makinen, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Makinen discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Makinen, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

> A computer having a reduced instruction computer (RISC) architecture has a RISC central processing unit (CPU)(1) coupled to a RAM memory (3) and to a flash ROM memory (4). A set of compressed operating instructions (6,8), including a subset defining a compression method (8), are stored in the flash ROM (4) together with a set of uncompressed instructions (7) defining a compression algorithm. Upon booting of the computer, the uncompressed instructions (7) are read from the ROM (4) by the CPU (1) which then also reads the compressed instructions (6,8), decompresses them according to the decompression process (7), and writes the decompressed instructions (6′,8′) to the RAM (3). The compressed instructions (6,8) can be dynamically altered by the CPU (1), by generating an altered set of uncompressed instructions, compressing these in accordance with the now decompressed compression method (8′), and writing these to the flash ROM (4).

Makinen, Abstract

> It will be appreciated that the operating system of the computer may be altered by directly accessing the flash ROM 5 to erase and/or rewrite compressed instructions 6 stored therein. However, in some circumstances it may be desirable for the end-user to be able to alter the compressed operating instructions 6, or indeed for the computer itself to be able to 'dynamically' alter the instructions. To this end, the Pkzip compression method may also be stored in the flash ROM 4.

Makinen, 4:10-17

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Makinen, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

> Software instructions defining the basic operation of a microprocessor are usually stored in non-volatile read only memory (ROM). Until recently, the preferred choice for storing operating instructions was UV light erasable programmable read only memory. More recently however, the preferred choice for storing operating instructions, especially in embedded devices (e.g. mobile phones, personal digital assistants, etc) has become flash ROM. Flash ROM is both non-volatile and electrically erasable and is used, to a large extent, because it can be programmed following assembly of the PCB containing the flash ROM oar of the completed device. A further advantage is that it is possible to upgrade operating instruction stored in the flash ROM at some future date.

Makinen, 1:20-33

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Makinen, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 1.1, 5, 9.1 and 8 above.

> The above second aspect of the present invention makes it possible to amend the stored compressed instructions in a dynamic manner. This may, for example, allow a user to configure the computer according to his specific needs.
>
> Preferably, the method comprises the step of reading a set of operating instructions from the ROM into the CPU, which instructions define a program for compressing said replacement or additional instruction(s). More preferably, the instructions defining the compression program form part of said set of compressed operating instructions.

Makinen, 2:29-39

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Makinen, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Makinen discloses this limitation:

See Claims 5.1, 8.1 and 9.1 above.

> According to a second aspect of the present invention there is provided a method of operating apparatus having a central processing unit (CPU) and a read only memory (ROM), the method comprising reading a set of compressed operating instructions from the ROM into the CPU, decompressing the compressed instructions in the CPU, and thereafter operating the apparatus in accordance with the decompressed instructions, the method further comprising generating one or more replacement or additional compressed instructions in the CPU and writing the compressed instruction(s) to the ROM.

Makinen, 2:18-28

> According to a fourth aspect of the present invention there is provided apparatus comprising a central processing unit (CPU), a read only memory (ROM), and a set of compressed operating instructions stored in the ROM, the CPU being arranged in use to read the compressed instructions from the ROM, decompress the compressed instructions, and thereafter operate the apparatus in accordance with the decompressed instructions, the apparatus being further arranged in use to compress replacement or additional operating instructions and to write these compressed instructions to the ROM.

Makinen, 2:60-3:2

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

U.S. Patent No. 5,793,943 to Noll ("Noll") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Noll, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the following detailed description and accompanying figures."

Noll, 1:29-42.

> "The present invention is embodied in a system for the automatic

> recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Noll, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable

> the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."
>
> Noll, 1:44-63.
>
> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."
>
> Noll, 4:66-5:2.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Noll, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the

> second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Noll, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the

following detailed description and accompanying figures."

Noll, 1:29-42.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

1.4.1 updating the boot data list,	Noll, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Noll, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the

following detailed description and accompanying figures."

Noll, 1:29-42.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Noll, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 1.4.1 above.

> The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory.

Noll, 1:14-26

> In one embodiment, the first BIOS memory is programmable and has a programming enable input. The error detection circuit verifies that the second BIOS memory does not have altered data and reprograms the first BIOS memory using the computer instructions stored within the second BIOS memory. As part of the computer instructions to initialize the computer, the computer executes a power-on self-test (POST). During the POST, the computer copies the series of computer instructions from the second BIOS memory to a random access memory (RAM). In this embodiment, the system includes a memory loader to copy the series of instructions to the RAM, and the first BIOS memory is reprogrammed using the series of computer instructions that have been copied into the RAM. The first BIOS memory may be a flash

programmable read-only memory. Although not necessary, the second BIOS memory may also be a flash-programmable read-only memory.

Noll, 1:64-2:13

An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 5:2-24

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Noll, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 1.4.1 above.

> FIGS. 2A and 2B are flowcharts illustrating the software controlling the operation of the CPU 12 in the preferred embodiment of the present invention. At a start 200, the computer 10 (see FIG. 1) has been turned on, or reset such that the computer will undergo an initialization. In step 201, the CPU 12 resets the error flag 20. In step 202, the CPU 12 checks to determine whether the primary BIOS ROM 22 contains a data error. If there is no data error, the result of decision 202 is NO. In that event, the computer 10 continues with the POST procedure in step 204 and boots the computer 10 in the conventional fashion. If the primary BIOS ROM 22 contains an error, the result of decision 202 is YES. In that event, the CPU 12 sets the error flag 20 in step 206. In step 210, the CPU 12 causes the chip enable circuit 36 to change the level of the ROMSEL2 control line 42, thus disabling operation of the primary BIOS ROM 22 and enabling operating of the secondary BIOS ROM 30.

Noll, 6:65- 7:14

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Noll, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.4.1, and 2 above.

> An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 5:2-24

5 (Preamble) A method for booting a computer system, the method comprising:	Noll, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1-1.4.2 above.

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the

> second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the following detailed description and accompanying figures."

Noll, 1:29-42.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Noll, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 1.1 above.

> Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it. An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 4:66-5:24

5.2 loading the stored compressed boot data from the first memory;	Noll, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Noll, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Noll, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Noll, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the

following detailed description and accompanying figures."

Noll, 1:29-42.

"Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

5.6 updating the boot data list;	Noll, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Noll, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1-1.3 above.

> Thus, the instructions contained within the primary BIOS ROM 22 are copied into the RAM 14 so that the instructions may be executed more rapidly by the CPU 12. Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it.

Noll, 4:63-5:2

6 (Preamble) A system comprising: a processor;	Noll, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the following detailed description and accompanying figures."

Noll, 1:29-42.

6.1 a memory; and	Noll, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Noll, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.1, 1.2, and Claim 6 (Preamble) above.

> A typical computer contains a single BIOS ROM (not shown) that contains a series of instructions to initialize the computer. If the BIOS ROM fails for whatever reason, the computer will not function properly. In contrast, the present invention has a backup, or secondary BIOS ROM that is automatically enabled upon detection of an error in the first or primary BIOS ROM. At the same time, the primary BIOS ROM is disabled. While the examples presented herein illustrate separate devices for the primary and secondary BIOS ROMs, it is possible to have a single memory device with one portion used as the primary BIOS and a second portion used as the secondary BIOS.

Noll, 2:33-44

> The computer 10 also includes a secondary or backup BIOS ROM 30. The secondary BIOS ROM could be the same chip type as the primary BIOS ROM 22 to reduce the number of different components required to assemble the computer 10. However, successful operation of the computer 10 does not require that the secondary BIOS ROM 30 be programmable. The secondary BIOS ROM 30 also has a chip enable input 32 to activate operation of the secondary BIOS ROM.

Noll, 3:17-25

> Thus, the instructions contained within the primary BIOS ROM 22 are copied into the RAM 14 so that the instructions may be executed more rapidly by the CPU 12. Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it.

Noll, 4:63-5:2

6.3 wherein the processor is configured:	Noll, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 6 (Preamble) above	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Noll, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Noll, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Noll, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Noll, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.4.1 above.	

Noll Claim 6.7
"to update the boot data list."

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Noll, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Noll, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Noll, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Noll, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Noll, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Noll, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the

following detailed description and accompanying figures."

Noll, 1:29-42.

"Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

8.6 updating the boot data list,	Noll, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.4.1 above.	

Noll Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Noll, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1-1.3 and 5.7 above.

> Thus, the instructions contained within the primary BIOS ROM 22 are copied into the RAM 14 so that the instructions may be executed more rapidly by the CPU 12. Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it.

Noll, 4:63-5:2

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Noll, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 1.1 above.

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the

> computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

9.2 storing the additional portion of the operating system in the first memory, and	Noll, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 8.1 above.

> In one embodiment, the present invention is embodied in a computer 10 illustrated in the functional block diagram of FIG. 1. The computer 10 includes a central processing unit (CPU) 12, such as an Intel Pentium® microprocessor. The computer 10 also includes a random access memory (RAM) 14 and a low power RAM 18, such as a CMOS RAM. The CMOS RAM 18 is used to store configuration data for the computer 10 and is typically powered by a back-up battery (not shown) when the computer is turned off. As will be discussed in detail below, a memory location in the CMOS RAM 18 stores an error flag 20 as an error signal that is written into the CMOS RAM 18 when the computer 10 detects errors. However, the error flag 20 can be stored in the RAM 14 or other suitable storage location.

Noll, 1:64-2:13

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Noll, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 8.5 above.

> In one embodiment, the present invention is embodied in a computer 10 illustrated in the functional block diagram of FIG. 1. The computer 10 includes a central processing unit (CPU) 12, such as an Intel Pentium® microprocessor. The computer 10 also includes a random access memory (RAM) 14 and a low power RAM 18, such as a CMOS RAM. The CMOS RAM 18 is used to store configuration data for the computer 10 and is typically powered by a back-up battery (not shown) when the computer is turned off. As will be discussed in detail below, a memory location in the CMOS RAM 18 stores an error flag 20 as an error signal that is written into the CMOS RAM 18 when the computer 10 detects errors. However, the error flag 20 can be stored in the RAM 14 or other suitable storage location.

Noll, 1:64-2:13

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Noll, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 9.1 above.	

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Noll, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Noll, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 1.1 and 6(Preamble) above.

> As discussed above, the contents of the primary BIOS ROM 22 are copied into the low portion of the RAM 14 as part of the shadowing POST procedure. However, if an error was detected in the primary BIOS ROM 22, the secondary BIOS ROM 30 is activated. Thus, it is the data contents of the secondary BIOS ROM 30 that are copied into the low portion of the RAM 14. Initialization and operation of the computer 10 proceed as described above with respect to the primary BIOS ROM 22, with the exception that the instructions themselves are provided by the secondary BIOS ROM 30. The error detection circuit 54 also performs an error detection test on the secondary BIOS ROM 30 using the definition of error previously discussed with respect to the primary BIOS ROM 22. If errors are detected in both the primary BIOS ROM 22 and the secondary BIOS ROM 30, the computer 10 cannot be properly initialized. However, a simultaneous failure of both devices is not likely to occur.

Noll, 6:1-17

11.2 accessing the loaded boot data in compressed form from the memory;	Noll, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Noll, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Noll, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system, such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the

following detailed description and accompanying figures."

Noll, 1:29-42.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

11.5 updating the boot data list.	Noll, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Noll, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Noll, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Noll, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Noll, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Noll, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Noll, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Noll, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.1 and 1.2 above.

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is

included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory."

Noll, 1:44-63.

14.2 loading the boot data into a memory; and	Noll, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Noll, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.2 and 1.3 above.

> "A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself."

Noll, Abstract.

> "For example, when the computer is first powered up or reset, a software program, typically designated as a "basic input-output system" (BIOS) initializes the computer and permits the startup of an operating system,

> such as Microsoft MS-DOS®. The BIOS program typically resides in a read-only memory (ROM). If the BIOS ROM is defective for any reason, the computer will not function properly. Therefore, it can be appreciated that there is a significant need for a system to recover from a BIOS ROM failure in a manner that does not require user intervention. The present invention provides this and other advantages as will be apparent from the following detailed description and accompanying figures."

Noll, 1:29-42.

> "Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it."

Noll, 4:66-5:2.

14.4 updating the boot data list.	Noll, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Noll, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1 (Preamble) and 1.1 above.

> A computer system includes a dual basic input-output system (BIOS) read-only memory (ROM) system to initialize the computer. When the computer is first powered on or reset, the primary BIOS ROM is initially enabled. The computer analyzes the entire program contents of the primary BIOS memory to detect data errors. If a data error is detected, a chip enable circuit disables the primary BIOS ROM and enables a secondary BIOS ROM containing essentially the same initialization instructions as the primary BIOS ROM. If no errors are detected in the secondary BIOS ROM, the initialization of the computer proceeds using the secondary BIOS ROM. As part of the initialization procedure, the contents of the secondary BIOS ROM are copied to a random access memory. The primary BIOS ROM can then be reprogrammed with the contents of the secondary BIOS ROM using the copy in random access memory, or from the secondary BIOS ROM itself.

Noll, Abstract

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Noll, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Noll, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.1 and 1.2 above

> The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory.

Noll, 1:14-26

> In one embodiment, the first BIOS memory is programmable and has a programming enable input. The error detection circuit verifies that the second BIOS memory does not have altered data and reprograms the first BIOS memory using the computer instructions stored within the second BIOS memory. As part of the computer instructions to initialize the computer, the computer executes a power-on self-test (POST). During the POST, the computer copies the series of computer instructions from the second BIOS

> memory to a random access memory (RAM). In this embodiment, the system includes a memory loader to copy the series of instructions to the RAM, and the first BIOS memory is reprogrammed using the series of computer instructions that have been copied into the RAM. The first BIOS memory may be a flash programmable read-only memory. Although not necessary, the second BIOS memory may also be a flash-programmable read-only memory.

Noll, 1:64-2:13

> An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 5:2-24

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Noll, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claim 2 above.

> The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory.

Noll, 1:14-26

> In one embodiment, the first BIOS memory is programmable and has a programming enable input. The error detection circuit verifies that the second BIOS memory does not have altered data and reprograms the first BIOS memory using the computer instructions stored within the second BIOS memory. As part of the computer instructions to initialize the computer, the computer executes a power-on self-test (POST). During the POST, the computer copies the series of computer instructions from the second BIOS memory to a random access memory (RAM). In this embodiment, the system includes a memory loader to copy the series of instructions to the RAM, and the first BIOS memory is reprogrammed using the series of computer instructions that have been copied into the RAM. The first BIOS memory may be a flash

programmable read-only memory. Although not necessary, the second BIOS memory may also be a flash-programmable read-only memory.

Noll, 1:64-2:13

An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 5:2-24

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.1 and 1.2 above

> In one embodiment, the first BIOS memory is programmable and has a programming enable input. The error detection circuit verifies that the second BIOS memory does not have altered data and reprograms the first BIOS memory using the computer instructions stored within the second BIOS memory. As part of the computer instructions to initialize the computer, the computer executes a power-on self-test (POST). During the POST, the computer copies the series of computer instructions from the second BIOS memory to a random access memory (RAM). In this embodiment, the system includes a memory loader to copy the series of instructions to the RAM, and the first BIOS memory is reprogrammed using the series of computer instructions that have been copied into the RAM. The first BIOS memory may be a flash programmable read-only memory. Although not necessary, the second BIOS memory may also be a flash-programmable read-only memory.

Noll, 1:64-2:13

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.1 and 1.2 above

> As part of the normal setup procedures, instructions contained within the primary BIOS ROM 22 cause the CPU 12 to perform a "Power-On Self Test" (POST). The POST procedure may include routines such as a check of the RAM 14, peripheral devices (not shown), and the like. The POST procedure is well known to those of ordinary skill in the art, and need not be described in detail herein. As part of the POST procedure, the data in the primary BIOS ROM 22 is copied to the RAM 14. This process is sometimes known as shadowing to reflect the fact that a complete copy of the data in the primary BIOS ROM 22 is copied or shadowed into the RAM 14. One reason for copying the primary BIOS ROM 22 into the RAM 14 is that the access time for the RAM is typically much less than the access time for the primary BIOS ROM. Thus, the instructions contained within the primary BIOS ROM 22 are copied into the RAM 14 so that the instructions may be executed more rapidly by the CPU 12. Another reason that the primary BIOS ROM 22 is copied into the RAM 14 is that some data in the primary BIOS ROM may be in a compressed format and must be decompressed before the CPU 12 can use it. An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not

shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 4:49-5:24

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 1.1 above	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Noll, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 1.1 and 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.1 and 1.2 above

> A typical computer contains a single BIOS ROM (not shown) that contains a series of instructions to initialize the computer. If the BIOS ROM fails for whatever reason, the computer will not function properly. In contrast, the present invention has a backup, or secondary BIOS ROM that is automatically enabled upon detection of an error in the first or primary BIOS ROM. At the same time, the primary BIOS ROM is disabled. While the examples presented herein illustrate separate devices for the primary and secondary BIOS ROMs, it is possible to have a single memory device with one portion used as the primary BIOS and a second portion used as the secondary BIOS.

Noll, 2:33-44

> The computer 10 also includes a secondary or backup BIOS ROM 30. The secondary BIOS ROM could be the same chip type as the primary BIOS ROM 22 to reduce the number of different components required to assemble the computer 10. However, successful operation of the computer 10 does not require that the secondary BIOS ROM 30 be programmable. The secondary BIOS ROM 30 also has a chip enable input 32 to activate operation of the secondary BIOS ROM.

Noll, 3:17-25

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation:	

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Noll, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

Appendix C14
Invalidity of U.S. Patent 8,880,862 based on Noll

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Noll, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Noll, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Noll, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Noll, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Noll, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 32, 33 and 49 above.	

Noll **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 31 above.	

Noll **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 32, 33 and 49 above.	

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Noll, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claim 32, 33, and 49 above.	

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Noll, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Noll, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Noll, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

> The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory.

Noll, 1:14-26

> In one embodiment, the first BIOS memory is programmable and has a programming enable input. The error detection circuit verifies that the second BIOS memory does not have altered data and reprograms the first BIOS memory using the computer instructions stored within the second BIOS memory. As part of the computer instructions to initialize the computer, the computer executes a power-on self-test (POST). During the POST, the computer copies the series of computer instructions from the second BIOS memory to a random access memory (RAM). In this embodiment, the system includes a memory loader to copy the series of instructions to the RAM, and the

first BIOS memory is reprogrammed using the series of computer instructions that have been copied into the RAM. The first BIOS memory may be a flash programmable read-only memory. Although not necessary, the second BIOS memory may also be a flash-programmable read-only memory.

Noll, 1:64-2:13

An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 5:2-24

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Noll, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Noll discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

> The present invention is embodied in a system for the automatic recovery of a BIOS ROM failure. In one embodiment, the system includes a first BIOS memory that contains a series of computer instructions to initialize the computer. The first BIOS memory has a chip enable input that is initially enabled. An error detection circuit analyzes data contained within the first BIOS memory and detects errors therein. The error detection circuit generates an error signal upon detection of errors in the first BIOS memory. The system also includes a second BIOS memory containing the series of computer instructions to initialize the computer and also having a chip enable input. An enabling circuit is included to disable the first BIOS memory chip enable input and to enable the second BIOS memory chip enable input in response to the error signal. This effectively causes the computer system to switch to the second BIOS memory so that the series of computer instructions to initialize the computer are executed from the second BIOS memory rather the first BIOS memory.

Noll, 1:14-26

> In one embodiment, the first BIOS memory is programmable and has a programming enable input. The error detection circuit verifies that the second BIOS memory does not have altered data and reprograms the first BIOS memory using the computer instructions stored within the second BIOS memory. As part of the computer instructions to initialize the computer, the computer executes a power-on self-test (POST). During the POST, the computer copies the series of computer instructions from the second BIOS memory to a random access memory (RAM). In this embodiment, the system includes a memory loader to copy the series of instructions to the RAM, and the first BIOS memory is reprogrammed using the series of computer instructions

that have been copied into the RAM. The first BIOS memory may be a flash programmable read-only memory. Although not necessary, the second BIOS memory may also be a flash-programmable read-only memory.

Noll, 1:64-2:13

An exact copy of the data in the primary BIOS ROM 22 is initially copied into a low address portion of the RAM 14. After the primary BIOS ROM 22 has been copied into the low address portion of the RAM 14, the chip enable circuit 36 sets the ROMSEL1 control line 40 to a low level, thus disabling both the primary BIOS ROM 22 and the secondary BIOS ROM 30. The CPU 12 subsequently executes instructions from the low address portion of the RAM 14. A different portion (not shown) of the RAM 14, typically having an address space that overlap with the address space of the primary BIOS ROM 22, is designated as a "shadow RAM." Once the CPU 12 is executing instructions from the low address portion of the RAM 14, the data in the low address portion is decompressed if necessary and copied to the shadow RAM (not shown). At this point in time, the CPU 12 will begin executing instructions from the addresses in the RAM 14 that correspond with the addresses in the BIOS ROM 22 until the POST procedure is completed. The computer 10 can use the shadow RAM address space because the primary BIOS ROM 22 and the secondary BIOS ROM 30 have both been disabled by the low logic level on the ROMSEL1 control line 40. At that point, the computer 10 will boot the operating system, such as MS-DOS®.

Noll, 5:2-24

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Noll, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Noll, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Noll discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

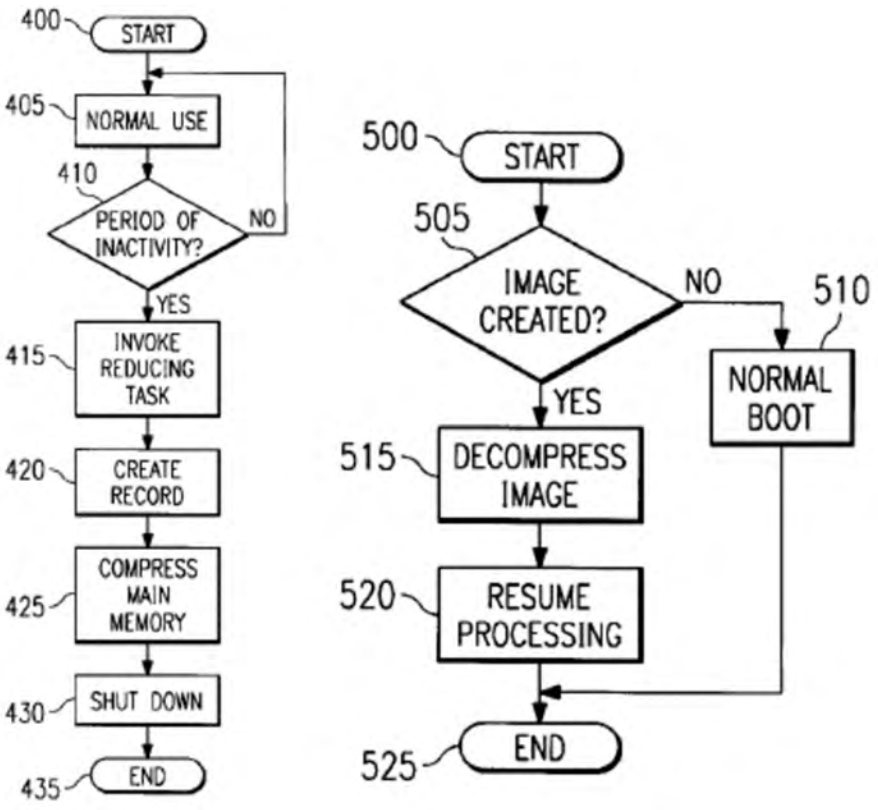
U.S. Patent No. 5,828,877 to Pearce ("Pearce") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Pearce, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:



Pearce, Fig. 4, Fig. 5.

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in

> the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it was when power was interrupted."

Pearce, 1:65-2:10.

> "Execution begins in a start block 300 wherein the portable PC 100 is started (or "booted") and proceeds to a block 305 wherein the user engages the portable PC 100 in normal use. During normal use, an operating system is loaded into the main memory 210."

Pearce, 5:52-55, Fig. 3.

See also Pearce 4:16-22.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Pearce, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

Pearce discloses this claim limitation.

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it was when power was interrupted."

Pearce, 1:65-2:10.

> "Accordingly, a first embodiment of the present invention provides, in a computer system having a CPU, a main memory (either real or virtual) divisible into allocable units, a secondary storage unit and an operating system for allocating the allocable units to tasks for use thereby, a suspend circuit for creating an optimized compressed image of data in the main memory."

Pearce, 2:36-43.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back

> into the main memory 210, allowing the CPU 200 to continue processing at the point where it stopped prior to interruption of power."
>
> Pearce, 5:36-44.
>
> "Execution begins in a start block 500 wherein the portable PC 100 boots, loading initial portions of code commonly stored in nonvolatile memory within the portable PC 100."
>
> Pearce, 8:39-41, Fig. 5.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Pearce, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it was when power was interrupted."

Pearce, 1:65-2:10.

> "Accordingly, a first embodiment of the present invention provides, in a computer system having a CPU, a main memory (either real or virtual) divisible into allocable units, a secondary storage unit and an operating system for allocating the allocable units to tasks for use thereby, a suspend circuit for creating an optimized compressed image of data in the main memory."

Pearce, 2:36-43.

> "Once the contents of main memory have been compressed and stored as an image in the secondary storage device, power to the computer system is interrupted."

Pearce, 2:57-59.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power

> to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back into the main memory 210, allowing the CPU 200 to continue processing at the point where it stopped prior to interruption of power."
>
> Pearce, 5:36-44.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Pearce, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it was when power was interrupted."

Pearce, 1:65-2:10.

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back into the main memory 210, allowing the CPU 200 to continue processing

at the point where it stopped prior to interruption of power."

Pearce, 5:36-44.

> "If a compressed image file is detected, execution proceeds to a block 515, Wherein a corresponding (e.g., run length decoding) decompression routine decompresses and restores the main memory 210 to the state in which it was prior to shutdown. In the first embodiment, the units previously allocated to the reducing task remain filled with the bit pattern. In the second embodiment, the stored record is retrieved and used as a guide by the decompression routine to restore the compressed allocable units to their proper position in main memory."

Pearce, 8:52-61, Fig. 5.

1.4.1 updating the boot data list,	Pearce, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "As a part of the block 515, the units previously allocated to the reducing task are marked as unallocated and therefore free for subsequent allocation by the operating system."

Pearce, 8:67-9:3, Fig. 5.

See also Pearce 4:16-22.

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Pearce, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Pearce, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Pearce, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Pearce, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Pearce, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1-1.4.2 above.

See also

> "During compression of the main memory, the record containing the identity of the allocable units compressed is also stored for use by the main memory restoration method detailed in FIG. 5."

Pearce, 8:27-30, Fig. 5.

> "As a part of the block 515, the units previously allocated to the reducing task are marked as unallocated and therefore free for subsequent allocation by the operating system."

Pearce, 8:67-9:3, Fig. 5.

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Pearce, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claim 1.1 above.

See also

> "The present invention is directed, in general, to computer systems and, more particularly, to a circuit and method for conserving power used by the computer system by storing a compressed image of the contents of the computer system's main memory on a nonvolatile secondary storage device, allowing power to the main memory to be interrupted."

Pearce, 1:6-11.

5.2 loading the stored compressed boot data from the first memory;	Pearce, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Pearce, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Pearce, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Pearce, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back into the main memory 210, allowing the CPU 200 to continue processing

at the point where it stopped prior to interruption of power."

Pearce, 5:36-44.

> "If a compressed image file is detected, execution proceeds to a block 515, wherein a corresponding (e.g., run length decoding) decompression routine decompresses and restores the main memory 210 to the state in which it was prior to shutdown. In the first embodiment, the units previously allocated to the reducing task remain filled with the bit pattern. In the second embodiment, the stored record is retrieved and used as a guide by the decompression routine to restore the compressed allocable units to their proper position in main memory."

Pearce, 8:52-61, Fig. 5.

See also Pearce 4:16-22.

5.6 updating the boot data list;	Pearce, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Pearce, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

6 (Preamble) A system comprising: a processor;	Pearce, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

6.1 a memory; and	Pearce, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Pearce, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

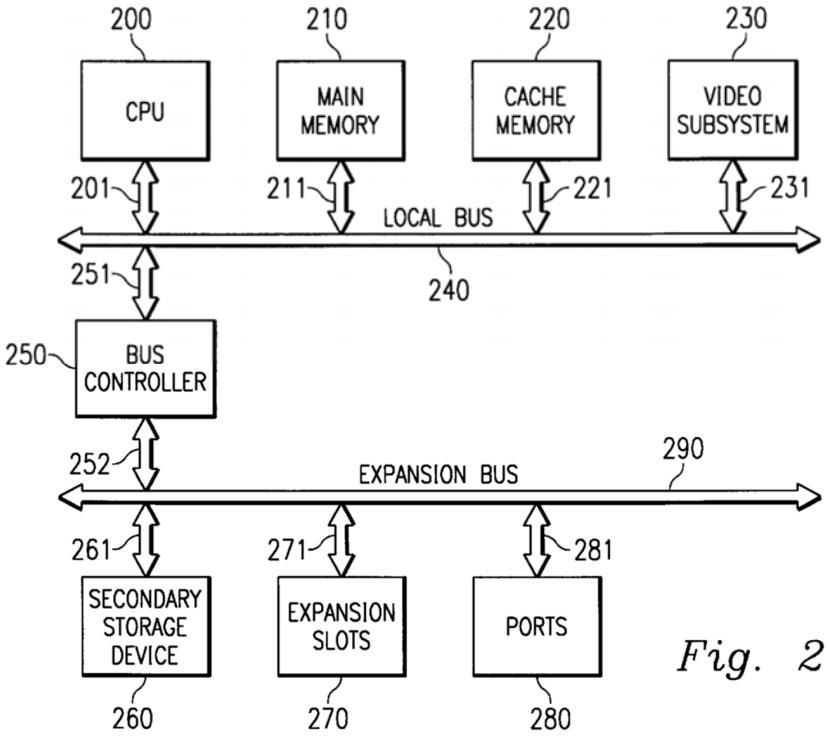


Fig. 2.

"The present invention is directed, in general, to computer systems and, more particularly, to a circuit and method for conserving power used by the computer system by storing a compressed image of the contents of

> the computer system's main memory on a nonvolatile secondary storage device, allowing power to the main memory to be interrupted."

Pearce, 1:6-11.

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

6.3 wherein the processor is configured:	Pearce, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Pearce, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Pearce, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.2 above.	

Pearce Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Pearce, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Pearce, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

Pearce Claim 6.7
"to update the boot data list."

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Pearce, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Pearce, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Pearce, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Pearce, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Pearce, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Pearce, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back

> into the main memory 210, allowing the CPU 200 to continue processing at the point where it stopped prior to interruption of power."

Pearce, 5:36-44.

> "If a compressed image file is detected, execution proceeds to a block 515, wherein a corresponding (e.g., run length decoding) decompression routine decompresses and restores the main memory 210 to the state in which it was prior to shutdown. In the first embodiment, the units previously allocated to the reducing task remain filled with the bit pattern. In the second embodiment, the stored record is retrieved and used as a guide by the decompression routine to restore the compressed allocable units to their proper position in main memory."

Pearce, 8:52-61, Fig. 5.

See also Pearce 4:16-22.

8.6 updating the boot data list,	Pearce, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

Pearce Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Pearce, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Pearce, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Pearce, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Pearce, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claim 9.1 above.

> "(3) a circuit for executing the data compression process to store a compressed image of the main memory in the secondary storage unit, the bit pattern allowing a size of the compressed image to be reduced and a time required to compress and store the compressed image to be minimized."

Pearce, 2:49-53. See also Pearce, 3:36-42, 9:25-31.

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Pearce, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Pearce, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claim 1.1 above.

See also ***Look for additional references where loading occurs upon initialization of the computer system***

See also ***Add disclosure from '608 Patent Claim 1.2***

> "Execution begins in a start block 500 Wherein the portable PC 100 boots, loading initial portions of code commonly stored in nonvolatile memory Within the portable PC 100."

Pearce, 8:39-41.

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

11.2 accessing the loaded boot data in compressed form from the memory;	Pearce, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Pearce, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Pearce, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it Was When power was interrupted."

Pearce, 1:65-2:10.

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back into the main memory 210, allowing the CPU 200 to continue processing at the point where it stopped prior to interruption of power."

Pearce, 5:36-44.

> "If a compressed image file is detected, execution proceeds to a block 515, wherein a corresponding (e.g., run length decoding) decompression routine decompresses and restores the main memory 210 to the state in which it was prior to shutdown. In the first embodiment, the units previously allocated to the reducing task remain filled with the bit pattern. In the second embodiment, the stored record is retrieved and used as a guide by the decompression routine to restore the compressed allocable units to their proper position in main memory."

Pearce, 8:52-61, Fig. 5.

See also Pearce 4:16-22.

11.5 updating the boot data list.	Pearce, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Pearce, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Pearce, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Pearce, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Pearce, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Pearce, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Pearce, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Pearce, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.1 and 1.2 above.

> "During compression of the main memory, the record containing the identity of the allocable units compressed is also stored for use by the main memory restoration method detailed in FIG. 5."

Pearce, 8:27-30, Fig. 5.

> "As a part of the block 515, the units previously allocated to the reducing task are marked as unallocated and therefore free for subsequent allocation by the operating system."

Pearce, 8:67-9:3, Fig. 5.

14.2 loading the boot data into a memory; and	Pearce, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Pearce, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

> "The most effective solution to power saving following a long period of user inactivity is to turn off the PC completely. However, it is unacceptable simply to interrupt has not yet been stored in non-volatile memory, particularly the hard disk drive. Rather, it is advantageous to store the contents of the volatile main memory of the PC as an image in the nonvolatile secondary storage device (a so-called "suspend-to-disk" or "STD"). When the user restores power to the PC, the image is restored to the main memory (a "resume-from-disk" or "RFD"), placing the PC in exactly the same functional state as it was when power was interrupted."

Pearce, 1:65-2:10.

> "Accordingly, a first embodiment of the present invention provides, in a computer system having a CPU, a main memory (either real or virtual) divisible into allocable units, a secondary storage unit and an operating system for allocating the allocable units to tasks for use thereby, a suspend circuit for creating an optimized compressed image of data in the

main memory."

Pearce, 2:36-43.

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "The present invention is designed to conserve computer power by storing the pertinent contents of the main memory 210 as a compressed image in the nonvolatile secondary storage device 260, allowing power to be interrupted to the portable PC 100 in its entirety. When the user so directs, power is restored to the PC, and the image is decompressed back into the main memory 210, allowing the CPU 200 to continue processing at the point where it stopped prior to interruption of power."

Pearce, 5:36-44.

> "If a compressed image file is detected, execution proceeds to a block 515, Wherein a corresponding (e.g., run length decoding) decompression routine decompresses and restores the main memory 210 to the state in which it was prior to shutdown. In the first embodiment, the units previously allocated to the reducing task remain filled with the bit pattern. In the second embodiment, the stored record is retrieved and used as a guide by the decompression routine to restore the compressed allocable units to their proper position in main memory."

Pearce, 8:52-61, Fig. 5.

14.4 updating the boot data list.	Pearce, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Pearce, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See ***Add disclosure where operating system comprises a plurality of files***

> "Accordingly, a first embodiment of the present invention provides, in a computer system having a CPU, a main memory (either real or virtual) divisible into allocable units, a secondary storage unit and an operating system for allocating the allocable units to tasks for use thereby, a suspend circuit for creating an optimized compressed image of data in the main memory."

Pearce, 2:36-42.

> "One of the many functions of an operating system is to grant requests for allocable units of memory by the programs that run Within the framework of the operating system."

Pearce, 2:57-60.

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Pearce, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Pearce, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Pearce, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.1 and 1.2.

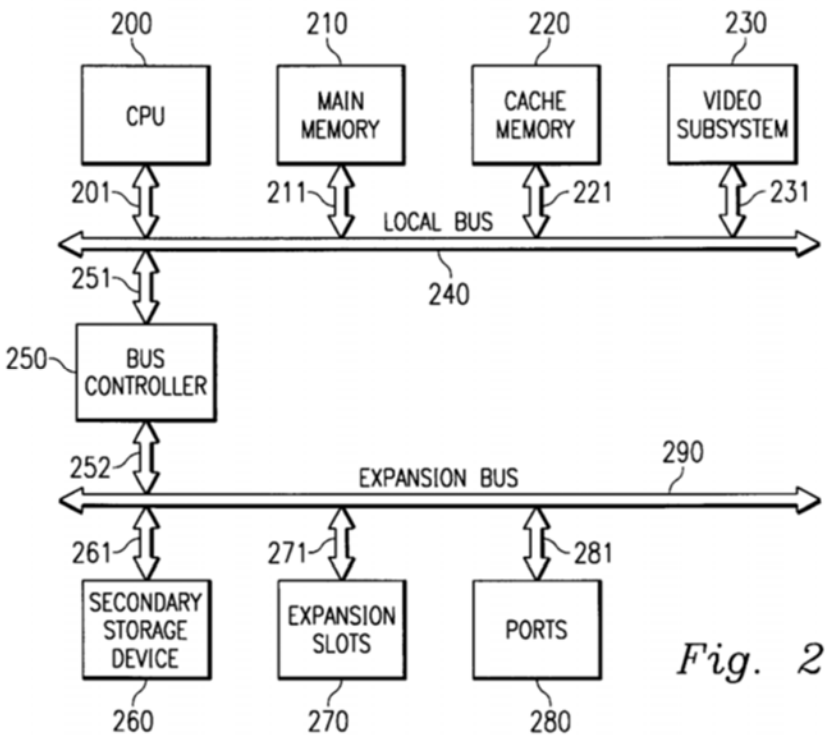


Fig. 2.

"The present invention is directed, in general, to computer systems and, more particularly, to a circuit and method for conserving power used by the computer system by storing a compressed image of the contents of the computer system's main memory on a nonvolatile secondary storage device, allowing power to the main memory to be interrupted."

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

Pearce, 1:6-11.

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

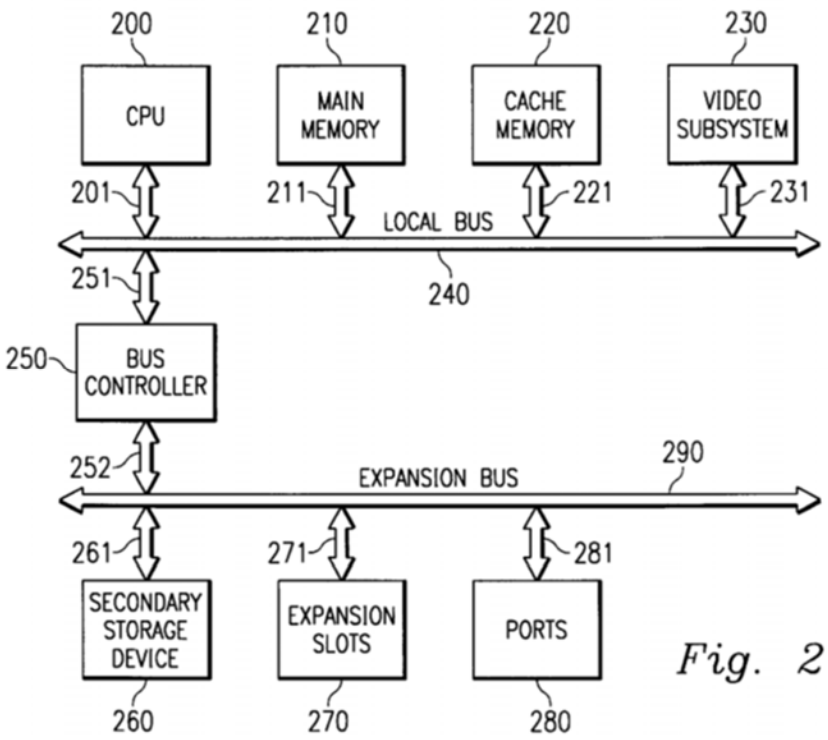


Fig. 2

Pearce, Fig. 2.

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "There are a wide variety of conventional data compression algorithms that are suitable to compress the contents of main memory to create a compressed image to be stored in the secondary storage device. Those skilled in the art are familiar With standard compression algorithms such as run length encoding, adaptive pattern substitution, variable length character encoding (such as Huffman coding), restricted variability codes, dictionary substitution, differencing and ordered data schemes."

Pearce, 7:11-19.

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Pearce, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claim 32 above.

> "There are a wide variety of conventional data compression algorithms that are suitable to compress the contents of main memory to create a compressed image to be stored in the secondary storage device. Those skilled in the art are familiar With standard compression algorithms such as run length encoding, adaptive pattern substitution, variable length character encoding (such as Huffman coding), restricted variability codes, dictionary substitution, differencing and ordered data schemes."

Pearce, 7:11-19.

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

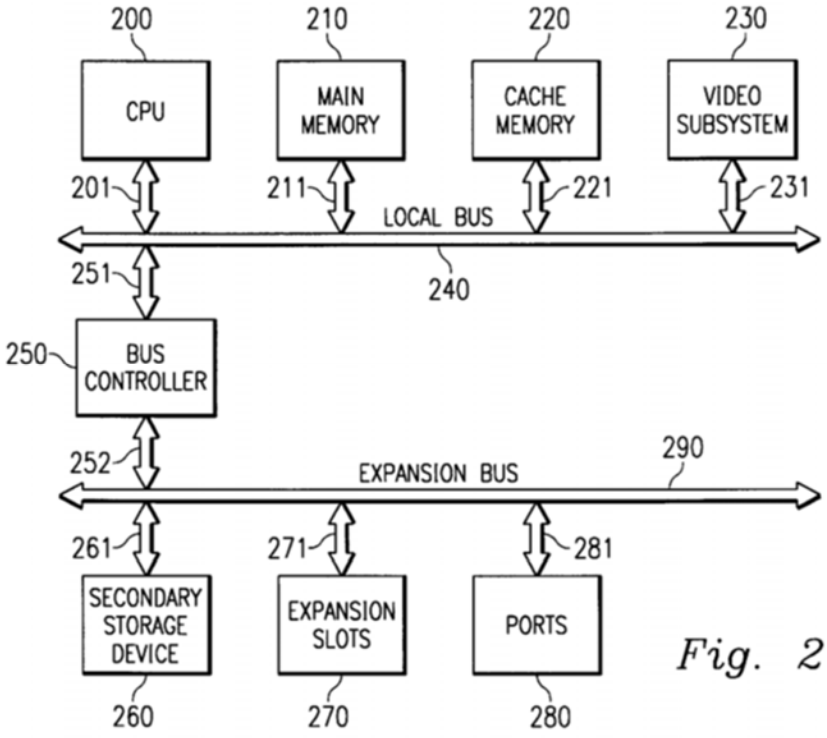


Fig. 2

Pearce, Fig. 2

Appendix C15
Invalidity of U.S. Patent 8,880,862 based on Pearce

"The present invention is directed, in general, to computer systems and, more particularly, to a circuit and method for conserving power used by the computer system by storing a compressed image of the contents of the computer system's main memory on a nonvolatile secondary storage device, allowing power to the main memory to be interrupted." Pearce, 1:6-11.

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory."

Pearce, 3:46-49.

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Pearce, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Pearce, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Pearce, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 32, 33 and 49 above.	

Pearce **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

Pearce **Claim 83**
"The method of claim 13, wherein the boot data in the compressed form represents a plurality of files."

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 15, 17 and 24 above.	

Pearce **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 32, 33 and 49 above.	

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Pearce, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claim 32, 33, and 49 above.	

Pearce **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Pearce, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

Pearce **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Pearce, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

See also

> "In a preferred embodiment, the suspend circuit further comprises a circuit for restoring the main memory by decompressing the compressed image into the main memory. The allocable units allocated to the reducing task are designated as unallocated units in the decompressed image."

Pearce, 3:47-52.

> "As a part of the block 515, the units previously allocated to the reducing task are marked as unallocated and therefore free for subsequent allocation by the operating system."

Pearce, 8:67-9:3, Fig. 5.

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Pearce, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Pearce discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

See also

> "During compression of the main memory, the record containing the identity of the allocable units compressed is also stored for use by the main memory restoration method detailed in FIG. 5."

Pearce, 8:27-30.

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Pearce, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Pearce, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Pearce discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C16
Invalidity of U.S. Patent 8,880,862 based on Rahman

U.S. Patent No. 5,901,310 Rahman ("Rahman") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

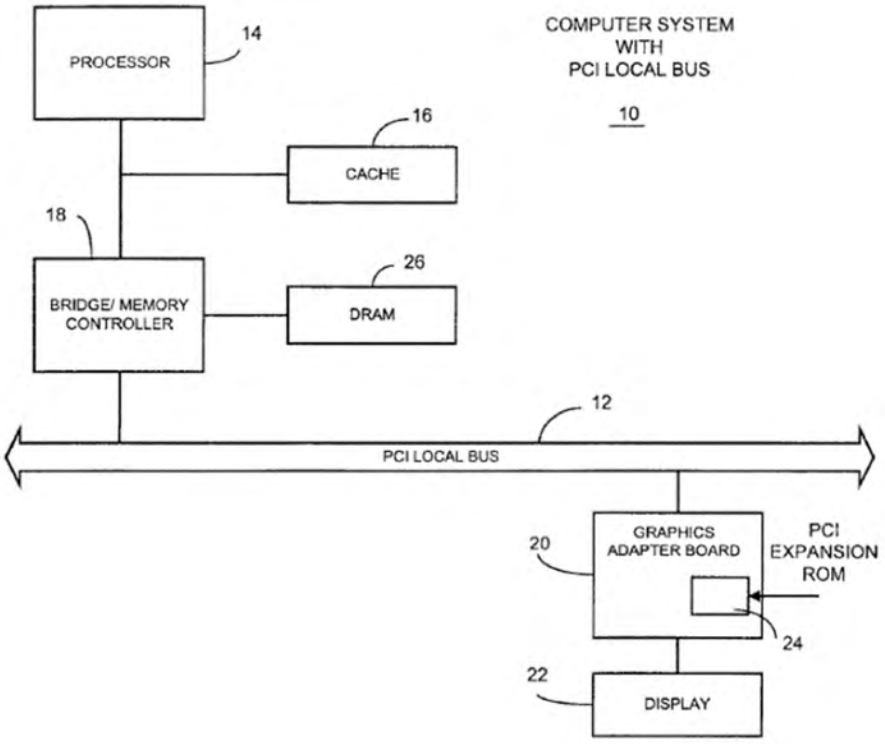
The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C16
Invalidity of U.S. Patent 8,880,862 based on Rahman

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Rahman, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically located on the adapter board, and the firmware may be firmware for

initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Rahman, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory."

Rahman, Abstract.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Rahman, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory."

Rahman, Abstract.

> "Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory."

Rahman, Abstract.

> "The invention virtually increases the size of the nonvolatile semiconductor memory (e.g., a ROM) available for storing firmware by storing the firmware in compressed form and quickly decompressing it on startup. This permits the firmware memory to hold more instructions than would otherwise be possible. The invention is able to decompress firmware quickly, reliably, and fully automatically, so that the fact of the firmware being compressed is substantially invisible to the end-user."

Rahman, 1:48-56.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and

> supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Rahman, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory."

Rahman, Abstract.

> "The invention virtually increases the size of the nonvolatile semiconductor memory (e.g., a ROM) available for storing firmware by storing the firmware in compressed form and quickly decompressing it on startup. This permits the firmware memory to hold more instructions than would otherwise be possible. The invention is able to decompress firmware quickly, reliably, and fully automatically, so that the fact of the firmware being compressed is substantially invisible to the end-user."

Rahman, 1:48-56.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

1.4.1 updating the boot data list,	Rahman, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Rahman, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Rahman, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Rahman, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

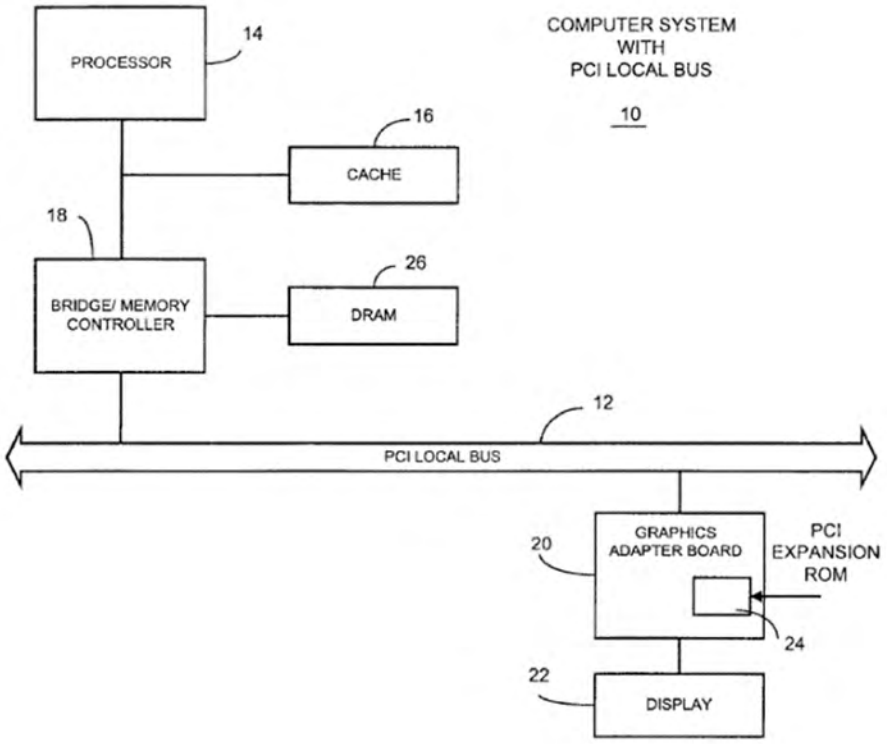
4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Rahman, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Rahman, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claims 1-1.4.2 above.



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically

> located on the adapter board, and the firmware may be firmware for initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Rahman, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Rahman, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Rahman, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Rahman, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Rahman, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory."

Rahman, Abstract.

> "Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory."

Rahman, Abstract.

> "The invention virtually increases the size of the nonvolatile semiconductor memory (e.g., a ROM) available for storing firmware by storing the firmware in compressed form and quickly decompressing it on startup. This permits the firmware memory to hold more instructions than would otherwise be possible. The invention is able to decompress firmware quickly, reliably, and fully automatically, so that the fact of the firmware being compressed is substantially invisible to the end-user."

Rahman, 1:48-56.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression

> program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

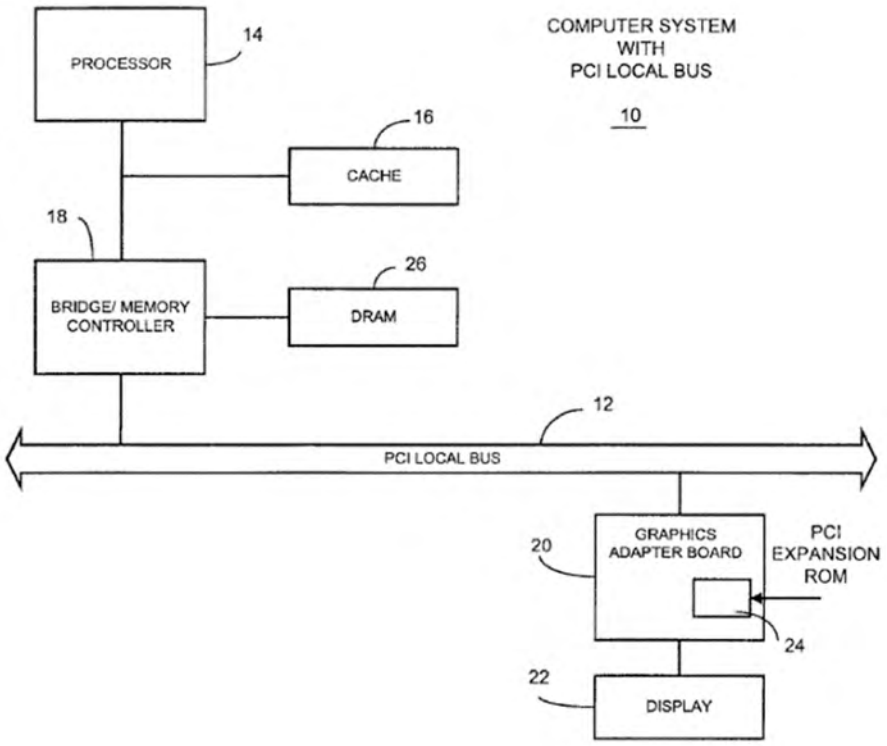
5.6 updating the boot data list;	Rahman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Rahman, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1-1.3 above.	

6 (Preamble) A system comprising: a processor;	Rahman, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically located on the adapter board, and the firmware may be firmware for initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

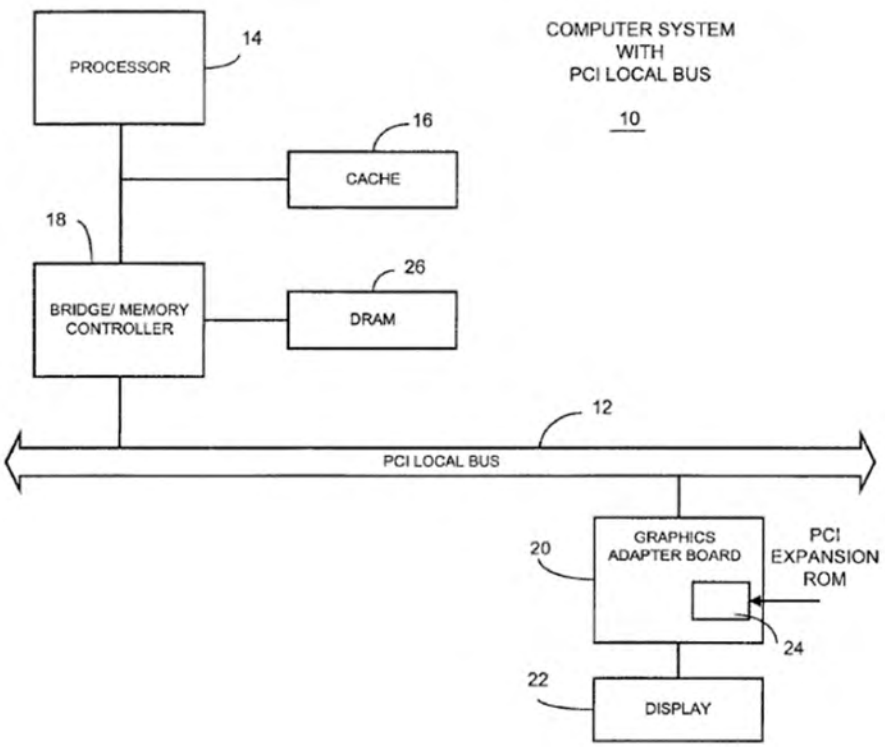
6.1 a memory; and	Rahman, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Rahman, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claims 1.1 and 1.2 above.



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the

> firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically located on the adapter board, and the firmware may be firmware for initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

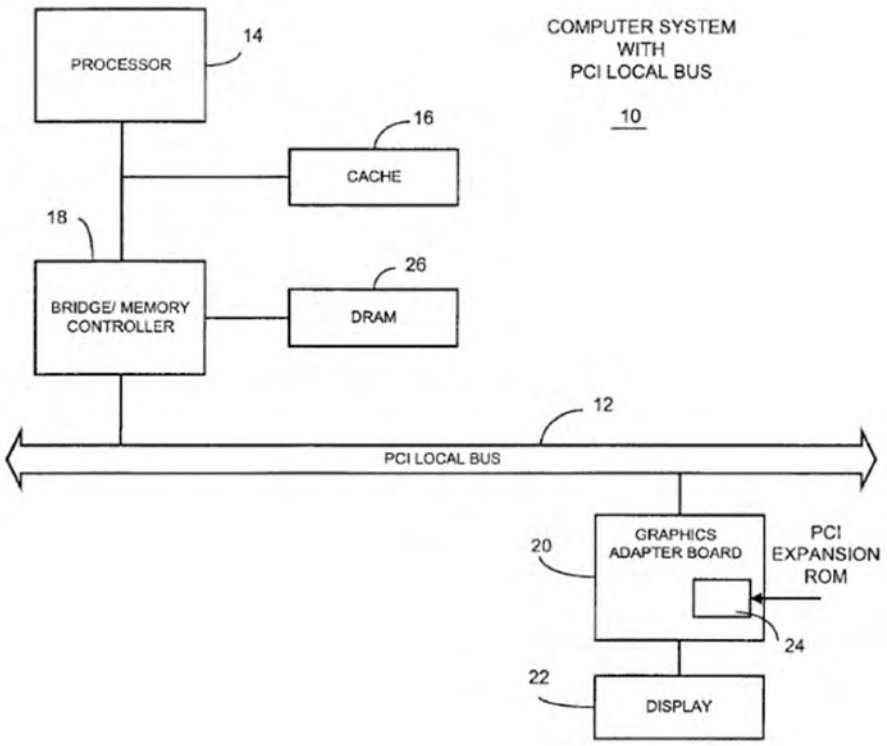
> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

6.3 wherein the processor is configured:	Rahman, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically located on the adapter board, and the firmware may be firmware for initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Rahman, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Rahman, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.2 above.	

Rahman Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Rahman, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Rahman, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

Rahman Claim 6.7
"to update the boot data list."

Appendix C16
Invalidity of U.S. Patent 8,880,862 based on Rahman

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Rahman, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Rahman, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Rahman, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Rahman, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Rahman, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Rahman Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Rahman, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory."

Rahman, Abstract.

> "Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory."

Rahman, Abstract.

> "The invention virtually increases the size of the nonvolatile semiconductor memory (e.g., a ROM) available for storing firmware by storing the firmware in compressed form and quickly decompressing it on startup. This permits the firmware memory to hold more instructions than would otherwise be possible. The invention is able to decompress firmware quickly, reliably, and fully automatically, so that the fact of the firmware being compressed is substantially invisible to the end-user."

Rahman, 1:48-56.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code

> 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

8.6 updating the boot data list,	Rahman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

Rahman Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Rahman, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Rahman, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Rahman, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Rahman, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claim 9.1 above.

> "The compression technique used may include both run-length encoding and pattern compression, and may operate at the bit level."

Rahman, Abstract.

> "Compression techniques using both run-length encoding and pattern compression may be used (e.g., the Ross Compression technique). The compression technique may scan an input file. First, it may determine if it can perform run-length-encoding compression on the current character. If not, the compression technique may determine if it can compress a pattern of characters. If the run-length-encoding and pattern matching Were not successful, the compression technique may copy the character directly to the compressed file."

Rahman, 2:7-16.

> "The compression and decompression techniques utilize four 3-byte formats. FIG. 3 illustrates the formats, which are a short run-length-encoded format 30, a long run-length encoded format 32, a short patterned format 34, and a long patterned format 36."

Rahman, 3:12-16. *See also* 3:17-17, Appendix.

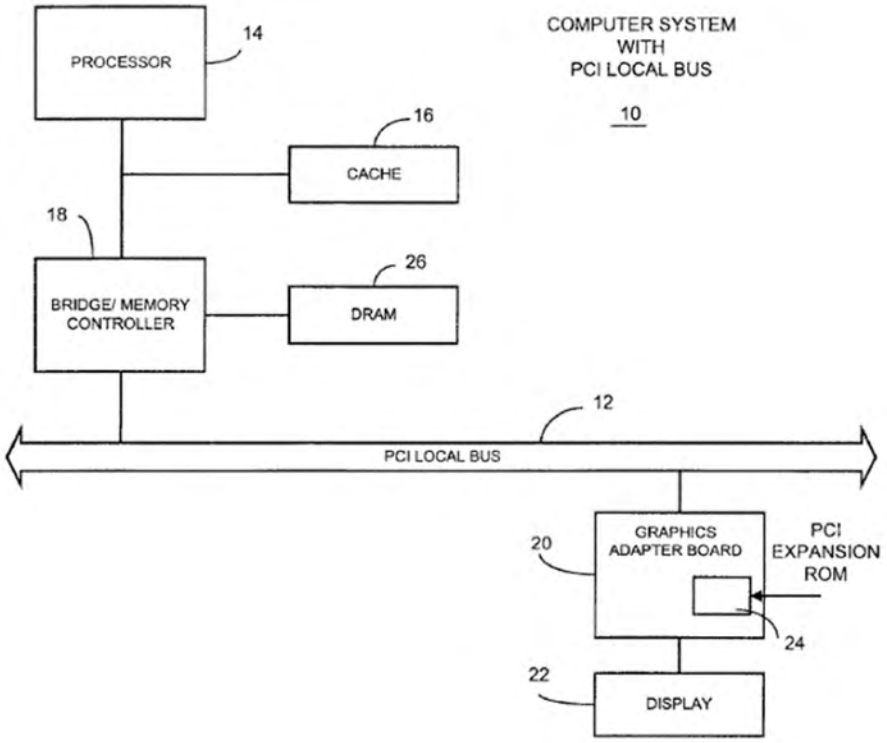
11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Rahman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Rahman, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claim 1.1 above.



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the

Rahman Claim 11.1
"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

> firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically located on the adapter board, and the firmware may be firmware for initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

11.2 accessing the loaded boot data in compressed form from the memory;	Rahman, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Rahman, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Rahman, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory."

Rahman, Abstract.

> "Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory."

Rahman, Abstract.

> "The invention virtually increases the size of the nonvolatile semiconductor memory (e.g., a ROM) available for storing firmware by storing the firmware in compressed form and quickly decompressing it on startup. This permits the firmware memory to hold more instructions than would otherwise be possible. The invention is able to decompress firmware quickly, reliably, and fully automatically, so that the fact of the firmware being compressed is substantially invisible to the end-user."

Rahman, 1:48-56.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The

> compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

11.5 updating the boot data list.	Rahman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C16
Invalidity of U.S. Patent 8,880,862 based on Rahman

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Rahman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Rahman, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Rahman, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Rahman, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Rahman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Rahman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1 (Preamble) above.	

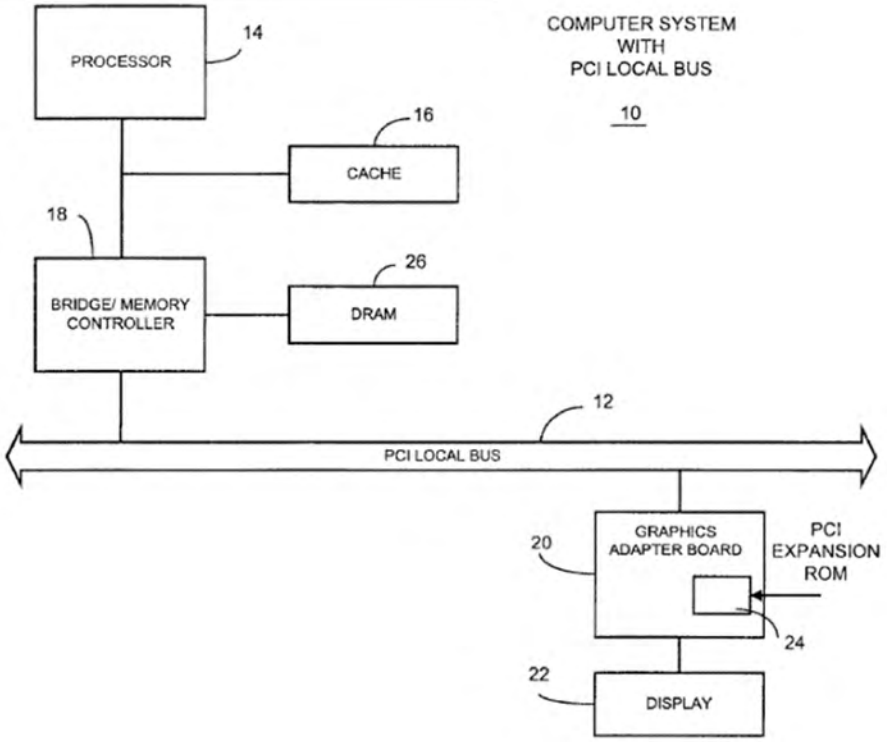
14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Rahman, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claims 1.1 and 1.2 above.

See ***Add disclosure from '608 Patent Claim 1.1***



Rahman, Fig. 1.

> "Initializing and configuring computer hardware with firmware stored in compressed form in nonvolatile semiconductor memory. Upon startup of the computer hardware, decompression software decompress the

> firmware, which is then stored in another memory. The computer hardware may be an adapter board (e.g., a graphics board connected to PCI bus), the nonvolatile semiconductor memory may be physically located on the adapter board, and the firmware may be firmware for initializing and configuring the adapter board."

Rahman, Abstract.

> "The firmware may be the BIOS for initializing and configuring a personal computer."

Rahman, 2:1-2.

> "Devices that connect to the PCI bus provide initialization code and runtime code for the device. This code resides in ROM 24 on the device. When the computer system starts up or initializes, it detects the device, reads the code from ROM into the host system memory, such as a RAM or dynamic random access memory 26 (DRAM), and interprets the code."

Rahman, 2:51-57.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

14.2 loading the boot data into a memory; and	Rahman, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Rahman, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claims 1.2 and 1.3 above.

> "Upon startup of the computer hardware, decompression software decompress the firmware, which is then stored in another memory."

Rahman, Abstract.

> "The invention virtually increases the size of the nonvolatile semiconductor memory (e.g., a ROM) available for storing firmware by storing the firmware in compressed form and quickly decompressing it on startup. This permits the firmware memory to hold more instructions than would otherwise be possible. The invention is able to decompress firmware quickly, reliably, and fully automatically, so that the fact of the firmware being compressed is substantially invisible to the end-user."

Rahman, 1:48-56.

> "When the computer system starts up or initializes, it detects the adapter board connected to the PCI bus. It then locates the decompression code 29 in the adapter board's ROM. The computer system processor runs an FCode interpreter, which interprets the instructions in the decompression program as it reads the code from the adapter board's ROM. The

> compressed code 28 is decompressed, using the computer system's memory (such as RAM or DRAM 26) to store the image. After the image is decompressed, the system recognizes the decompressed image as a device driver that configures and initializes the adapter board, and supplies the runtime set of instructions for the adapter board."

Rahman, 2:66-3:11.

> "Other embodiments are within the scope of the following claims. For example, a personal computer's basic input/output system (BIOS) is firmware stored in ROM and could be stored in a compressed format. Furthermore, other types of nonvolatile semiconductor memory, such as programmable ROMs (PROMS) and erasable programmable ROMs (EPROMs), could be used to store the compressed firmware."

Rahman, 4:63-5:3.

14.4 updating the boot data list.	Rahman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Rahman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Rahman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Rahman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Rahman, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.1 and 1.2	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.1 and 1.2	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "The compression technique used may include both run-length encoding and pattern compression, and may operate at the bit level."

Rahman, Abstract.

> "Compression techniques using both run-length encoding and pattern compression may be used (e.g., the Ross Compression technique). The compression technique may scan an input file. First, it may determine if it can perform run-length-encoding compression on the current character. If not, the compression technique may determine if it can compress a pattern of characters. If the run-length-encoding and pattern matching Were not successful, the compression technique may copy the character directly to the compressed file."

Rahman, 2:7-16.

> "The compression and decompression techniques utilize four 3-byte formats. FIG. 3 illustrates the formats, which are a short run-length-encoded format 30, a long run-length encoded format 32, a short patterned format 34, and a long patterned format 36."

Rahman, 3:12-16. *See also* 3:17-17, Appendix.

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Rahman, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

See Claim 32 above.

> "The compression technique used may include both run-length encoding and pattern compression, and may operate at the bit level."

Rahman, Abstract.

> "Compression techniques using both run-length encoding and pattern compression may be used (e.g., the Ross Compression technique). The compression technique may scan an input file. First, it may determine if it can perform run-length-encoding compression on the current character. If not, the compression technique may determine if it can compress a pattern of characters. If the run-length-encoding and pattern matching Were not successful, the compression technique may copy the character directly to the compressed file."

Rahman, 2:7-16.

> "The compression and decompression techniques utilize four 3-byte formats. FIG. 3 illustrates the formats, which are a short run-length-encoded format 30, a long run-length encoded format 32, a short patterned format 34, and a long patterned format 36."

Rahman, 3:12-16. *See also* 3:17-17, Appendix.

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 32 and 33 above.	

Rahman **Claim 45**

"The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Rahman discloses this limitation:

> "The compression technique used may include both run-length encoding and pattern compression, and may operate at the bit level."

Rahman, Abstract.

> "Compression techniques using both run-length encoding and pattern compression may be used (e.g., the Ross Compression technique). The compression technique may scan an input file. First, it may determine if it can perform run-length-encoding compression on the current character. If not, the compression technique may determine if it can compress a pattern of characters. If the run-length-encoding and pattern matching Were not successful, the compression technique may copy the character directly to the compressed file."

Rahman, 2:7-16.

> "The compression and decompression techniques utilize four 3-byte formats. FIG. 3 illustrates the formats, which are a short run-length-encoded format 30, a long run-length encoded format 32, a short patterned format 34, and a long patterned format 36."

Rahman, 3:12-16. *See also* 3:17-17, Appendix.

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Rahman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

Rahman **Claim 59**

"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Rahman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Rahman, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

Rahman **Claim 84**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system."

87. The method of claim 13, wherein the memory comprises: a physical memory.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 15, 17 and 24 above.	

Rahman **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 31 above.	

Rahman **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 32, 33 and 49 above.	

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Rahman, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claim 32, 33, and 49 above.	

Rahman **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Rahman, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Rahman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Rahman, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Rahman, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

Rahman **Claim 108**
"The method of claim 2, further comprising: compressing at least a portion of the additional boot data."

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Rahman, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Rahman discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Rahman **Claim 109**
"The method of claim 108, further comprising: storing the compressed additional boot data."

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

U.S. Patent No. 6,374,353 to Settsu ("Settsu") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

In addition, Apple incorporates by reference, as if set forth fully herein, all arguments related to Settsu in pending inter partes review petitions IPR2016-01737, IPR2016-01738, and IPR2016-01739.

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Settsu, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this claim limitation:

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

See also Settsu, 1:43-2:25, 3:6-25, 10:43-12:16, 13:49-15:4.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Settsu, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this claim limitation:

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system

> in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

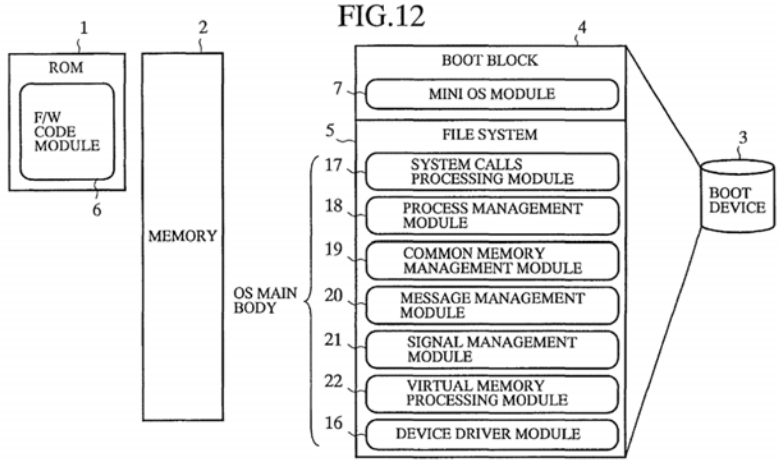
Settsu, 3:6-25

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

FIG.12



Settsu, Fig. 12

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of

the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 8:21-35, 8:66-9:11, 11:7-9, 11:18-39, 13:49-15:5, 16:7-17:62, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Settsu, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this claim limitation:

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded

functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 13:49-15:5, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Settsu, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this claim limitation:

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

> Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

> Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS

initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 1:23-26, 4:28-37, 10:43-12:16, 13:49-15:5, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

1.4.1 updating the boot data list,	Settsu, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this claim limitation:

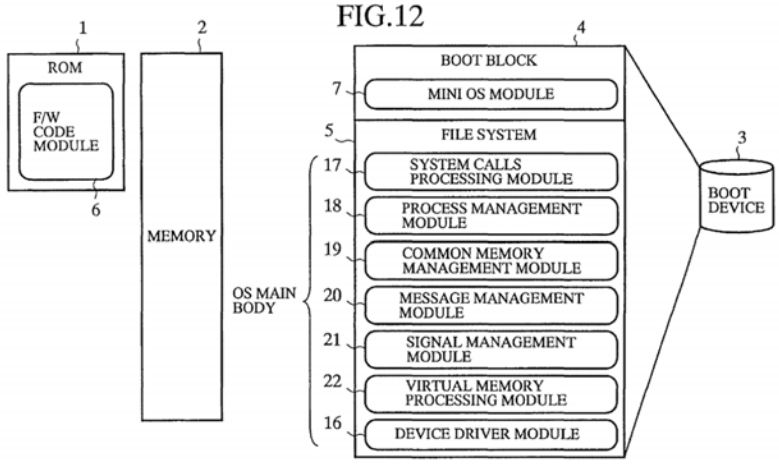
> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> In accordance with another preferred embodiment of the present invention, the mini OS module further includes an address resolve table used for linking the mini OS module with the OS main body module. Further, after the mini OS module generates and starts execution of a thread for the OS loading and initialization processing module, the OS loading and initialization processing module loads the OS main body module into the memory and then initializes it, loads a first process to be executed first, into the memory, loads code portions of the mini kernel module and the boot device driver module into the memory, and writes addresses of the code portions loaded into the memory into the address resolve table.

Settsu, 5:39-51

FIG.12



Settsu, Fig. 12

> Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

> The OS loading and decompression processing module 50 then, in step ST185, checks whether or not the whole of the main body of the OS has been loaded, that is, whether or not all the functional modules 16 to 22 have been loaded into the memory 2. If all the functional modules 16 to 22 have not been loaded into the memory 2 yet, the OS loading and decompression processing module 50 returns to step ST181 in which it continues to load the remaining functional modules of the OS main body.

Settsu, 14:44-52

See also Settsu, 16:7-17:62 and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Settsu, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this claim limitation:

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded

functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 1:23-26, 4:28-37, 13:49-15:5, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Settsu, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Settsu, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Settsu, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.4.1, and 2 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

5 (Preamble) A method for booting a computer system, the method comprising:	Settsu, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1-1.4.2 above.

See also

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The

FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7

then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

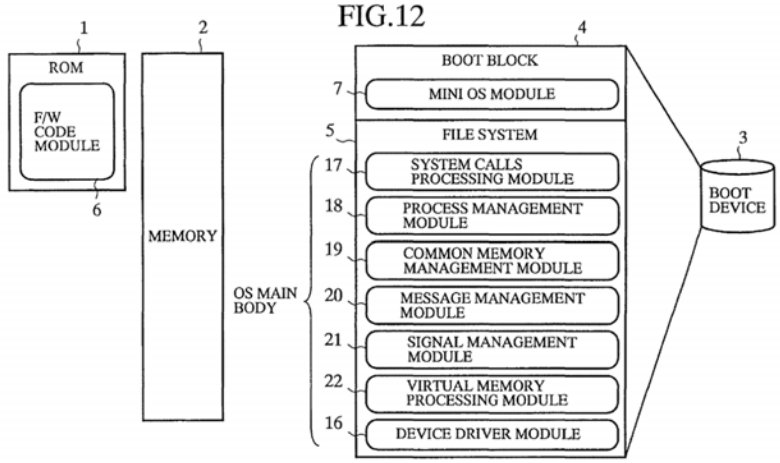
The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

FIG.12



Settsu, Fig. 12

> Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

> Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu **Claim 5 (Preamble)**

"A method for booting a computer system, the method comprising:"

Settsu, 13:66-14:12

See also Settsu, 16:7-17:62, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Settsu, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 1.1 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

5.2 loading the stored compressed boot data from the first memory;	Settsu, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Settsu, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Settsu, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Settsu, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system

> in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

> Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

> Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

> Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

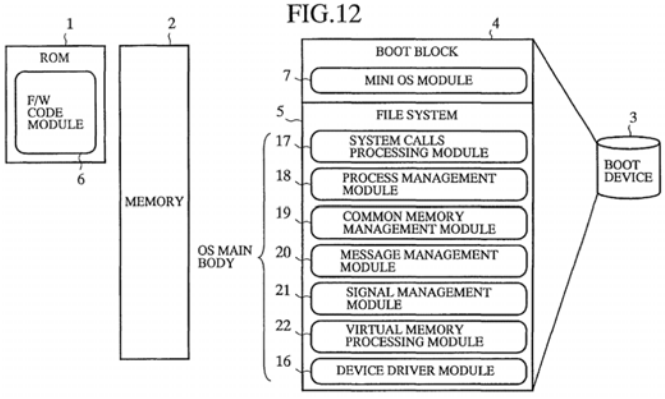
The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

FIG.12



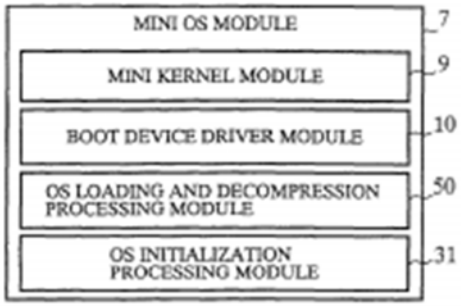
Settsu, Fig. 12

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be

omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

FIG.13



Settsu, Fig. 13.

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body

> of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

> As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 8:21-35, 8:66-9:11, 11:7-9, 11:18-39, 13:49-15:5, 16:7-17:62, and Figs. 1-4, 6-9, 14, 20 & 35-36.

5.6 updating the boot data list;	Settsu, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Settsu, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1-1.3 above.

See also

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

> As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

6 (Preamble) A system comprising: a processor;	Settsu, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and

then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management

module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression

processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

See also Figs. 1-4, 6-9, 12-14, 20 & 35-36.

6.1 a memory; and	Settsu, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Settsu, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot

device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing

apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention.

> Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

See also Figs. 1-4, 6-9, 12-14, 20 & 35-36.

6.3 wherein the processor is configured:	Settsu, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register.

Settsu Claim 6.3
"wherein the processor is configured"

Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are

Settsu Claim 6.3
"wherein the processor is configured"

initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression

processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

See also Figs. 1-4, 6-9, 12-14, 20 & 35-36.

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Settsu, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Settsu, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.2 above.	

Settsu Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Settsu, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Settsu, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

Settsu Claim 6.7
"to update the boot data list."

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Settsu, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1 (Preamble) above.	

Settsu **Claim 8 (Preamble)**

"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Settsu, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 6.2 above.

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Settsu, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Settsu, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Settsu, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Settsu, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system

in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing

apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

Settsu, 9:7-21.

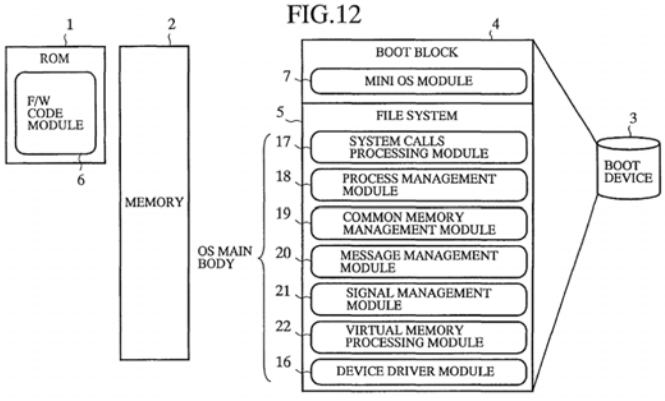
> The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

> As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

FIG.12



Settsu, Fig. 12

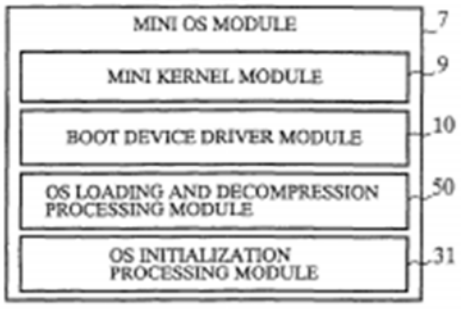
Settsu Claim 8.5

"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

FIG.13



Settsu, Fig. 13.

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Settsu Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

> Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

> As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 8:21-35, 8:66-9:11, 11:7-9, 11:18-39, 13:49-15:5, 16:7-17:62, and Figs. 1-4, 6-9, 14, 20 & 35-36.

8.6 updating the boot data list,	Settsu, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Settsu, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1-1.3 and 5.7 above.

See also

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further

reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 1.1 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

9.2 storing the additional portion of the operating system in the first memory, and	Settsu, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 8.1 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

> In accordance with another preferred embodiment of the present invention, the OS loading and initialization processing module of the mini OS module is divided into an OS loading processing module and an OS initialization module, the OS main body module is divided into a plurality of blocks of arbitrary record size, each of which includes a loading flag consisting of a plurality of bits respectively corresponding to the plurality of functional modules included in the OS main body module. Further, the loading flag of one of the plurality of blocks

including the end of any one of the plurality of functional modules has a corresponding bit set to a predetermined value. After the mini OS module is loaded into the memory, the mini OS module initializes the mini kernel module and the boot device driver module and then generates and starts execution of a thread for the OS loading processing module. After the thread for the OS loading processing module is started, the loading processing module loads each of the plurality of blocks of the OS main body module into the memory, and refers to the loading flag every time it loads each of the plurality of blocks into the memory. Only if a bit of the loading flag is set to a predetermined value, the OS loading processing module generates and starts execution of a thread for the OS initialization module. And, after the thread for the OS initialization module is started, the OS initialization module initializes a corresponding one of the plurality of functional modules loaded into the memory.

Settsu, 4:15-40

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Settsu, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 8.5 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

> In accordance with another preferred embodiment of the present invention, the OS loading and initialization processing module of the mini OS module is divided into an OS loading processing module and an OS initialization module, the OS main body module is divided into a plurality of blocks of arbitrary record size, each of which includes a

loading flag consisting of a plurality of bits respectively corresponding to the plurality of functional modules included in the OS main body module. Further, the loading flag of one of the plurality of blocks including the end of any one of the plurality of functional modules has a corresponding bit set to a predetermined value. After the mini OS module is loaded into the memory, the mini OS module initializes the mini kernel module and the boot device driver module and then generates and starts execution of a thread for the OS loading processing module. After the thread for the OS loading processing module is started, the loading processing module loads each of the plurality of blocks of the OS main body module into the memory, and refers to the loading flag every time it loads each of the plurality of blocks into the memory. Only if a bit of the loading flag is set to a predetermined value, the OS loading processing module generates and starts execution of a thread for the OS initialization module. And, after the thread for the OS initialization module is started, the OS initialization module initializes a corresponding one of the plurality of functional modules loaded into the memory.

Settsu, 4:15-40

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Settsu, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 9.1 above.	

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Settsu, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1 (Preamble) above.	

Settsu **Claim 11 (Preamble)**

"A method for providing accelerated loading of an operating system in a computer system, comprising"

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Settsu, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 1.1 above.

See also

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a

Settsu Claim 11.1
"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Settsu Claim 11.1

"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

> Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

> The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

> As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

Settsu Claim 11.1

"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

> Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

See also Figs. 1-4, 6-9, 12-14, 20 & 35-36.

11.2 accessing the loaded boot data in compressed form from the memory;	Settsu, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Settsu, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Settsu, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system

in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

> Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

> Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

> Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

Settsu, 9:7-21.

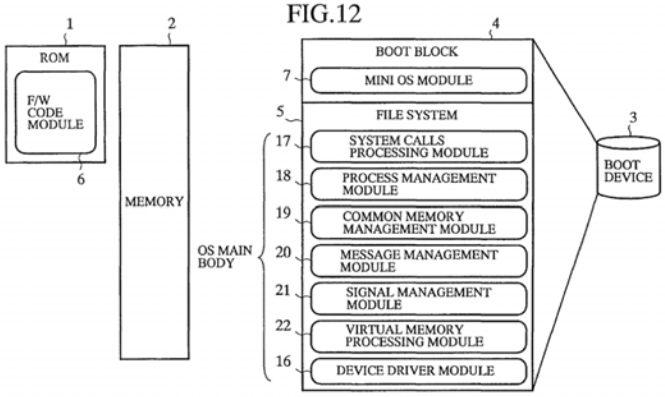
The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.

FIG.12



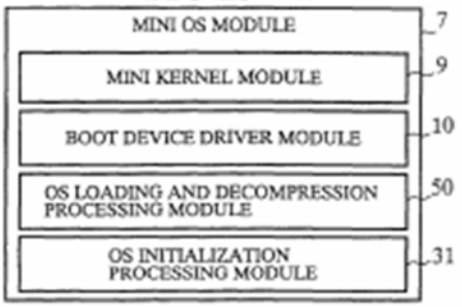
Settsu, Fig. 12

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like

elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

FIG.13



Settsu, Fig. 13.

Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu, 13:66-14:12

Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and

> decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

> As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5

See also Settsu, 8:21-35, 8:66-9:11, 11:7-9, 11:18-39, 13:49-15:5, 16:7-17:62, and Figs. 1-4, 6-9, 14, 20 & 35-36

11.5 updating the boot data list.	Settsu, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Settsu, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Settsu, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Settsu, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Settsu, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Settsu, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Settsu, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Settsu, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.1 and 1.2 above.

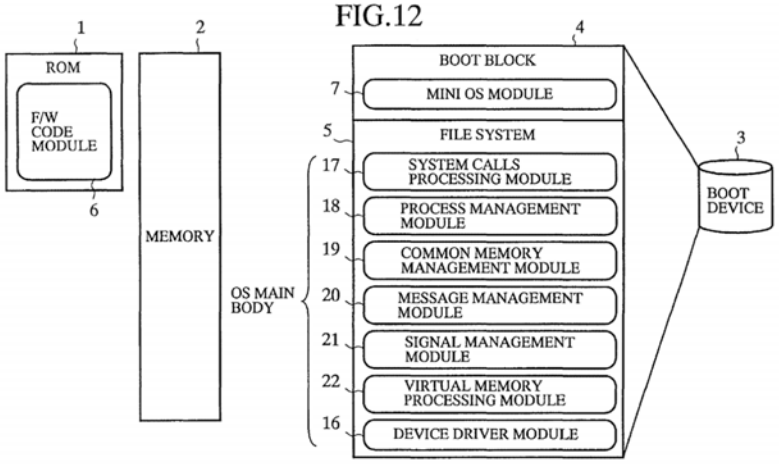
See also

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.
>
> Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini

OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

FIG.12



Settsu, Fig. 12

Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

See also Settsu, 16:7-17:62, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

14.2 loading the boot data into a memory; and	Settsu, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Settsu, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

> A method of booting up an information processing apparatus is provided. An operating system is divided into a mini operating system (OS) module having a function of bootstrap and an OS main body module having functions other than the function of bootstrap. The mini OS module can be located in a boot block of a boot device, whereas the OS main body module can be located in a file system of the boot device. A firmware or F/W code module stored in a ROM loads the mini OS module into memory when booting up the information processing apparatus. The mini OS module then loads the OS main body module into memory and then initializes the OS main body module.

Settsu, Abstract

> The present invention relates to an information processing apparatus capable of reducing the time required for booting itself when it is powered on, and a method of booting an information processing apparatus at a high speed.

Settsu, 1:8-12.

> In accordance with an aspect of the present invention, there is provided an information processing apparatus comprising: a boot device divided into a boot block in which a mini operating system (OS) module having a function required for bootstrap processing is located and a file system in which an operating system (OS) main body module having functions other than the function of bootstrap. . . .

Settsu, 1:51-57

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

> Referring next to FIG. 1, there is illustrated a block diagram showing the structure of an information processing apparatus according to a first embodiment of th present invention. In the figure, reference numeral 1 denotes a ROM of the information processing apparatus, 2 denotes a memory of the information processing apparatus, 3 denotes 5 a boot device of the information processing apparatus, 4 denotes a boot block in the boot device 3, 5 denotes a file system I the boot device 3, and 6 denotes a firmware or FI/W code module stored in the ROM 1. The FI/W code module 6 is directly executed on the ROM 1 so as to load data from the boot block 4 in the boot device 3 into the memory 2 and then assume that the loaded data is a code and execute the code after setting up and running diagnostic checks on a hardware or H/W register. Further, reference numeral 7 denotes a mini operating system (OS) module complied and liked in the same way as ordinary program files

and located in the boot block 4 within the boot device, the mini OS module having OS functions required for bootstrap processing, and 8 denotes an OS main body module located in the file system 5 within the boot device and provided with OS functions except the OS functions included in the mini OS module 7. When the information processing apparatus is powered on, it goes through initialization and transfers control to the F/W code module 6 stored in the ROM 1.

Referring next to FIG. 2, there is illustrated a diagram showing the structure of the mini OS module 7 stored in the boot block of the boot device of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the mini OS module 7 consists of a mini kernel module 9 which is a basic part of the OS, a boot device driver module 10 for performing I/O operations on the boot device 3, and an OS loading and initialization processing module 11 for loading the OS main body module 8 from the boot device 3 into the memory 2 and for executing the initialization of the OS main body module 8.

Settsu, 7:65-8:35

Referring next to FIG. 3, there is illustrated a diagram showing the structure of the OS main body module 8 of the information processing apparatus according to the first embodiment of the present invention. As shown in the figure, the OS main body module 8 consists of a kernel module 15 having kernel functions except the functions included in the mini kernel module 9 of FIG. 2, and a device driver module 16 for performing I/O operations on devices (not shown) except the boot device 3. . . . The OS main body module 8 is located within the file system 5 of the boot device 3 and is loaded into the memory 2 by the mini OS module 7. The OS main body module 8 then goes through initialization.

Settsu, 8:47-9:3

Referring next to FIG. 4, there is illustrated a flow chart showing operations of the mini OS module 7 of the information processing apparatus according to the first embodiment of the present invention. The F/W code module 6 loads the mini OS module 7 into the memory 2 and transfers control to the mini OS module 7. The mini OS module 7 then, in step ST101, executes initialization of the mini kernel module 9. In this case, the thread management module 12, the I/O management module 13, and the thread communication management module 14 are initialized and their functions are available now. Next, the mini OS module 7, in step ST102, executes initialization of the boot device

> driver module 10. The boot device driver module 10 registers an interrupt request from the boot device into an interrupt table of the I/O management module 13 so as to handle the interrupt request, and generates and starts execution of a thread for boot device I/O processing by using the thread management module 12.

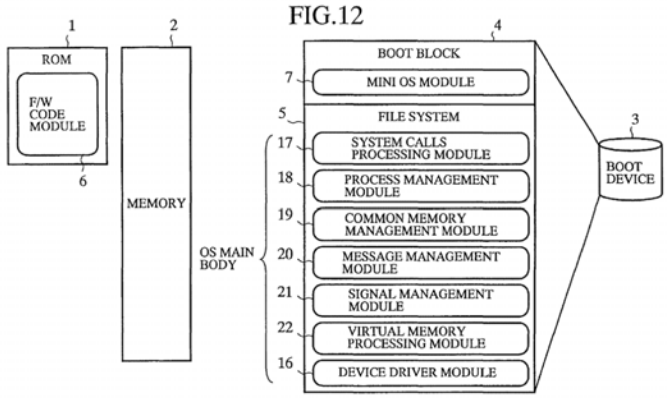
Settsu, 9:7-21.

> The mini OS module 7, in step ST103, generates a thread for the OS loading and initialization processing module 11 by using the thread management module 12. The mini OS module 7 further, in step ST104, starts execution of the thread by using the thread management module 12. The OS loading and initialization processing module 11 is thus started up as the thread. After that, the mini OS module 7 transfers control to the OS loading and initialization processing module 11.

Settsu, 9:30-39.

> As previously mentioned, in accordance with the second embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body of the OS. Further, the plurality of functional modules are separately stored in the file system 5. In addition, the OS load processing and the OS initialization processing can be performed in parallel with each other after any one of the plurality of functional modules of the OS main body is loaded into the memory. As a result, while the CPU waits for the occurrence of an event in performing the OS load or initialization processing, the CPU does not idle but the CPU performs another processing. Accordingly, the second embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 12:1-16.



Settsu, Fig. 12

> Referring next to FIG. 12, there is illustrated a block diagram showing the structure of an information processing apparatus according to a fourth embodiment of the present invention. In the figure, the same reference numerals as shown in FIG. 5 designate the same or like elements, and therefore the description of those elements will be omitted hereinafter. Like the OS of the second embodiment, the OS of the second embodiment is divided into a mini OS module 7 and a main body of the OS, and the main body is further divided into a plurality of functional modules, such as a system call processing module 17, a process management module 18, a common memory management module 19, a message management module 20, a signal management module 21, a virtual memory processing module 22, and a device driver module 16. The plurality of functional modules are separately stored as compressed files in a file system 5 of a boot device 3.

Settsu, 13:55-65

> Referring next to FIG. 13, there is illustrated a block diagram showing the structure of the mini OS module 7 of the information processing apparatus according to the fourth embodiment of the present invention. Like the mini OS module 7 of the second embodiment as shown in FIG. 6, the mini OS module 7 of the fourth embodiment is provided with a mini kernel module 9, a boot device driver module 10, and an OS initialization processing module 31. The mini OS module 7 of the fourth embodiment further comprises an OS loading and decompression processing module 50 having a function of decompressing a loaded functional module in addition to the function of the OS load processing module 30 of the second embodiment, instead of the OS load processing module 30.

Settsu **Claim 14.3**

"servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

Settsu, 13:66-14:12

> Referring next to FIG. 14, there is illustrated a flow chart showing operations of the OS loading and decompression processing module 50 of the information processing apparatus according to the fourth embodiment of the present invention. The OS loading and decompression processing module 50 to which control from the mini OS module 7 has been transferred, in step ST181, loads the main body of the OS stored in the file system 5 of the boot device 3, such as the system call processing module 17, the process management module 18, the common memory management module 19, the message management module 20, the signal management module 21, the virtual memory processing module 22, and the device driver module 16, into the memory 2. In performing step ST181, the OS loading and decompression processing module 50 loads any one of those functional modules 16 to 22 first. The OS loading and decompression processing module 50 then, in step ST182, decompresses one functional module loaded into the memory. Since the plurality of functional modules 16 to 22 are stored as compresses files in the file system 5 of the boot device, the functional module loaded into the memory 2 is compressed data. Therefore, in performing step ST182, the OS loading and decompression processing module 50 decompresses the compressed data so as to convert it into executable code and data.

Settsu, 14:13-37

> As previously mentioned, in accordance with the fourth embodiment of the present invention, the main body of the OS is divided into a plurality of functional modules according a plurality of functions to be performed by the main body, and the plurality of functional modules are stored as compressed files in the file system **5** of the boot device. Further, the OS loading and decompression processing module **50** decompresses each of the plurality of functional modules each time it loads each of them into memory. As a result, the time required for I/O processing can be reduced. Accordingly, the fourth embodiment of the present invention provides an advantage of being able to further reduce the time required for booting up the information processing apparatus.

Settsu, 14:58-15:5
See also Settsu, 8:21-35, 8:66-9:11, 11:7-9, 11:18-39, 13:49-15:5, 16:7-17:62, and Figs. 1-4, 6-9, 13-14, 20 & 35-36.

14.4 updating the boot data list.	Settsu, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Settsu, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 1 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

> In accordance with another preferred embodiment of the present invention, the OS loading and initialization processing module of the mini OS module is divided into an OS loading processing module and an OS initialization module, the OS main body module is divided into a plurality of blocks of arbitrary record size, each of which includes a loading flag consisting of a plurality of bits respectively corresponding to the plurality of functional modules included in the OS main body

module. Further, the loading flag of one of the plurality of blocks including the end of any one of the plurality of functional modules has a corresponding bit set to a predetermined value. After the mini OS module is loaded into the memory, the mini OS module initializes the mini kernel module and the boot device driver module and then generates and starts execution of a thread for the OS loading processing module. After the thread for the OS loading processing module is started, the loading processing module loads each of the plurality of blocks of the OS main body module into the memory, and refers to the loading flag every time it loads each of the plurality of blocks into the memory. Only if a bit of the loading flag is set to a predetermined value, the OS loading processing module generates and starts execution of a thread for the OS initialization module. And, after the thread for the OS initialization module is started, the OS initialization module initializes a corresponding one of the plurality of functional modules loaded into the memory.

Settsu, 4:15-40

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Settsu, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 15 above.

See also

> The present invention is made to overcome the above problems. It is therefore an object of the present invention to provide an information processing apparatus and a method capable of reducing the time required for booting up itself when it is powered on, and also reducing the time required to start execution of applications to be started automatically when the information processing apparatus is booted up.

Settsu, 1:44-50

> In accordance with another preferred embodiment of the present invention, the OS loading processing module of the mini OS module is an application (AP) execution and OS loading processing module for starting execution of at least a predetermined application module which is located in the file system and which can automatically be started and run on the operating system when booting up the information processing apparatus, and for loading each of the plurality of functional modules into the memory. Further, the predetermined application module includes a function definition file in which some functional modules required for the application module to run on the operating system are listed. After the mini OS module is loaded into the memory, the mini OS module initializes the mini kernel module and the boot device driver module and then generates and starts execution of a thread for the AP execution and OS loading processing module. After the thread for the AP execution and OS loading processing module is started, the AP execution and OS loading processing module loads the application nodule from the file system into the memory and further loads some

functional modules required for the application module into the memory according to the function definition file included in the application module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is started, the OS initialization module then initializes each of the some functional modules loaded into the memory. And, after the initialization of all of the some functional modules is completed, the application execution and OS loading processing module further loads the remainder of all functional modules included in the OS main body module into the memory and initializes the remainder using the OS initialization processing module while starting execution of the application module as a process.

Settsu, 3:48-4:14

Settsu, Fig. 18.

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Settsu, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.1 and 1.2	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Settsu, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.1 and 1.2 above	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.1 and 1.2	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 1.1

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Settsu, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 32 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.1 and 1.2 above	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 10 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claim 6 above

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Settsu, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Settsu, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 31 above.	

Appendix C17
Invalidity of U.S. Patent 8,880,862 based on Settsu

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 32, 33 and 49 above.	

Settsu **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Settsu, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 32, 33 and 49 above.	

Settsu **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 15, 17 and 24 above.	

Settsu **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 31 above.	

Settsu **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 32, 33 and 49 above.	

Settsu **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Settsu, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claim 32, 33, and 49 above.	

Settsu **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Settsu, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.4.1, 2, 4, 5.6, and 8.6 above.

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Settsu, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Settsu, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Settsu discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Settsu, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 1.1, 5, 9.1 and 8 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Settsu, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Settsu discloses this limitation:

See Claims 5.1, 8.1 and 9.1 above.

See also

> In accordance with another preferred embodiment of the present invention, the plurality of functional modules, into which the OS main body module is divided, are stored as compressed files in the file system and the loading and initialization processing module of the mini OS module is divided into an OS loading and decompression processing module and an OS initialization module. Further, the mini OS module generates and starts execution of a thread for the OS loading and decompression processing module after the mini OS module initializes the mini kernel module and the boot device driver module. After the thread for the OS loading and decompression processing module is started, the OS loading and decompression processing module loads each of the plurality of functional modules into the memory and decompresses the loaded functional module, and then generates and starts execution of a thread for the OS initialization module. After the thread for the OS initialization module is executed, the OS initialization module initializes each of the plurality of functional modules loaded into the memory and decompressed.

Settsu, 3:6-25

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

U.S. Patent Number 5,269,022 to Shinjo ("Shinjo") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

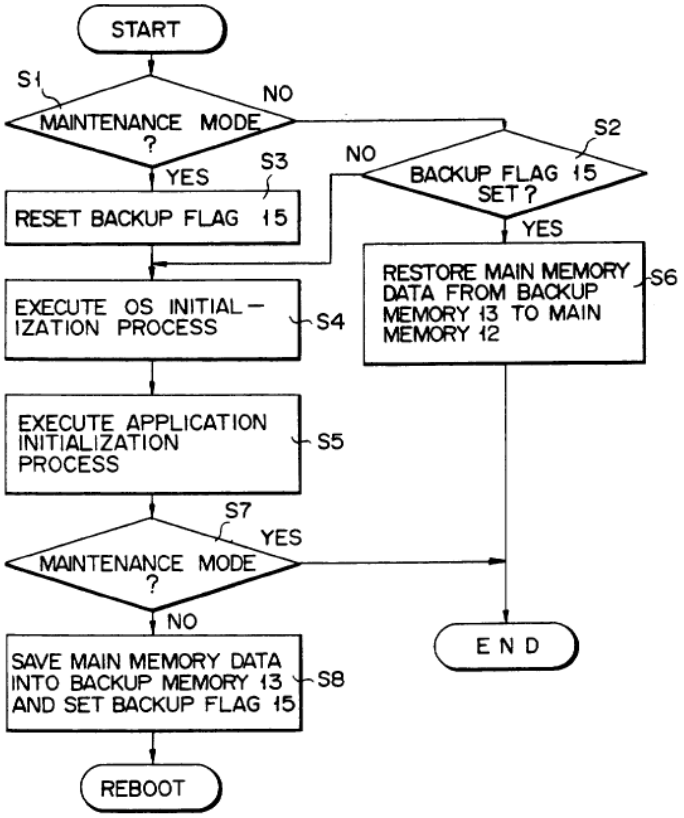
1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Shinjo, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> In a computer system, when the system is first booted in a normal mode, main memory data stored in a main memory immediately after the system is booted, is stored as backup data in a backup memory or the like. A backup flag representing whether or not the backup data can be restored is set and the system is rebooted. When the system is next booted in the normal mode, the backup data stored in the backup memory or the like is restored as the main memory data in the main memory. The backup flag is automatically reset in a maintenance mode.

Shinjo, Abstract



F I G. 2

Shinjo, Figure 2

> The above-described process requires long time since the number of times of access to a recording medium such as a disk is very large and an overhead for retrieval of file, production of process or the like is increased. In other words, boot time required until an operation as a computer system starts, is lengthened. It is thus desirable to achieve an apparatus capable of booting a computer system at high speed.

Shinjo, 1:22-29

> It is an object of the present invention to provide a method and an apparatus for booting a computer system.

Shinjo, 1:32-34

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is

restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:32-48

The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

Therefore, the boot process of the present invention can be executed at higher speed than the conventional boot process.

Shinjo, 3:62-64

As described above, according to the present invention, when the system is first booted, the saving mode is selected. Immediately after the system is booted, the main memory data stored in the main memory is saved as backup data into the backup memory (backup file), and the backup flag is set. The system is then rebooted. Therefore, when the system is next booted in the normal mode, the quick start mode is selected and the backup data saved into the backup memory (backup file) is restored as the main memory data in the main memory. The boot process of the system is completed only by the above process, and the state immediately after the system is booted is restored.

Shinjo, 4:45-57

> In the conventional system, a very large number of times of disk access are necessary for the OS initialization process such as setting of the running environment of an OS and for the application initialization process such as setting of the running environment of an application program, and long time is required for the boot process. In the present invention, however, the system can be booted at high speed.
>
> Shinjo, 4:58-65

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Shinjo, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> In the initialization process, a resident load module is loaded into a main memory, various control blocks are produced, the running environment of an operating system (OS) is set, and the running environment of an application system is set (for example, a control process is produced).

Shinjo, 1:16-21

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:35-48

> In step S7, when the designated boot mode is not the maintenance mode, i.e., when it is the normal mode, the saving mode is selected, and the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data and the backup flag 15 of the backup memory 13 is set (step S8). The process of step S8 is completed and then a reboot is executed.

Shinjo, 3:28-34

> Since the backup flag 15 has been set, the boot process is started in the quick start

mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Shinjo, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> In a computer system, when the system is first booted in a normal mode, main memory data stored in a main memory immediately after the system is booted, is stored as backup data in a backup memory or the like. A backup flag representing whether or not the backup data can be restored is set and the system is rebooted. When the system is next booted in the normal mode, the backup data stored in the backup memory or the like is restored as the main memory data in the main memory. The backup flag is automatically reset in a maintenance mode.

Shinjo, Abstract

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:32-48

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected

and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

In the computer system according to the first embodiment wherein the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data, when the system power source is turned off, the backup data is erased. It is thus necessary to boot the system in the same manner as the conventional apparatus and save the main memory data stored in the main memory 12 into the backup memory 13 as the backup data immediately after the system is booted. By backing up the backup memory 13 by a battery or the like, such boot process in the system is executed only at the first time.

Shinjo, 3:65-4:8

Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

A disk unit 19 of a computer system according to the second embodiment as shown in FIG. 3 can be used in place of the backup memory 13 shown in FIG. 1. The disk unit 19 includes a backup file 23 for storing the main memory data stored in the main memory 12 as backup data and a backup flag 25. Even through the system power source is turned off, the backup data stored in the backup file 23 is not erased. In the system according to the second embodiment, since the backup data as the main memory data is saved and restored between the main memory 12 and disk unit 19, disk access occurs. Therefore, the time required for the boot process of the system according to the first embodiment is longer than the time required for that of the system according to the first embodiment.

Shinjo, 4:9-24

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Shinjo, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> Therefore, the boot process of the present invention can be executed at higher speed than the conventional boot process.

Shinjo, 3:62-64

> In the conventional system, a very large number of times of disk access are necessary for the OS initialization process such as setting of the running environment of an OS and for the application initialization process such as setting of the running environment of an application program, and long time is required for the boot process. In the present invention, however, the system can be booted at high speed.

Shinjo, 4:58-65

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:52-68

1.4.1 updating the boot data list,	Shinjo, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:52-68

> When software maintenance such as replacement of programs and patch is performed, the boot process is executed in the maintenance mode and thus the backup data cannot be automatically restored. When the system is next booted, the saving mode is selected again. Therefore, the update backup data can always be stored in the backup memory (backup file).

Shinjo, 5:3-9

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Shinjo, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> In the initialization process, a resident load module is loaded into a main memory, various control blocks are produced, the running environment of an operating system (OS) is set, and the running environment of an application system is set (for example, a control process is produced).

Shinjo, 1:16-21

> In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

> After the backup flag 15 is reset in step S3, the boot process is executed as in the conventional system, i.e., the OS initialization process and the application initialization process are executed (steps S4 and S5).

Shinjo, 3:54-57

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Shinjo, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 1.4.1 above.

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Shinjo, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 1.4.1 above.

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Shinjo, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Shinjo, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1-1.4.2 above.

> The present invention relates to a method and an apparatus for booting a computer system.

Shinjo, 1:9-10

> In a computer system, generally, whenever the system is booted, a boot process for loading firmware, an initial program loader (IPL) program and an initialize (INZ) program, an initial program loader process, and an initialization process are executed.

Shinjo, 1:11-16

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:35-48

> According to another aspect of the present invention, there is provided an apparatus for booting a computer system, the apparatus comprising: a main memory for storing main memory data; means for setting a boot mode for booting the computer system; determining means for determining whether or not the boot mode is a normal mode; a flag to be set/reset in accordance with a determination result by the determining means; and a backup memory for storing the main memory data to be stored in the main memory immediately after the computer system is booted, as backup data when the boot mode is the normal mode and the flag is reset, and wherein the backup data stored in the

backup memory is restored as the main memory data into the main memory when the boot mode is the normal mode and the flag is set.

Shinjo, 1:49-64

The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-68

When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

In step S1, it is determined whether or not the boot mode designated by the boot command output from the boot mode setting unit 17 is the maintenance mode. If the boot mode is not the maintenance mode, i.e., if it is the normal mode, it is determined whether or not the backup flag 15 of the backup memory 13 is set (step S2). That is, it is determined whether a boot process is executed in the quick start mode or saving mode.

Shinjo, 3:8-15

In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the

> OS initialization process of step S3 or the application initialization process of step S4.
>
> Shinjo, 3:35-45
>
> In the computer system according to the first embodiment wherein the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data, when the system power source is turned off, the backup data is erased. It is thus necessary to boot the system in the same manner as the conventional apparatus and save the main memory data stored in the main memory 12 into the backup memory 13 as the backup data immediately after the system is booted. By backing up the backup memory 13 by a battery or the like, such boot process in the system is executed only at the first time.
>
> Shinjo, 3:65-4:8
>
> As described above, according to the present invention, when the system is first booted, the saving mode is selected. Immediately after the system is booted, the main memory data stored in the main memory is saved as backup data into the backup memory (backup file), and the backup flag is set. The system is then rebooted. Therefore, when the system is next booted in the normal mode, the quick start mode is selected and the backup data saved into the backup memory (backup file) is restored as the main memory data in the main memory. The boot process of the system is completed only by the above process, and the state immediately after the system is booted is restored.
>
> Shinjo, 4:45-67

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Shinjo, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 1.1 above.

> In a computer system, when the system is first booted in a normal mode, main memory data stored in a main memory immediately after the system is booted, is stored as backup data in a backup memory or the like. A backup flag representing whether or not the backup data can be restored is set and the system is rebooted. When the system is next booted in the normal mode, the backup data stored in the backup memory or the like is restored as the main memory data in the main memory. The backup flag is automatically reset in a maintenance mode.

Shinjo, Abstract

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

5.2 loading the stored compressed boot data from the first memory;	Shinjo, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Shinjo, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Shinjo, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Shinjo, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in

which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

In step S7, when the designated boot mode is not the maintenance mode, i.e., when it is the normal mode, the saving mode is selected, and the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data and the backup flag 15 of the backup memory 13 is set (step S8). The process of step S8 is completed and then a reboot is executed.

Shinjo, 3:28-34

In a system according to the third embodiment as shown in FIG. 4, the memory 14 includes the backup memory 13, and the disk unit 19 includes the backup file 23. In the saving mode, therefore, the main memory data stored in the main memory 12 can be saved as backup data into the backup memory 13 and backup file 23, and the backup flags 15 and 25 can be set.

Shinjo, 4:24-31

5.6 updating the boot data list;	Shinjo, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Shinjo, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1-1.3 above.

> Therefore, the boot process of the present invention can be executed at higher speed than the conventional boot process.

Shinjo, 3:62-64

> In the conventional system, a very large number of times of disk access are necessary for the OS initialization process such as setting of the running environment of an OS and for the application initialization process such as setting of the running environment of an application program, and long time is required for the boot process. In the present invention, however, the system can be booted at high speed.

Shinjo, 4:58-65

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the

normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:52-68

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

6 (Preamble) A system comprising: a processor;	Shinjo, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> The CPU 11 controls the entire computer system.

Shinjo, 2:32

> When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

6.1 a memory; and	Shinjo, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Shinjo, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.1 and 1.2 above.

> The CPU 11 controls the entire computer system.

Shinjo, 2:32

> When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

6.3 wherein the processor is configured:	Shinjo, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> The CPU 11 controls the entire computer system.

Shinjo, 2:32

> When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Shinjo, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Shinjo, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Shinjo, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Shinjo, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.4.1 above.	

Shinjo Claim 6.7
"to update the boot data list."

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Shinjo, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1 (Preamble) above.	

Shinjo **Claim 8 (Preamble)**

"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Shinjo, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 6.2 above.	

Shinjo Claim 8.1
"storing a portion of the operating system in a compressed form in a first memory"

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Shinjo, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Shinjo, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Shinjo, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Shinjo, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than

the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

In step S7, when the designated boot mode is not the maintenance mode, i.e., when it is the normal mode, the saving mode is selected, and the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data and the backup flag 15 of the backup memory 13 is set (step S8). The process of step S8 is completed and then a reboot is executed.

Shinjo, 3:28-34

In a system according to the third embodiment as shown in FIG. 4, the memory 14 includes the backup memory 13, and the disk unit 19 includes the backup file 23. In the saving mode, therefore, the main memory data stored in the main memory 12 can be saved as backup data into the backup memory 13 and backup file 23, and the backup flags 15 and 25 can be set.

Shinjo, 4:24-31

8.6 updating the boot data list,	Shinjo, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.4.1 above.	

Shinjo Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Shinjo, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 1.1 above.

> The present invention relates to a method and an apparatus for booting a computer system.

Shinjo, 1:9-10

> In a computer system, generally, whenever the system is booted, a boot process for loading firmware, an initial program loader (IPL) program and an initialize (INZ) program, an initial program loader process, and an initialization process are executed.

Shinjo, 1:11-16

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:35-48

> According to another aspect of the present invention, there is provided an apparatus for booting a computer system, the apparatus comprising: a main memory for storing main memory data; means for setting a boot mode for booting the computer system; determining means for determining whether or

not the boot mode is a normal mode; a flag to be set/reset in accordance with a determination result by the determining means; and a backup memory for storing the main memory data to be stored in the main memory immediately after the computer system is booted, as backup data when the boot mode is the normal mode and the flag is reset, and wherein the backup data stored in the backup memory is restored as the main memory data into the main memory when the boot mode is the normal mode and the flag is set.

Shinjo, 1:49-64

The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-68

When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

In step S1, it is determined whether or not the boot mode designated by the boot command output from the boot mode setting unit 17 is the maintenance mode. If the boot mode is not the maintenance mode, i.e., if it is the normal mode, it is determined whether or not the backup flag 15 of the backup memory 13 is set (step S2). That is, it is determined whether a boot process is executed in the quick start mode or saving mode.

Shinjo, 3:8-15

In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

Shinjo Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

> In the computer system according to the first embodiment wherein the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data, when the system power source is turned off, the backup data is erased. It is thus necessary to boot the system in the same manner as the conventional apparatus and save the main memory data stored in the main memory 12 into the backup memory 13 as the backup data immediately after the system is booted. By backing up the backup memory 13 by a battery or the like, such boot process in the system is executed only at the first time.

Shinjo, 3:65-4:8

> As described above, according to the present invention, when the system is first booted, the saving mode is selected. Immediately after the system is booted, the main memory data stored in the main memory is saved as backup data into the backup memory (backup file), and the backup flag is set. The system is then rebooted. Therefore, when the system is next booted in the normal mode, the quick start mode is selected and the backup data saved into the backup memory (backup file) is restored as the main memory data in the main memory. The boot process of the system is completed only by the above process, and the state immediately after the system is booted is restored.

Shinjo, 4:45-67

Shinjo Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

9.2 storing the additional portion of the operating system in the first memory, and	Shinjo, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 8.1 above.

> In the initialization process, a resident load module is loaded into a main memory, various control blocks are produced, the running environment of an operating system (OS) is set, and the running environment of an application system is set (for example, a control process is produced).

Shinjo, 1:16-21

> In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

> After the backup flag 15 is reset in step S3, the boot process is executed as in the conventional system, i.e., the OS initialization process and the application initialization process are executed (steps S4 and S5).

Shinjo, 3:54-57

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Shinjo, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 8.5 above.

> In the initialization process, a resident load module is loaded into a main memory, various control blocks are produced, the running environment of an operating system (OS) is set, and the running environment of an application system is set (for example, a control process is produced).

Shinjo, 1:16-21

> In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

> After the backup flag 15 is reset in step S3, the boot process is executed as in the conventional system, i.e., the OS initialization process and the application initialization process are executed (steps S4 and S5).

Shinjo, 3:54-57

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 9.1 above.

> The present invention relates to a method and an apparatus for booting a computer system.

Shinjo, 1:9-10

> In a computer system, generally, whenever the system is booted, a boot process for loading firmware, an initial program loader (IPL) program and an initialize (INZ) program, an initial program loader process, and an initialization process are executed.

Shinjo, 1:11-16

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:35-48

> According to another aspect of the present invention, there is provided an apparatus for booting a computer system, the apparatus comprising: a main memory for storing main memory data; means for setting a boot mode for booting the computer system; determining means for determining whether or

not the boot mode is a normal mode; a flag to be set/reset in accordance with a determination result by the determining means; and a backup memory for storing the main memory data to be stored in the main memory immediately after the computer system is booted, as backup data when the boot mode is the normal mode and the flag is reset, and wherein the backup data stored in the backup memory is restored as the main memory data into the main memory when the boot mode is the normal mode and the flag is set.

Shinjo, 1:49-64

The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-68

When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

In step S1, it is determined whether or not the boot mode designated by the boot command output from the boot mode setting unit 17 is the maintenance mode. If the boot mode is not the maintenance mode, i.e., if it is the normal mode, it is determined whether or not the backup flag 15 of the backup memory 13 is set (step S2). That is, it is determined whether a boot process is executed in the quick start mode or saving mode.

Shinjo, 3:8-15

In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

Shinjo Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

> In the computer system according to the first embodiment wherein the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data, when the system power source is turned off, the backup data is erased. It is thus necessary to boot the system in the same manner as the conventional apparatus and save the main memory data stored in the main memory 12 into the backup memory 13 as the backup data immediately after the system is booted. By backing up the backup memory 13 by a battery or the like, such boot process in the system is executed only at the first time.

Shinjo, 3:65-4:8

> As described above, according to the present invention, when the system is first booted, the saving mode is selected. Immediately after the system is booted, the main memory data stored in the main memory is saved as backup data into the backup memory (backup file), and the backup flag is set. The system is then rebooted. Therefore, when the system is next booted in the normal mode, the quick start mode is selected and the backup data saved into the backup memory (backup file) is restored as the main memory data in the main memory. The boot process of the system is completed only by the above process, and the state immediately after the system is booted is restored.

Shinjo, 4:45-67

Shinjo Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Shinjo, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Shinjo, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claim 1.1 above.

> The CPU 11 controls the entire computer system.

Shinjo, 2:32

> When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

11.2 accessing the loaded boot data in compressed form from the memory;	Shinjo, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Shinjo, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Shinjo, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in

which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

In step S7, when the designated boot mode is not the maintenance mode, i.e., when it is the normal mode, the saving mode is selected, and the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data and the backup flag 15 of the backup memory 13 is set (step S8). The process of step S8 is completed and then a reboot is executed.

Shinjo, 3:28-34

In a system according to the third embodiment as shown in FIG. 4, the memory 14 includes the backup memory 13, and the disk unit 19 includes the backup file 23. In the saving mode, therefore, the main memory data stored in the main memory 12 can be saved as backup data into the backup memory 13 and backup file 23, and the backup flags 15 and 25 can be set.

Shinjo, 4:24-31

11.5 updating the boot data list.	Shinjo, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Shinjo, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Shinjo, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Shinjo, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Shinjo, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Shinjo, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C18
Invalidity of U.S. Patent 8,880,862 based on Shinjo

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Shinjo, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Shinjo, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.1 and 1.2 above.

> The present invention relates to a method and an apparatus for booting a computer system.

Shinjo, 1:9-10

> In a computer system, generally, whenever the system is booted, a boot process for loading firmware, an initial program loader (IPL) program and an initialize (INZ) program, an initial program loader process, and an initialization process are executed.

Shinjo, 1:11-16

> According to one aspect of the present invention, there is provided a method for booting a computer system, the method comprising the steps of: setting a boot mode for booting the computer system; determining whether or not the set boot mode is a normal mode; determining whether or not a flag is set when the boot mode is the normal mode, the flag representing whether or not backup data is restorable; storing main memory data to be stored in a main memory immediately after the computer system is booted, as the backup data, into a backup memory when the flag is reset; and restoring the backup data stored in the backup memory, as the main memory data, into the main memory when the flag is set.

Shinjo, 1:35-48

> According to another aspect of the present invention, there is provided an apparatus for booting a computer system, the apparatus comprising: a main memory for storing main memory data; means for setting a boot mode for booting the computer system; determining means for determining whether or not the boot mode is a normal mode; a flag to be set/reset in accordance with a

determination result by the determining means; and a backup memory for storing the main memory data to be stored in the main memory immediately after the computer system is booted, as backup data when the boot mode is the normal mode and the flag is reset, and wherein the backup data stored in the backup memory is restored as the main memory data into the main memory when the boot mode is the normal mode and the flag is set.

Shinjo, 1:49-64

The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-68

When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

In step S1, it is determined whether or not the boot mode designated by the boot command output from the boot mode setting unit 17 is the maintenance mode. If the boot mode is not the maintenance mode, i.e., if it is the normal mode, it is determined whether or not the backup flag 15 of the backup memory 13 is set (step S2). That is, it is determined whether a boot process is executed in the quick start mode or saving mode.

Shinjo, 3:8-15

In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the

backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the OS initialization process of step S3 or the application initialization process of step S4.

Shinjo, 3:35-45

In the computer system according to the first embodiment wherein the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data, when the system power source is turned off, the backup data is erased. It is thus necessary to boot the system in the same manner as the conventional apparatus and save the main memory data stored in the main memory 12 into the backup memory 13 as the backup data immediately after the system is booted. By backing up the backup memory 13 by a battery or the like, such boot process in the system is executed only at the first time.

Shinjo, 3:65-4:8

As described above, according to the present invention, when the system is first booted, the saving mode is selected. Immediately after the system is booted, the main memory data stored in the main memory is saved as backup data into the backup memory (backup file), and the backup flag is set. The system is then rebooted. Therefore, when the system is next booted in the normal mode, the quick start mode is selected and the backup data saved into the backup memory (backup file) is restored as the main memory data in the main memory. The boot process of the system is completed only by the above process, and the state immediately after the system is booted is restored.

Shinjo, 4:45-67

14.2 loading the boot data into a memory; and	Shinjo, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Shinjo, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.2 and 1.3 above.

> When a boot command is output from the boot mode setting unit 17 to the CPU 11, an operation using the initial program loader (IPL) program is started by the CPU 11.

Shinjo, 3:4-7

> In step S1, it is determined whether or not the boot mode designated by the boot command output from the boot mode setting unit 17 is the maintenance mode. If the boot mode is not the maintenance mode, i.e., if it is the normal mode, it is determined whether or not the backup flag 15 of the backup memory 13 is set (step S2). That is, it is determined whether a boot process is executed in the quick start mode or saving mode.

Shinjo, 3:8-15

> Since the backup flag 15 has been set, the boot process is started in the quick start mode through steps S1 and S2 after the reboot. That is, in step S6, the backup data stored in the backup memory 13 is restored into the main memory 12 as the main memory data. Since the computer system is thus restored to the state immediately after the boot process and the running environment is set, the boot process of the computer system can be completed without executing the

OS initialization process of step S3 or the application initialization process of step S4. Shinjo, 3:35-45

14.4 updating the boot data list.	Shinjo, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1 (Preamble) and 1.1 above.

> In the initialization process, a resident load module is loaded into a main memory, various control blocks are produced, the running environment of an operating system (OS) is set, and the running environment of an application system is set (for example, a control process is produced).

Shinjo, 1:16-21

> In step S2, when the backup flag 15 is reset, it is determined that the backup data stored in the backup memory 13 cannot be restored. Since this determination is made in the first boot process, the same boot process as a conventional one is executed. In other words, the operating system (OS) initialization process and the application initialization process are executed by the initialization program (steps S4 and S5).

Shinjo, 3:16-23

> After the backup flag 15 is reset in step S3, the boot process is executed as in the conventional system, i.e., the OS initialization process and the application initialization process are executed (steps S4 and S5).

Shinjo, 3:54-57

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> A disk unit 19 of a computer system according to the second embodiment as shown in FIG. 3 can be used in place of the backup memory 13 shown in FIG. 1. The disk unit 19 includes a backup file 23 for storing the main memory data stored in the main memory 12 as backup data and a backup flag 25. Even through the system power source is turned off, the backup data stored in the backup file 23 is not erased. In the system according to the second embodiment, since the backup data as the main memory data is saved and restored between the main memory 12 and disk unit 19, disk access occurs. Therefore, the time required for the boot process of the system according to the first embodiment is longer than the time required for that of the system according to the first embodiment.

Shinjo, 4:9-24

> In a system according to the third embodiment as shown in FIG. 4, the memory 14 includes the backup memory 13, and the disk unit 19 includes the backup file 23. In the saving mode, therefore, the main memory data stored in the main memory 12 can be saved as backup data into the backup memory 13 and backup file 23, and the backup flags 15 and 25 can be set.

Shinjo, 4:24-37

> As described above, according to the present invention, when the system is first booted, the saving mode is selected. Immediately after the system is booted, the main memory data stored in the main memory is saved as backup data into the backup memory (backup file), and the backup flag is set. The system is then rebooted. Therefore, when the system is next booted in the normal mode, the quick start mode is selected and the backup data saved into the backup memory (backup file) is restored as the main memory data in the main memory. The boot process of the system is completed only by the above process, and the state immediately after the system is booted is restored.

Shinjo, 4:45-57

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Shinjo, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.1 and 1.2.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Shinjo, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> The disk unit 16 is constituted of, for example, a magnetic disk unit and stores various programs, data or the like which are to be booted.

Shinjo, 2:48-51

> A disk unit 19 of a computer system according to the second embodiment as shown in FIG. 3 can be used in place of the backup memory 13 shown in FIG. 1. The disk unit 19 includes a backup file 23 for storing the main memory data stored in the main memory 12 as backup data and a backup flag 25. Even through the system power source is turned off, the backup data stored in the backup file 23 is not erased. In the system according to the second embodiment, since the backup data as the main memory data is saved and restored between the main memory 12 and disk unit 19, disk access occurs. Therefore, the time required for the boot process of the system according to the first embodiment is longer than the time required for that of the system according to the first embodiment.

Shinjo, 4:9-23

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.1 and 1.2 above.

> A disk unit 19 of a computer system according to the second embodiment as shown in FIG. 3 can be used in place of the backup memory 13 shown in FIG. 1. The disk unit 19 includes a backup file 23 for storing the main memory data stored in the main memory 12 as backup data and a backup flag 25. Even through the system power source is turned off, the backup data stored in the backup file 23 is not erased. In the system according to the second embodiment, since the backup data as the main memory data is saved and restored between the main memory 12 and disk unit 19, disk access occurs. Therefore, the time required for the boot process of the system according to the first embodiment is longer than the time required for that of the system according to the first embodiment.

Shinjo, 4:9-24

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 1.1 above.	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

> The disk unit 16 is constituted of, for example, a magnetic disk unit and stores various programs, data or the like which are to be booted.

Shinjo, 2:48-51

> A disk unit 19 of a computer system according to the second embodiment as shown in FIG. 3 can be used in place of the backup memory 13 shown in FIG. 1. The disk unit 19 includes a backup file 23 for storing the main memory data stored in the main memory 12 as backup data and a backup flag 25. Even through the system power source is turned off, the backup data stored in the backup file 23 is not erased. In the system according to the second embodiment, since the backup data as the main memory data is saved and restored between the main memory 12 and disk unit 19, disk access occurs. Therefore, the time required for the boot process of the system according to the first embodiment is longer than the time required for that of the system according to the first embodiment.

Shinjo, 4:9-23

> The memory 14 includes a main memory 12 and a backup memory 13. The main memory 12 stores various programs and data as main memory data. The backup memory 13 stores the main memory data, which is stored in the main memory 12, as backup data. Since the backup memory 13 has approximately the same memory capacity as the main memory 12, the memory capacity of the memory 14 is about twice as large as that of the main memory 12.

Shinjo, 2:33-42

> The backup memory 13 includes a backup flag 15. The backup flag 15 is used to indicate whether or not the backup data stored in the backup memory 13 can be restored into the main memory 12 as the main memory data.

Shinjo, 2:43-47

> The boot mode setting unit 17 is constituted of, for example, a service processor (SVP) and used to set a boot mode in the computer system. The boot mode includes a maintenance mode for software maintenance such as replacement of the programs and patch and a mode (normal mode) other than the maintenance mode. The normal mode includes a quick start mode in which a high speed boot can be executed using the backup data and a saving mode in which the main memory data stored in the main memory 12 is saved in the backup memory 13 as the backup data, immediately after a normal boot is executed. It depends upon the set/reset state of the backup flag 15 which of the quick start mode and the saving mode is selected. More specifically, in the normal mode, when the backup flag 15 is set, the quick start mode is selected and, when the backup flag 15 is reset, the saving mode is selected.

Shinjo, 2:51-2:68

> In step S7, when the designated boot mode is not the maintenance mode, i.e., when it is the normal mode, the saving mode is selected, and the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data and the backup flag 15 of the backup memory 13 is set (step S8). The process of step S8 is completed and then a reboot is executed.

Shinjo, 3:28-34

> In the computer system according to the first embodiment wherein the main memory data stored in the main memory 12 is saved into the backup memory 13 as the backup data, when the system power source is turned off, the backup data is erased. It is thus necessary to boot the system in the same manner as the conventional apparatus and save the main memory data stored in the main memory 12 into the backup memory 13 as the backup data immediately after the system is booted. By backing up the backup memory 13 by a battery or the like, such boot process in the system is executed only at the first time.

Shinjo, 3:65-4:8

> In a system according to the third embodiment as shown in FIG. 4, the memory 14 includes the backup memory 13, and the disk unit 19 includes the backup file 23. In the saving mode, therefore, the main memory data stored in the main memory 12 can be saved as backup data into the backup memory 13 and backup file 23, and the backup flags 15 and 25 can be set.

Shinjo, 4:24-31

Shinjo **Claim 39**

"The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory."

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation:	

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Shinjo, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 16 and 23 above.	

Shinjo **Claim 88**
"The method of claim 13, wherein the operating system comprises: a plurality of files."

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 32, 33 and 49 above.	

Shinjo **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claim 32, 33, and 49 above.	

Shinjo **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Shinjo, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

Shinjo **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shinjo discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

> The disk unit 16 is constituted of, for example, a magnetic disk unit and stores various programs, data or the like which are to be booted.

Shinjo, 2:48-51

> A disk unit 19 of a computer system according to the second embodiment as shown in FIG. 3 can be used in place of the backup memory 13 shown in FIG. 1. The disk unit 19 includes a backup file 23 for storing the main memory data stored in the main memory 12 as backup data and a backup flag 25. Even through the system power source is turned off, the backup data stored in the backup file 23 is not erased. In the system according to the second embodiment, since the backup data as the main memory data is saved and restored between the main memory 12 and disk unit 19, disk access occurs. Therefore, the time required for the boot process of the system according to the first embodiment is longer than the time required for that of the system according to the first embodiment.

Shinjo, 4:9-23

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Shinjo, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shinjo discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

U.S. Patent No. 5,671,413 to Shipman ("Shipman") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Shipman, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

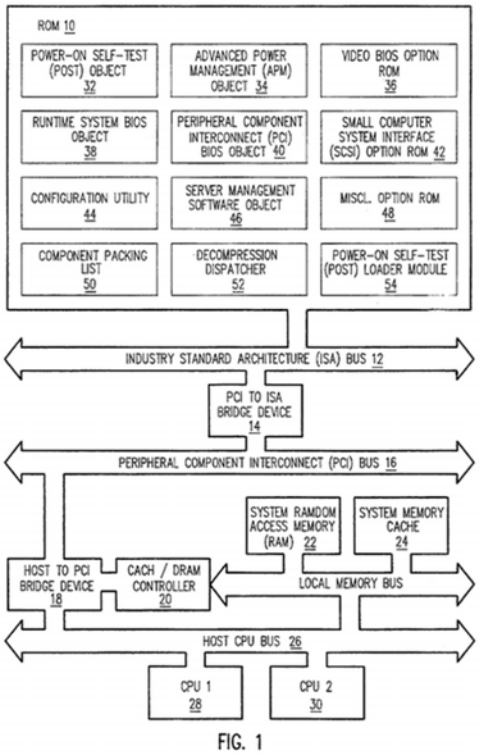


FIG. 1

Shipman, Fig. 1

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in

> a compressed state, allowing more services to be implemented without requiring more non-volatile storage.
>
> More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Shipman, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with

a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

> Component Dispatching
>
> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched

(506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

See also Shipman, Figs. 1-13

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Shipman, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the

> computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.
>
> More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

> Component Dispatching

> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510).

> If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

> Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.
>
> A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.
>
> If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Shipman, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.
>
> More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as

other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

> Component Dispatching
>
> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is

being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an

uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

1.4.1 updating the boot data list,	Shipman, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

> Component Dispatching
>
> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the

compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Shipman, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the

computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Shipman, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Shipman, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Shipman, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Shipman, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1-1.4.2 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image

(1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

Component Dispatching

Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.

. . .

When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).

If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

> Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.
>
> A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

> If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic

> input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.
>
> Shipman, 2:63-3:12

Shipman

Claim 5 (Preamble)

"A method for booting a computer system, the method comprising:"

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Shipman, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Shipman, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Shipman, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Shipman, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Shipman, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the

> service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.
>
> More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

Component Dispatching

Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.

. . .

When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).

If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list

(508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

5.6 updating the boot data list;	Shipman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Shipman, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1-1.3 above.	

6 (Preamble) A system comprising: a processor;	Shipman, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization

components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

See also Shipman, Figs. 1-13

6.1 a memory; and	Shipman, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Shipman, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.1, 1.2, and Claim 6 (Preamble) above.

> The ROM (10) is a non-volatile device used to store the BIOS components, some in a compressed state and some in an uncompressed state. The DRAM (22) stores a shadow RAM binary image of selected components stored in the ROM. For those components stored in the compressed state, they are decompressed before storing in the shadow RAM. In Intel 80386 and 80486 computers, shadow RAM is a portion of upper memory area set aside for programs retrieved from ROM.

Shipman, 3:13-21

6.3 wherein the processor is configured:	Shipman, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 6 (Preamble) above	

Shipman Claim 6.3
"wherein the processor is configured"

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Shipman, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above.	

Shipman Claim 6.4
"to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"

6.5 to access the loaded portion of the boot data in the compressed form,	Shipman, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.2 above.	

Shipman Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Shipman, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Shipman, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

Shipman Claim 6.7
"to update the boot data list."

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Shipman, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1 (Preamble) above.	

Shipman **Claim 8 (Preamble)**

"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Shipman, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Shipman, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Shipman, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.2 above.	

Shipman Claim 8.3
"accessing the loaded portion of the operating system from the second memory in the compressed form"

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Shipman, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Shipman Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Shipman, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently

executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as

other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

> Component Dispatching
>
> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

Shipman Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

8.6 updating the boot data list,	Shipman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

Shipman Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Shipman, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1-1.3 and 5.7 above.

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claim 1.1 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

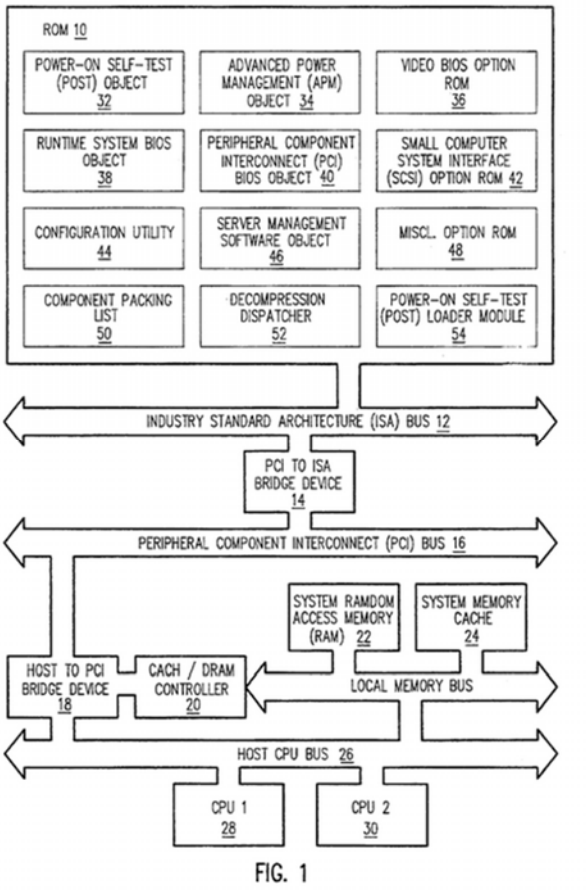


FIG. 1

Shipman, Fig. 1

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the

Shipman Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

> service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.
>
> More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman — Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

Shipman, 4:33-36

> Component Dispatching
>
> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass

Shipman Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

Shipman, 14:15-41

See also Shipman, Figs. 2-13

> The BIOS is organized into a set of components, each being dispatched as an independent executable object. Components fall into three major types: initialization components, runtime components and functional components. Initialization components are dispatched during what is commonly referred to as the Power On Self Test (POST) cycle. These components are active during initialization and terminated prior to loading the Operating System.
>
> Initialization components may contain direct runtime dependencies but these dependencies are limited to objects of the same Component Class. For the illustrated embodiment, all initialization components are segment relocatable. An example of an initialization component is the POST Object.
>
> Runtime components are objects that are uncompressed or remain decompressed and executable after loading of the Operating System. Typically, runtime components are combined to form the 64K System BIOS image located in the F0000h segment. For the illustrated embodiment, runtime are segment relocatable components do not contain any direct dependencies to any other component regardless of its type or class, but may contain indirect dependencies to other runtime components. An example of a runtime component is the Runtime System BIOS Object.
>
> Functional components are dispatched on an as needed basis. For the illustrated embodiment, a Functional component does not contain any direct dependencies to any other component regardless of type or class. An example of a Functional component is the Configuration Utility.

Shipman, 4:40-5:2

Shipman Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

9.2 storing the additional portion of the operating system in the first memory, and	Shipman, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claim 8.1 above.

> The BIOS is organized into a set of components, each being dispatched as an independent executable object. Components fall into three major types: initialization components, runtime components and functional components. Initialization components are dispatched during what is commonly referred to as the Power On Self Test (POST) cycle. These components are active during initialization and terminated prior to loading the Operating System.
>
> Initialization components may contain direct runtime dependencies but these dependencies are limited to objects of the same Component Class. For the illustrated embodiment, all initialization components are segment relocatable. An example of an initialization component is the POST Object.
>
> Runtime components are objects that are uncompressed or remain decompressed and executable after loading of the Operating System. Typically, runtime components are combined to form the 64K System BIOS image located in the F0000h segment. For the illustrated embodiment, runtime are segment relocatable components do not contain any direct dependencies to any other component regardless of its type or class, but may contain indirect dependencies to other runtime components. An example of a runtime component is the Runtime System BIOS Object.
>
> Functional components are dispatched on an as needed basis. For the illustrated embodiment, a Functional component does not contain any direct dependencies to any other component regardless of type or class. An example of a Functional component is the Configuration Utility.

Shipman, 4:40-5:2

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Shipman, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claim 8.5 above.

> The BIOS is organized into a set of components, each being dispatched as an independent executable object. Components fall into three major types: initialization components, runtime components and functional components. Initialization components are dispatched during what is commonly referred to as the Power On Self Test (POST) cycle. These components are active during initialization and terminated prior to loading the Operating System.
>
> Initialization components may contain direct runtime dependencies but these dependencies are limited to objects of the same Component Class. For the illustrated embodiment, all initialization components are segment relocatable. An example of an initialization component is the POST Object.
>
> Runtime components are objects that are uncompressed or remain decompressed and executable after loading of the Operating System. Typically, runtime components are combined to form the 64K System BIOS image located in the F0000h segment. For the illustrated embodiment, runtime are segment relocatable components do not contain any direct dependencies to any other component regardless of its type or class, but may contain indirect dependencies to other runtime components. An example of a runtime component is the Runtime System BIOS Object.
>
> Functional components are dispatched on an as needed basis. For the illustrated embodiment, a Functional component does not contain any direct dependencies to any other component regardless of type or class. An example of a Functional component is the Configuration Utility.

Shipman, 4:40-5:2

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Shipman, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claim 9.1 above.

See also

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

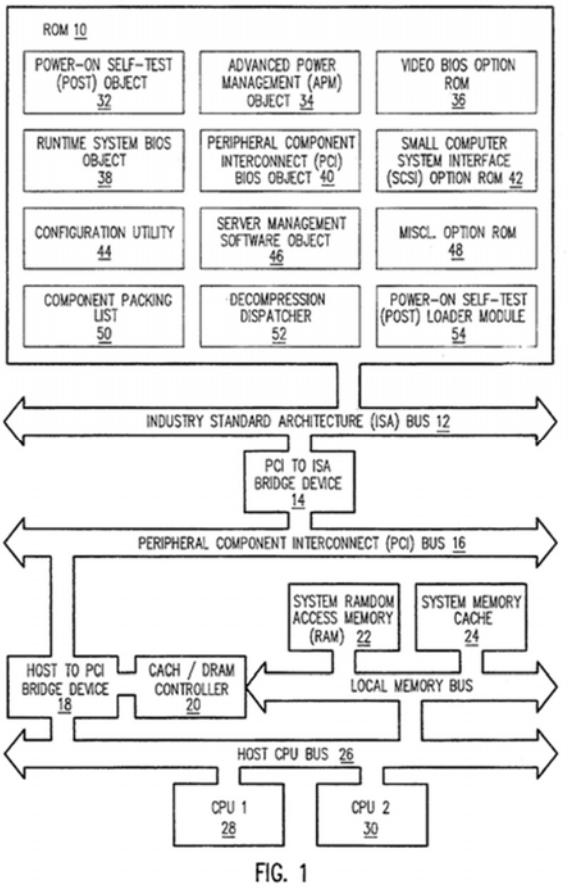


FIG. 1

Shipman, Fig. 1

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the

Shipman Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Shipman, 4:33-36

> Component Dispatching
>
> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass

Shipman Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Shipman, 14:15-41

See also Shipman, Figs. 2-13

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 4:63-5:12

Shipman Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Shipman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1 (Preamble) above.	

Shipman **Claim 11 (Preamble)**

"A method for providing accelerated loading of an operating system in a computer system, comprising"

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Shipman, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claim 1.1 and 6(Preamble) above.

> Initialization components are dispatched during what is commonly referred to as the Power On Self Test (POST) cycle. These components are active during initialization and terminated prior to loading the Operating System. Initialization components may contain direct runtime dependencies but these dependencies are limited to objects of the same Component Class. For the illustrated embodiment, all initialization components are segment relocatable. An example of an initialization component is the POST Object.

Shipman, 4:46-4:54

> Power On Self Test
>
> The Power On Self Test component (32) is the main POST executable, and is responsible for initialization of the system, and causing all necessary components to be dispatched. This component resides in a compressed state in the ROM image, and as described earlier, this component is decompressed and dispatched into DRAM by the decompression dispatcher.

Shipman, 6:60-6:67

> System BIOS Runtime Executable
>
> The System BIOS Runtime Executable component is responsible for all compatible System Services including Interrupt and Device Service Routines. This component contains the Shutdown Handler responsible for handling all software related resets and dispatching the POST Loader as needed. This

component resides in a compressed state in the ROM state, and it is decompressed and dispatched into DRAM by the decompression dispatcher.

Shipman, 7:1-7:10

Shipman Claim 11.1

"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

11.2 accessing the loaded boot data in compressed form from the memory;	Shipman, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Shipman, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Shipman, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM, accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the

computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

> Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

> Component Dispatching

Shipman Claim 11.4
"utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

> Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.
>
> . . .
>
> When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).
>
> If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

> Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510).

> If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

> Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.
>
> A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.
>
> If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

11.5 updating the boot data list.	Shipman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Shipman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Shipman, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Shipman, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Shipman, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Shipman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Shipman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Shipman, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.1 and 1.2 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility

(1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image (1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

Component Dispatching

Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.

. . .

When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).

If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end

of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are placed in the ROM uncompressed have identical Compressed and

Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

14.2 loading the boot data into a memory; and	Shipman, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Shipman, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.2 and 1.3 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> In IBM compatible personal computers a set of programs called basic input/output system (BIOS) are encoded in read-only memory (ROM). The BIOS facilitates the transfer of data and instructions between a central processing unit (CPU) and peripheral devices such as disk drives. Computer systems are designed to perform functional tests of the BIOS every time the computer is turned on. When the computer is turned on, BIOS is copied to an area of random access memory (RAM) set aside for it. Since a RAM is much faster acting than a ROM,

accessing the BIOS code from RAM results in much faster initialization of the computer.

Shipman, 1:12-22

The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:35-59

The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis.

Shipman, 1:60-2:7

> When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 2:14-22

> Refer to FIG. 1 which is a block diagram of a computer system in which the present invention is embodied. The computer includes a read only memory, or ROM, (10), a dynamic random access memory, or DRAM, (22), a system memory cache (24), a cache/DRAM controller (20), and central processing units (28, 30). A set of programs called basic input/output system (BIOS) are encoded in ROM (10). The BIOS facilitates the transfer of data and instructions between the central processing units (28, 30) and peripheral devices. In the present invention the BIOS is organized into a set of BIOS components (32 through 54), each of which is capable of being dispatched as an independent executable object. Each BIOS component is a self-contained (link independent) binary image that performs a certain task or function. The BIOS components fall into three major types: initialization components, runtime components and functional components.

Shipman, 2:63-3:12

> Compressed code is the code that resides in the ROM in a compressed state. Decompressed code is code that has been decompressed from its compressed state inside the ROM and now resides in DRAM (22) shadowed memory, in a decompressed state. Uncompressed code is code that resides inside the ROM in an uncompressed state. This is code that was never compressed.

Shipman, 3:22-28

> FIG. 13 is a flow diagram of a BIOS build process that automates compressing and packing components into a final ROM binary image. A list of binary images and associated attributes to include in the final ROM image (1300) and a number of binary image files (1302) are provided to a ROM Packing Utility (1304). The ROM Packing Utility (1304) creates a packed ROM image (1306) and a packing list (1308) that are merged by a merge utility (1310) into a final ROM image

(1312) and a packing map (1314) that are stored in the ROM (10) shown in FIG. 1.

Shipman, 3:55-65

Packing List--The Packing List is generated by the Packing Utility and describes all the attributes of the packed ROM image. This list is used by the decompression dispatcher to dispatch components.

Shipman, 4:33-36

Component Dispatching

Components are dispatched through a utility called the decompression dispatcher. There are two types of dispatching supported by the decompression dispatcher, active and passive.

. . .

When dispatching components the decompression dispatcher works from a Packing List provided in the ROM image by the BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The Packing List provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 5:3-34

Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).

If the required size is available (322), then the memory allocation table is updated (330). If the end of the component list in the packing list has not been reached (332), then the flow returns to block (310). If the end

of the component list in the packing list has been reached (332), then the flow ends (334).

Shipman, 12:17-35

Refer to FIG. 5 which is a flow diagram of how components are dispatched. The dispatcher assumes only one component from a class is being loaded, and hence the count of components is set to 1. The counter is used as an index into the packing list. The component of a class might contain a number of components referenced by a subclass identifier. A check (504) is made to determine if all components of the class are to be loaded. If yes, then the first subclass identifier is fetched (506) as described in FIG. 7. If the subclass is found in the packing list (508) then the count of the subclass index counter is incremented (510). If the subclass is not found in the packing list (508), then the count of the subclass index counter is not incremented and the flow proceeds to get packing information for the component indexed by the count of the component counter (512). The procedure next decompresses the component at the index count and places the decompressed Code in RAM (514). The details of block (514) are shown in more detail in FIG. 12. At the end of the decompressing procedure, a check (516) is made to determine if there are more components in this class to decompressed. If yes, the component index counter is decremented (518). If no, a check (520) is made to determine if this is the last component actively dispatched or if an option ROM is present. If not, the flow ends (524). If yes, then the return address is adjusted to execute the dispatched component (522).

Shipman, 12:49-13:7

Refer to FIG. 12 which is a flow diagram of the procedure for getting a specified component transferred and stored in RAM in an uncompressed and executable state.

A RAM memory block large enough to hold the specified component is found (1202) and allocated. After the memory space is allocated (1204), the size field in the packing list is accessed to get the size information (1206). The compressed and decompressed size information is obtained from the packing list entry (1206). The Compressed Size field specifies the amount of space (in bytes) consumed by a component when it is in the ROM image. Unitialized Data components use this field to specify the maximum amount of memory required and that they be aligned on a 64K boundary. The Decompressed Size field specifies the amount of space (in bytes) consumed by a component after it has been decompressed and dispatched. As described earlier, components that are

placed in the ROM uncompressed have identical Compressed and Decompressed sizes. Unitialized Data components specify this field to be identical to the Compressed Size field.

If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).

Shipman, 14:15-41

See also Shipman, Figs. 1-13

14.4 updating the boot data list.	Shipman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Shipman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1 (Preamble) and 1.1 above.

> The BIOS is organized into a set of components, each being dispatched as an independent executable object. Components fall into three major types: initialization components, runtime components and functional components. Initialization components are dispatched during what is commonly referred to as the Power On Self Test (POST) cycle. These components are active during initialization and terminated prior to loading the Operating System.
>
> Initialization components may contain direct runtime dependencies but these dependencies are limited to objects of the same Component Class. For the illustrated embodiment, all initialization components are segment relocatable. An example of an initialization component is the POST Object.
>
> Runtime components are objects that are uncompressed or remain decompressed and executable after loading of the Operating System. Typically, runtime components are combined to form the 64K System BIOS image located in the F0000h segment. For the illustrated embodiment, runtime are segment relocatable components do not contain any direct dependencies to any other component regardless of its type or class, but may contain indirect dependencies to other runtime components. An example of a runtime component is the Runtime System BIOS Object.
>
> Functional components are dispatched on an as needed basis. For the illustrated embodiment, a Functional component does not contain any direct dependencies to any other component regardless of type or class. An example of a Functional component is the Configuration Utility.

Shipman, 4:40-5:2

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The decompression dispatcher component (52) dispatches a Component into memory (22). Each Component is identified by a unique Class/Sub-Class code combination. Dispatched Components are recorded in an Allocation Table kept by the Memory Manager. The actual base address of a Component may be retrieved at any time by calling a Get Physical Address() function.

Shipman, 6:42-46

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Shipman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Shipman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.1 and 1.2 above

> The BIOS is organized into a set of components, each being dispatched as an independent executable object. Components fall into three major types: initialization components, runtime components and functional components. Initialization components are dispatched during what is commonly referred to as the Power On Self Test (POST) cycle. These components are active during initialization and terminated prior to loading the Operating System.
>
> Initialization components may contain direct runtime dependencies but these dependencies are limited to objects of the same Component Class. For the illustrated embodiment, all initialization components are segment relocatable. An example of an initialization component is the POST Object.
>
> Runtime components are objects that are uncompressed or remain decompressed and executable after loading of the Operating System. Typically, runtime components are combined to form the 64K System BIOS image located in the F0000h segment. For the illustrated embodiment, runtime are segment relocatable components do not contain any direct dependencies to any other component regardless of its type or class, but may contain indirect dependencies to other runtime components. An example of a runtime component is the Runtime System BIOS Object.
>
> Functional components are dispatched on an as needed basis. For the illustrated embodiment, a Functional component does not contain any direct dependencies to any other component regardless of type or class. An example of a Functional component is the Configuration Utility.

Shipman, 4:40-5:2

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Shipman, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.1 and 1.2 above

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, 1:35-46

> The ROM (10) is a non-volatile device used to store the BIOS components, some in a compressed state and some in an uncompressed state. The DRAM (22) stores a shadow RAM binary image of selected components stored in the ROM. For those components stored in the compressed state, they are decompressed before storing in the shadow RAM. In Intel 80386 and 80486

> computers, shadow RAM is a portion of upper memory area set aside for programs retrieved from ROM.
>
> Shipman, 3:13-21
>
> If the component is not compressed, then the component is copied (1212) to RAM from the non-volatile storage device used to store the compressed or uncompressed BIOS code. If the component is compressed, then the component is decompressed (1210), stored in an uncompressed state in the shadowed memory portion of RAM, and the procedure stops (1216).
>
> Shipman, 14:35-41

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.1 and 1.2 above	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 1.1 above	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Shipman, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.1 and 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.1 and 1.2 above

> The ROM (10) is a non-volatile device used to store the BIOS components, some in a compressed state and some in an uncompressed state. The DRAM (22) stores a shadow RAM binary image of selected components stored in the ROM. For those components stored in the compressed state, they are decompressed before storing in the shadow RAM. In Intel 80386 and 80486 computers, shadow RAM is a portion of upper memory area set aside for programs retrieved from ROM.

Shipman, 3:13-21

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claim 10 above.

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and optionally removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, Abstract

> The BIOS stored in a read only memory (ROM) is organized into a set of components, each being dispatched as an independent executable object after being copied to a shadow memory portion of a random access memory (RAM). Three types of components are defined, initialization components, runtime components and functional components. The initialization components are dispatched during a Power On Self Test (POST) cycle such that the initialization components are active during initialization. The initialization components are terminated prior to loading an operating system. The runtime components are maintained in the uncompressed/decompressed state and executable after loading of the operating system. The functional components are decompressed and dispatched on an as needed basis. The components can be actively or passively dispatched. Control over a dispatched object is returned to a requester by the decompression dispatcher upon a condition that the component is to be passively dispatched and the passively dispatched component is recorded in a memory manager. Control over a dispatched object is passed to the decompressed component upon a condition that the component

is to be actively dispatched. When dispatching components the decompression dispatcher works from a packing list provided in the ROM image by a BIOS build process. To optimize device usage, the binary images that comprise the different components are packed into the final ROM image. The packing list provides a detailed description of the packing as well as other important pieces of information regarding the types of components that have been packed into the ROM.

Shipman, 1:60-2:22

Shipman **Claim 49**

"The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form."

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

> The services to be provided by a basic input/output system (BIOS) of a computer system are implemented via a number of independently executable service components. Additionally, the BIOS is provided with a decompression dispatcher for decompressing and dispatching the service components into random access memory (RAM) of the computer system for execution on an as needed basis, and removing the dispatched service components when they are no longer needed. As a result, the service components may be stored in a non-volatile storage in a compressed state, allowing more services to be implemented without requiring more non-volatile storage.

Shipman, 1:34-46

> More specifically, different basic input/output system (BIOS) functions are split into components, or objects, and selected ones of the components are stored in a compressed state and these compressed components are only decompressed and executed when needed. A decompression dispatcher software takes a request from currently executing BIOS code to decompress another portion of BIOS code that may be needed. The requester can specify whether or not to just decompress the portion of code into memory or to decompress and initialize the decompressed BIOS. The decompression dispatcher can be used to dispatch different portions of BIOS code to different processors in a multi-processor system.

Shipman, 1:47-59

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Shipman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

Shipman **Claim 59**

"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Shipman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 31 above.	

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 32, 33 and 49 above.	

Shipman **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

Appendix C19
Invalidity of U.S. Patent 8,880,862 based on Shipman

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Shipman, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 32, 33 and 49 above.	

Shipman **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

Shipman **Claim 83**
"The method of claim 13, wherein the boot data in the compressed form represents a plurality of files."

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

Shipman **Claim 84**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system."

87. The method of claim 13, wherein the memory comprises: a physical memory.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 27 above.	

Shipman **Claim 87**
"The method of claim 13, wherein the memory comprises: a physical memory."

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 15, 17 and 24 above.	

Shipman **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 31 above.	

Shipman **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 32, 33 and 49 above.	

Shipman **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Shipman, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claim 32, 33, and 49 above.	

Shipman **Claim 93**
"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Shipman, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

Shipman **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Shipman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Shipman discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Shipman, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

The memory manager is responsible for managing memory allocation for the BIOS' memory. Memory testing and subsequent update of status are all done in `Real Big` mode as opposed to constantly switching between Real and Protected Mode. `Real Big` mode is a mode where the data segment registers of the processor are loaded with Protected Mode type characteristics while the code and stack segment registers retain their Real Mode characteristics. The processor essentially runs in Real Mode while allowing a full 4 GB access to memory.

Shipman, 10:25-34

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Shipman, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 1.1, 5, 9.1 and 8 above.

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).

Shipman, 12:17-29

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Shipman, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Shipman discloses this limitation:

See Claims 5.1, 8.1 and 9.1 above.

> Refer to FIG. 3 which is a flow diagram of initializing the decompression dispatcher of FIG. 2. The dispatcher data and executable code is copied (302) from the ROM to the RAM. The dispatcher interrupt vector is installed (304), the memory allocation table is cleared (306) and the component handle table is cleared (308). The compressed BIOS component information in the packing list is fetched from RAM (310). The relocation type field is examined to find what type of relocation is supported, below 1 meg. (312, 314), above 1 meg. (316, 318), or anywhere (320). If the required size is not available (322) allocation error flags are set (324). If the specified load address is not available (326) allocation error flags are set (328).

Shipman, 12:17-29

> The ROM (10) is a non-volatile device used to store the BIOS components, some in a compressed state and some in an uncompressed state. The DRAM (22) stores a shadow RAM binary image of selected components stored in the ROM. For those components stored in the compressed state, they are decompressed before storing in the shadow RAM. In Intel 80386 and 80486 computers, shadow RAM is a portion of upper memory area set aside for programs retrieved from ROM.

Shipman, 3:13-21

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

U.S. Patent No. 5,860,083 to Sukegawa ("Sukegawa") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

In addition, Apple incorporates by reference, as if set forth fully herein, all arguments related to Sukegawa in pending inter partes review petitions IPR2016-1365, IPR2016-01366, IPR2016-01737, and IPR2016-01738.

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Sukegawa, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this claim limitation:

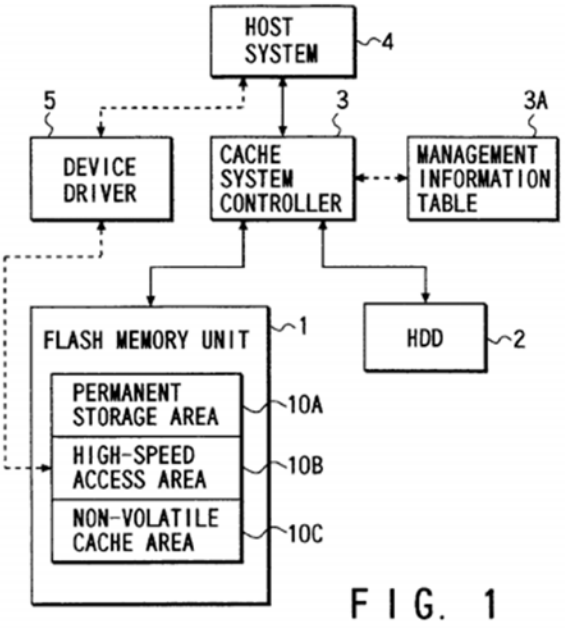


FIG. 1

Sukegawa, Fig. 1. *See also* Sukegawa, 5:14-6:17, 6:19-58, 7:28-55.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Sukegawa, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this claim limitation:

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Sukegawa, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this claim limitation:

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."

Sukegawa, 4:1-21.

> "The first embodiment relates to a system wherein the permanent storage area 10A of flash memory unit 1 is used as a cache memory area."

Sukegawa, 5:10-12.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data

> storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig 1, Fig. 4, Fig. 5.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Sukegawa, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this claim limitation:

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."

Sukegawa, 4:1-21.

> "The first embodiment relates to a system wherein the permanent storage area 10A of flash memory unit 1 is used as a cache memory area."

Sukegawa, 5:10-12.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig 1, Fig. 4, Fig. 5.

1.4.1 updating the boot data list,	Sukegawa, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's

Invalidity Contentions.

Sukegawa discloses this claim limitation:

> "The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:22-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when

> the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

> "The permanent storage area 10A and non-volatile cache area 10C function as cache memory areas of the HDD 2. Normally, each time the data in the cache memory area is updated, the updated data is written in the HDD 2. In this embodiment, the user can set the mode of each of the areas 10A and 10C, thereby determining whether or not the updated data should be written in the HDD 2 each time the data is updated."

Sukegawa, 9:19-29.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 9:53-10:52, 11:7-19, Fig 1, Fig. 4, Fig. 5

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Sukegawa, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this claim limitation:

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 9:53-10:52, 11:7-19, Fig 1, Fig. 4, Fig. 5

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Sukegawa, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Sukegawa, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claims 1-1.4.2 above.

See also

Sukegawa discloses this claim limitation:

> "The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:22-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Sukegawa, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Sukegawa, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Sukegawa, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Sukegawa, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Sukegawa, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver).
>
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "The first embodiment relates to a system wherein the permanent storage area 10A of flash memory unit 1 is used as a cache memory area."

Sukegawa, 5:10-12.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area lOA of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig 1, Fig. 4, Fig. 5.

5.6 updating the boot data list;	Sukegawa, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Sukegawa, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

6 (Preamble) A system comprising: a processor;	Sukegawa, as evidenced by the example citations below, discloses "a system comprising: a processor:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* ***Add disclosure from '608 Patent Claim 1.2***	

6.1 a memory; and	Sukegawa, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Sukegawa, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> "The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:22-30.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control

> information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.
See also Sukegawa, 6:19-58

6.3 wherein the processor is configured:	Sukegawa, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See also

> "The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:22-30.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.
See also Sukegawa, 6:19-58

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Sukegawa, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Sukegawa, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.2 above.	

Sukegawa Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Sukegawa, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

Sukegawa Claim 6.7
"to update the boot data list."

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Sukegawa, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1 (Preamble) above.	

Sukegawa **Claim 8 (Preamble)**
"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Sukegawa, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claim 6.2 above.

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Sukegawa, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Sukegawa, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Sukegawa, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Sukegawa Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Sukegawa, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver).
>
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "The first embodiment relates to a system wherein the permanent storage area 10A of flash memory unit 1 is used as a cache memory area."

Sukegawa, 5:10-12.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area lOA of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig 1, Fig. 4, Fig. 5.

8.6 updating the boot data list,	Sukegawa, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Sukegawa, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

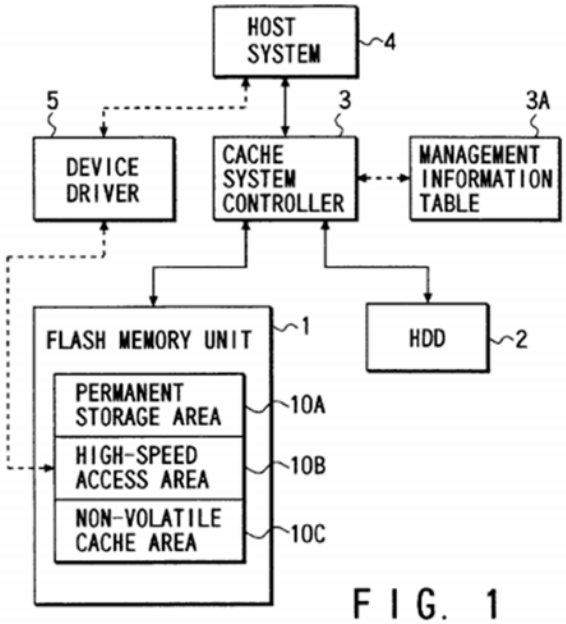
9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claim 1.1 above.

See also



F I G. 1

Sukegawa, Fig. 1

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as

shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.

The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver).”

Sukegawa, 4:1-21.

“The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD.”

Sukegawa, 4:22-30.

“According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data.”

Sukegawa, 2:65-3:3.

“(First Modification of the First Embodiment)
FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area lOA of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation.”

Sukegawa Claim 9.1

“The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list”

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 4, Fig. 5.

9.2 storing the additional portion of the operating system in the first memory, and	Sukegawa, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 8.1 above.	

Sukegawa Claim 9.2
"storing the additional portion of the operating system in the first memory"

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Sukegawa, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 8.5 above.	

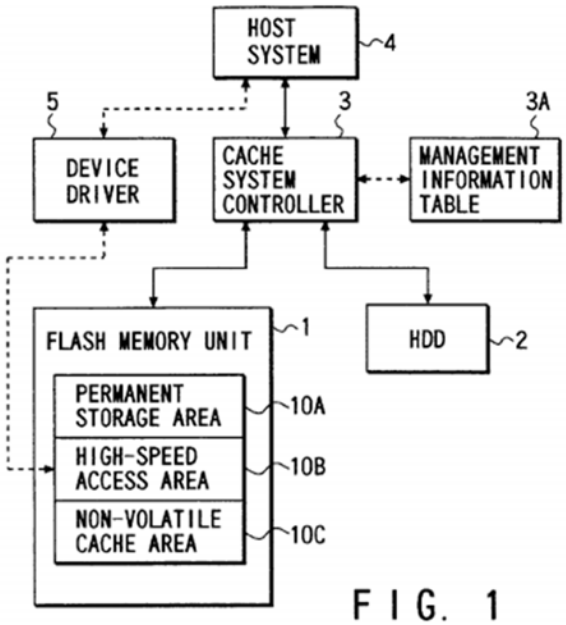
10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claim 9.1 above.

See also



F I G. 1

Sukegawa, Fig. 1

"(Data Storage System)
It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as

Sukegawa Claim 10
"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.

The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver).”

Sukegawa, 4:1-21.

“The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD.”

Sukegawa, 4:22-30.

“According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data.”

Sukegawa, 2:65-3:3.

“(First Modification of the First Embodiment)
FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area lOA of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation.”

Sukegawa Claim 10

“The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder”

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 4, Fig. 5.

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Sukegawa, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1 (Preamble) above.	

Sukegawa **Claim 11 (Preamble)**
"A method for providing accelerated loading of an operating system in a computer system, comprising"

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Sukegawa, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claim 1.1 above.

See also

> "The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:22-30.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area lOA of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control

> information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa, 6:19-58.

11.2 accessing the loaded boot data in compressed form from the memory;	Sukegawa, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Sukegawa, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Sukegawa, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver).
>
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "The first embodiment relates to a system wherein the permanent storage area 10A of flash memory unit 1 is used as a cache memory area."

Sukegawa, 5:10-12.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area lOA of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig 1, Fig. 4, Fig. 5.

11.5 updating the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Sukegawa, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Sukegawa, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Sukegawa, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Sukegawa, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claim 1.3 above.

13.4 updating the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Sukegawa, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Sukegawa, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> "The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:22-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5.

14.2 loading the boot data into a memory; and	Sukegawa, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Sukegawa, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area 10B or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."

Sukegawa, 4:1-21.

> "The first embodiment relates to a system wherein the permanent storage area 10A of flash memory unit 1 is used as a cache memory area."

Sukegawa, 5:10-12.

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

See also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig 1, Fig. 4, Fig. 5.

14.4 updating the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: Claims 1 (Preamble) and 1.1 above.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See Claim 15 above.

See also

> The first embodiment relates to a system wherein the permanent storage area lOA of flash memory unit 1 is used as a cache memory area. In this embodiment, it is supposed that the user desires to start a frequently used application program (AP) at high speed at all times.
>
> The user starts a data storage utility program of the cache system controller 3 via a user interface provided in the host system 4 (step S1). The data storage utility program reads specified data from the HDD 2 and stores the read data in a specified storage area in the flash memory unit 1. In this case, it is assumed that the user sets the permanent storage area 10A in the flash memory unit 1 as the data storage area, at the time of instructing the start of the data storage utility program (step S2).
>
> Then, the user instructs the host system 4 to start the AP (step S3). The host system 4, upon receiving the AP start instruction, issues a read command to the controller 3 in order to read control information necessary for the start of the AP from the HDD 2.
>
> The controller 3 controls the HDD 2, reads out the control information necessary for the start of the AP and transfers the read-out control information to the host system 4 (step S4). At this time, according to the started-up data storage utility program, the controller 3 stores the AP control information read out from the HDD 2 in the permanent storage area 10A of flash memory unit 1 (step S5). When the AP has been prepared to start, the data storage utility program is stopped by the instruction from the user ("YES" in step S6; step S7). Through these

operations, the control information necessary for starting the AP is stored in the permanent storage area 10A in the flash memory unit 1.

Sukegawa 5:10-40

Sukegawa **Claim 17**

"The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are

> frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5.

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See also

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

"According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3

The first embodiment relates to a system wherein the permanent storage area lOA of flash memory unit 1 is used as a cache memory area. In this embodiment, it is supposed that the user desires to start a frequently used application program (AP) at high speed at all times.

The user starts a data storage utility program of the cache system controller 3 via a user interface provided in the host system 4 (step S1). The data storage utility program reads specified data from the HDD 2 and stores the read data in a specified storage area in the flash memory unit 1. In this case, it is assumed that the user sets the permanent storage area 10A in the flash memory unit 1 as the data storage area, at the time of instructing the start of the data storage utility program (step S2).

Then, the user instructs the host system 4 to start the AP (step S3). The host system 4, upon receiving the AP start instruction, issues a read command to the controller 3 in order to read control information necessary for the start of the AP from the HDD 2.

The controller 3 controls the HDD 2, reads out the control information necessary for the start of the AP and transfers the read-out control information to the host system 4 (step S4). At this time, according to the started-up data storage utility program, the controller 3 stores the AP control information read out from the HDD 2 in the permanent storage area 10A of flash memory unit 1 (step S5). When the AP has been prepared to start, the data storage utility program is stopped by the instruction from the user ("YES" in step S6; step S7). Through these operations, the control information necessary for starting the AP is stored in the permanent storage area 10A in the flash memory unit 1.

Sukegawa 5:10-40

"(First Modification of the First Embodiment)
FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of

Sukegawa **Claim 32**

"The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form."

> operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa, 6:45-58.

see also Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5.

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Sukegawa discloses this limitation:

See

> "(Data Storage System)
> It is assumed that the data storage system of the present invention is applied to a computer system such as a personal computer. Thus, as shown in FIG. 1, this data storage system has a flash memory unit 1 constituted by a flash EEPROM, a disk drive or an HDD (Hard Disk Drive) 2, and a controller. The controller includes a cache system controller 3 (hereinafter called "controller") for performing a cache function of using the flash memory unit 1 as a cache memory, and a device driver (software) 5 with no cache function.
>
> The device driver 5 has a function of controlling the flash memory unit 1 under the management of the OS (Operation System) of a host system 4. In this invention, in particular, the device driver 5 performs the access control (to be described later) of a highspeed access area lOB or a specific storage area in the flash memory unit 1. The controller 3 performs data input/output control (including cache operation control) for the flash memory unit 1 and HDD 2 via respective device drivers (i.e. a flash memory driver and a hard disk driver)."
> The host system 4 refers to a computer body comprising a CPU of the computer system, a main memory storing the OS and an application program (AP), and other various structural elements. The controller 3 is provided between the host system 4 and the storage units 1 and 2. The controller 3 controls the flash memory unit 1 and HDD 2, as an

> integrated storage system, in accordance with access requests (read/write commands) issued from the host system 4 to the HDD."

Sukegawa, 4:1-30.

> "According to this system, for example, control information necessary for starting an application program (AP) and an OS, which are frequently used, is stored in the first storage area. Thus, the storage area of the flash memory can be effectively used in accordance with the function and the condition of use of data."

Sukegawa, 2:65-3:3;

> "(First Modification of the First Embodiment)
> FIG. 4 is a flow chart illustrating a first modification of the first embodiment. This modification relates to a system having a mode (data storage mode) for storing the control information necessary for starting the OS in the permanent storage area 10A of flash memory unit 1, for example, when the OS of the host system 4 is started in a series of operations from the turn-on of power to the completion of the starting operation."

Sukegawa, 6:18-26.

> "According to the data storage utility program, the control information, which is pre-stored in the HDD 2 and necessary for starting the OS, is read out and stored in the permanent storage area 10A (steps S16 and S17). The controller 3 transfers to the host system 4 the control information necessary for starting the OS read out from the HDD. Based on the control information, the host system 4 starts the OS."

Sukegawa, 6:35-42.

> "According to this system, when the OS is automatically started by the control information read out from the HDD 2 at the time of turning-on of power, the control information is stored in the permanent storage area 10 used as the cache memory area for the HDD 2. Accordingly, when the OS is started at the time of the next turning-on of power, the control information necessary for starting the OS is read out not from the HDD 2 but from the permanent storage area 10 or cache memory area, and the read-out control information is transferred to the host system 4. Thus, the control information can be accessed from the permanent storage area 10A in the flash memory unit 1 having a higher access speed than the HDD 2. As a result, the OS can be started at higher speed."

Sukegawa **Claim 39**

"The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory."

Sukegawa, 6:45-58. *see also* Sukegawa 5:1-7:2, 7:28-55, 9:1-10, 10:33-52, 11:7-19, Fig. 1, Fig. 4, Fig. 5.

Sukegawa **Claim 39**

"The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory."

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* claim 6 above.	

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Sukegawa, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

Sukegawa **Claim 59**
"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 31 above.	

Appendix C20
Invalidity of U.S. Patent 8,880,862 based on Sukegawa

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 32, 33 and 49 above.	

Sukegawa **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 31 above.	

Sukegawa **Claim 91**
"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 32, 33 and 49 above.	

Sukegawa **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claim 32, 33, and 49 above.	

Sukegawa **Claim 93**
"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

Sukegawa **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Sukegawa, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Sukegawa discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C21
Invalidity of U.S. Patent 8,880,862 based on Surine

U.S. Patent No. 6,212,632 to Surine ("Surine") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C21
Invalidity of U.S. Patent 8,880,862 based on Surine

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Surine, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At boot time, or power-up, boot code 202 executes, decompresses compressed code 201 into camera system code 203 and loads camera system code 203 into RAM 102."

Surine, 2:4-7, Fig. 2.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Surine, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "Typically, as shown in FIG. 2, the camera system code 203 (e.g., operating system software and its associated data structures, resources, etc.) is stored as compressed code 201 in non-volatile ROM 104."

Surine, 2:1-4, Fig. 2.

> "At boot time, or power-up, boot code 202 executes, decompresses compressed code 201 into camera system code 203 and loads camera system code 203 into RAM 102."

Surine, 2:4-7, Fig. 2.

> "Consequently, these digital cameras and other performance-oriented types of embedded system consumer electronic devices transfer a compressed image of their system code from a non-volatile ROM to a faster RAM at power up. The system code then executes from RAM."

Surine, 2:21-25.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

> "Patch manager 405 subsequently executes. Patch manager 405 includes decompression software which decompresses compressed high-use functions 402 and, as described in greater detail in the discussion of FIG.

> 5 below, loads the resulting decompressed high-use functions 416 into RAM 315."

Surine, 5:35-40.

> "Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for high-use functions 416."

Surine: 6:32-34.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Surine, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At boot time, or power-up, boot code 202 executes, decompresses compressed code 201 into camera system code 203 and loads camera system code 203 into RAM 102."

Surine, 2:4-7, Fig. 2.

> "Consequently, these digital cameras and other performance-oriented types of embedded system consumer electronic devices transfer a compressed image of their system code from a non-volatile ROM to a faster RAM at power up. The system code then executes from RAM."

Surine, 2:21-25.

> "In this embodiment, in addition to instantiating certain high-use functions, a memory configuration manager decompresses new software functions out of ROM, instantiates them in RAM, and updates the function pointer table to link them dynamically, as the capability of the new software functions are needed."

Surine, 3:27-32.

> "Patch manager 405 subsequently executes. Patch manager 405 includes decompression software which decompresses compressed high-use functions 402 and, as described in greater detail in the discussion of FIG. 5 below, loads the resulting decompressed high-use functions 416 into RAM 315."

Surine, 5:35-40.

> "Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for

high-use functions 416." Surine: 6:32-34.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Surine, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

> "At boot time, or power-up, boot code 202 executes, decompresses compressed code 201 into camera system code 203 and loads camera system code 203 into RAM 102."

Surine, 2:4-7, Fig. 2.

> "Additionally, the system of the present invention maintains the speed and responsiveness of the device while reducing the amount of RAM used in the device. In comparison to prior art embedded system devices, a device in accordance with the present invention either uses less RAM and is thus less expensive, or runs faster using the same amount of RAM."

Surine, 2:49-55.

> "In this embodiment, in addition to instantiating certain high-use functions, a memory configuration manager decompresses new software

functions out of ROM, instantiates them in RAM, and updates the function pointer table to link them dynamically, as the capability of the new software functions are needed."

Surine, 3:27-32.

"Additionally, the system of the present invention maintains the speed and responsiveness of the device while reducing the amount of RAM used in the device. In comparison to prior art embedded system devices, a device in accordance with the present invention either uses less RAM and is thus less expensive, or runs faster using the same amount of RAM."

Surine, 3:39-45.

"In accordance with the present invention, the size of RAM 315 is minimized by running the majority of the software of embedded system 300 (e.g., boot code 401 and operating system code 403) from ROM 310. However, the overall speed of system 300 is largely maintained by running the most frequently-used software (e.g., high-use functions 416) from RAM 315"

Surine, 5:24-30.

"Patch manager 405 subsequently executes. Patch manager 405 includes decompression software which decompresses compressed high-use functions 402 and, as described in greater detail in the discussion of FIG. 5 below, loads the resulting decompressed high-use functions 416 into RAM 315."

Surine, 5:35-40.

"Consequently, since RAM 315 is accessed at least three times faster that ROM 310, system 300 runs much faster than a prior art embedded system running the entirety of its operating system code from ROM. The majority of RAM 315 is occupied by capture buffer 414, display buffer 415, and working memory 420. By using a relatively small portion of RAM 315 to run the uncompressed high-use functions 416, the speed of computer system 300 is greatly increased."

Surine, 5:62-6:3.

"Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for high-use functions 416."

Surine: 6:32-34. *See also* 8:14-17, 8:36-43.

Appendix C21
Invalidity of U.S. Patent 8,880,862 based on Surine

1.4.1 updating the boot data list,	Surine, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

> "The patch manager subsequently updates the function pointer table to incorporate an entry for the high-use functions, thereby linking the high-use functions with the rest of the instantiated functions."

Surine, 3:15-18.

> "Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for high-use functions 416."

Surine: 6:32-34.

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Surine, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Surine, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claim 1.4.1 above.

See also

> "Memory configuration manager then dynamically updates function pointer table 510 with entries reflecting review & play code 751."

Surine, 8:23-25.

> "If more space is required, the software for the newly selected mode is instantiated over (e. g., Written over) the software for the previous, no longer needed, mode. Then, in step 911, memory configuration manager 755 dynamically updates the function pointer table and links the required functions, enabling the new mode."

Surine, 9:53-59.

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Surine, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claim 1.4.1 above.

See also

> "Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315. Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for high-use functions 416."

Surine, 6:29-35.

> "Memory configuration manager then dynamically updates function pointer table 510 with entries reflecting review & play code 751."

Surine, 8:23-25.

> "If software for the new mode overwrites software for the previous mode, Memory configuration manager updates the function pointer table to de-link those functions which have been replaced (e.g., written over)."

Surine, 9: 59-63.

> "In step 1052, any processing for review & play mode is completed, and in step 1053, memory configuration manager 755 updates function pointer table 510 is updated to de-link the review & play code 751."

Surine, 10:44-48.

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Surine, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Surine, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1-1.4.2 above.

See also

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At power-up, boot code stored in the ROM is executed and begins instantiating the initial operating environment of the embedded system. A function pointer table is instantiated in the RAM. The function pointer table has entries, or function pointers, for each instantiated function such that they can each call each other and pass execution. The function pointer table has entries for functions which are instantiated in ROM and entries for functions which are instantiated in RAM."

Surine, 2:66-3:7.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

> "Function pointer table 510 includes a plurality of entries, or function pointers, which, when called by processing unit 305, redirect program

> execution to the memory address of a selected routine. The function pointers of function pointer table allow instantiated functions, whether executing from ROM 310 or RAM 315, to call one another. Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315."

Surine, 6:24-29. *See also* 8:54-56.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Surine, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Surine, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Surine, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Surine, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Surine, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

> "In accordance with the present invention, a set of high-use functions are decompressed out of ROM and instantiated in RAM using a patch manager."

Surine, 3:7-9. *See also* 8:14-17, 8:36-43.

5.6 updating the boot data list;	Surine, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Surine, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1-1.3 above.	

6 (Preamble) A system comprising: a processor;	Surine, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At power-up, boot code stored in the ROM is executed and begins instantiating the initial operating environment of the embedded system. A function pointer table is instantiated in the RAM. The function pointer table has entries, or function pointers, for each instantiated function such that they can each call each other and pass execution. The function pointer table has entries for functions which are instantiated in ROM and entries for functions which are instantiated in RAM."

Surine, 2:66-3:7.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

> "Function pointer table 510 includes a plurality of entries, or function pointers, which, when called by processing unit 305, redirect program execution to the memory address of a selected routine. The function pointers of function pointer table allow instantiated functions, whether

executing from ROM 310 or RAM 315, to call one another. Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315."

Surine, 6:24-29. *See also* 8:54-56.

6.1 a memory; and	Surine, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Surine, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At power-up, boot code stored in the ROM is executed and begins instantiating the initial operating environment of the embedded system. A function pointer table is instantiated in the RAM. The function pointer table has entries, or function pointers, for each instantiated function such that they can each call each other and pass execution. The function pointer table has entries for functions which are instantiated in ROM and entries for functions which are instantiated in RAM."

Surine, 2:66-3:7.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

> "Function pointer table 510 includes a plurality of entries, or function pointers, which, when called by processing unit 305, redirect program execution to the memory address of a selected routine. The function pointers of function pointer table allow instantiated functions, whether executing from ROM 310 or RAM 315, to call one another. Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315."

Surine, 6:24-29. *See also* 8:54-56.

6.3 wherein the processor is configured:	Surine, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At power-up, boot code stored in the ROM is executed and begins instantiating the initial operating environment of the embedded system. A function pointer table is instantiated in the RAM. The function pointer table has entries, or function pointers, for each instantiated function such that they can each call each other and pass execution. The function pointer table has entries for functions which are instantiated in ROM and entries for functions which are instantiated in RAM."

Surine, 2:66-3:7.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

> "Function pointer table 510 includes a plurality of entries, or function pointers, which, when called by processing unit 305, redirect program execution to the memory address of a selected routine. The function pointers of function pointer table allow instantiated functions, whether executing from ROM 310 or RAM 315, to call one another. Initially, the function pointers are initialized to addresses in ROM 310, but after

copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315."

Surine, 6:24-29. *See also* 8:54-56.

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Surine, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Surine, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.2 above.	

Surine Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Surine, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.3 above.	

Surine Claim 6.6
"to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and"

6.7 to update the boot data list.	Surine, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.4.1 above.	

Surine Claim 6.7
"to update the boot data list."

Appendix C21
Invalidity of U.S. Patent 8,880,862 based on Surine

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Surine, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1 (Preamble) above.	

Surine **Claim 8 (Preamble)**
"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Surine, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Surine, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Surine, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Surine, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Surine Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Surine, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1.3 and 1.4.2 above.

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

> "In accordance with the present invention, a set of high-use functions are decompressed out of ROM and instantiated in RAM using a patch manager."

Surine, 3:7-9. *See also* 8:14-17, 8:36-43.

8.6 updating the boot data list,	Surine, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.4.1 above.	

Surine Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Surine, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Surine, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Surine, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Surine, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Surine, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 9.1 above.	

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Surine, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1 (Preamble) above.	

Surine **Claim 11 (Preamble)**
"A method for providing accelerated loading of an operating system in a computer system, comprising"

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Surine, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claim 1.1 above.

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At power-up, boot code stored in the ROM is executed and begins instantiating the initial operating environment of the embedded system. A function pointer table is instantiated in the RAM. The function pointer table has entries, or function pointers, for each instantiated function such that they can each call each other and pass execution. The function pointer table has entries for functions which are instantiated in ROM and entries for functions which are instantiated in RAM."

Surine, 2:66-3:7.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

Surine Claim 11.1
"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

> "Function pointer table 510 includes a plurality of entries, or function pointers, which, when called by processing unit 305, redirect program execution to the memory address of a selected routine. The function pointers of function pointer table allow instantiated functions, whether executing from ROM 310 or RAM 315, to call one another. Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315."

Surine, 6:24-29. *See also* 8:54-56.

11.2 accessing the loaded boot data in compressed form from the memory;	Surine, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Surine, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Surine, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

> "In accordance with the present invention, a set of high-use functions are decompressed out of ROM and instantiated in RAM using a patch manager."

Surine, 3:7-9. *See also* 8:14-17, 8:36-43.

11.5 updating the boot data list.	Surine, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.4.1 above.	

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Surine, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Surine, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Surine, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Surine, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Surine, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.4.1 above.	

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Surine, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Surine, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1.1 and 1.2 above.

> "At power-up, boot code stored in the non-volatile memory is executed and begins instantiating the initial operating environment of the embedded computer system. A function pointer table is instantiated in the volatile memory, wherein the function pointer table includes a plurality of entries for a corresponding plurality of instantiated functions, wherein at least one entry is for operating system code stored in the non-volatile memory."

Surine, Abstract.

> "At power-up, boot code stored in the ROM is executed and begins instantiating the initial operating environment of the embedded system. A function pointer table is instantiated in the RAM. The function pointer table has entries, or function pointers, for each instantiated function such that they can each call each other and pass execution. The function pointer table has entries for functions which are instantiated in ROM and entries for functions which are instantiated in RAM."

Surine, 2:66-3:7.

> "Referring now to FIG. 4, a memory diagram depicting the contents of ROM 310 and RAM 315 is shown. As shown in FIG. 4, ROM 310 stores software including boot code 401, compressed high-use functions 402, and operating system code 403."

Surine, 5:15-19.

> "Function pointer table 510 includes a plurality of entries, or function

> pointers, which, when called by processing unit 305, redirect program execution to the memory address of a selected routine. The function pointers of function pointer table allow instantiated functions, whether executing from ROM 310 or RAM 315, to call one another. Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use function pointers are updated to addresses in RAM 315."

Surine, 6:24-29. *See also* 8:54-56.

14.2 loading the boot data into a memory; and	Surine, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Surine, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1.2 and 1.3 above.

> "At least one high-use function is decompressed out of the non-volatile memory and instantiated in volatile memory. The function pointer table is updated using a patch manager to incorporate an entry for the high-use function(s). The operating system code is executed from the non-volatile memory while the high-use function is executed from the volatile memory."

Surine, Abstract.

> "At boot time, or power-up, boot code 202 executes, decompresses compressed code 201 into camera system code 203 and loads camera system code 203 into RAM 102."

Surine, 2:4-7, Fig. 2.

> "Consequently, these digital cameras and other performance-oriented types of embedded system consumer electronic devices transfer a compressed image of their system code from a non-volatile ROM to a faster RAM at power up. The system code then executes from RAM."

Surine, 2:21-25.

> "Additionally, the system of the present invention maintains the speed and responsiveness of the device while reducing the amount of RAM used in the device. In comparison to prior art embedded system devices, a device in accordance with the present invention either uses less RAM and is thus less expensive, or runs faster using the same amount of RAM."

Surine, 2:49-55.

> "In accordance with the present invention, a set of high-use functions are decompressed out of ROM and instantiated in RAM using a patch manager."

Surine, 3:7-9.

> "In this embodiment, in addition to instantiating certain high-use functions, a memory configuration manager decompresses new software functions out of ROM, instantiates them in RAM, and updates the function pointer table to link them dynamically, as the capability of the new software functions are needed."

Surine, 3:27-32.

> "Additionally, the system of the present invention maintains the speed and responsiveness of the device while reducing the amount of RAM used in the device. In comparison to prior art embedded system devices, a device in accordance with the present invention either uses less RAM and is thus less expensive, or runs faster using the same amount of RAM."

Surine, 3:39-45.

> "In accordance with the present invention, the size of RAM 315 is minimized by running the majority of the software of embedded system 300 (e.g., boot code 401 and operating system code 403) from ROM 310. However, the overall speed of system 300 is largely maintained by running the most frequently-used software (e.g., high-use functions 416) from RAM 315"

Surine, 5:24-30.

> "Patch manager 405 subsequently executes. Patch manager 405 includes decompression software which decompresses compressed high-use functions 402 and, as described in greater detail in the discussion of FIG.

> 5 below, loads the resulting decompressed high-use functions 416 into RAM 315."

Surine, 5:35-40.

> "Consequently, since RAM 315 is accessed at least three times faster that ROM 310, system 300 runs much faster than a prior art embedded system running the entirety of its operating system code from ROM. The majority of RAM 315 is occupied by capture buffer 414, display buffer 415, and working memory 420. By using a relatively small portion of RAM 315 to run the uncompressed high-use functions 416, the speed of computer system 300 is greatly increased."

Surine, 5:62-6:3.

> "Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for high-use functions 416."

Surine, 6:32-34. *See also* Surine, 7:5-43, 8:14-17, 8:36-43.

14.4 updating the boot data list.	Surine, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Surine, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "Typically, as shown in FIG. 2, the camera system code 203 (e.g., operating system software and its associated data structures, resources, etc.) is stored as compressed code 201 in non-volatile ROM 104."

Surine, 2:1-4.

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Surine, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Surine, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.1 and 1.2 above.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Surine, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.1 and 1.2 above.	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.1 and 1.2 above.	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Surine, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.1 and 1.2 above.	

Surine **Claim 39**

"The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory."

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

> "Patch manager 405 includes decompression software which decompresses compressed high-use functions 402 and, as described in greater detail in the discussion of FIG. 5 below, loads the resulting decompressed high-use functions 416 into RAM 315. This is shown by arrow 410."

Surine, 5:36-41.

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Surine, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

Surine **Claim 59**

"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Surine, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Surine, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Surine, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Surine, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 32, 33 and 49 above.	

Surine **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Surine, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 32, 33 and 49 above.	

Surine **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 27 above.	

Surine **Claim 87**
"The method of claim 13, wherein the memory comprises: a physical memory."

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 15, 17 and 24 above.	

Surine **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 31 above.	

Surine **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 32, 33 and 49 above.	

Surine **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Surine, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claim 32, 33, and 49 above.	

Surine **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Surine, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Surine, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Surine, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Surine discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

> "Initially, the function pointers are initialized to addresses in ROM 310, but after copying to RAM 315, the high-use pointers are updated to addresses in RAM 315. Patch manager 405 then loads the decompressed high-use functions 416 into RAM 315 and updates function pointer table 510 with entries for high-use functions 416."

Surine, 6:29-35.

> "Memory configuration manager then dynamically updates function pointer table 510 with entries reflecting review & play code 751."

Surine, 8:23-25.

> "If software for the new mode overwrites software for the previous mode, Memory configuration manager updates the function pointer table to de-link those functions which have been replaced (e.g., written over)."

Surine, 9: 59-63.

> In step 1052, any processing for review & play mode is completed, and in step 1053, memory configuration manager 755 updates function pointer table 510 is updated to de-link the review & play code 751."

Surine, 10:44-48.

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Surine, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Surine, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Surine, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Surine discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

U.S. Patent No. 6,370,614 Teoman ("Teoman") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Teoman, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

> "In one embodiment, the user-cache is non-volatile and large enough to hold several hundred megabytes worth of data. Consequently, by configuring the user-cache to store program code and configuration information used for computer system startup, the user-cache may be used as a boot source to provide much faster system boot up than can be achieved by booting out of mass storage media such as disk or tape."

Teoman, 3:10-17. *See also* Teoman, 3:18-45.

> "According to one embodiment, the user cache 25 includes boot firmware 66 for storing program code that is used to support operation of the user cache at system startup."

Teoman, 7:50-52.

> "The user interface 123 also permits the user to configure the computer system to use the user cache as a boot device, meaning that the user cache will be treated as a source of boot software at system startup. In one embodiment, if the user enables the "Use as a boot device" option, the user cache manager process prompts the user to specify the logical drive that is ordinarily the source of boot software. The user cache manager software responds by interacting with the BIOS configuration to treat the user-cache as the user-specified logical drive and thereby to boot out of the user-cache at system startup."

Teoman, 13:29-39.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the

entire boot sequence."

Teoman, 13:46-51.

> "In yet another embodiment for using the user cache to support system startup, before the OS boot sequence is begun, boot program code in the user cache firmware is executed to notify the system BIOS that the user cache is a bootable mass storage device. Also, when executed, the boot program code loads software into system memory for operating the user cache as a bootable mass storage device. In this embodiment, the user selects the user cache as the boot device in the system BIOS settings. Then, during the boot sequence, the operating system accesses boot up files in the user cache."

Teoman, 13:52-62.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Teoman, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

> "When an I/O request 40 to access the mass storage 46 is issued (e.g., a file read or write request issued in the course of executing an application program), the I/O request 40 is first applied to the OS cache maintained in system memory 16. If the I/O request 40 hits the OS cache (i.e., the data sought to be accessed is cached in the OS cache), the access is performed in the OS cache. If the I/O request 40 is a read request, the data is returned to the requestor. If the I/O request 40 does not hit the OS cache, the I/O request 40 is redirected from the mass storage 46 to the user cache 25 by software mechanisms described below. If the I/O request 40 hits the user cache 25, the access is performed in the user cache 25 without having to access the mass storage 46, thereby substantially reducing the overall access time. Also, because the user cache 25 is significantly larger than the OS cache and supports data preloading (discussed below), much higher hit rates can be achieved in the user cache than in the OS cache."

Teoman, 4:57-5:7. *See also* Teoman, 5:16-20, 5:48-51.

> "According to one embodiment, the user cache 25 includes boot firmware 66 for storing program code that is used to support operation of the user cache at system startup."

Teoman, 7:50-52.

> "In general, there are two types of storage operations that take place in the user cache: preloading and responsive caching. Responsive caching refers to the storage of data in the user cache in response to I/O requests from application processes. In a preload operation, by contrast, data is retrieved from mass storage and stored in the user cache before being

requested for use in an application process."

Teoman, 9:24-31. *See also* Teoman, 9:31-44, 10:40-60.

> "The user interface 123 also permits the user to configure the computer system to use the user cache as a boot device, meaning that the user cache will be treated as a source of boot software at system startup. In one embodiment, if the user enables the "Use as a boot device" option, the user cache manager process prompts the user to specify the logical drive that is ordinarily the source of boot software. The user cache manager software responds by interacting with the BIOS configuration to treat the user-cache as the user-specified logical drive and thereby to boot out of the user-cache at system startup."

Teoman, 13:29-39.

> "In another embodiment, the user cache driver is loaded very early in the OS boot sequence and is operable for most of the sequence. Most operating systems support loading device drivers early in the boot sequence so that, in most cases, this mode of operation requires no special hardware or software support."

Teoman, 13:40-45.

> "In yet another embodiment for using the user cache to support system startup, before the OS boot sequence is begun, boot program code in the user cache firmware is executed to notify the system BIOS that the user cache is a bootable mass storage device. Also, when executed, the boot program code loads software into system memory for operating the user cache as a bootable mass storage device. In this embodiment, the user selects the user cache as the boot device in the system BIOS settings. Then, during the boot sequence, the operating system accesses boot up files in the user cache."

Teoman, 13:52-62.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Teoman, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

> "When an I/O request 40 to access the mass storage 46 is issued (e.g., a file read or write request issued in the course of executing an application program), the I/O request 40 is first applied to the OS cache maintained in system memory 16. If the I/O request 40 hits the OS cache (i.e., the data sought to be accessed is cached in the OS cache), the access is performed in the OS cache. If the I/O request 40 is a read request, the data is returned to the requestor. If the I/O request 40 does not hit the OS cache, the I/O request 40 is redirected from the mass storage 46 to the user cache 25 by software mechanisms described below. If the I/O request 40 hits the user cache 25, the access is performed in the user cache 25 without having to access the mass storage 46, thereby substantially reducing the overall access time. Also, because the user cache 25 is significantly larger than the OS cache and supports data preloading (discussed below), much higher hit rates can be achieved in the user cache than in the OS cache."

Teoman, 4:57-5:7. *See also* Teoman, 5:16-20, 5:48-51.

> "After a user has specified a set of commanded preload parameters, the user cache manager 90 responds by generating I/O requests to retrieve the data identified by the preload parameters from mass storage 46"

Teoman, 9:56-59.

> "In yet another embodiment for using the user cache to support system startup, before the OS boot sequence is begun, boot program code in the user cache firmware is executed to notify the system BIOS that the user cache is a bootable mass storage device. Also, when executed, the boot program code loads software into system memory for operating the user cache as a bootable mass storage device. In this embodiment, the user selects the user cache as the boot device in the system BIOS settings.

Then, during the boot sequence, the operating system accesses boot up files in the user cache." Teoman, 13:52-62.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Teoman, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.1 updating the boot data list,	Teoman, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

> "The requested blocks are then passed back to the user cache driver 45 which writes them to the user cache 25 and updates the user cache directory."

Teoman, 10:20-23.

> "After loading data into the user cache 25, the user cache driver 45 updates the user cache directory to indicate the newly cached blocks."

Teoman, 12:50-52.

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Teoman, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Teoman, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Teoman, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Teoman, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Teoman, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claims 1-1.4.2 above.

See also ***Add disclosure from '608 Patent Claims 1.1 and 1.2.***

> "According to one embodiment, the user cache 25 includes boot firmware 66 for storing program code that is used to support operation of the user cache at system startup."

Teoman, 7:50-52.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

> "In yet another embodiment for using the user cache to support system startup, before the OS boot sequence is begun, boot program code in the user cache firmware is executed to notify the system BIOS that the user cache is a bootable mass storage device. Also, when executed, the boot program code loads software into system memory for operating the user cache as a bootable mass storage device. In this embodiment, the user selects the user cache as the boot device in the system BIOS settings. Then, during the boot sequence, the operating system accesses boot up files in the user cache."

Teoman, 13:52-62.

> "The program code is returned to the bus interface circuitry 61 which outputs it to the expansion bus where it is routed to its ultimate destination

(e.g., the processor used to execute system boot code)."

Teoman, 7:61-64.

> "Still referring to FIG. 1, the processing unit 12 includes one or more processors that fetch program code from system memory 16 and execute the code to operate on data and to read and Write data to the system memory 16 and to the I/O devices on the expansion bus 18."

Teoman, 4:8-12.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Teoman, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claim 1.1 above.

See also

> "In a preload operation, by contrast, data is retrieved from mass storage and stored in the user cache before being requested for use in an application process."

Teoman, 9:28-31.

> "The user cache driver 45 determines whether the requested blocks are already stored in the user cache 25."

Teoman, 10:10-12.

> "Consequently, by configuring the user-cache to store program code and configuration information used for computer system startup, the user-cache may be used as a boot source to provide much faster system boot up than can be achieved by booting out of mass storage media such as disk or tape."

Teoman, 3:12-17.

> "According to one embodiment, the user cache 25 includes boot firmware 66 for storing program code that is used to support operation of the user cache at system startup."

Teoman, 7:50-53.

5.2 loading the stored compressed boot data from the first memory;	Teoman, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Teoman, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Teoman, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Teoman, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

5.6 updating the boot data list;	Teoman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Teoman, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

6 (Preamble) A system comprising: a processor;	Teoman, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See

> "The program code is returned to the bus interface circuitry 61 which outputs it to the expansion bus where it is routed to its ultimate destination (e.g., the processor used to execute system boot code)."

Teoman, 7:61-64.

> "Still referring to FIG. 1, the processing unit 12 includes one or more processors that fetch program code from system memory 16 and execute the code to operate on data and to read and Write data to the system memory 16 and to the I/O devices on the expansion bus 18."

Teoman, 4:8-12.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

6.1 a memory; and	Teoman, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Teoman, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claims 1.1 and 1.2 above.

See

> "The program code is returned to the bus interface circuitry 61 which outputs it to the expansion bus where it is routed to its ultimate destination (e.g., the processor used to execute system boot code)."

Teoman, 7:61-64.

> "Still referring to FIG. 1, the processing unit 12 includes one or more processors that fetch program code from system memory 16 and execute the code to operate on data and to read and Write data to the system memory 16 and to the I/O devices on the expansion bus 18."

Teoman, 4:8-12.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

6.3 wherein the processor is configured:	Teoman, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See

> "The program code is returned to the bus interface circuitry 61 which outputs it to the expansion bus where it is routed to its ultimate destination (e.g., the processor used to execute system boot code)."

Teoman, 7:61-64.

> "Still referring to FIG. 1, the processing unit 12 includes one or more processors that fetch program code from system memory 16 and execute the code to operate on data and to read and Write data to the system memory 16 and to the I/O devices on the expansion bus 18."

Teoman, 4:8-12.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Teoman, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Teoman, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.2 above.	

Teoman Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Teoman, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Teoman, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

Teoman Claim 6.7
"to update the boot data list."

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Teoman, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Teoman, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Teoman, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Teoman, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Teoman, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Teoman Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Teoman, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.6 updating the boot data list,	Teoman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

Teoman Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Teoman, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Teoman, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 8.1 above.	

Teoman Claim 9.2
"storing the additional portion of the operating system in the first memory"

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Teoman, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Teoman, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 9.1 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Teoman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1 (Preamble) above	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Teoman, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claim 1.1 above.

See also

> "The program code is returned to the bus interface circuitry 61 which outputs it to the expansion bus where it is routed to its ultimate destination (e.g., the processor used to execute system boot code)."

Teoman, 7:61-64.

> "Still referring to FIG. 1, the processing unit 12 includes one or more processors that fetch program code from system memory 16 and execute the code to operate on data and to read and Write data to the system memory 16 and to the I/O devices on the expansion bus 18."

Teoman, 4:8-12.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

11.2 accessing the loaded boot data in compressed form from the memory;	Teoman, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Teoman, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Teoman, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

11.5 updating the boot data list.	Teoman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Teoman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Teoman, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Teoman, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Teoman, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Teoman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Teoman, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Teoman, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> "According to one embodiment, the user cache 25 includes boot firmware 66 for storing program code that is used to support operation of the user cache at system startup."

Teoman, 7:50-52.

> "In another embodiment, boot program code in the user cache firmware is executed to redirect boot time I/O requests before the operating system is loaded. In this way, boot time I/O requests are redirected to the user cache, making the user cache operable as a source of boot files during the entire boot sequence."

Teoman, 13:46-51.

> "In yet another embodiment for using the user cache to support system startup, before the OS boot sequence is begun, boot program code in the user cache firmware is executed to notify the system BIOS that the user cache is a bootable mass storage device. Also, when executed, the boot program code loads software into system memory for operating the user cache as a bootable mass storage device. In this embodiment, the user selects the user cache as the boot device in the system BIOS settings. Then, during the boot sequence, the operating system accesses boot up files in the user cache."

Teoman, 13:52-62.

14.2 loading the boot data into a memory; and	Teoman, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Teoman, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

> "When an I/O request 40 to access the mass storage 46 is issued (e.g., a file read or write request issued in the course of executing an application program), the I/O request 40 is first applied to the OS cache maintained in system memory 16. If the I/O request 40 hits the OS cache (i.e., the data sought to be accessed is cached in the OS cache), the access is performed in the OS cache. If the I/O request 40 is a read request, the data is returned to the requestor. If the I/O request 40 does not hit the OS cache, the I/O request 40 is redirected from the mass storage 46 to the user cache 25 by software mechanisms described below. If the I/O request 40 hits the user cache 25, the access is performed in the user cache 25 without having to access the mass storage 46, thereby substantially reducing the overall access time. Also, because the user cache 25 is significantly larger than the OS cache and supports data preloading (discussed below), much higher hit rates can be achieved in the user cache than in the OS cache."

Teoman, 4:57-5:7. *See also* Teoman, 5:16-20, 5:48-51.

> "After a user has specified a set of commanded preload parameters, the

> user cache manager 90 responds by generating I/O requests to retrieve the data identified by the preload parameters from mass storage 46"

Teoman, 9:56-59.

> "In yet another embodiment for using the user cache to support system startup, before the OS boot sequence is begun, boot program code in the user cache firmware is executed to notify the system BIOS that the user cache is a bootable mass storage device. Also, when executed, the boot program code loads software into system memory for operating the user cache as a bootable mass storage device. In this embodiment, the user selects the user cache as the boot device in the system BIOS settings. Then, during the boot sequence, the operating system accesses boot up files in the user cache."

Teoman, 13:52-62.

14.4 updating the boot data list.	Teoman, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Teoman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

See Claims 1 (Preamble) and 1.1 above.

See also

> "Consequently, by configuring the user-cache to store program code and configuration information used for computer system startup, the user-cache may be used as a boot source to provide much faster system boot up than can be achieved by booting out of mass storage media such as disk or tape."

Teoman, 3:12-17.

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Teoman discloses this limitation:

> "For example, a user may specify to cache the contents of a folder or directory (e. g., the system directory that includes much of the operating system and system configuration files), files having a particular file type identifier (e.g., files with a given filename extension), files having particular file names, files accessed by a particular user and so forth."

Teoman, 3:20-26.

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Teoman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Teoman, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Teoman, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Teoman, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.1 and 1.2 above.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 32 and 33 above.	

Teoman **Claim 45**

"The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Teoman, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

Teoman **Claim 59**

"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Teoman, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 32, 33 and 49 above.	

Teoman **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Teoman, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

Appendix C22
Invalidity of U.S. Patent 8,880,862 based on Teoman

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 31 above.	

Teoman **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 32, 33 and 49 above.	

Teoman **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Teoman, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claim 32, 33, and 49 above.	

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Teoman, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Teoman, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Teoman, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Teoman, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Teoman, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Teoman discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Teoman **Claim 109**
"The method of claim 108, further comprising: storing the compressed additional boot data."

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

German Patent No. DE19721786 to Michael Vers ("Vers") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Vers, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> From the prior art it is known that, in a data processing apparatus, both the system and the application software is stored in a nonvolatile memory. The system software can run on the one hand directly from the non-volatile memory or on the other hand copied from the nonvolatile memory to the volatile memory and run from this. The start of the system software from non-volatile memory has the disadvantage that either a quicker and thereby more expensive non-volatile memory must be used, or that the system performance is degraded by inserting wait states. At the start of the system software from the volatile memory, a nonvolatile memory full size of the system software must be available, but which is no longer needed after the copying of the system software.

Vers, 1

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory

or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Vers, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after

decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the

operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Vers, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the

beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a

checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Vers, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header

in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element,

Vers Claim 1.3

"decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form."

decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

1.4.1 updating the boot data list,	Vers, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

> In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable

code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

If a user software from the volatile memory operates 18 may at any time provide the required memory locations for storing online modifications available memory management. For an online modified user program is again even with reclosing of the automation device in the previously amended version is available, it must be stored before switching off the automation device in the non-volatile memory, so that all previous changes and variable initialization are stored.

Vers, 4

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Vers, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the

beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a

checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Vers, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Vers, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Vers, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Vers, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1-1.4.2 above.

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

> In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can

also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.
Vers, 3

Vers **Claim 5 (Preamble)**

"A method for booting a computer system, the method comprising:"

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Vers, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Vers, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Vers, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Vers, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Vers, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header

in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element,

decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

5.6 updating the boot data list;	Vers, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Vers, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

6 (Preamble) A system comprising: a processor;	Vers, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

> In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is

shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

6.1 a memory; and	Vers, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Vers, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.1, 1.2, and Claim 6 (Preamble) above.

6.3 wherein the processor is configured:	Vers, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 6 (Preamble) above	

Vers Claim 6.3
"wherein the processor is configured"

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Vers, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Vers, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.2 above.	

Vers Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Vers, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Vers, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

Vers Claim 6.7
"to update the boot data list."

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Vers, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1 (Preamble) above.	

Vers **Claim 8 (Preamble)**

"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Vers, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claim 6.2 above.

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Vers, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Vers, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Vers, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Vers, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files

in the volatile memories.

Vers, 2

Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

Vers Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

Vers Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

8.6 updating the boot data list,	Vers, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Vers, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Vers, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claim 1.1 above.

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files

in the volatile memories.

Vers, 2

Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

Vers Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

Vers Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

9.2 storing the additional portion of the operating system in the first memory, and	Vers, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Vers, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Vers, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 9.1 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Vers, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Vers, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claim 1.1 and 6 (Preamble) above.

11.2 accessing the loaded boot data in compressed form from the memory;	Vers, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Vers, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Vers, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC

checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the

Vers Claim 11.4

"utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

11.5 updating the boot data list.	Vers, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Vers, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Vers, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Vers, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Vers, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Vers, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Vers, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Vers, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

> In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used,

which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

14.2 loading the boot data into a memory; and	Vers, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Vers, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there

and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in

Vers **Claim 14.3**

"servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

14.4 updating the boot data list.	Vers, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Vers, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1 (Preamble), 1.1, and 14 above.

See also

> Preferably, in the header in the nonvolatile memory a destination address in the volatile memory is stored, is where the system software such as operating system copied or stored decompressed.

Vers, 2

> In FIG. 2, the memory map of the nonvolatile memory 16 is shown. Invention is provided in accordance with that system software, such as operating system is stored in a top portion 22. Following the operating system followed by a free space 24, the memory area 26 joins with a reset initialization and a decompression Algorithm mechanism as executable code.

Vers, 3

> Fig. 3 shows the structure of a file system first header 28 in a region at the beginning of the establishment in which not the actual data but important structural information is saved to the file. So the CRC checksum (Cyclical Redundancy Check) is stored in a storage area 30 through the first four bytes of the first header 28. In a storage area 32 of the header, the length of the operating system is stored in non-volatile memory without the header itself. In a further memory area 34, the flag element is stored. It has the value "ONE" when the operating system is stored in compressed form and the value "zero" when the operating system in a transparent, i.e, unmodified form is saved. By a further storage area 36, the destination address is specified in the volatile memory to which copies the operating system is to be stored or

decompressed. Finally, the filing of the operating system begins in compressed or transparent form. Vers, 3

Vers **Claim 15**

"the method of claim 14, wherein the boot data comprises: a program code associated with the operating system"

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1 (Preamble), 1.1, and 14 above.

See also

> Preferably, in the header in the nonvolatile memory a destination address in the volatile memory is stored, is where the system software such as operating system copied or stored decompressed.

Vers, 2

> In FIG. 2, the memory map of the nonvolatile memory 16 is shown. Invention is provided in accordance with that system software, such as operating system is stored in a top portion 22. Following the operating system followed by a free space 24, the memory area 26 joins with a reset initialization and a decompression Algorithm mechanism as executable code.

Vers, 3

> Fig. 3 shows the structure of a file system first header 28 in a region at the beginning of the establishment in which not the actual data but important structural information is saved to the file. So the CRC checksum (Cyclical Redundancy Check) is stored in a storage area 30 through the first four bytes of the first header 28. In a storage area 32 of the header, the length of the operating system is stored in non-volatile memory without the header itself. In a further memory area 34, the flag element is stored. It has the value "ONE" when the operating system is stored in compressed form and the value "zero" when the operating system in a transparent, i.e, unmodified form is saved. By a further storage area 36, the destination address is specified in the volatile memory to which copies the operating system is to be stored or

decompressed. Finally, the filing of the operating system begins in compressed or transparent form. Vers, 3

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Vers, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claim 15 above.

See also

> The invention relates to a method for operating a data processing device, in particular automation device, with volatile and non-volatile memories, said system and / or application software is copied from non-volatile to a volatile memory.

Vers, 1

> From the prior art it is known that in a data processing device, both the system and the application software is stored in a nonvolatile memory. The system software can run directly on the one hand from the non-volatile memory or on the other hand copies of the non-volatile memory to the volatile memory and run from this. The start of the system software from non-volatile memory has the disadvantage that either a quicker and thereby more expensive non-volatile memory must be used, or that the system power is discontinued by inserting wait states ago. At the start of the system software from the volatile memory non-volatile memory must be in full size of the system software, however, is no longer needed after copying the system software.

Vers, 1

> The problem is inventively achieved in that the system software stored compression form in the in the non-volatile memory after reset depending on a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and

started there and / or that the application software associated, are compressed files containing changes the user software or by appointment by the programmer before power failure and stored in non-volatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers 2

In particular, programmable logic controllers or PLCs can be stored in non-volatile memory in compressed form and application software.

Vers, 3

Vers **Claim 17**

"The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Vers, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.1 and 1.2 above	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Vers, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.1 and 1.2 above

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> From the prior art it is known that in a data processing device, both the system and the application software is stored in a nonvolatile memory. The system software can run directly on the one hand from the non-volatile memory or on the other hand copies of the non-volatile memory to the volatile memory and run from this. The start of the system software from non-volatile memory has the disadvantage that either a quicker and thereby more expensive non-volatile memory must be used, or that the system power is discontinued by inserting wait states ago. At the start of the system software from the volatile memory non-volatile memory must be in full size of the system software, however, is no longer needed after copying the system software.
>
> It is also known that application software can run in non-volatile memory or can be loaded from a programming unit via a controlled by the system software programming in the volatile memory directly. While executing the application software from non-volatile memory often occurs a

problem that the application software on-line changes are difficult to achieve or only with a considerable increase in the cycle time. In a sequence of user software from the volatile memory is also known that a memory management can always provide the necessary resources for online changes available. If the modified data upon reconnection of the data processing apparatus provided in its amended form available to use must be assured that they were previously stored in non-volatile memory so that the previous online changes are saved. However, this would require an unnecessarily large non-volatile memory which would not be used during the lifetime of the data processing device.

Vers, 1-2

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.1 and 1.2 above	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claim 1.1 above

See also

> A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Vers, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.1 and 32 above.

See also

> A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.1 and 1.2 above	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 32 and 33 above.	

Vers **Claim 45**

"The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> The invention is based on the problem to provide a method for operating a data processing device in such a way that non-volatile at least consistent performance storage elements with a minimum memory capacity can be used. the problem is inventively solved in that the system software stored in compressed form in the non-volatile memory and to reset in response to a flag element either is decompressed and stored in the volatile memory or copied without decompression in the volatile memory and started there and / or that the application software associated, are compressed changes of application software files containing or by instructing from the programming device before power failure and stored in the nonvolatile memory and decompressed when power is restored to regenerate the files in the volatile memories.

Vers, 2

> Preferably, a destination address in the volatile memory is in the header in the non-volatile memory, where the system software such as operating system copied or stored decompressed. It is also provided that after

decompression or copying of the system software calculated CRC checksum compared with a in a second header in the region of the beginning of the file system software in the volatile memory, and if they match a start routine is started.

Vers, 2

A particularly preferred compressing algorithm is the LZW algorithm used.

Vers, 2

In Fig. 1 the basic construction of a programmable controller is shown with a bus 12 to which a microprocessor 14, non-volatile and volatile memories 16, 18 and one or more peripheral devices 20 are connected. as non-volatile memory 16 preferably flash memory modules are used, which can be written and read, and in contrast to the volatile memory 18 may be retained their content when they are not supplied with operating voltage. Other non-volatile memory such as EPROM and EEPROM can also be used. Preferably as a volatile memory SRAM and / or DRAM's are used. In Fig. 2, the memory map of the nonvolatile memory 16 is shown. According to the invention, that the system software as the operating system stored in an initial region 22nd Following the operating system is followed by a free space 24, which is a storage area 26 connects with a reset initialization and a decompression algorithm as executable code. Especially with programmable logic controllers or PLCs can also use the software in the nonvolatile memory is stored in compressed form be.

Vers, 3

By a further storage area 36, the destination address is specified in the volatile memory to which the operating system copies or store decompressed. Finally, the tray of the operating system begins in compressed or transparent form.

Vers, 3

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the

Vers **Claim 50**

"The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form."

operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

Vers **Claim 50**

"The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form."

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Vers, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Vers, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Vers, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Vers, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Vers, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 27 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 31 above.	

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 32, 33 and 49 above.	

Vers **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Vers, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 32, 33 and 49 above.	

Vers **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

Vers **Claim 83**
"The method of claim 13, wherein the boot data in the compressed form represents a plurality of files."

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 27 above.	

Vers **Claim 87**
"The method of claim 13, wherein the memory comprises: a physical memory."

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 16 and 23 above.	

Vers **Claim 88**
"The method of claim 13, wherein the operating system comprises: a plurality of files."

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 31 above.	

Vers **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 32, 33 and 49 above.	

Vers **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

Appendix C23
Invalidity of U.S. Patent 8,880,862 based on Vers

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Vers, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claim 32, 33, and 49 above.	

Vers **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Vers, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Vers, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Vers, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Vers, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Vers discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

See also

> The method involves placing system software into a non-volatile memory (16) in compressed form. After a reset, the software is copied into the volatile memory (18) and is started there. This is done either after decompression or without decompression, depending on the characteristic element. Data corresponding to changes in the user software prior to voltage drop-out are compressed and stored in the non-volatile memory and after reset are decompressed for regeneration of the data in volatile memory.

Vers, Abstract

> From the prior art it is known that in a data processing device, both the system and the application software is stored in a nonvolatile memory. The system software can run directly on the one hand from the non-volatile memory or on the other hand copies of the non-volatile memory to the volatile memory and run from this. The start of the system software from non-volatile memory has the disadvantage that either a quicker and thereby more expensive non-volatile memory must be used, or that the system power is discontinued by inserting wait states ago. At the start of the system software from the volatile memory non-volatile memory must be in full size of the system software, however, is no longer needed after copying the system software.
>
> It is also known that application software can run in non-volatile memory or can be loaded from a programming unit via a controlled by the system software programming in the volatile memory directly. While executing

the application software from non-volatile memory often occurs a problem that the application software on-line changes are difficult to achieve or only with a considerable increase in the cycle time. In a sequence of user software from the volatile memory is also known that a memory management can always provide the necessary resources for online changes available. If the modified data upon reconnection of the data processing apparatus provided in its amended form available to use must be assured that they were previously stored in non-volatile memory so that the previous online changes are saved. However, this would require an unnecessarily large non-volatile memory which would not be used during the lifetime of the data processing device.

Vers, 1-2

To inventively to guarantee the use of non-volatile memories of a minimum memory size is provided that the stored in compressed form in the non-volatile memory operating system during initialization in the volatile memory is loaded. For this purpose it is provided that the non-volatile memory compresses stored operating system, ie, when the flag element the value "ONE" has a function of the marker element, decompressed by a stored in nonvolatile memory uncompressed routine and is stored in the nonvolatile memory. Before the decompression of the operating system a CRC checksum is calculated and compared with the likewise in the header 28 stored in the storage area 30 by means of a checksum stored in the memory area 32 of the first header 28 length information. Is the calculated checksum matches the checksum stored, the process continues with the decompression. Here, the operating system is decompressed without the header 28 and stored decompressed to the stored in the memory area 36 of the header 28 destination address in the volatile memory. Does the marker element the value "zero" on, is the operating system directly, ie without decompression and without header in the volatile memory copied. After decompress or copying of the operating system as previously described a CRC checksum is determined and compared with a in a second header 40 in volatile memory which is present in decompressed form. If the two checksums match, a startup routine of the operating system is started, whose address is also stored in the header 40 in the storage area 46th

Vers, 3-4

Vers **Claim 107**

"The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Vers, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

Vers **Claim 108**
"The method of claim 2, further comprising: compressing at least a portion of the additional boot data."

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Vers, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Vers discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Vers **Claim 109**
"The method of claim 108, further comprising: storing the compressed additional boot data."

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

U.S. Patent No. 6,317,818 to Zwiegincew ("Zwiegincew") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

In addition, Apple incorporates by reference, as if set forth fully herein, all arguments related to Zwiegincew in pending inter partes review petitions IPR2016-01737, IPR2016-01738, and IPR2016-01739.

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Zwiegincew, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

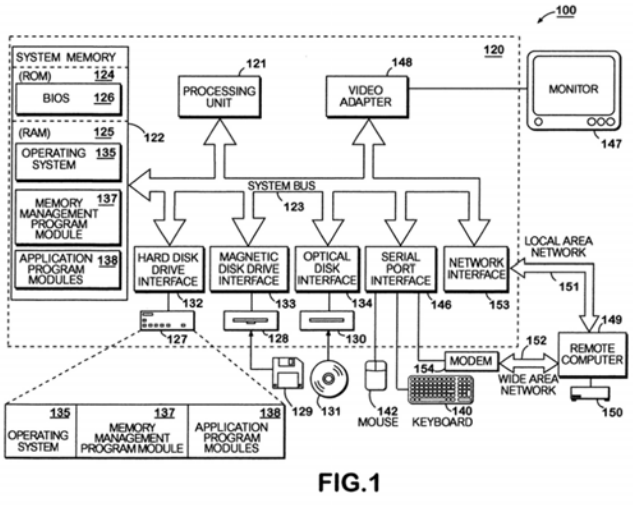


FIG.1

Zwiegincew, Fig. 1

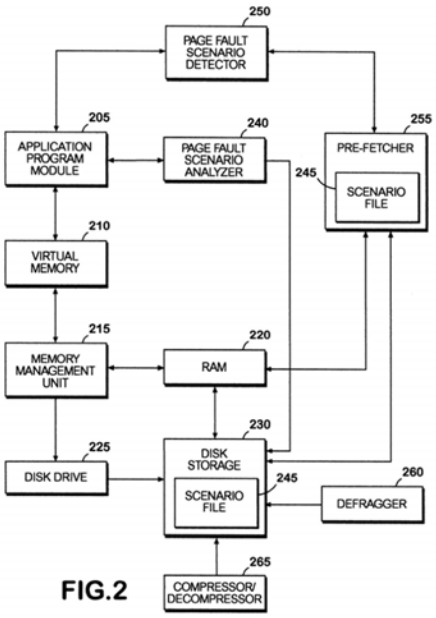


FIG.2

Zwiegincew. Fig 2

There are various potential solutions to the performance bottleneck caused by disk access time during hard page fault scenarios. An obvious potential solution is to reduce disk access time. The reduction of disk access time is primarily a hardware consideration and is not easily

> accomplished. However, other potential solutions involve the manipulation of memory storage through software program modules.

Zwiegincew, 1:45-51

> In an exemplary embodiment, the present invention may comprise one or more memory management program modules 137 stored on the drives or RAM 125 of the computer 100. Specifically, program modules 137 of the present invention may comprise computer implemented instructions for determining which pages will have to be retrieved from disk during a potential hard page fault scenario and pre-fetching the determined pages into RAM prior to the occurrence of the potential hard page fault sequence.

Zwiegincew, 5:50-51.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Zwiegincew, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

> There are various potential solutions to the performance bottleneck caused by disk access time during hard page fault scenarios. An obvious potential solution is to reduce disk access time. The reduction of disk access time is primarily a hardware consideration and is not easily accomplished. However, other potential solutions involve the manipulation of memory storage through software program modules.

Zwiegincew, 1:45-51

> In an exemplary embodiment, the present invention may comprise one or more memory management program modules 137 stored on the

drives or RAM 125 of the computer 100. Specifically, program modules 137 of the present invention may comprise computer implemented instructions for determining which pages will have to be retrieved from disk during a potential hard page fault scenario and pre-fetching the determined pages into RAM prior to the occurrence of the potential hard page fault sequence.

Zwiegincew, 5:50-51.

The present invention meets the needs described above by providing a system and method for improving the performance of an application program module by reducing the occurrence of hard page faults during the operation of an application program module. The present invention may be embodied in an add-on software program module that operates in conjunction with the application program module. In this manner, no effort is required on the part of the application programmer to manipulate or modify the application program module in order to improve performance. Furthermore, the add-on software program module does not detract from the intended operation of the application program module.

According to one aspect of the present invention, a scenario file is created which comprises ordered copies of pages that are likely to be retrieved from disk storage by an application program module during a hard page fault. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the scenario file is fetched from disk storage and the ordered copies of the pages are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. In another aspect of the invention, a hard page fault scenario analyzer is provided for analyzing a hard page fault scenario of an application program module in order to determine the pages that will be retrieved from disk storage upon the occurrence of a hard page fault scenario. The hard page fault scenario analyzer creates a scenario file comprising copies of the pages in the determined order. A hard page fault scenario detector is provided for monitoring the execution of the application module, detecting a hard page fault scenario and sending a message to a pre-fetcher. The hard page fault scenario detector may be manual or automatic. A pre-fetcher retrieves a scenario file from disk storage and transfers the copies of the

> determined pages into RAM. The copies of the determined pages are placed on a standby list in RAM. Accordingly, the determined pages will be available in RAM during a hard page fault scenario and will be soft-faulted into the working set of the application program module when they are requested by the application program module, thereby avoiding a hard page fault.
>
> According to another aspect of the invention, a scenario file is created which comprises ordered references to pages that are likely to be retrieved from disk storage by an application program module during a hard page fault scenario, rather than the actual pages themselves. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the pages referenced by the scenario file are fetched from disk storage in the optimal manner and are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. This aspect of the invention will result in more seek operations on disk, but will still allow reading of the required pages in an optimal manner, rather than the 'as needed' ordering if the pages are hard faulted into RAM. This aspect of the invention also reduces the disk space requirements over the previously mentioned aspect.

Zwiegincew, 2:43-3:49.

> Further, an exemplary embodiment includes a disk compressor/decompressor 265. Well known compression algorithms may be employed to achieve approximately 50% compression with 25 MB/s decompression throughput. These results may be achieved with as little as 64 KB extra memory. Average disk transfer rates are about 8 MB/s. So, for an illustrative 3 MB pre-fetch scenario, comparative pre-fetch times are as follows:
>
> No compression: 0.012 s (seek)+3 MB/8 MB/s (read)=0.3870 s.
>
> 50% compression: 0.012 s (seek)+1.5 MB/8 MB/s (read)+3 MB/25 MB/s (decompress)=0.3195 s.
>
> Thus, there is a 17.5% improvement in pre-fetch time using 50% compression.

Zwiegincew, 8:66-9:13 *See also* Zwiegincew. 1:5-2:40, Figs 1-2

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Zwiegincew, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

> The present invention meets the needs described above by providing a system and method for improving the performance of an application program module by reducing the occurrence of hard page faults during the operation of an application program module. The present invention may be embodied in an add-on software program module that operates in conjunction with the application program module. In this manner, no effort is required on the part of the application programmer to manipulate or modify the application program module in order to improve performance. Furthermore, the add-on software program module does not detract from the intended operation of the application program module.

According to one aspect of the present invention, a scenario file is created which comprises ordered copies of pages that are likely to be retrieved from disk storage by an application program module during a hard page fault. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the scenario file is fetched from disk storage and the ordered copies of the pages are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. In another aspect of the invention, a hard page fault scenario analyzer is provided for analyzing a hard page fault scenario of an application program module in order to determine the pages that will be retrieved from disk storage upon the occurrence of a hard page fault scenario. The hard page fault scenario analyzer creates a scenario file comprising copies of the pages in the determined order. A hard page fault scenario detector is provided for monitoring the execution of the application module, detecting a hard page fault scenario and sending a message to a pre-fetcher. The hard page fault scenario detector may be manual or automatic. A pre-fetcher retrieves a scenario file from disk storage and transfers the copies of the determined pages into RAM. The copies of the determined pages are placed on a standby list in RAM. Accordingly, the determined pages will be available in RAM during a hard page fault scenario and will be soft-faulted into the working set of the application program module when they are requested by the application program module, thereby avoiding a hard page fault.

According to another aspect of the invention, a scenario file is created which comprises ordered references to pages that are likely to be retrieved from disk storage by an application program module during a hard page fault scenario, rather than the actual pages themselves. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the pages referenced by the scenario file are fetched from disk storage in the optimal manner and are transferred into a standby list in RAM. In this manner, the requested

> pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. This aspect of the invention will result in more seek operations on disk, but will still allow reading of the required pages in an optimal manner, rather than the 'as needed' ordering if the pages are hard faulted into RAM. This aspect of the invention also reduces the disk space requirements over the previously mentioned aspect.

Zwiegincew, 2:43-3:49.

> Further, an exemplary embodiment includes a disk compressor/decompressor 265. Well known compression algorithms may be employed to achieve approximately 50% compression with 25 MB/s decompression throughput. These results may be achieved with as little as 64 KB extra memory. Average disk transfer rates are about 8 MB/s. So, for an illustrative 3 MB pre-fetch scenario, comparative pre-fetch times are as follows:
>
> No compression: 0.012 s (seek)+3 MB/8 MB/s (read)=0.3870 s.
>
> 50% compression: 0.012 s (seek)+1.5 MB/8 MB/s (read)+3 MB/25 MB/s (decompress)=0.3195 s.
>
> Thus, there is a 17.5% improvement in pre-fetch time using 50% compression.

Zwiegincew, 8:66-9:13

See also Zwiegincew. 1:5-2:40, 5:50-51, Figs 1-2

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Zwiegincew, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

> There are various potential solutions to the performance bottleneck caused by disk access time during hard page fault scenarios. An obvious potential solution is to reduce disk access time. The reduction of disk access time is primarily a hardware consideration and is not easily

> accomplished. However, other potential solutions involve the manipulation of memory storage through software program modules.

Zwiegincew, 1:45-51

> In an exemplary embodiment, the present invention may comprise one or more memory management program modules 137 stored on the drives or RAM 125 of the computer 100. Specifically, program modules 137 of the present invention may comprise computer implemented instructions for determining which pages will have to be retrieved from disk during a potential hard page fault scenario and pre-fetching the determined pages into RAM prior to the occurrence of the potential hard page fault sequence.

Zwiegincew, 5:50-51.

> The present invention meets the needs described above by providing a system and method for improving the performance of an application program module by reducing the occurrence of hard page faults during the operation of an application program module. The present invention may be embodied in an add-on software program module that operates in conjunction with the application program module. In this manner, no effort is required on the part of the application programmer to manipulate or modify the application program module in order to improve performance. Furthermore, the add-on software program module does not detract from the intended operation of the application program module.
>
> According to one aspect of the present invention, a scenario file is created which comprises ordered copies of pages that are likely to be retrieved from disk storage by an application program module during a hard page fault. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the scenario file is fetched from disk storage and the ordered copies of the pages are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. In another aspect of the invention, a hard page fault scenario analyzer is provided for analyzing a hard page fault scenario of an application program module in order to determine the pages that will be retrieved from disk storage upon the

> occurrence of a hard page fault scenario. The hard page fault scenario analyzer creates a scenario file comprising copies of the pages in the determined order. A hard page fault scenario detector is provided for monitoring the execution of the application module, detecting a hard page fault scenario and sending a message to a pre-fetcher. The hard page fault scenario detector may be manual or automatic. A pre-fetcher retrieves a scenario file from disk storage and transfers the copies of the determined pages into RAM. The copies of the determined pages are placed on a standby list in RAM. Accordingly, the determined pages will be available in RAM during a hard page fault scenario and will be soft-faulted into the working set of the application program module when they are requested by the application program module, thereby avoiding a hard page fault.
>
> According to another aspect of the invention, a scenario file is created which comprises ordered references to pages that are likely to be retrieved from disk storage by an application program module during a hard page fault scenario, rather than the actual pages themselves. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the pages referenced by the scenario file are fetched from disk storage in the optimal manner and are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. This aspect of the invention will result in more seek operations on disk, but will still allow reading of the required pages in an optimal manner, rather than the 'as needed' ordering if the pages are hard faulted into RAM. This aspect of the invention also reduces the disk space requirements over the previously mentioned aspect.

Zwiegincew, 2:43-3:49.

> Further, an exemplary embodiment includes a disk compressor/decompressor 265. Well known compression algorithms may be employed to achieve approximately 50% compression with 25 MB/s decompression throughput. These results may be achieved with as little as 64 KB extra memory. Average disk transfer rates are about 8 MB/s. So, for an illustrative 3 MB pre-fetch scenario, comparative pre-fetch times are as follows:

No compression: 0.012 s (seek)+3 MB/8 MB/s (read)=0.3870 s.

50% compression: 0.012 s (seek)+1.5 MB/8 MB/s (read)+3 MB/25 MB/s (decompress)=0.3195 s.

Thus, there is a 17.5% improvement in pre-fetch time using 50% compression.

Zwiegincew, 8:66-9:13

See also Zwiegincew. 1:5-2:40, 5:50-51, Figs 1-2

1.4.1 updating the boot data list,	Zwiegincew, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

> The present invention meets the needs described above by providing a system and method for improving the performance of an application program module by reducing the occurrence of hard page faults during the operation of an application program module. The present invention may be embodied in an add-on software program module that operates in conjunction with the application program module. In this manner, no effort is required on the part of the application programmer to manipulate or modify the application program module in order to improve performance. Furthermore, the add-on software program module does not detract from the intended operation of the application program module.
>
> According to one aspect of the present invention, a scenario file is created which comprises ordered copies of pages that are likely to be retrieved from disk storage by an application program module during a hard page fault. The scenario file is stored in the disk storage. Then, the

execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the scenario file is fetched from disk storage and the ordered copies of the pages are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. In another aspect of the invention, a hard page fault scenario analyzer is provided for analyzing a hard page fault scenario of an application program module in order to determine the pages that will be retrieved from disk storage upon the occurrence of a hard page fault scenario. The hard page fault scenario analyzer creates a scenario file comprising copies of the pages in the determined order. A hard page fault scenario detector is provided for monitoring the execution of the application module, detecting a hard page fault scenario and sending a message to a pre-fetcher. The hard page fault scenario detector may be manual or automatic. A pre-fetcher retrieves a scenario file from disk storage and transfers the copies of the determined pages into RAM. The copies of the determined pages are placed on a standby list in RAM. Accordingly, the determined pages will be available in RAM during a hard page fault scenario and will be soft-faulted into the working set of the application program module when they are requested by the application program module, thereby avoiding a hard page fault.

According to another aspect of the invention, a scenario file is created which comprises ordered references to pages that are likely to be retrieved from disk storage by an application program module during a hard page fault scenario, rather than the actual pages themselves. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the pages referenced by the scenario file are fetched from disk storage in the optimal manner and are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. This aspect of the invention will result in more seek operations on disk, but will still allow reading of the required pages in an optimal manner, rather than the ‘as needed’

> ordering if the pages are hard faulted into RAM. This aspect of the invention also reduces the disk space requirements over the previously mentioned aspect.

Zwiegincew, 2:43-3:49.

> A hard page fault scenario analyzer 240 anticipates and analyzes hard page fault scenarios. As mentioned, a hard page fault scenario is a situation in which a hard page fault sequence is highly likely to occur. The hard page fault scenario analyzer logs various hard page fault scenarios that occur during operation of the application program module 205. The logged hard page fault scenarios are then analyzed to determine if they re-occur frequently, and if they do, they are put in a scenario file. This analysis can occur programmatically on the end-user's computer system, or in advance by the developer of a particular software product. As an example, suppose the application program module 205 is a word processor and that an anticipated hard page fault scenario is the situation in which the user selects a well known "file open" command. In response to the "file open" command, the application program will display a graphical representation of a file directory. However, in order to display the graphical representation of the file directory, a sequence of hard page faults will occur because the word processor must retrieve a particular set of pages from disk. In accordance with an exemplary embodiment of the present invention, the hard page fault scenario analyzer 240 anticipates the "open file" hard page fault scenario of the example and determines the set of pages that will need to be retrieved from disk upon the occurrence of the hard page fault. The determination of pages that will need to be retrieved from disk will be described in greater detail below. The detection of particular classes of hard page fault scenarios may be built into the system. For example, application launch is virtually always a hard page fault scenario, so an exemplary embodiment of the present invention may be configured such that any launch operation of an application program will be considered to be a hard page fault scenario.

Zwiegincew, 6:29-61.

> The hard page fault scenario analyzer 240 may comprise functionality for automatically analyzing hard page fault scenarios and generating corresponding scenario files. By way of illustration, the hard page fault analyzer 240 may log hard page faults that occur upon execution of a process during operation of an application program module 205. During idle time of the application program module 205, the hard page fault scenario analyzer 240 may write the log of hard page faults to a log file. Then, a pattern matching algorithm may be used to find a pattern of hard page faults based on all log files generated for the process same. If

> a pattern of hard page faults is found, a new scenario file may be generated based on the pages that are retrieved from disk during the pattern. Automatically generated scenario files may be subject to subsequent refinement, i.e., they may be input into the pattern-matching algorithm.

Zwiegincew, 7:24-40

> Further, an exemplary embodiment includes a disk compressor/decompressor 265. Well known compression algorithms may be employed to achieve approximately 50% compression with 25 MB/s decompression throughput. These results may be achieved with as little as 64 KB extra memory. Average disk transfer rates are about 8 MB/s. So, for an illustrative 3 MB pre-fetch scenario, comparative pre-fetch times are as follows:
>
> No compression: 0.012 s (seek)+3 MB/8 MB/s (read)=0.3870 s.
>
> 50% compression: 0.012 s (seek)+1.5 MB/8 MB/s (read)+3 MB/25 MB/s (decompress)=0.3195 s.
>
> Thus, there is a 17.5% improvement in pre-fetch time using 50% compression.

Zwiegincew, 8:66-9:13

See also Zwiegincew. 1:5-2:40, 5:50-51, Figs 1-2

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.4.1, and 2 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

5 (Preamble) A method for booting a computer system, the method comprising:	Zwiegincew, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1-1.4.2 above.	

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Zwiegincew, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Zwiegincew, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Zwiegincew, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Zwiegincew, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Zwiegincew, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

5.6 updating the boot data list;	Zwiegincew, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

6 (Preamble) A system comprising: a processor;	Zwiegincew, as evidenced by the example citations below, discloses "a system comprising: a processor:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.1 a memory; and	Zwiegincew, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Zwiegincew, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1, and 1.2 above.	

6.3 wherein the processor is configured:	Zwiegincew, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Zwiegincew, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Zwiegincew, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Zwiegincew, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

Zwiegincew Claim 6.7
"to update the boot data list."

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Zwiegincew, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Zwiegincew, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Zwiegincew, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Zwiegincew, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Zwiegincew, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Zwiegincew Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Zwiegincew, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.6 updating the boot data list,	Zwiegincew, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

Zwiegincew Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Zwiegincew, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Zwiegincew, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 9.1 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Zwiegincew, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Zwiegincew, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

See Claim 1.1 above.

11.2 accessing the loaded boot data in compressed form from the memory;	Zwiegincew, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Zwiegincew, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Zwiegincew, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

> There are various potential solutions to the performance bottleneck caused by disk access time during hard page fault scenarios. An obvious potential solution is to reduce disk access time. The reduction of disk access time is primarily a hardware consideration and is not easily accomplished. However, other potential solutions involve the manipulation of memory storage through software program modules.

Zwiegincew, 1:45-51

> In an exemplary embodiment, the present invention may comprise one or more memory management program modules 137 stored on the drives or RAM 125 of the computer 100. Specifically, program modules 137 of the present invention may comprise computer implemented

> instructions for determining which pages will have to be retrieved from disk during a potential hard page fault scenario and pre-fetching the determined pages into RAM prior to the occurrence of the potential hard page fault sequence.

Zwiegincew, 5:50-51.

> The present invention meets the needs described above by providing a system and method for improving the performance of an application program module by reducing the occurrence of hard page faults during the operation of an application program module. The present invention may be embodied in an add-on software program module that operates in conjunction with the application program module. In this manner, no effort is required on the part of the application programmer to manipulate or modify the application program module in order to improve performance. Furthermore, the add-on software program module does not detract from the intended operation of the application program module.
>
> According to one aspect of the present invention, a scenario file is created which comprises ordered copies of pages that are likely to be retrieved from disk storage by an application program module during a hard page fault. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the scenario file is fetched from disk storage and the ordered copies of the pages are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. In another aspect of the invention, a hard page fault scenario analyzer is provided for analyzing a hard page fault scenario of an application program module in order to determine the pages that will be retrieved from disk storage upon the occurrence of a hard page fault scenario. The hard page fault scenario analyzer creates a scenario file comprising copies of the pages in the determined order. A hard page fault scenario detector is provided for monitoring the execution of the application module, detecting a hard page fault scenario and sending a message to a pre-fetcher. The hard page fault scenario detector may be manual or automatic. A pre-fetcher retrieves a scenario file from disk storage and transfers the copies of the determined pages into RAM. The copies of the determined pages are placed on a standby list in RAM. Accordingly, the determined pages

will be available in RAM during a hard page fault scenario and will be soft-faulted into the working set of the application program module when they are requested by the application program module, thereby avoiding a hard page fault.

According to another aspect of the invention, a scenario file is created which comprises ordered references to pages that are likely to be retrieved from disk storage by an application program module during a hard page fault scenario, rather than the actual pages themselves. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the pages referenced by the scenario file are fetched from disk storage in the optimal manner and are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. This aspect of the invention will result in more seek operations on disk, but will still allow reading of the required pages in an optimal manner, rather than the 'as needed' ordering if the pages are hard faulted into RAM. This aspect of the invention also reduces the disk space requirements over the previously mentioned aspect.

Zwiegincew, 2:43-3:49.

Further, an exemplary embodiment includes a disk compressor/decompressor 265. Well known compression algorithms may be employed to achieve approximately 50% compression with 25 MB/s decompression throughput. These results may be achieved with as little as 64 KB extra memory. Average disk transfer rates are about 8 MB/s. So, for an illustrative 3 MB pre-fetch scenario, comparative pre-fetch times are as follows:

No compression: 0.012 s (seek)+3 MB/8 MB/s (read)=0.3870 s.

50% compression: 0.012 s (seek)+1.5 MB/8 MB/s (read)+3 MB/25 MB/s (decompress)=0.3195 s.

Thus, there is a 17.5% improvement in pre-fetch time using 50% compression.

Zwiegincew, 8:66-9:13

See also Zwiegincew. 1:5-2:40, Figs 1-2

11.5 updating the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Zwiegincew, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Zwiegincew, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

See Claim 1.1 above.

13.2 accessing the loaded boot data in the compressed form;	Zwiegincew, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Zwiegincew, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

See Claim 1.3 above.

13.4 updating the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Zwiegincew, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Zwiegincew, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 1.2 above.	

14.2 loading the boot data into a memory; and	Zwiegincew, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Zwiegincew, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

See Claims 1.2 and 1.3 above.

14.4 updating the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 1.2 above	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 1.2 above	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 1.2 above	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 1.1 above	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1 and 1.2 above	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 32 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 32 and 33 above.	

Zwiegincew **Claim 45**

"The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Zwiegincew discloses this limitation:

> Hard page fault patterns of an application program module are analyzed in order to determine the pages that will be retrieved from disk storage during a common hard page fault scenario. Copies of, or references to, the determined pages are stored in a scenario file, along with an index referred to as a page sequence. The scenario file may also include a prologue indicating events that lead to a hard page fault scenario and an epilogue that may indicate subsequent hard page fault scenarios. Execution of the application program module is monitored to detect the occurrence of a hard page fault scenario. When a hard page fault scenario is detected, a corresponding scenario file is fetched from disk storage and the determined pages, or copies thereof, are transferred into RAM. The determined pages, or copies thereof, may be placed on a stand-by list in RAM and later soft-faulted into the working set of the application program upon request by the application program module, thereby avoiding a sequence of hard page faults.

Zwiegincew, Abstract

> There are various potential solutions to the performance bottleneck caused by disk access time during hard page fault scenarios. An obvious potential solution is to reduce disk access time. The reduction of disk access time is primarily a hardware consideration and is not easily accomplished. However, other potential solutions involve the manipulation of memory storage through software program modules.

Zwiegincew, 1:45-51

> In an exemplary embodiment, the present invention may comprise one or more memory management program modules 137 stored on the drives or RAM 125 of the computer 100. Specifically, program modules

137 of the present invention may comprise computer implemented instructions for determining which pages will have to be retrieved from disk during a potential hard page fault scenario and pre-fetching the determined pages into RAM prior to the occurrence of the potential hard page fault sequence.

Zwiegincew, 5:50-51.

The present invention meets the needs described above by providing a system and method for improving the performance of an application program module by reducing the occurrence of hard page faults during the operation of an application program module. The present invention may be embodied in an add-on software program module that operates in conjunction with the application program module. In this manner, no effort is required on the part of the application programmer to manipulate or modify the application program module in order to improve performance. Furthermore, the add-on software program module does not detract from the intended operation of the application program module.

According to one aspect of the present invention, a scenario file is created which comprises ordered copies of pages that are likely to be retrieved from disk storage by an application program module during a hard page fault. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the scenario file is fetched from disk storage and the ordered copies of the pages are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. In another aspect of the invention, a hard page fault scenario analyzer is provided for analyzing a hard page fault scenario of an application program module in order to determine the pages that will be retrieved from disk storage upon the occurrence of a hard page fault scenario. The hard page fault scenario analyzer creates a scenario file comprising copies of the pages in the determined order. A hard page fault scenario detector is provided for monitoring the execution of the application module, detecting a hard page fault scenario and sending a message to a pre-fetcher. The hard page fault scenario detector may be manual or automatic. A pre-fetcher retrieves a scenario file from disk storage and transfers the copies of the determined pages into RAM. The copies of the determined pages are

Zwiegincew **Claim 50**

"The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form."

placed on a standby list in RAM. Accordingly, the determined pages will be available in RAM during a hard page fault scenario and will be soft-faulted into the working set of the application program module when they are requested by the application program module, thereby avoiding a hard page fault.

According to another aspect of the invention, a scenario file is created which comprises ordered references to pages that are likely to be retrieved from disk storage by an application program module during a hard page fault scenario, rather than the actual pages themselves. The scenario file is stored in the disk storage. Then, the execution of the application program module is monitored until either an explicit begin-of-scenario instruction is detected, or a hard page fault scenario is detected. A hard page fault scenario may comprise any situation or event that is likely to trigger a hard page fault, i.e., one or more requested pages will not be available in RAM and will be retrieved from disk storage. In response to the detection of a begin-of-scenario instruction or a hard page fault scenario, the pages referenced by the scenario file are fetched from disk storage in the optimal manner and are transferred into a standby list in RAM. In this manner, the requested pages will be soft faulted into a working set of the application program module, and no hard page fault will occur. This aspect of the invention will result in more seek operations on disk, but will still allow reading of the required pages in an optimal manner, rather than the 'as needed' ordering if the pages are hard faulted into RAM. This aspect of the invention also reduces the disk space requirements over the previously mentioned aspect.

Zwiegincew, 2:43-3:49.

Further, an exemplary embodiment includes a disk compressor/decompressor 265. Well known compression algorithms may be employed to achieve approximately 50% compression with 25 MB/s decompression throughput. These results may be achieved with as little as 64 KB extra memory. Average disk transfer rates are about 8 MB/s. So, for an illustrative 3 MB pre-fetch scenario, comparative pre-fetch times are as follows:

No compression: 0.012 s (seek)+3 MB/8 MB/s (read)=0.3870 s.

50% compression: 0.012 s (seek)+1.5 MB/8 MB/s (read)+3 MB/25 MB/s (decompress)=0.3195 s.

Thus, there is a 17.5% improvement in pre-fetch time using 50% compression.

Zwiegincew **Claim 50**

"The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form."

Zwiegincew, 8:66-9:13 *See also* Zwiegincew. 1:5-2:40, Figs 1-2

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Zwiegincew, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 32, 33 and 49 above.	

Zwiegincew **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 32, 33 and 49 above.	

Zwiegincew **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

Zwiegincew **Claim 83**
"The method of claim 13, wherein the boot data in the compressed form represents a plurality of files."

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 15, 17 and 24 above.	

Zwiegincew **Claim 89**
"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 31 above.	

Zwiegincew **Claim 91**

"The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access."

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 32, 33 and 49 above.	

Zwiegincew **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claim 32, 33, and 49 above.	

Zwiegincew **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

Appendix C24
Invalidity of U.S. Patent 8,880,862 based on Zwiegincew

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

Zwiegincew **Claim 97.1**

"The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,"

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

Zwiegincew **Claim 108**
"The method of claim 2, further comprising: compressing at least a portion of the additional boot data."

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Zwiegincew, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Zwiegincew discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

Anyimi, "Implementing a Plug and Play BIOS Using Intel's Boot Block Flash Memory," Feb. 1995 ("Anyimi") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Anyimi, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this claim limitation:

Plug and Play can make the usual experience of adding new functionality to an existing system as easy as:

1. Turn the system off.
2. Insert the new device.
3. Turn the system on.

PnP flash BIOS can also extend the life of the PC. By enabling simple updates to the BIOS (simply insert the upgrade disk and the software programs the new BIOS), the user can get more out of their PC investment.

Anyimi, 1.0

3.1 PnP Components

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

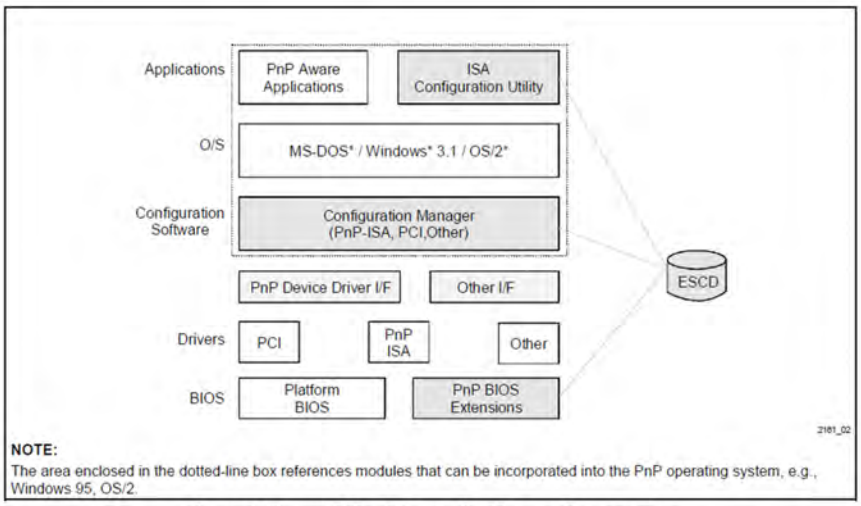


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

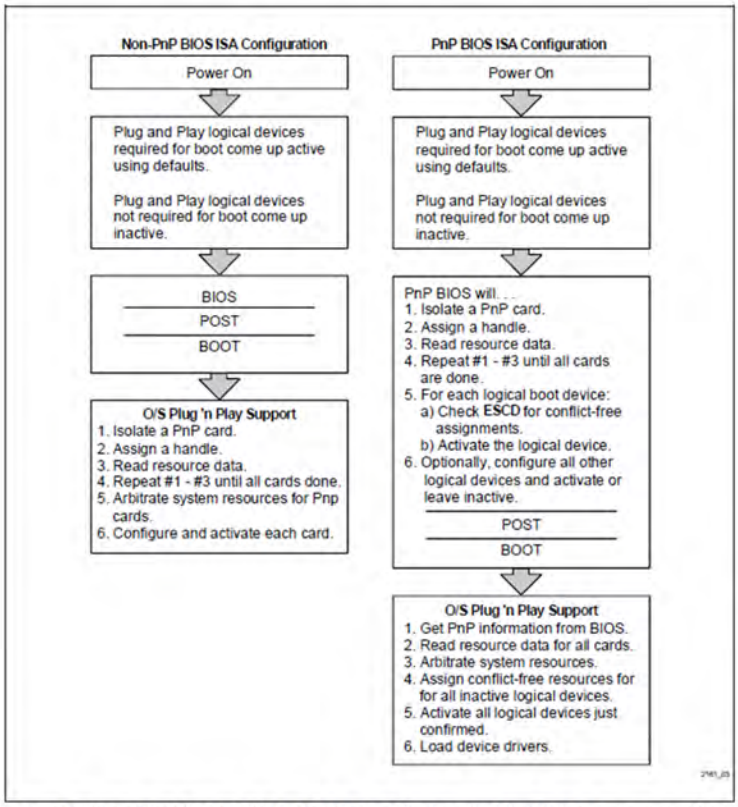


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Anyimi, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this claim limitation:

3.1 PnP Components

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

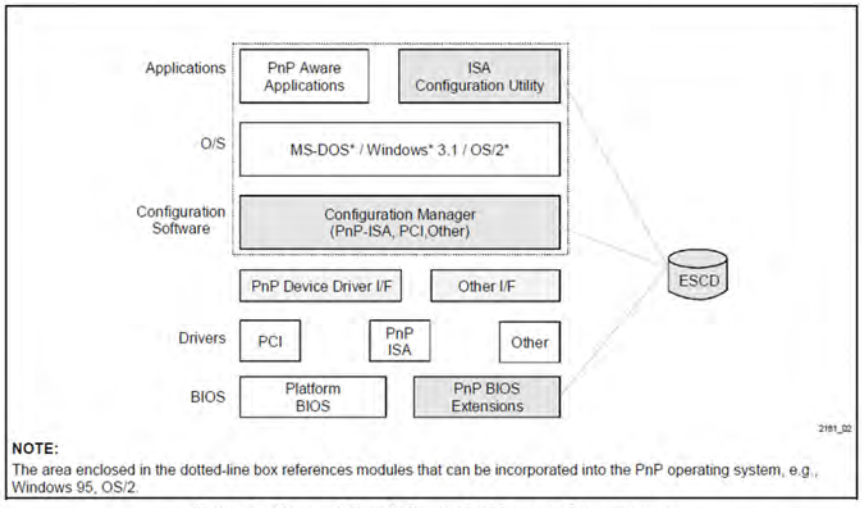


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

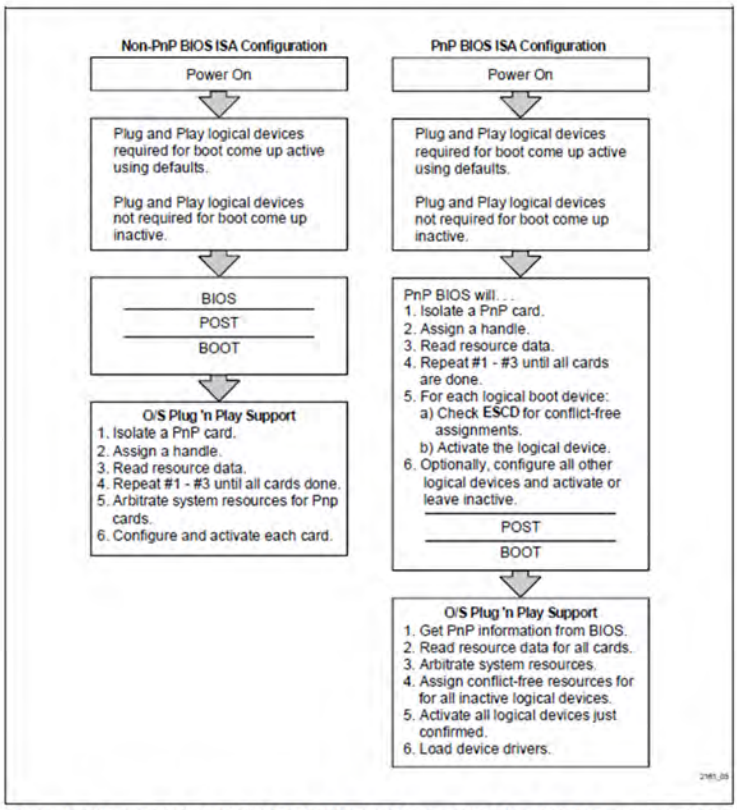


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Anyimi Claim 1.1

"loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory."

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Anyimi, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this claim limitation:

> Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

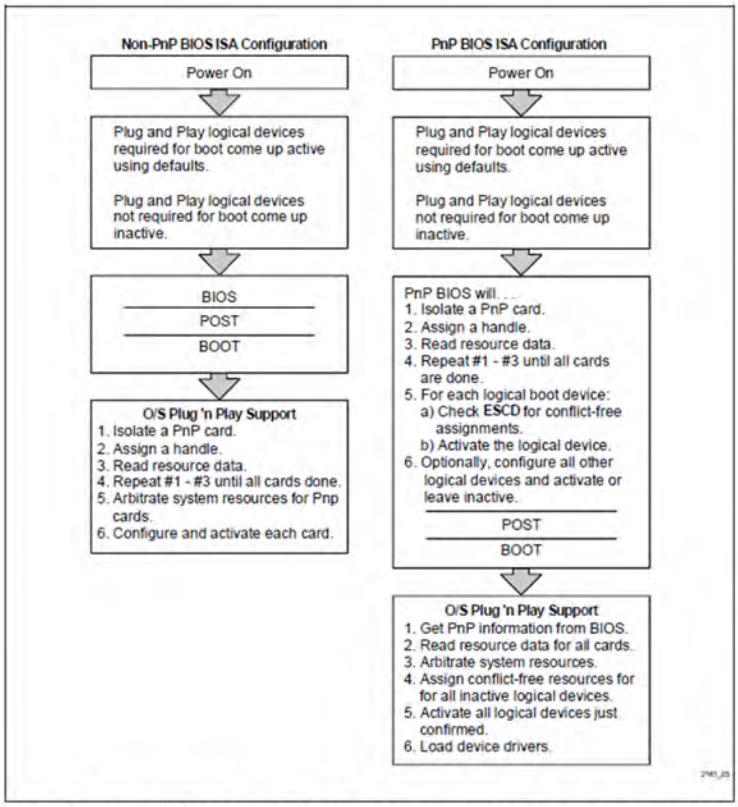


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Anyimi, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this claim limitation:

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

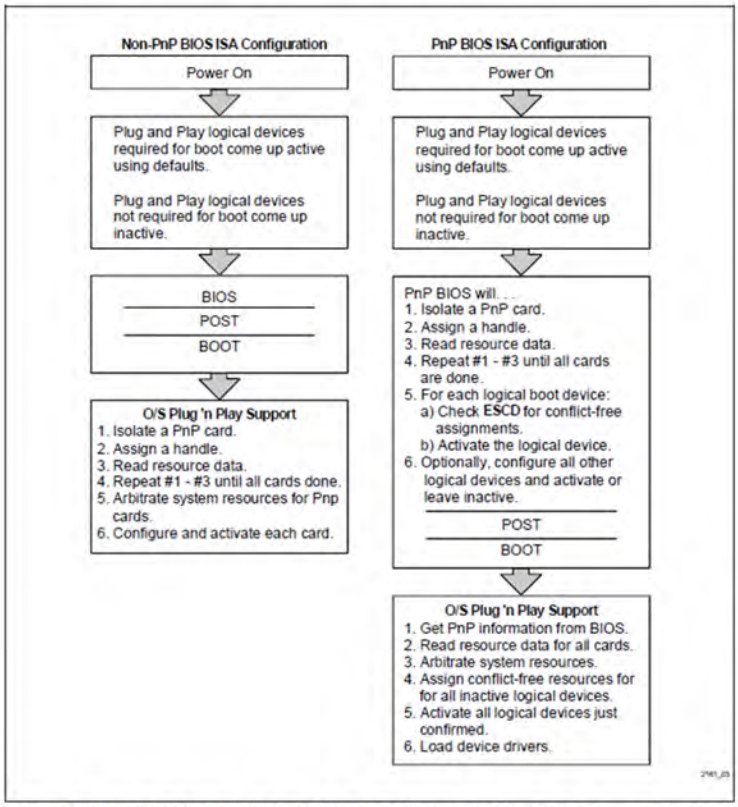


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Anyimi Claim 1.3

"decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form."

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

1.4.1 updating the boot data list,	Anyimi, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this claim limitation:

Plug and Play can make the usual experience of adding new functionality to an existing system as easy as:

1. Turn the system off.
2. Insert the new device.
3. Turn the system on.

PnP flash BIOS can also extend the life of the PC. By enabling simple updates to the BIOS (simply insert the upgrade disk and the software programs the new BIOS), the user can get more out of their PC investment.

Anyimi, 1.0

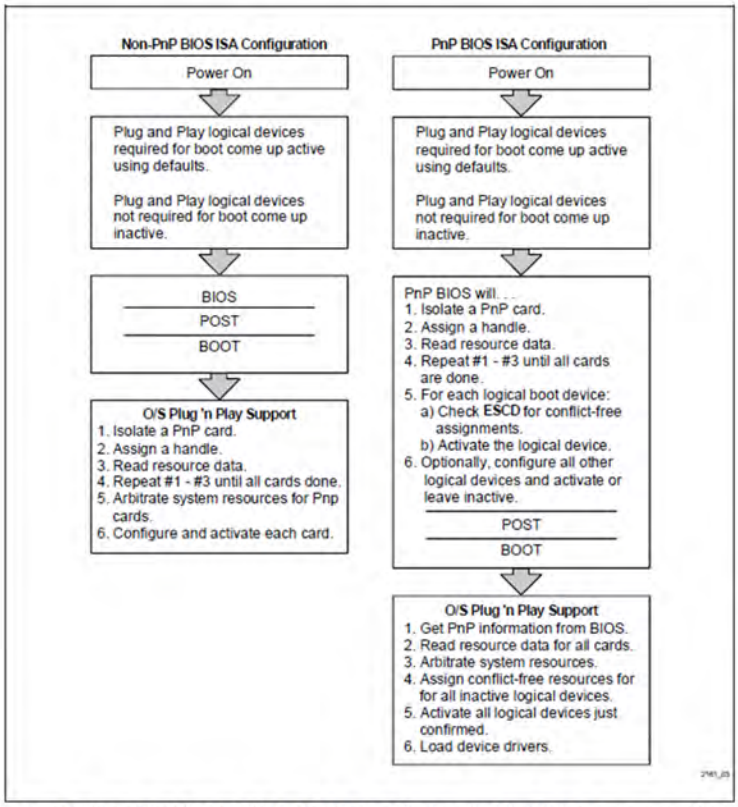


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Anyimi, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this claim limitation:

> Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

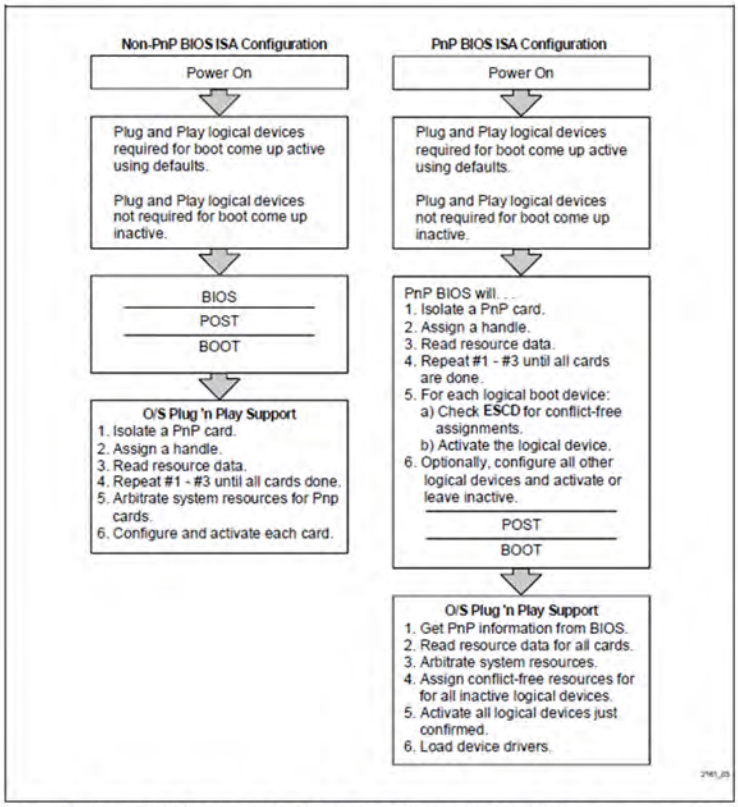


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Anyimi, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Anyimi, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Anyimi, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Anyimi, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claims 1-1.4.2 above.

See also

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

3.1 PnP Components

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

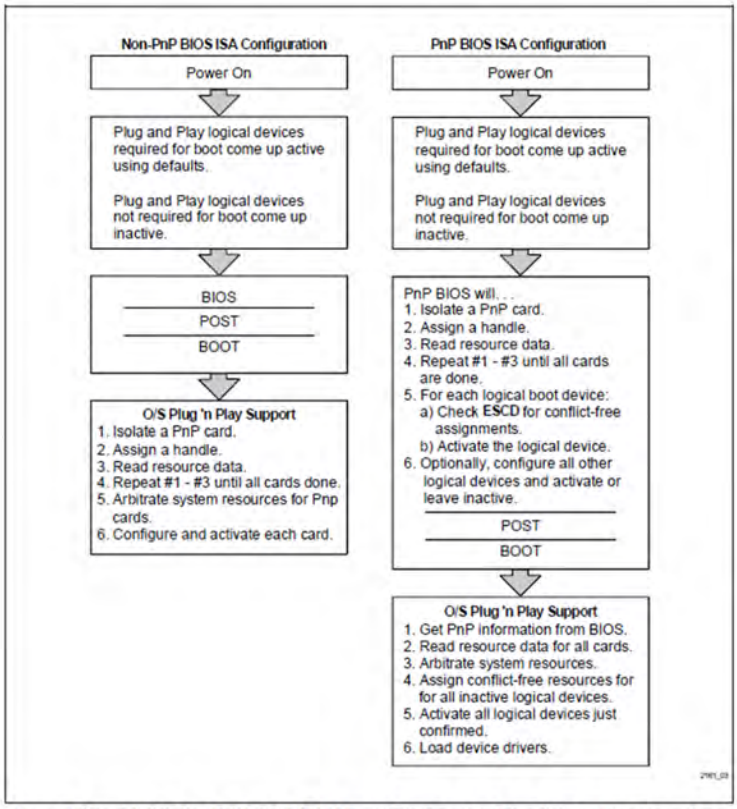


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Anyimi **Claim 5 (Preamble)**

"A method for booting a computer system, the method comprising:"

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Anyimi, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Anyimi, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Anyimi, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Anyimi, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Anyimi, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

Anyimi discloses this claim limitation:

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

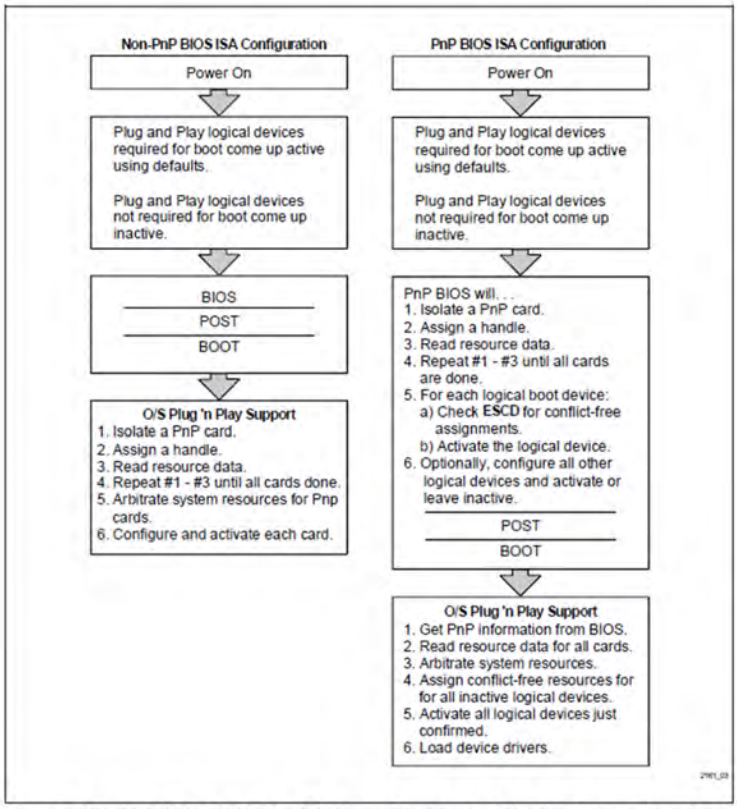


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

5.6 updating the boot data list;	Anyimi, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Anyimi, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1-1.3 above.	

6 (Preamble) A system comprising: a processor;	Anyimi, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

3.1 PnP Components

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

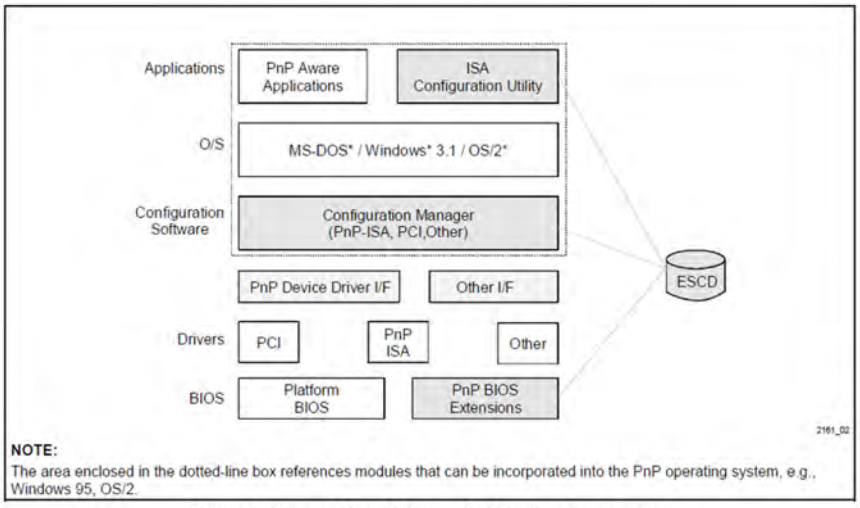


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

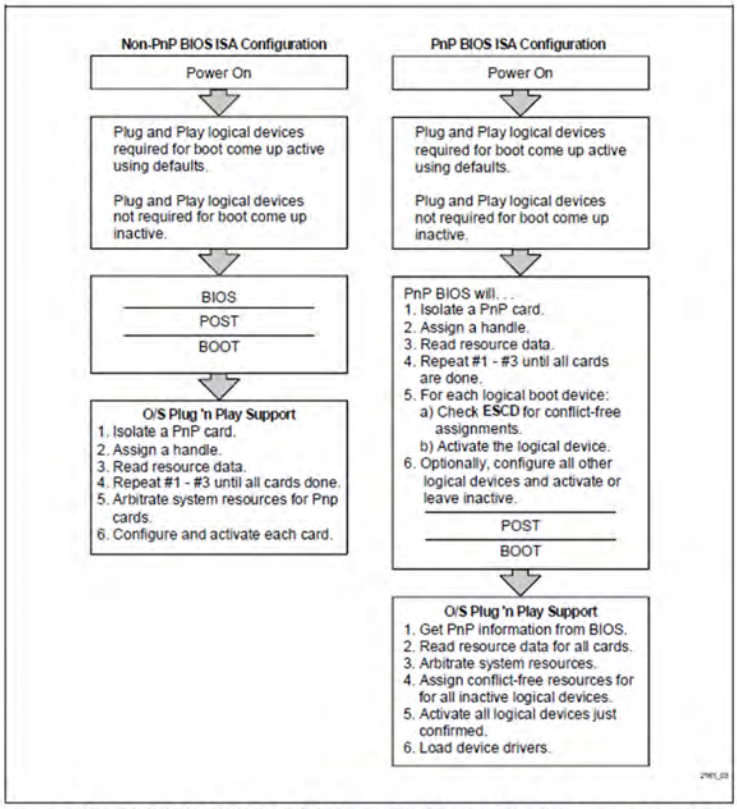


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

6.1 a memory; and	Anyimi, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Anyimi, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

3.1 PnP Components

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

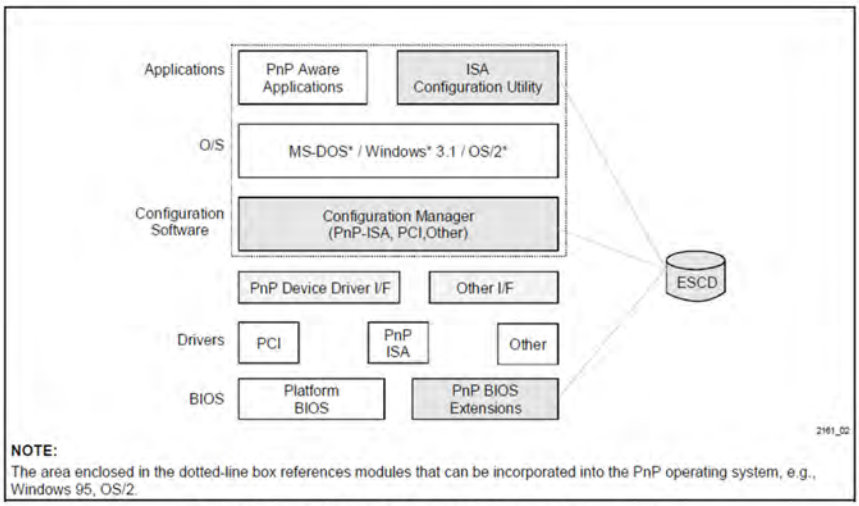


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

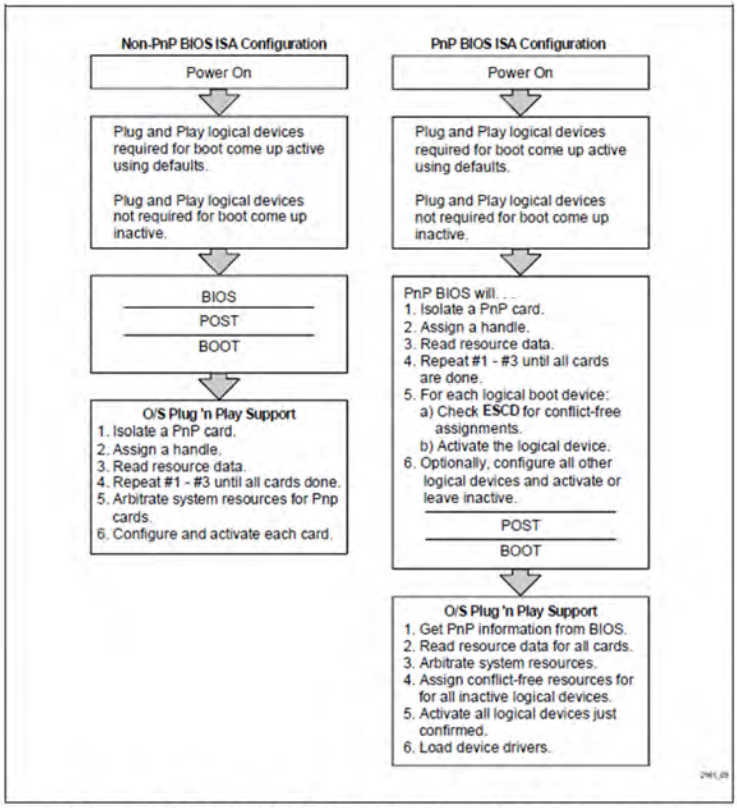


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

6.3 wherein the processor is configured:	Anyimi, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

3.1 PnP Components

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

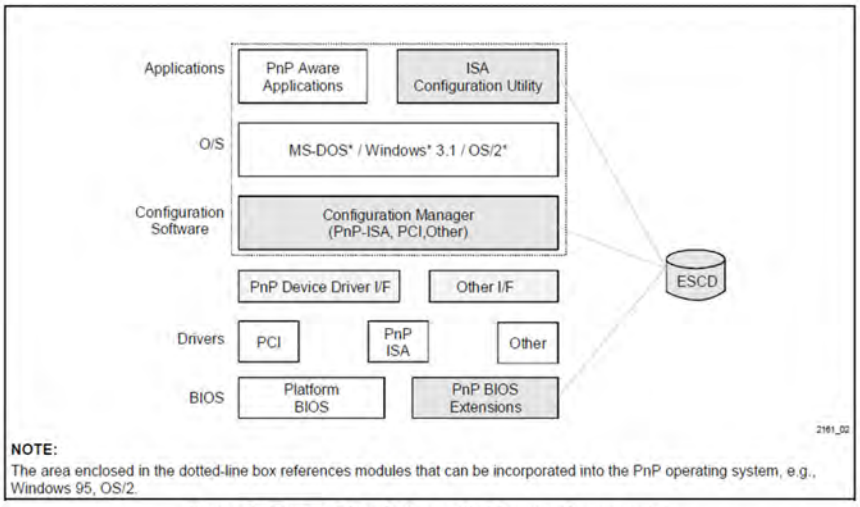


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi Claim 6.3
"wherein the processor is configured"

Anyimi, Fig. 2

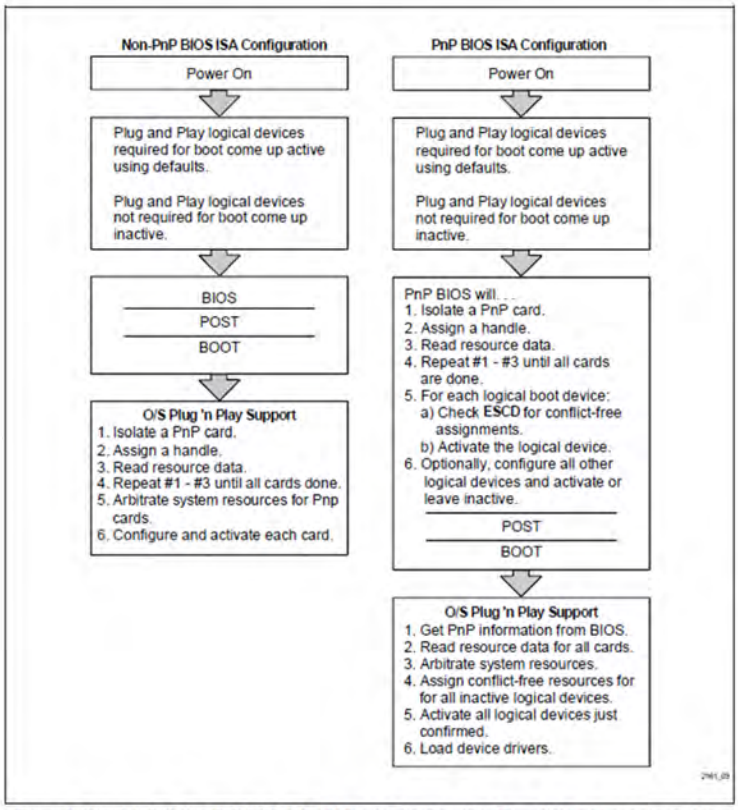


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Anyimi, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Anyimi, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.2 above.	

Anyimi Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Anyimi, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Anyimi, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

Anyimi Claim 6.7
"to update the boot data list."

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Anyimi, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1 (Preamble) above.	

Anyimi **Claim 8 (Preamble)**
"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Anyimi, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Anyimi, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Anyimi, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Anyimi, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Anyimi, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

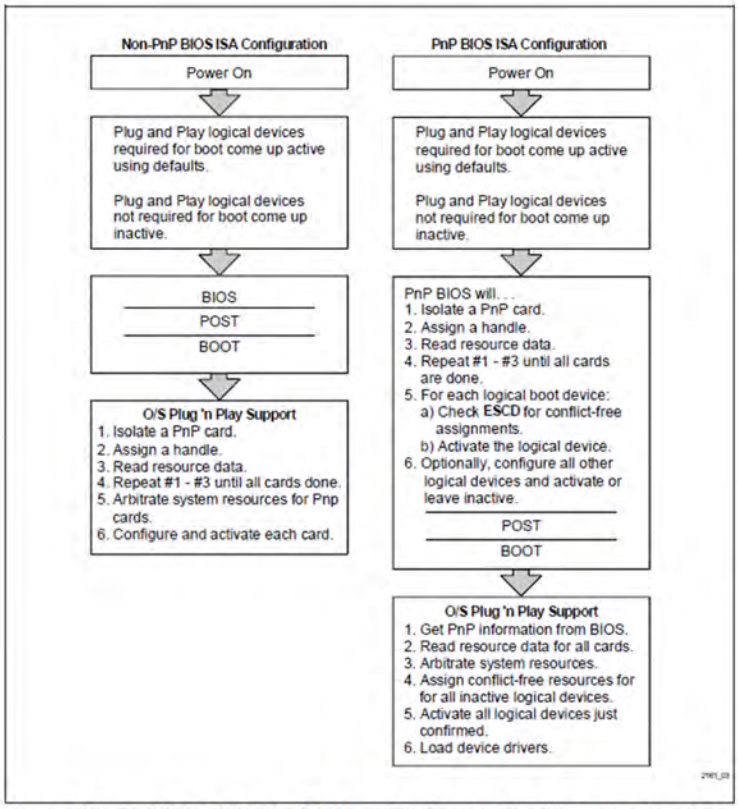


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Anyimi Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

8.6 updating the boot data list,	Anyimi, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

Anyimi Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Anyimi, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Anyimi, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 8.1 above.	

Anyimi Claim 9.2
"storing the additional portion of the operating system in the first memory"

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Anyimi, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 9.1 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Anyimi, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1 (Preamble) above.	

Anyimi **Claim 11 (Preamble)**

"A method for providing accelerated loading of an operating system in a computer system, comprising"

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Anyimi, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claim 1.1 above.

See also

Anyimi discloses this claim limitation:

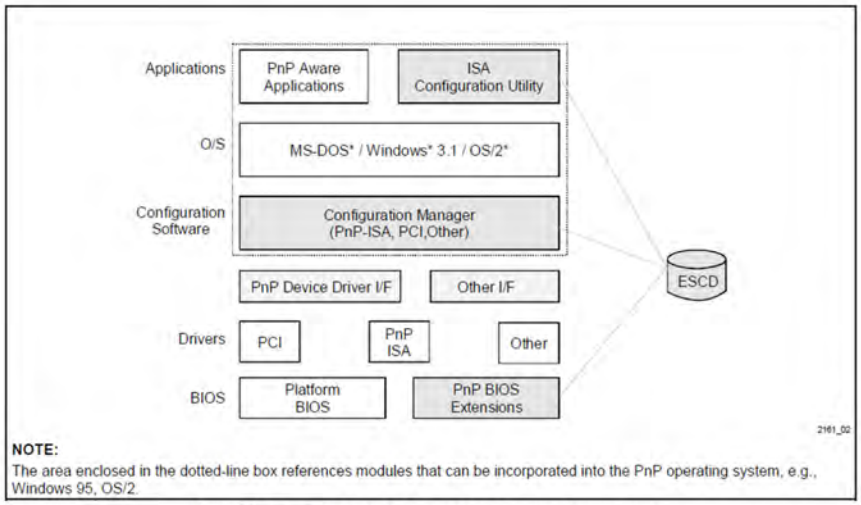


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

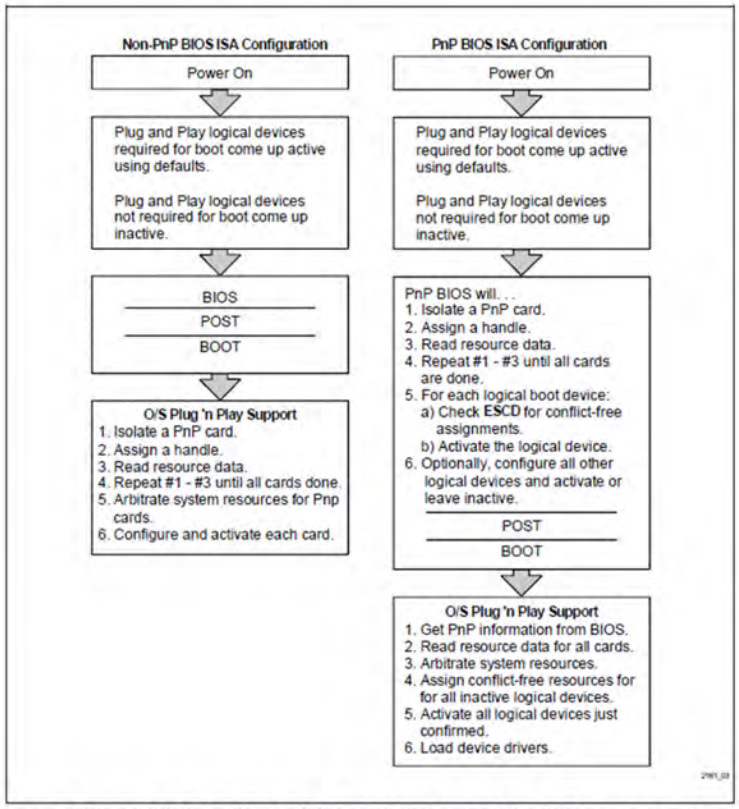


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Anyimi Claim 11.1

"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

11.2 accessing the loaded boot data in compressed form from the memory;	Anyimi, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Anyimi, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Anyimi, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

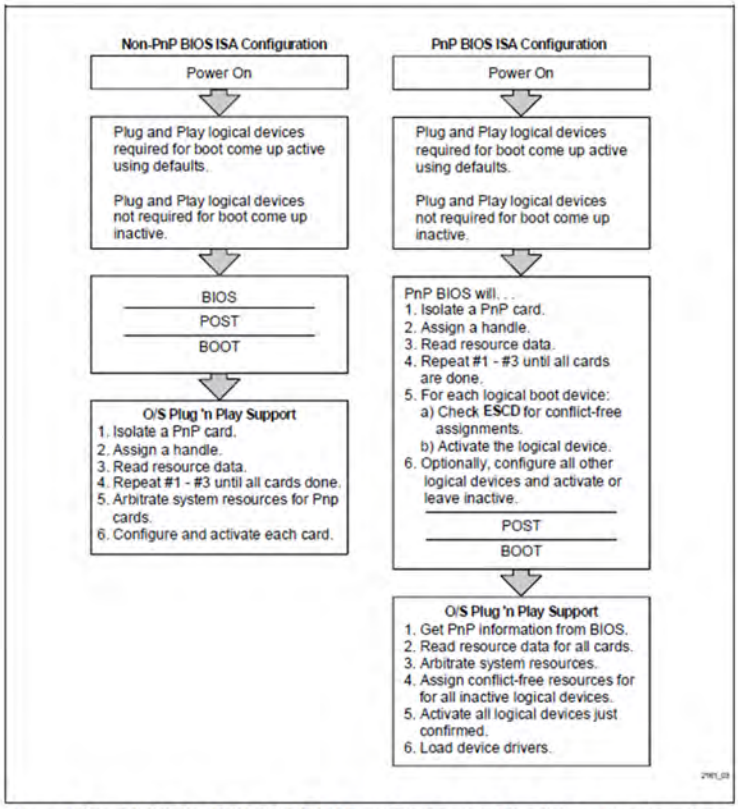


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

11.5 updating the boot data list.	Anyimi, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Anyimi, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Anyimi, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Anyimi, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Anyimi, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claim 1.3 above.

13.4 updating the boot data list.	Anyimi, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Anyimi, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Anyimi, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

Anyimi discloses this claim limitation:

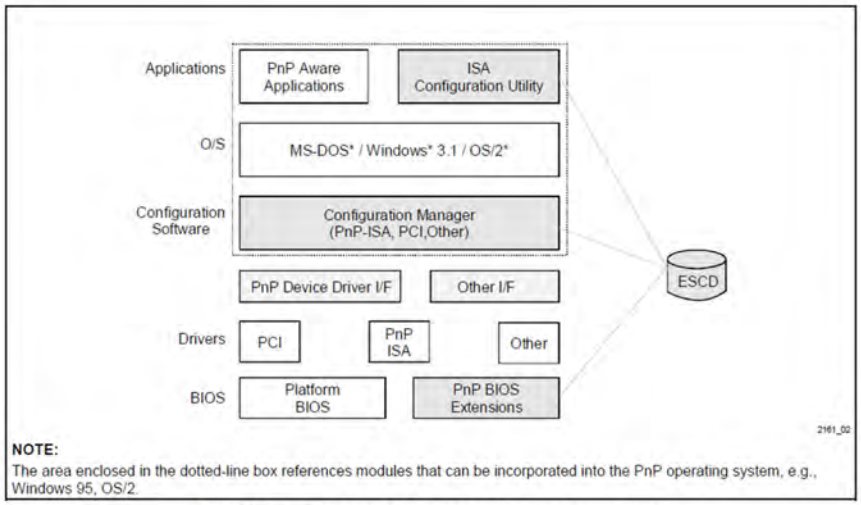


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

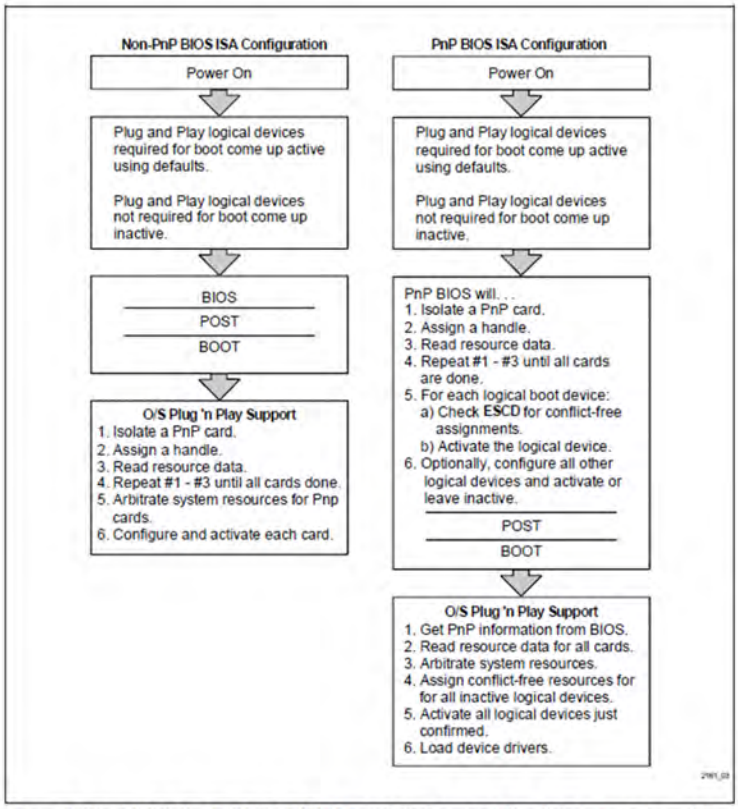


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

14.2 loading the boot data into a memory; and	Anyimi, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Anyimi, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

Anyimi discloses this claim limitation:

Today, PnP BIOS requirements may extend beyond 128 KB. A standard system BIOS consumes roughly 64 KB; PnP support is another 12 KB; automatic power management support adds 2 to 10 KB; PCI (Peripheral Components Interconnect) extensions hoard some 20 KB of the BIOS; and on-board VGA (Video Graphics Adapter) controllers utilize an extra 30 KB to 40 KB of space. The potential size of the BIOS for such a system is almost 150 KB. Although most of this code may be compressed, certain code segments cannot; besides even compression has its limits. Since new developments for the PC show no signs of abatement, it seems clear that a means of updating as well as expanding BIOS quickly and easily has become a necessity. Plug and Play is only one of the many technologies that can be implemented better with flash. As users realize the benefits of PnP, the demand generated will drive the number of Plug and Play systems upward.

Anyimi, 2.2

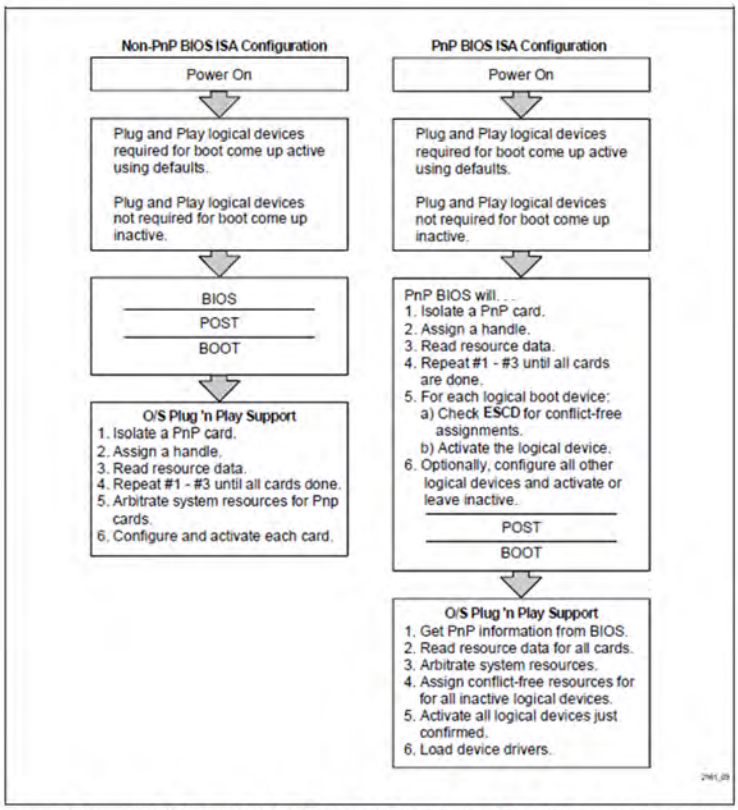


Figure 3. Possible PnP/Non-PnP BIOS ISA Add-In Card Configuration Flow. Note the importance of the ESCD in the Plug and Play Configuration Flow

Anyimi, Fig. 3

Furthermore, the PnP BIOS is capable of event management. Through its event management interfaces, the PnP BIOS can alert the system about new devices, like a notebook docking station, added or removed during runtime.

The POST procedure of the PnP BIOS identifies, tests, and configures the system before passing control to the operating system. The POST process must maintain previous POST compatibility, configure all legacy devices known to the PnP BIOS, arbitrate resources, initialize the IPL (Initial Program Load—this is how the operating system is launched), and support both PnP and non-PnP operating systems. Upon completion of the POST process, the BIOS attempts to have all necessary system devices initialized and enabled before the operating system is loaded; the PnP BIOS aspires further to provide the operating system a conflict-free environment in which to boot.

Anyimi, 3.2

Anyimi **Claim 14.3**

"servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

Without an ESCD, the PnP BIOS must perform resource allocation as well as conflict detection and resolution each and every time the system is booted, and any locking of devices must be done by the supporting PnP operating system. Although the need for nonvolatile storage cannot be disputed, the amount of storage required depends on whether or not the PnP system needs real time updates. Ultimately, it will be decided by the methodology selected for resource management. A static or dynamic allocation scheme requires a fixed amount of nonvolatile memory for the ESCD and other BIOS parameters. A combined scheme for allocation constantly adds or removes from the ESCD data structure. Other parameters may need to be updated as a result. Regardless of the storage method chosen, the BIOS must know how to resolve any untimely interruption of a crucial system or BIOS function. The underlying mechanism for appeasing all these requirements is flash, and Intel's boot block flash is the solution for today's demands and tomorrow's inclinations.

Anyimi, 3.2

Figure 1 showed that either a 1-Mb or 2-Mb flash device can be used to implement BIOS. The choice of device depends on the contents of the BIOS and the level of sophistication desired in the design. The 1-Mb boot block flash memory is an established standard for implementing flash BIOS, not just for PnP. However, more BIOS space will be needed to support standardization of current features (like power management and virus aids) and impending future enhancements (like Windows 95 and Desktop Management Interfaces*, DMI). As consumers clamor for "more bang for the buck," more features and functions will be integrated into the standard PC. The migration beyond a 128-KB BIOS is inevitable. Already, some vendors have adopted code compression of BIOS in order to adhere to this 128-KB space limitation. This is a viable alternative that requires additional code decompression algorithms and consumes additional RAM space to store the full BIOS. Fortunately, Intel provides a secure means of code storage as well as a built-in growth path with its boot block architecture.

Anyimi, 5.1

14.4 updating the boot data list.	Anyimi, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

Perhaps you are wondering why flash is the medium of choice advocated in this application note as the means for storing system BIOS, particularly for Plug and Play? From a PC user's perspective, a PnP flash BIOS enables users to install new hardware without having to call the support number. This translates to ease-of-use:

- Easily updatable code assuring optimal BIOS performance
- No need to edit the CONFIG.SYS file.
- No need to determine system type and match, somehow, to jumper settings.
- No need to investigate available system resources and muddle through reassigning them.
- No need to fiddle with system memory reallocation.
- No need to buy a new system every time something changes (increases system's useful lifespan).

Anyimi, 2.0

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

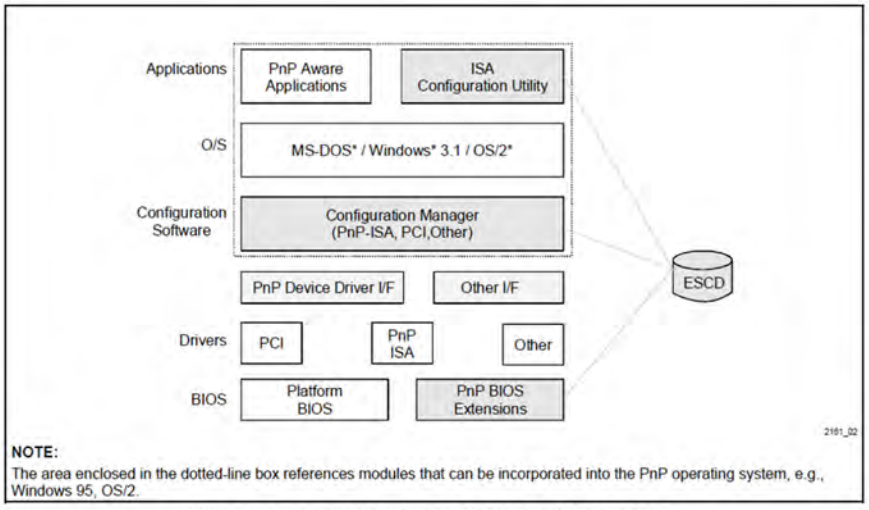


Figure 2. Plug and Play Software Architecture Components.
Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

The Intel boot block (BB) flash memory products are particularly well suited for BIOS applications. Boot block flash is segmented into a lockable boot block, two parameter blocks and one or more main blocks. All boot

block devices are manufactured on Intel's ETOX™ flash memory process technology. An on-chip Write State Machine (WSM) provides automated program and erase algorithms with an SRAM-compatible write interface. The key feature of the boot block architecture that differentiates it from other flash memories is its hardware-lockable boot block, which allows system recovery from fatal crashes. Additional features of boot block flash include:

- Hardware write protection via pin
- Hardware locking of boot block
- Hardware pin for system reset during write
- High performance reads (for speedy access to data)
- Deep power-down mode (a key feature for "green" PCs)
- Extended cycling capability (100,000 block/erase cycles)

Anyimi, 4.0

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Anyimi discloses this limitation:

See Claim 15 above.

See also

Perhaps you are wondering why flash is the medium of choice advocated in this application note as the means for storing system BIOS, particularly for Plug and Play? From a PC user's perspective, a PnP flash BIOS enables users to install new hardware without having to call the support number. This translates to ease-of-use:

- Easily updatable code assuring optimal BIOS performance
- No need to edit the CONFIG.SYS file.
- No need to determine system type and match, somehow, to jumper settings.
- No need to investigate available system resources and muddle through reassigning them.
- No need to fiddle with system memory reallocation.
- No need to buy a new system every time something changes (increases system's useful lifespan).

Anyimi, 2.0

For a system to be fully PnP-compliant, four basic elements are required:

1. System/PnP BIOS
2. PnP operating system
3. PnP hardware devices
4. PnP application software

Anyimi, 3.1

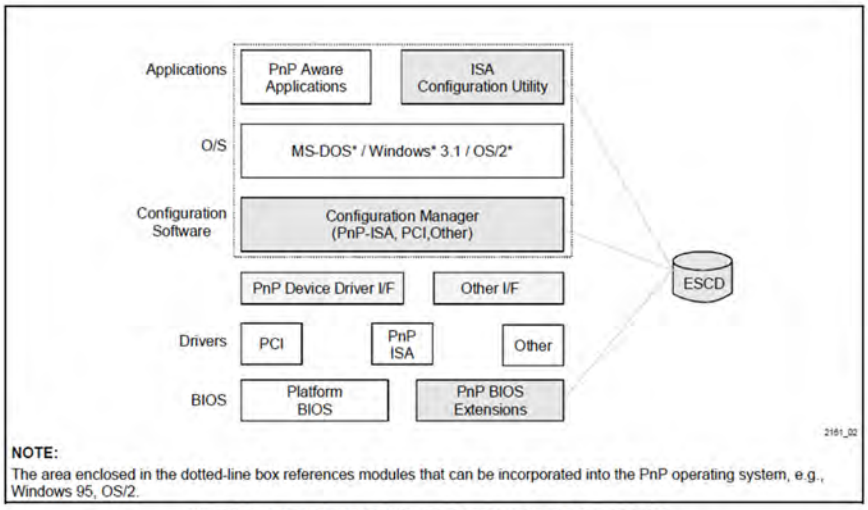


Figure 2. Plug and Play Software Architecture Components. Note the many layers that depend on the ESCD. The parameter blocks of the boot block flash memory enable this nonvolatile storage capability of Plug and Play.

Anyimi, Fig. 2

The Intel boot block (BB) flash memory products are particularly well suited for BIOS applications. Boot block flash is segmented into a lockable boot block, two parameter blocks and one or more main blocks. All boot block devices are manufactured on Intel's ETOX™ flash memory process technology. An on-chip Write State Machine (WSM) provides automated program and erase algorithms with an SRAM-compatible write interface. The key feature of the boot block architecture that differentiates it from other flash memories is its hardware-lockable boot block, which allows system recovery from fatal crashes. Additional features of boot block flash include:

- Hardware write protection via pin
- Hardware locking of boot block
- Hardware pin for system reset during write
- High performance reads (for speedy access to data)
- Deep power-down mode (a key feature for "green" PCs)
- Extended cycling capability (100,000 block/erase cycles)

Anyimi, 4.0

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Anyimi, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 and 1.2 above	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Anyimi, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 and 1.2 above	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 and 1.2 above	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 above	

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 and 32 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1 and 1.2 above.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 1.1 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 27 and 39 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Anyimi, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

Anyimi **Claim 59**

"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 31 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 32, 33 and 49 above.	

Anyimi **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

Anyimi **Claim 84**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system."

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

87. The method of claim 13, wherein the memory comprises: a physical memory.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 15, 17 and 24 above.	

Anyimi **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 31 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 32, 33 and 49 above.	

Anyimi **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claim 32, 33, and 49 above.	

Anyimi **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Anyimi, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

Appendix C25
Invalidity of U.S. Patent 8,880,862 based on Anyimi

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Anyimi, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Anyimi discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

The publication D. Bennett, "Booting Linux from EPROM," Linux Journal, January 1997 ("Bennett") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Bennett, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Bennett, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Bennett, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Bennett, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

1.4.1 updating the boot data list,	Bennett, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Bennett, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Bennett, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Bennett, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. :	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Bennett, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

5 (Preamble) A method for booting a computer system, the method comprising:	Bennett, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1-1.4.2 above.	

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Bennett, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Bennett, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Bennett, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Bennett, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Bennett, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

5.6 updating the boot data list;	Bennett, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Bennett, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1-1.3 above.	

6 (Preamble) A system comprising: a processor;	Bennett, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, initializing a central processing unit of the computer system), Apple contends that one of skill in the art would understand the operation of loading of an operating system, to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art about the loading of an operating system during prosecution. *See* Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

6.1 a memory; and	Bennett, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Bennett, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

6.3 wherein the processor is configured:	Bennett, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Bennett, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Bennett, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Bennett, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Bennett, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

Bennett Claim 6.7
"to update the boot data list."

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Bennett, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Bennett, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Bennett, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Bennett, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Bennett, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Bennett, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.6 updating the boot data list,	Bennett, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Bennett, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Bennett, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Bennett, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Bennett, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claim 9.1 above.

See also

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

EPROM Disk Compression
(/files/linuxjournal.com/linuxjournal/articles/002/0243/0243f1.jpg)

Bennett, 1

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Bennett, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Bennett, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claim 1.1 above.

See also

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

Bennett Claim 11.1

"loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

11.2 accessing the loaded boot data in compressed form from the memory;	Bennett, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Bennett, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Bennett, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this claim limitation:

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

11.5 updating the boot data list.	Bennett, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Bennett, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Bennett, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Bennett, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Bennett, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Bennett, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. .	

Appendix C26
Invalidity of U.S. Patent 8,880,862 based on Bennett

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Bennett, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Bennett, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

14.2 loading the boot data into a memory; and	Bennett, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Bennett, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

System Operation

For booting, two options were considered:

- booting DOS, then running the loadlin program (to load Linux) from autoexec.bat
- installing LILO and booting Linux directly

The advantage of the second option would be a slightly shorter boot time. However, we implemented the first option, because we wanted to use a programmable keyboard—the software for programming the keyboard runs under DOS.

Bennett, 1

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

Bennett, 1

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

The first phase of disk image development was identifying the required and the desired items. The first step was to come up with a minimal system and then add the items required for the operator interface. Not being a Unix expert, coming up with the minimal system ended up being something of a trial and error process. I started with what I thought was needed, then tried running it. When an error occurred because of a missing program or library, that file was added. This process went on until the system ran happily.

The bulk of this was done by copying files from the "full" Linux partition to the 6MB partition, booting DOS and using the loadlin line:

```
loadlin zimage root=/dev/hda2 ro
```

Bennett, 3

Once the system was fairly stable, the 6MB partition was loaded into the RAM disk. This is very similar to how the RAM disk is loaded from EPROM, but development went faster since EPROMs weren't being programmed. To test the system without programming EPROMs, the system booted DOS and ran loadlin with the line:

```
loadlin zimage root=/dev/hda2 ramdisk=6144 ro
```

Because of the modification to ramdisk.c, the /dev/hda2 disk image is loaded into the RAM disk, then the root file system is switched to the RAM disk. The process of refining the disk image continues until everything is "perfect".

Bennett, 4

14.4 updating the boot data list.	Bennett, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Bennett, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 14 above.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Bennett, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claim 15 above.

See also

This is a quick look at making Linux bootable from EPROM on a 486 single board computer.

This article describes one way to run Linux in an embedded system with no hard disk. The application described is an Operator Interface in a monitor and display system developed by Boeing Flight Test. The airborne environment requires something fairly rugged which can withstand common power interruptions. To meet these requirements we decided to build the operator interface without a hard disk.

Bennett, 1

In order to automatically run the operator interface application, a program was written to replace getty. This program (dboot.c) will run login for a given user, and set the stdin, stdout and stderr to the specified virtual console.

The boot sequence is:

- Power up and run memory tests
- load DOS which executes AUTOEXEC.BAT
  - run the keyboard programming application
  - run loadlin—this reads a linux kernel from the DOS disk & executes it
- the linux kernel takes over
- load the RAM disk from the EPROM disk
- switch the root file system to the RAM disk
- init will read inittab which executes dboot instead of getty
- the operator interface application is started

Bennett, 2

Bennett **Claim 17**

"The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Bennett, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.1 and 1.2 above.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Bennett, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.1 and 1.2 above.	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.1 and 1.2 above.	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claim 1.1 above

See also

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

EPROM Disk Compression
(/files/linuxjournal.com/linuxjournal/articles/002/0243/0243f1.jpg)

Bennett, 1

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Bennett, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Bennett discloses this limitation:

See Claim 32 above.

See also

When deciding how to implement the EPROM device driver, the first idea was to create an image of a disk in the EPROM. This would provide a RAM disk of the same size as the EPROM, 3.5MB in this case (the DOS portion of the SSD takes 1/2 MB). Instead, to allow a larger RAM disk, a compressed disk image is used. The compression used is simple—any sectors which are identical are only stored once. The primary advantage this gives is blank areas of the disk image don't need to take up EPROM space. Listing 1 shows the SSD disk compression used.

EPROM Disk Compression
(/files/linuxjournal.com/linuxjournal/articles/002/0243/0243f1.jpg)

Bennett, 1

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.1 and 1.2 above.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation:	

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Bennett, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Bennett, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Bennett, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 32, 33 and 49 above.	

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Bennett, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claim 32, 33, and 49 above.	

Bennett **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Bennett, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

Bennett **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Bennett, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Bennett, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Bennett, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Bennett, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Bennett discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C27
Invalidity of U.S. Patent 8,880,862 based on Burrows

The publication Michael Burrows, Charles Jerian, Butler Lampson, and Timothy Mann, On-line data compression in a log-structured file System ("Burrows") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C27
Invalidity of U.S. Patent 8,880,862 based on Burrows

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Burrows, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Burrows, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "The module that reads a disk block given its logical address needs a way to find the physical address of the compressed bytes. We keep a logical block map for each segment, which is simply an array indexed by compression block number, whose entries are the physical byte addresses of the blocks relative to the start of the segment. The block map is constructed in memory as the segment is being compressed, and written to the end of the segment when the segment is full. The maps are needed for all file reads, so they are cached in memory whenever possible."

Burrows, 5.

> "Unfortunately, this procedure reads and decompresses a full compression block even if the caller wanted only some smaller unit, such as a file system block or a physical sector. We alleviate this problem by caching the entire decompressed block in memory, rather than just caching the requested sectors. The data could be placed in the file system buffer cache, but for simplicity in our prototype, we cached the last decompressed block within the read routine. Sprite LFS reads files sequentially in 4 KByte units, so this simple caching strategy typically achieves three hits for each 16 KByte compression block when reading

large files.

When the file system is reading non-sequentially, the additional time to read a full compression block cannot be hidden by caching. Fortunately, this time is small compared to the rotational latency. The time needed to decompress the full block in software is several milliseconds; it would be much smaller if decompression were implemented in hardware."

Burrows, 6.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Burrows, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

> "The module that reads a disk block given its logical address needs a way to find the physical address of the compressed bytes. We keep a logical block map for each segment, which is simply an array indexed by compression block number, whose entries are the physical byte addresses of the blocks relative to the start of the segment. The block map is constructed in memory as the segment is being compressed, and written to the end of the segment when the segment is full. The maps are needed for all file reads, so they are cached in memory whenever possible."

Burrows, 5.

> "The procedure for finding the compressed data associated with a logical address is as follows:
>
> > 1. Extract the segment number from the logical address. Use it to find the logical block map for the segment.
> >
> > 2. Extract the compression block number from the address. Use it to index the logical block map. This yields the physical byte offset of the compressed data within the segment.
> >
> > 3. Examine the next entry in the map, to find the start of the next block. This determines how much data should be read from the disk.
> >
> > 4. Read the compressed data from the disk and decompress it.
> >
> > 5. Extract the sector number from the logical address. Use it to identify the desired sector within the decompressed block."

Burrows, 5-6.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Burrows, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "Hardware compression devices mesh well with this design. Chips are already available that operate at speeds exceeding disk transfer rates, which indicates that hardware compression would not only remove the performance degradation we observed, but might well increase the effective disk transfer rate beyond that obtainable from a system without compression."

Burrows, Abstract.

> "Building a file system that compresses the data it stores on disk is clearly an attractive idea. First, more data would fit on the disk. Also, if a fast hardware data compressor could be put into the data path to disk, it would increase the effective disk transfer rate by the compression factor, thus speeding up the system."

Burrows, 1.

> "The procedure for finding the compressed data associated with a logical

address is as follows:

> 1. Extract the segment number from the logical address. Use it to find the logical block map for the segment.
>
> 2. Extract the compression block number from the address. Use it to index the logical block map. This yields the physical byte offset of the compressed data within the segment.
>
> 3. Examine the next entry in the map, to find the start of the next block. This determines how much data should be read from the disk.
>
> 4. Read the compressed data from the disk and decompress it.
>
> 5. Extract the sector number from the logical address. Use it to identify the desired sector within the decompressed block."

Burrows, 5-6.

> "Unfortunately, this procedure reads and decompresses a full compression block even if the caller wanted only some smaller unit, such as a file system block or a physical sector. We alleviate this problem by caching the entire decompressed block in memory, rather than just caching the requested sectors. The data could be placed in the file system buffer cache, but for simplicity in our prototype, we cached the last decompressed block within the read routine. Sprite LFS reads files sequentially in 4 KByte units, so this simple caching strategy typically achieves three hits for each 16 KByte compression block when reading large files.
>
> When the file system is reading non-sequentially, the additional time to read a full compression block cannot be hidden by caching. Fortunately, this time is small compared to the rotational latency. The time needed to decompress the full block in software is several milliseconds; it would be much smaller if decompression were implemented in hardware."

Burrows, 6.

1.4.1 updating the boot data list,	Burrows, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Burrows, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Burrows, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Burrows, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Burrows, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Burrows, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1-1.4.2 above.	

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Burrows, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Burrows, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Burrows, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Burrows, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Burrows, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "The procedure for finding the compressed data associated with a logical address is as follows:
>
> > 1. Extract the segment number from the logical address. Use it to find the logical block map for the segment.
> >
> > 2. Extract the compression block number from the address. Use it to index the logical block map. This yields the physical byte offset of the compressed data within the segment.
> >
> > 3. Examine the next entry in the map, to find the start of the next block. This determines how much data should be read from the disk.
> >
> > 4. Read the compressed data from the disk and decompress it.
> >
> > 5. Extract the sector number from the logical address. Use it to identify the desired sector within the decompressed block."

Burrows, 5-6.

> "Unfortunately, this procedure reads and decompresses a full compression block even if the caller wanted only some smaller unit, such as a file system block or a physical sector. We alleviate this problem by caching the entire decompressed block in memory, rather than just caching the requested sectors. The data could be placed in the file system buffer cache, but for simplicity in our prototype, we cached the last decompressed block within the read routine. Sprite LFS reads files sequentially in 4 KByte units, so this simple caching strategy typically achieves three hits for each 16 KByte compression block when reading large files.
>
> When the file system is reading non-sequentially, the additional time to read a full compression block cannot be hidden by caching. Fortunately, this time is small compared to the rotational latency. The time needed to decompress the full block in software is several milliseconds; it would be much smaller if decompression were implemented in hardware."

Burrows, 6.

5.6 updating the boot data list;	Burrows, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Burrows, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C27
Invalidity of U.S. Patent 8,880,862 based on Burrows

6 (Preamble) A system comprising: a processor;	Burrows, as evidenced by the example citations below, discloses "a system comprising: a processor:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.1 a memory; and	Burrows, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Burrows, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.3 wherein the processor is configured:	Burrows, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Burrows, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Burrows, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Burrows, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Burrows, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

Burrows Claim 6.7
"to update the boot data list."

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Burrows, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1 (Preamble) above.	

Burrows **Claim 8 (Preamble)**
"A method of loading an operating system for booting a computer system, comprising:"

8.1 storing a portion of the operating system in a compressed form in a first memory;	Burrows, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Burrows, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Burrows, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Burrows, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Burrows, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "The procedure for finding the compressed data associated with a logical address is as follows:
>
> > 1. Extract the segment number from the logical address. Use it to find the logical block map for the segment.
> >
> > 2. Extract the compression block number from the address. Use it to index the logical block map. This yields the physical byte offset of the compressed data within the segment.
> >
> > 3. Examine the next entry in the map, to find the start of the next block. This determines how much data should be read from the disk.
> >
> > 4. Read the compressed data from the disk and decompress it.
> >
> > 5. Extract the sector number from the logical address. Use it to

identify the desired sector within the decompressed block."

Burrows, 5-6.

> "Unfortunately, this procedure reads and decompresses a full compression block even if the caller wanted only some smaller unit, such as a file system block or a physical sector. We alleviate this problem by caching the entire decompressed block in memory, rather than just caching the requested sectors. The data could be placed in the file system buffer cache, but for simplicity in our prototype, we cached the last decompressed block within the read routine. Sprite LFS reads files sequentially in 4 KByte units, so this simple caching strategy typically achieves three hits for each 16 KByte compression block when reading large files.
>
> When the file system is reading non-sequentially, the additional time to read a full compression block cannot be hidden by caching. Fortunately, this time is small compared to the rotational latency. The time needed to decompress the full block in software is several milliseconds; it would be much smaller if decompression were implemented in hardware."

Burrows, 6.

8.6 updating the boot data list,	Burrows, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

Burrows Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Burrows, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

See Claim 1.1 above.

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "The module that reads a disk block given its logical address needs a way to find the physical address of the compressed bytes. We keep a logical block map for each segment, which is simply an array indexed by compression block number, whose entries are the physical byte addresses of the blocks relative to the start of the segment. The block map is constructed in memory as the segment is being compressed, and written to the end of the segment when the segment is full. The maps are needed for all file reads, so they are cached in memory whenever possible."

Burrows, 5.

9.2 storing the additional portion of the operating system in the first memory, and	Burrows, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Burrows, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Burrows, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

See Claim 9.1 above.

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "The module that reads a disk block given its logical address needs a way to find the physical address of the compressed bytes. We keep a logical block map for each segment, which is simply an array indexed by compression block number, whose entries are the physical byte addresses of the blocks relative to the start of the segment. The block map is constructed in memory as the segment is being compressed, and written to the end of the segment when the segment is full. The maps are needed for all file reads, so they are cached in memory whenever possible."

Burrows, 5.

Appendix C27
Invalidity of U.S. Patent 8,880,862 based on Burrows

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Burrows, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Burrows, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

11.2 accessing the loaded boot data in compressed form from the memory;	Burrows, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Burrows, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Burrows, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "The procedure for finding the compressed data associated with a logical address is as follows:
>
> 1. Extract the segment number from the logical address. Use it to find the logical block map for the segment.
>
> 2. Extract the compression block number from the address. Use it to index the logical block map. This yields the physical byte offset of the compressed data within the segment.
>
> 3. Examine the next entry in the map, to find the start of the next block. This determines how much data should be read from the disk.
>
> 4. Read the compressed data from the disk and decompress it.
>
> 5. Extract the sector number from the logical address. Use it to identify the desired sector within the decompressed block."

Burrows, 5-6.

> "Unfortunately, this procedure reads and decompresses a full compression block even if the caller wanted only some smaller unit, such

as a file system block or a physical sector. We alleviate this problem by caching the entire decompressed block in memory, rather than just caching the requested sectors. The data could be placed in the file system buffer cache, but for simplicity in our prototype, we cached the last decompressed block within the read routine. Sprite LFS reads files sequentially in 4 KByte units, so this simple caching strategy typically achieves three hits for each 16 KByte compression block when reading large files.

When the file system is reading non-sequentially, the additional time to read a full compression block cannot be hidden by caching. Fortunately, this time is small compared to the rotational latency. The time needed to decompress the full block in software is several milliseconds; it would be much smaller if decompression were implemented in hardware."

Burrows, 6.

11.5 updating the boot data list.	Burrows, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Burrows, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Burrows, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Burrows, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Burrows, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Burrows, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Burrows, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Burrows, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1 and 1.2 above.	

14.2 loading the boot data into a memory; and	Burrows, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Burrows, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

See Claims 1.2 and 1.3 above.

See ***Add disclosure from '608 Patent Claim 1.4***

> "We have incorporated on-line data compression into the low levels of a log-structured file system (Rosenblum's Sprite LFS). Each block of data or meta-data is compressed as it is written to the disk and decompressed as it is read."

Burrows, Abstract.

> "Hardware compression devices mesh well with this design. Chips are already available that operate at speeds exceeding disk transfer rates, which indicates that hardware compression would not only remove the performance degradation we observed, but might well increase the effective disk transfer rate beyond that obtainable from a system without compression."

Burrows, Abstract.

> "Building a file system that compresses the data it stores on disk is clearly an attractive idea. First, more data would fit on the disk. Also, if a fast hardware data compressor could be put into the data path to disk, it would increase the effective disk transfer rate by the compression factor, thus speeding up the system."

Burrows, 1.

> "The module that reads a disk block given its logical address needs a way to find the physical address of the compressed bytes. We keep a logical block map for each segment, which is simply an array indexed by compression block number, whose entries are the physical byte addresses of the blocks relative to the start of the segment. The block map is constructed in memory as the segment is being compressed, and written to the end of the segment when the segment is full. The maps are needed for all file reads, so they are cached in memory whenever possible."

Burrows, 5.

> "The procedure for finding the compressed data associated with a logical address is as follows:
>
> 1. Extract the segment number from the logical address. Use it to find the logical block map for the segment.
>
> 2. Extract the compression block number from the address. Use it to index the logical block map. This yields the physical byte offset of the compressed data within the segment.
>
> 3. Examine the next entry in the map, to find the start of the next block. This determines how much data should be read from the disk.
>
> 4. Read the compressed data from the disk and decompress it.
>
> 5. Extract the sector number from the logical address. Use it to identify the desired sector within the decompressed block."

Burrows, 5-6.

> "Unfortunately, this procedure reads and decompresses a full compression block even if the caller wanted only some smaller unit, such as a file system block or a physical sector. We alleviate this problem by caching the entire decompressed block in memory, rather than just

caching the requested sectors. The data could be placed in the file system buffer cache, but for simplicity in our prototype, we cached the last decompressed block within the read routine. Sprite LFS reads files sequentially in 4 KByte units, so this simple caching strategy typically achieves three hits for each 16 KByte compression block when reading large files.

When the file system is reading non-sequentially, the additional time to read a full compression block cannot be hidden by caching. Fortunately, this time is small compared to the rotational latency. The time needed to decompress the full block in software is several milliseconds; it would be much smaller if decompression were implemented in hardware."

Burrows, 6.

14.4 updating the boot data list.	Burrows, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Burrows, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Burrows, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Burrows, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Burrows, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 15 above.	

Appendix C27
Invalidity of U.S. Patent 8,880,862 based on Burrows

27. The method of claim 1, wherein the memory comprises: a physical memory.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1 and 1.2 above.	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1 and 1.2 above.	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Burrows discloses this limitation:

> "Besides the algorithms based on Wheeler's ideas, we tried the LZRW1-A and LZRW3-A algorithms due to Williams. A full description and implementation are available elsewhere [15, 16], so we omit the details here."

Burrows, 11. *See also* Table 2.

> "For comparison, we include figures for the popular compress utility, which uses the LZC algorithm, and the zoo archiver, which uses the LZSS algorithm. The table shows that the algorithms we chose are quite fast, but better compression could be obtained by sacrificing speed.
>
> Bell, Witten, and Cleary give a more thorough comparison of compression algorithms and their effectiveness on different sorts of data [3]. They also describe the LZC and LZSS algorithms."

Burrows, 11-12.

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Burrows, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1 and 1.2 above.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Burrows, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

Burrows **Claim 59**

"The method of claim 8, wherein the operating system in the compressed form represents a plurality of files."

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Burrows, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 32, 33 and 49 above.	

Burrows **Claim 80**

"The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form."

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Burrows, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 32, 33 and 49 above.	

Burrows **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

Burrows **Claim 84**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system."

87. The method of claim 13, wherein the memory comprises: a physical memory.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 15, 17 and 24 above.	

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 32, 33 and 49 above.	

Burrows **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Burrows, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claim 32, 33, and 49 above.	

Burrows **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Burrows, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

Burrows **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Burrows, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Burrows, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Burrows, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Burrows, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Burrows discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

The publication Cheng et al., Fast and highly reliable IBMLZ1 compression chip and algorithm for storage, Hot Chips V11, August 1995 ("Cheng") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Cheng, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

> "Early Objectives: 16X speed-up, 80% cost reduction"

Cheng, 144.

> "Compression Benefits for Storage System
>
> • Compression x Compaction = 4.5 X - 6.0 X
>
> •System Performance"

Cheng, 144. *See also* Cheng, 144 Fig.1, Fig. 2.

> "Data compression allows more efficient use of storage media and communication bandwidth. Standard compression offerings for tape storage have been well-established since the late 1980s."

Cheng, 155.

> "The desire to extend these cost/performance benefits to higher data-rate media and broader media forms, such as DASD storage subsystems, motivated the design and development of the IBMLZ1 compression algorithm and technology."

Cheng, 155.

> "The addition of compression capability to the DASD subsystem facilitates more efficient use of subsystem resources: cache, data path bandwidth, and disk capacity in a transparent manner to the system storing the data."

Cheng, 155.

> "If the data is compressed as it enters the subsystem, see Fig. 1, the cache resource has effectively been tripled. Customers can benefit either from improved performance due to better hit ratios, or reduce their cost by configuring smaller amounts of cache. An added benefit of data compression upon entry to the subsystem is the reduced utilization of internal buses and DASD paths as data flows through the subsystem. Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by th compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155. *See also* Fig. 1.

> "The IBMLZl algorithm and technology was designed for high compression/decompression throughput with efficient hardware implementation, high reliability, low system overhead, and robust compression. Data integrity and reliability are ensured by coupled compression-decompression checking, scrubbing operation, and extensive build-in checkings. Extremely low CPB=l[6] compression and decompression have been achieved. The extremely high compressing/decompressing throughput of 30 MB/sec-50 MB/sec allows transparent mode of operation and hence achieved minimal system overhead. The IBMLZl algorithm compresses well over the VM, MVS, RS6000, and PC test cases."

Cheng, 163.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Cheng, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions..	

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Cheng, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Cheng, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

> "Data compression allows more efficient use of storage media and communication bandwidth. Standard compression offerings for tape storage have been well-established since the late 1980s."

Cheng, 155.

> "The desire to extend these cost/performance benefits to higher data-rate media and broader media forms, such as DASD storage subsystems, motivated the design and development of the IBMLZ1 compression algorithm and technology."

Cheng, 155.

> "The addition of compression capability to the DASD subsystem facilitates more efficient use of subsystem resources: cache, data path bandwidth, and disk capacity in a transparent manner to the system storing the data."

Cheng, 155.

> "If the data is compressed as it enters the subsystem, see Fig. 1, the cache resource has effectively been tripled. Customers can benefit either from improved performance due to better hit ratios, or reduce their cost by configuring smaller amounts of cache. An added benefit of data compression upon entry to the subsystem is the reduced utilization of internal buses and DASD paths as data flows through the subsystem.

> Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by th compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155. *See also*, Fig. 1.

> "If the data is compressed as it enters the subsystem, see Fig. 1, the cache resource has effectively been tripled."

Cheng, 155.

> "Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by the compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155.

> "The IBMLZl algorithm and technology was designed for high compression/decompression throughput with efficient hardware implementation, high reliability, low system overhead, and robust compression. Data integrity and reliability are ensured by coupled compression-decompression checking, scrubbing operation, and extensive build-in checkings. Extremely low CPB=l[6] compression and decompression have been achieved. The extremely high compressing/decompressing throughput of 30 MB/sec-50 MB/sec allows transparent mode of operation and hence achieved minimal system overhead. The IBMLZl algorithm compresses well over the VM, MVS, RS6000, and PC test cases."

Cheng, 163.

1.4.1 updating the boot data list,	Cheng, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Cheng, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Cheng, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Cheng, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Cheng, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.4.1, and 2 above.	

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

5 (Preamble) A method for booting a computer system, the method comprising:	Cheng, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1-1.4.2 above.	

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Cheng, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Cheng, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Cheng, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Cheng, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Cheng, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

5.6 updating the boot data list;	Cheng, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Cheng, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

See Claims 1-1.3 above.

> "Early Objectives: 16X speed-up, 80% cost reduction"

Cheng, 144.

> "Compression Benefits for Storage System
>
> • Compression x Compaction = 4.5 X - 6.0 X
>
> •System Performance"

Cheng, 144. *See also* Cheng, 144 Fig.1, Fig. 2.

> "Data compression allows more efficient use of storage media and communication bandwidth. Standard compression offerings for tape storage have been well-established since the late 1980s."

Cheng, 155.

> "The desire to extend these cost/performance benefits to higher data-rate media and broader media forms, such as DASD storage subsystems, motivated the design and development of the IBMLZ1 compression algorithm and technology."

Cheng, 155.

> "The addition of compression capability to the DASD subsystem facilitates more efficient use of subsystem resources: cache, data path bandwidth, and disk capacity in a transparent manner to the system storing the data."

Cheng, 155.

> "Compression allows existing platforms and applications to benefit from a lower storage cost and potentially higher performance without any change to system hardware or software."

Cheng, 155.

> "If the data is compressed as it enters the subsystem, see Fig. 1, the cache resource has effectively been tripled. Customers can benefit either from improved performance due to better hit ratios, or reduce their cost by configuring smaller amounts of cache. An added benefit of data compression upon entry to the subsystem is the reduced utilization of internal buses and DASD paths as data flows through the subsystem. Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by th compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155. *See also* Fig. 1.

> "Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by the compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155.

> "The IBMLZl algorithm and technology was designed for high compression/decompression throughput with efficient hardware implementation, high reliability, low system overhead, and robust compression. Data integrity and reliability are ensured by coupled

> compression-decompression checking, scrubbing operation, and extensive build-in checkings. Extremely low CPB=l^6 compression and decompression have been achieved. The extremely high compressing/decompressing throughput of 30 MB/sec-50 MB/sec allows transparent mode of operation and hence achieved minimal system overhead. The IBMLZl algorithm compresses well over the VM, MVS, RS6000, and PC test cases."
>
> Cheng, 163.

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

6 (Preamble) A system comprising: a processor;	Cheng, as evidenced by the example citations below, discloses "a system comprising: a processor:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.1 a memory; and	Cheng, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Cheng, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

See Claims 1.1 and 1.2 above.

6.3 wherein the processor is configured:	Cheng, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Cheng, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Cheng, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.2 above.	

Cheng Claim 6.5
"to access the loaded portion of the boot data in the compressed form"

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Cheng, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.3 above.	

Cheng Claim 6.6
"to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and"

6.7 to update the boot data list.	Cheng, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

Cheng Claim 6.7
"to update the boot data list."

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Cheng, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Cheng, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Cheng, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Cheng, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Cheng, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Cheng Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Cheng, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.6 updating the boot data list,	Cheng, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

Cheng Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Cheng, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

See Claims 1-1.3 and 5.7 above.

> "Early Objectives: 16X speed-up, 80% cost reduction"

Cheng, 144.

> "Compression Benefits for Storage System
>
> • Compression x Compaction = 4.5 X - 6.0 X
>
> •System Performance"

Cheng, 144. *See also* Cheng, 144 Fig.1, Fig. 2.

> "Data compression allows more efficient use of storage media and communication bandwidth. Standard compression offerings for tape storage have been well-established since the late 1980s."

Cheng, 155.

> "The desire to extend these cost/performance benefits to higher data-rate media and broader media forms, such as DASD storage subsystems,

> motivated the design and development of the IBMLZ1 compression algorithm and technology."

Cheng, 155.

> "The addition of compression capability to the DASD subsystem facilitates more efficient use of subsystem resources: cache, data path bandwidth, and disk capacity in a transparent manner to the system storing the data."

Cheng, 155.

> "Compression allows existing platforms and applications to benefit from a lower storage cost and potentially higher performance without any change to system hardware or software."

Cheng, 155.

> "If the data is compressed as it enters the subsystem, see Fig. 1, the cache resource has effectively been tripled. Customers can benefit either from improved performance due to better hit ratios, or reduce their cost by configuring smaller amounts of cache. An added benefit of data compression upon entry to the subsystem is the reduced utilization of internal buses and DASD paths as data flows through the subsystem. Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by th compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155. *See also* Fig. 1.

> "Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by the compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155.

> "The IBMLZl algorithm and technology was designed for high compression/decompression throughput with efficient hardware implementation, high reliability, low system overhead, and robust compression. Data integrity and reliability are ensured by coupled compression-decompression checking, scrubbing operation, and extensive build-in checkings. Extremely low CPB=l^6 compression and decompression have been achieved. The extremely high compressing/decompressing throughput of 30 MB/sec-50 MB/sec allows transparent mode of operation and hence achieved minimal system overhead. The IBMLZl algorithm compresses well over the VM, MVS, RS6000, and PC test cases."

Cheng, 163.

Cheng Claim 8.7

"wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Cheng, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 8.1 above.	

Cheng Claim 9.2
"storing the additional portion of the operating system in the first memory"

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Cheng, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Cheng, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

See Claim 9.1 above.

> "Common methods used for redundancy reduction are run-length encoding, pattern matching, transformation, transform coding, and so on."

Cheng, 156.

> "The Huffman code, however, remains as the most popular encoding method for its simplicity and its effectiveness in general."

Cheng, 156. *See also generally*, Cheng, 156-165, Fig. 6.

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Cheng, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Cheng, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

11.2 accessing the loaded boot data in compressed form from the memory;	Cheng, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Cheng, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Cheng, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

11.5 updating the boot data list.	Cheng, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Cheng, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Cheng, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Cheng, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Cheng, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Cheng, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Cheng, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Cheng, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.1 and 1.2 above.	

14.2 loading the boot data into a memory; and	Cheng, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Cheng, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

See Claims 1.2 and 1.3 above.

> "Data compression allows more efficient use of storage media and communication bandwidth. Standard compression offerings for tape storage have been well-established since the late 1980s."

Cheng, 155.

> "The addition of compression capability to the DASD subsystem facilitates more efficient use of subsystem resources: cache, data path bandwidth, and disk capacity in a transparent manner to the system storing the data."

Cheng, 155.

> "Compression allows existing platforms and applications to benefit from a lower storage cost and potentially higher performance without any change to system hardware or software."

Cheng, 155.

> "If the data is compressed as it enters the subsystem, see Fig. 1, the cache

> resource has effectively been tripled. Customers can benefit either from improved performance due to better hit ratios, or reduce their cost by configuring smaller amounts of cache. An added benefit of data compression upon entry to the subsystem is the reduced utilization of internal buses and DASD paths as data flows through the subsystem. Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by th compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155. *See also* Fig. 1.

> "Finally, sequential performance can be improved. In general, on high end systems with ESCON attached DASD, the throughput of sequential operations is gated by the DASD data rate, typically less than the 18 MB/sec capability of the channel. When the device is transferring compressed data, the transfer rate is effectively multiplied by the compression ratio, allowing the full capability of the channel to be realized."

Cheng, 155.

> "The IBMLZl algorithm and technology was designed for high compression/decompression throughput with efficient hardware implementation, high reliability, low system overhead, and robust compression. Data integrity and reliability are ensured by coupled compression-decompression checking, scrubbing operation, and extensive build-in checkings. Extremely low CPB=l^6 compression and decompression have been achieved. The extremely high compressing/decompressing throughput of 30 MB/sec-50 MB/sec allows transparent mode of operation and hence achieved minimal system overhead. The IBMLZl algorithm compresses well over the VM, MVS, RS6000, and PC test cases."

Cheng, 163.

14.4 updating the boot data list.	Cheng, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Cheng, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Cheng, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Cheng, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Cheng, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

Appendix C28
Invalidity of U.S. Patent 8,880,862 based on Cheng

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

"DASO (DISK) Controller Compression and IBMLZ23

•Lossless and General-Purpose Compression

•Simple Algorithm Format and Robust

• Optimized for High-Thruput Hardware Execution: 1 Byte/Cycle,

40MH/sec.

• Algorithm Accepted as QIC 154 Standard"

Cheng, 144.

"IBMLZ1 Algorithm and Hardware Development Paradigm

Extremely High Reliability One undetected Error in 1030

Efficiencies Robust Compression, Speed, Latency

System Requirement Intra-Parallel instead Inter-Parallel, good

Compression, Speed

Silicon Technology Regularity"

Cheng, 146. *See also generally*, Cheng, 147-154.

"The IBMLZ1 compression algorithm was designed not only for robust

> and highly efficient compression, but also for extremely high reliability. As compression removes redundancy in the source, the compressed data becomes extremely vulnerable to data corruption. Key design objectives for the IBMLZ1 development were: efficient hardware execution and efficient use of silicon technology; and minimum system integration overhead. Through new observations of pattern matching match-length distribution and use of graph vertex coloring for evaluating data flows, the IBMLZ1 compression algorithm and technology achieved the above objectives."

Cheng, 155.

> "The LZl and the LZ2 compression algorithms are commonly regarded as Lempel and Ziv's compression algorithm 1 [ZL 77] and algorithm 2 [ZL 78] . The LZl and the LZ2, in their original form, expressed the notion of coding Model and bounds on compression. Professor Lempel noted that more than 90 percent of the compression software in the PC world is derived from either LZl or LZ2 class algorithms."

Cheng, 157. *See also generally*, Cheng, 157-165.

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Cheng, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Cheng discloses this limitation:

> "Common methods used for redundancy reduction are run-length encoding, pattern matching, transformation, transform coding, and so on."

Cheng, 156.

> "The Huffman code, however, remains as the most popular encoding method for its simplicity and its effectiveness in general."

Cheng, 156. *See also generally*, Cheng, 156-165, Fig. 6.

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Cheng, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Cheng, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Cheng, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 32, 33 and 49 above.	

Cheng **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

Cheng **Claim 83**
"The method of claim 13, wherein the boot data in the compressed form represents a plurality of files."

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 27 above.	

Cheng **Claim 87**
"The method of claim 13, wherein the memory comprises: a physical memory."

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 16 and 23 above.	

Cheng **Claim 88**
"The method of claim 13, wherein the operating system comprises: a plurality of files."

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 15, 17 and 24 above.	

Cheng **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 32, 33 and 49 above.	

Cheng **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Cheng, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claim 32, 33, and 49 above.	

Cheng **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Cheng, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

Cheng **Claim 97.2**

"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Cheng, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Cheng, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

Cheng **Claim 107**
"The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Cheng, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

Cheng **Claim 108**
"The method of claim 2, further comprising: compressing at least a portion of the additional boot data."

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Cheng, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Cheng discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Cheng **Claim 109**
"The method of claim 108, further comprising: storing the compressed additional boot data."

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

The publication Craft, A Fast hardware data compression algorithm and some algorithmic extension, IBM J. Res. Develop., Vol 43, Nov. 1998, ("Craft") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Craft, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

> "This paper reports on work at IBM's Austin and Burlington laboratories concerning fast hardware implementations of general-purpose lossless data compression algorithms, particularly for use in enhancing the data capacity of computer storage devices or systems, and transmission data rates for networking or telecommunications channels"

Craft, 733.

> "The scope of the work quickly expanded, however, as it became apparent that data compression technology could have tremendous implications for some major IBM business segments. In particular, its deployment within computer systems to enhance DASD storage capacity, or to increase the effective bandwidth of networking data channels, would present a major competitive advantage."

Craft, 734.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Craft, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Craft, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Craft, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

> "This paper reports on work at IBM's Austin and Burlington laboratories concerning fast hardware implementations of general-purpose lossless data compression algorithms, particularly for use in enhancing the data capacity of computer storage devices or systems, and transmission data rates for networking or telecommunications channels"

Craft, 733.

> "The scope of the work quickly expanded, however, as it became apparent that data compression technology could have tremendous implications for some major IBM business segments. In particular, its deployment within computer systems to enhance DASD storage capacity, or to increase the effective bandwidth of networking data channels, would present a major competitive advantage."

Craft, 734.

1.4.1 updating the boot data list,	Craft, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Craft, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Craft, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Craft, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Craft, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.4.1, and 2 above.	

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

5 (Preamble) A method for booting a computer system, the method comprising:	Craft, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1-1.4.2 above.	

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Craft, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Craft, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Craft, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Craft, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Craft, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

5.6 updating the boot data list;	Craft, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Craft, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

6 (Preamble) A system comprising: a processor;	Craft, as evidenced by the example citations below, discloses "a system comprising: a processor:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.1 a memory; and	Craft, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Craft, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.3 wherein the processor is configured:	Craft, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Craft, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Craft, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Craft, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Craft, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

Craft Claim 6.7
"to update the boot data list."

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Craft, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Craft, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Craft, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Craft, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Craft, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Craft, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.6 updating the boot data list,	Craft, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

Craft Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Craft, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

See Claims 1-1.3 and 5.7 above.

> "This paper reports on work at IBM's Austin and Burlington laboratories concerning fast hardware implementations of general-purpose lossless data compression algorithms, particularly for use in enhancing the data capacity of computer storage devices or systems, and transmission data rates for networking or telecommunications channels"

Craft, 733.

> "The scope of the work quickly expanded, however, as it became apparent that data compression technology could have tremendous implications for some major IBM business segments. In particular, its deployment within computer systems to enhance DASD storage capacity, or to increase the effective bandwidth of networking data channels, would present a major competitive advantage."

Craft, 734.

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Craft, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Craft, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Craft, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Craft, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

See Claim 9.1 above.

> "However, in our later designs, we chose to add a conventional address decoder to the CAM so that it can also function as an SRAM during an LZ1 decode function. This results in a very compact hardware encoder/decoder, which we call a CRAM design, as it combines a compression engine CAM and a decompression engine RAM into one silicon array."

Craft, 738.

> "An LZ1 decompressor builds and maintains an identical history copy, which it updates in the same manner as the encoder, as each LITERAL or COPY_POINTER is processed."

Craft, 738.

> "The code byte-stream output from this preprocessor is then fed into a standard ALDC-1 encoder. For decompression, an ALDC decoder generates code byte values which are then fed into a hardware postprocessor to reconstruct the original data bit stream."

Craft, 741.

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Craft, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Craft, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

11.2 accessing the loaded boot data in compressed form from the memory;	Craft, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Craft, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Craft, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

11.5 updating the boot data list.	Craft, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Craft, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Craft, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Craft, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Craft, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Craft, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Craft, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Craft, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.1 and 1.2 above.	

14.2 loading the boot data into a memory; and	Craft, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Craft, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

See Claims 1.2 and 1.3 above.

> "This paper reports on work at IBM's Austin and Burlington laboratories concerning fast hardware implementations of general-purpose lossless data compression algorithms, particularly for use in enhancing the data capacity of computer storage devices or systems, and transmission data rates for networking or telecommunications channels"

Craft, 733.

> "The scope of the work quickly expanded, however, as it became apparent that data compression technology could have tremendous implications for some major IBM business segments. In particular, its deployment within computer systems to enhance DASD storage capacity, or to increase the effective bandwidth of networking data channels, would present a major competitive advantage."

Craft, 734.

14.4 updating the boot data list.	Craft, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Craft, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Craft, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Craft, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Craft, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

> "Then, two main classes of adaptive Lempel-Ziv algorithm, now known as LZ1 and LZ2, are introduced. An outline of early work comparing these two types of algorithm is presented, together with some fundamental distinctions which led to the choice and development of an IBM variant of the LZ1 algorithm, ALDC, and its implementation in hardware."

Craft, 733.

> "Adaptive data compression techniques try to construct models, or look for data sequences derived in some fashion from recent experience. The algorithm thus adapts dynamically to different types of data. There are two classes of adaptive algorithm which are generally acknowledged to be among the most effective, yielding good compression over a wide range of data types. These were both first proposed by A. Lempel and J. Ziv, in 1977 and 1978, and are commonly now referred to as LZ1 and LZ2 respectively [1-3].

Craft, 734. See also generally, Craft 734-744.

> "This algorithm employs two cascaded pre/postprocessors, one designed to recode a run of identical byte values, the other to detect and recode Unicode-like data sequences. Unicode [13] is used by the Java language and other Web-based applications."

Craft, 742.

> "The CRAM compression technology is clearly able to cover an extremely wide range of applicability. Its small size allows integration

into the smallest portable, handheld, or wireless applications, where it is much faster and consumes far less power than software."

Craft, 744.

Craft **Claim 32**

"The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form."

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Craft, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Craft discloses this limitation:

> "However, in our later designs, we chose to add a conventional address decoder to the CAM so that it can also function as an SRAM during an LZ1 decode function. This results in a very compact hardware encoder/decoder, which we call a CRAM design, as it combines a compression engine CAM and a decompression engine RAM into one silicon array."

Craft, 738.

> "An LZ1 decompressor builds and maintains an identical history copy, which it updates in the same manner as the encoder, as each LITERAL or COPY_POINTER is processed."

Craft, 738.

> "The code byte-stream output from this preprocessor is then fed into a standard ALDC-1 encoder. For decompression, an ALDC decoder generates code byte values which are then fed into a hardware postprocessor to reconstruct the original data bit stream."

Craft, 741.

> "LZ1 decompression is relatively simple, requiring only a byte-wide SRAM of the same size as the history. An address counter is used, in much the same way as the WS pointer in the CAM, to store each output data byte sequentially into this SRAM. This satisfies the history update requirement. Decoding of COPY-POINTERS simply requires that the specified string be copied out of the SRAM once its start address and the byte count have been decoded. Output of each decompressed data

byte thus requires one READ-WRITE cycle of the SRAM, analogous to the CAM's SEARCH-WRITE cycle during compression."

Craft, 738.

Craft **Claim 50**

"The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form."

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Craft, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Craft, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Craft, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Craft, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Craft, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 31 above.	

Appendix C29
Invalidity of U.S. Patent 8,880,862 based on Craft

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Craft, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 32, 33 and 49 above.	

Craft **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 15, 17 and 24 above.	

Craft **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 32, 33 and 49 above.	

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Craft, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claim 32, 33, and 49 above.	

Craft **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Craft, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Craft, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Craft, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Craft, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Craft, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Craft, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Craft discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

Douglis, "One the Role of Compression in Distributed Systems" ("Douglis 1") and Douglis, "The Compression Cache: Using On-Line Compression to Extend Physical Memory," Winter 1993 USENIX Conference, Jan 1993 ("Douglis 2") (collectively, "Douglis"), alone or in combination, invalidate claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Douglis, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this claim limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Douglis, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this claim limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Douglis, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this claim limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Douglis, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this claim limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on-line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

1.4.1 updating the boot data list,	Douglis, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. *See* Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Douglis, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this claim limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on-line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Douglis, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 1.4.1 above.

See also

> Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Douglis, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 1.4.1 above.

See also

> Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Douglis, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.4.1, and 2 above.

See also

> Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

5 (Preamble) A method for booting a computer system, the method comprising:	Douglis, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1-1.4.2 above.

See also

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

Douglis **Claim 5 (Preamble)**

"A method for booting a computer system, the method comprising:"

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Douglis, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Douglis, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Douglis, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Douglis, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Douglis, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

5.6 updating the boot data list;	Douglis, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Douglis, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1-1.3 above.

See also ***Look for additional references where loading, accessing and decompressing occurs in less time than the time to access the boot data from the first memory if the boot data were stored in an uncompressed form.***

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

6 (Preamble) A system comprising: a processor;	Douglis, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the

Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on-line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

6.1 a memory; and	Douglis, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Douglis, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 1.1, 1.2, and Claim 6 (Preamble) above.	

6.3 wherein the processor is configured:	Douglis, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 6 (Preamble) above	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Douglis, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Douglis, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Douglis, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Douglis, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.4.1 above.	

Douglis Claim 6.7
"to update the boot data list."

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Douglis, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Douglis, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Douglis, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Douglis, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Douglis, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

Douglis Claim 8.4
"decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Douglis, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.3 and 1.4.2 above.

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

Douglis Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the

Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

Douglis Claim 8.5
"utilizing the decompressed portion of the operating system to at least partially boot the computer system"

8.6 updating the boot data list,	Douglis, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.4.1 above.	

Douglis Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Douglis, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1-1.3 and 5.7 above.

3.1 Overview

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

Douglis 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1

Compression speed. Compressing a page, and later decompressing it, must be significantly faster than transferring it to or from backing store. Otherwise, one might as well do traditional paging without compression.

Douglis 2

- To service a page fault for a page that is not already uncompressed and resident in memory, the VM system checks to see whether the page is compressed in memory or on the backing store. If it is on backing store, it is first brought into memory and stored in the compression cache, then it is decompressed and made accessible to the faulting process. The compressed copy in memory can be freed at any time, since there is already a copy on backing store.

Douglis 2

Douglis Claim 8.7

"wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 1.1 above.

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on-line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

Douglis Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

See also

There is also overhead for the space managed by the compression cache itself. Th
uses 8 bytes per page in the range of addresses the compression cache might occupy (a
in Figure 2). This overhead is determined at boot time based on the maximum possibl
the cache. The kernel also allocates a 24-byte header within each physical page frame
mapped into the cache (0.6% overhead), and a 36-byte header for each virtual page that h
compressed and placed in the cache. These overheads occur only when the compression ca
data in it, and are offset by the savings in memory usage due to compression.

Douglis 2

Douglis Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

9.2 storing the additional portion of the operating system in the first memory, and	Douglis, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 8.1 above.

There is also overhead for the space managed by the compression cache itself. Th uses 8 bytes per page in the range of addresses the compression cache might occupy (a in Figure 2). This overhead is determined at boot time based on the maximum possibl the cache. The kernel also allocates a 24-byte header within each physical page frame mapped into the cache (0.6% overhead), and a 36-byte header for each virtual page that h compressed and placed in the cache. These overheads occur only when the compression ca data in it, and are offset by the savings in memory usage due to compression.

Douglis 2

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Douglis, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 8.5 above.

There is also overhead for the space managed by the compression cache itself. Th uses 8 bytes per page in the range of addresses the compression cache might occupy (a in Figure 2). This overhead is determined at boot time based on the maximum possibl the cache. The kernel also allocates a 24-byte header within each physical page frame mapped into the cache (0.6% overhead), and a 36-byte header for each virtual page that h compressed and placed in the cache. These overheads occur only when the compression c data in it, and are offset by the savings in memory usage due to compression.

Douglis 2

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Douglis, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 9.1 above.

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on-line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

Douglis Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

See also

Burrows *et al.* integrated compression with Sprite LFS [12], also primarily to reduce disk space requirements [4]. They argued that LFS is a better vehicle for compressing files than traditional file systems, since files are not overwritten in place and a change to one block within a file would not cause changes to compressed data later in the file. Multiple file blocks may be compressed as a unit, providing better compression than if each block were compressed separately using a dynamic compression algorithm such as LZRW1 [16]. Burrows *et al.* found that on-line compression halved disk space requirements, as in Cate and Gross's system, without the delays that could be incurred by decompressing a large file as a single unit. The system had an acceptable performance degradation when compression was performed in software, and was well-suited to hardware compression.

Douglis 2

Douglis Claim 10

"The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder"

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Douglis, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Douglis, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 1.1 and 6(Preamble) above.

There is also overhead for the space managed by the compression cache itself. The kernel uses 8 bytes per page in the range of addresses the compression cache might occupy (as shown in Figure 2). This overhead is determined at boot time based on the maximum possible size of the cache. The kernel also allocates a 24-byte header within each physical page frame that is mapped into the cache (0.6% overhead), and a 36-byte header for each virtual page that has been compressed and placed in the cache. These overheads occur only when the compression cache has data in it, and are offset by the savings in memory usage due to compression.

Douglis 2

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2

11.2 accessing the loaded boot data in compressed form from the memory;	Douglis, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Douglis, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Douglis, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

11.5 updating the boot data list.	Douglis, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Douglis, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Douglis, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Douglis, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Douglis, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Douglis, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Douglis, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Douglis, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1 and 1.2 above.

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

14.2 loading the boot data into a memory; and	Douglis, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Douglis, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.2 and 1.3 above.

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1, Abstact

Data compression has long been used as a method to reduce the number of bits being stored or transmitted [7], but stored where and transmitted over what? The answer to this question is changing as the gap between processing speed and I/O bandwidth increases, and the uses for computers shift toward multimedia and other high-data-rate applications.

Traditionally, compression was used to reduce disk storage demands, by selectively compressing files that had not been accessed for a long time, and to reduce bandwidth requirements over such media as wide-area networks and modems. Some forms of compression are now nearly ubiquitous: for example, the vast majority of "anonymous FTP" archives store all but the most trivial of files in compressed format, saving disk space while making retrievals over the Internet faster. (Other sites sometimes store the original files in uncompressed format but allow on-the-fly compression upon request.) Images impose especially

Douglis 1, at 1

Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1, 2

memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

One question, of course, is whether paging is even interesting or desirable. In some environments, paging is largely passé due to the large amounts of memory available [4]. However, I believe there are still systems and applications that could benefit greatly from a cheaper method of paging. Mobile computers are especially likely to have less memory available than needed, and the cost of paging over a wireless network or onto a small local disk might be much more than the cost of compressing the page. Even in a workstation-based environment, a main-memory database that is within a factor of two or three of the size of available physical memory might fit completely in memory in compressed format. The important issues are how effective compression is at saving memory and how costly compression is by comparison to network or disk I/O.

In the best case, compression can be done in hardware, and the cost of compression will be limited only by memory-to-memory bandwidth. In this case I would expect the CCACHE to provide a substantial improvement in performance. Beyond this, however, I argue that on-line compression in software can still provide benefits to a wide class of applications if designed properly. In this section I describe an implementation of the CCACHE in software, and the performance of some benchmarks, to support this claim.

Douglis 1, 2

Douglis **Claim 14.3**

"servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

There are a number of potential benefits of using a compression cache. First, compressing a page and retaining it in memory can be cheaper than writing it to backing store. Second, reading it back from memory is likely to be much cheaper than reading it from backing store, even taking the cost of decompression into account. Finally, if the page is eventually written to backing store, it can be written in compressed format, requiring less bandwidth.

The CCACHE has some potential disadvantages as well. Most importantly, it competes with user processes (and perhaps the file system cache) for memory. Putting a page in the CCACHE has the effect of freeing up only part of a page, so more pages must be transferred from uncompressed memory to the CCACHE in order to make room for one new page. Processes will take additional page faults because they are given less memory. (This suggests that the CCACHE should be of a variable size, and should be used only when its presence improves performance rather than degrading it.) Additionally, the CCACHE adds overhead in the form of added complexity during page faults, as well as extra code and data associated with the cache.

Douglis 1, 3.1

As was mentioned above, the CCACHE is primarily targeted for mobile computers, which typically have less memory than desk-based computers of the same generation. Nevertheless, the idea of the CCACHE extends naturally to any environment that pages when applications use more virtual memory than physical memory. I have added a compression cache to the Sprite operating system [8], running on DECstation[2] 5000/200 workstations (approximately 18 SPECmarks, running a 25 MHz MIPS R3000 processor). Sprite is a particularly suitable system for the CCACHE, because it already supports the idea of trading physical memory between the virtual memory system and the file system [6]. Trading memory a third way, including the compression cache, would be a natural extension of this technique.

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1, 3.2

As a second test, I ran an application that responds to queries based on the contents of a hash table. The database containing the hash table was over 40 Mbytes, all of which was cached in memory, and the total address space of the process was over 60 Mbytes. The benchmark consisted of running a set of 9 queries against this database; the queries were run once to load the process's address space with the appropriate parts of its database and a second time to evaluate performance. The benchmark was run on the same DECstation as the previous measurements, but without the artificial restriction, so it had about 25 Mbytes of memory available. Running with a 12-Mbyte CCACHE was 32% faster than without the compression cache (9.28 minutes versus 13.7).

Douglis 1, 3.4

Abstract

This paper describes a method for trading off computation for disk or network I/O by using less expensive on-line compression. By using some memory to store data in compressed format, it may be possible to fit the working set of one or more large applications in relatively small memory. For working sets that are too large to fit in memory even when compressed, compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavior and the relative costs of compression and I/O. Measurements using Sprite on a DECstation[1] 5000/200 workstation with a local disk indicate that some memory-intensive applications running with a compression cache can run two to three times faster than on an unmodified system. Better speedups would be expected in a system with a greater disparity between the speed of its processor and the bandwidth to its backing store.

Douglis 2, Abstract

4 Design

This section describes the design and implementation of a compression cache in Sprite. Sprite is largely compatible with 4.3 BSD UNIX, but its virtual memory system has an interesting difference from most versions of UNIX: physical memory is traded dynamically between VM for application processes and the file system's buffer cache [9]. Since the compression cache must vary in size dynamically as well, Sprite provides a good framework for prototyping the compression cache. The idea of the compression cache should extend naturally to UNIX,[3] Mach, or other systems; in fact, Mach's external pager interface [7] should be an excellent foundation for future work in this area.

The target environment for this research consists of mobile computers with limited memory and network bandwidth, and with small local disks or no disks at all. Because of the limited availability of Sprite, the compression cache has been prototyped in a workstation environment, running on DECstation 5000/200 workstations, paging to a local RZ57 disk. The Sprite kernel is configurable at boot-time to allow the system to use a variable amount of physical memory, so a 32-Mbyte machine can behave as though it has as little as 12 Mbytes. About 6 Mbytes are used by the kernel for code, page tables, and some forms of tracing that cannot currently be disabled.

Douglis 2, 4

14.4 updating the boot data list.	Douglis, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Douglis, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1 (Preamble) and 1.1 above.

3 The Compression Cache: Extending Memory via Compression

Another way in which distributed systems can benefit from compression is when compression allows computers to avoid slow I/O completely. Just like compressing data on disk can allow a user to save larger files (or more files) without buying another disk, compressing data in memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

Douglis 1

The sharp leap in speedup when all pages fit in memory, as in Figure 1(b), demonstrates the potential difference between the compression cache and a system that compresses pages en route to the backing store. In practice, this improvement is not fully realized, because access patterns are not so pathological. The performance of sample applications is given after the description of a specific implementation of a compression cache for the Sprite operating system [11].

Douglis 2

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

2.1 File Systems

A number of systems have replaced individual users' *ad hoc* techniques for manual comp with a mechanism for automatically compressing some or all files. Cate and Gross use pressed files as a level in a hierarchy, with recently-accessed files being in uncompressed and less-recently-used ones compressed [5]. Since frequently-used files were never com and compression was performed in the background, the overall impact on interactive perfo (delays due to decompression) was minimal: less than 50 seconds per user per day. At th time, disk space requirements were roughly halved.

Burrows *et al.* integrated compression with Sprite LFS [12], also primarily to redu space requirements [4]. They argued that LFS is a better vehicle for compressing fil traditional file systems, since files are not overwritten in place and a change to one block a file would not cause changes to compressed data later in the file. Multiple file blocks

compressed as a unit, providing better compression than if each block were compressed se using a dynamic compression algorithm such as LZRW1 [16]. Burrows *et al.* found that compression halved disk space requirements, as in Cate and Gross's system, without the del could be incurred by decompressing a large file as a single unit. The system had an ac performance degradation when compression was performed in software, and was well-s hardware compression.

Douglis 1

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Douglis, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 15 above.

3.3 Target Applications

I next address the issue of what applications might make effective use of a compr cache. Obviously, many applications will fit into memory without the need for compr and other applications may cause excessive I/O even with compression. The applic that can best make use of a CCACHE are those that will not comfortably fit into p memory without compression but will fit if some of their pages are compressed.

One can imagine a contrived scenario in which the CCACHE would provide sign performance benefits: if an application cycles linearly through a working set that is or larger than the maximum number of pages allowed to be resident, and a least-recentl algorithm is used for page replacement, then the process will take a page fault on ea page. With the compression cache, the process will still fault on each page, but eac will be satisfied by a compression and a decompression rather than a pair of disk I/

Another scenario, as mentioned above, is the application that spreads its access formly in an address space much larger than physical memory. Although taking s memory for the CCACHE will result in additional page faults that would otherwise not the average cost of a page fault will be less with the compression cache in use as compression is much less expensive than disk I/O (say, one-third the cost) and the in the compression cache is reasonably high (say, 80–90%).

Douglis 1

Abstract

This paper describes a method for trading off computation for disk or network I/
using less expensive on-line compression. By using some memory to store data in compr
format, it may be possible to fit the working set of one or more large applications in rela
small memory. For working sets that are too large to fit in memory even when compre
compression still provides a benefit by reducing bandwidth and space requirements.

Overall, the effectiveness of this *compression cache* depends on application behavio
the relative costs of compression and I/O. Measurements using Sprite on a DECsta
5000/200 workstation with a local disk indicate that some memory-intensive applica
running with a compression cache can run two to three times faster than on an unmo
system. Better speedups would be expected in a system with a greater disparity betwee
speed of its processor and the bandwidth to its backing store.

Douglis 2

1 Introduction

Over the past decade, the processing power and physical memory size of typical comput
increased dramatically. Even as workstation memory sizes are increasing, however, a ne
nology trend is pushing toward small memories: mobile computers that are smaller tha
desk-top counterparts and are typically configured with significantly less memory. App
designers are sometimes forced to squeeze their applications to fit into available memory, a
not succeed. Therefore, in a general-purpose mobile computer, as with many computers, p
needed to enable a wider range of applications to run—as long as it can be performed effi

Douglis 2

The potential benefits of the compression cache depend on the relationship between the speed of compression and the I/O bandwidth of the system, as well as the compression ratio (anywhere from barely over 1:1 to about 4:1 in the experiments reported below). If the cost of compressing and copying a page were negligible, and pages compressed well, the compression cache could be used to give a computer the appearance of having additional physical memory. In practice, compressing and copying have costs associated with them, and the benefit of reducing traffic to the backing store is offset by the overhead of the compression cache. Overhead comes not only from the compression itself but from the additional page faults an application will experience when some memory is used for compressed pages (as well as the data structures used to support compressed pages). Furthermore, as mentioned above, not all applications compress well: for poorly suited applications, the effort to compress memory will be wasted and degrade rather than improve performance. Thus, depending on the application and the hardware environment, the benefits of reduced I/O may outweigh the costs of compression and additional faults, or vice-versa. Configuring the compression cache to improve performance in the first case while staying out of the way in the second case is an interesting, and difficult, problem.

Douglis 2

Douglis **Claim 17**

"The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Douglis, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1 and 1.2 above

> The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

Douglis 1

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Douglis, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1 and 1.2 above

Abstract

Compression has been used in numerous ways for many years, but recently two factors have combined in a way to push compression to the forefront of distributed systems. First, the disparity between processor speeds and I/O rates is ever-increasing, making it possible to perform compression in software to a much greater extent than was previously feasible. Second, the growth of new applications demanding enormous data rates, such as digital video and audio, makes hardware compression increasingly desirable. I discuss the importance of compression in various environments and describe how compression may be used not only to reduce the demand for disk space, disk bandwidth, and network bandwidth, but also to appear to extend physical memory.

Douglis 1

3 The Compression Cache: Extending Memory via Compression

Another way in which distributed systems can benefit from compression is when compression allows computers to avoid slow I/O completely. Just like compressing data on disk can allow a user to save larger files (or more files) without buying another disk, compressing data in memory can allow a user to run larger processes without buying more memory. In this case, virtual memory pages would be compressed and written back into physical memory rather than to a backing store. This idea was briefly mentioned by Appel and Li, who claimed that user-level support for compression would be more effective than generic operating system support [1]. Nevertheless, I believe that OS support for compressed virtual memory can prove useful. I refer to the technique of trading regular physical memory for data stored in compressed format as a *compression cache* (CCACHE).

Douglis 1

The Compression Cache: Using On-line Compression to Extend Physical Memory*

Douglis 2

Appendix C30
Invalidity of U.S. Patent 8,880,862 based on Douglis

1 Introduction

Over the past decade, the processing power and physical memory size of typical computers have increased dramatically. Even as workstation memory sizes are increasing, however, a new technology trend is pushing toward small memories: mobile computers that are smaller than their desk-top counterparts and are typically configured with significantly less memory. Application designers are sometimes forced to squeeze their applications to fit into available memory, and may not succeed. Therefore, in a general-purpose mobile computer, as with many computers, paging is needed to enable a wider range of applications to run—as long as it can be performed efficiently.

Douglis 2

Douglis **Claim 27**

"The method of claim 1, wherein the memory comprises: a physical memory"

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1 and 1.2 above

> Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

> Compression is already widely used to reduce demand for secondary storage and networks. I suggest that it is now feasible to use compression to reduce the demand for memory as well. The basic idea is to take some memory that would normally be used directly by an application, and use it instead to hold a larger number of pages in compressed format. I call the area used for compressed data a *compression cache*. If the pages touched by a process could not normally fit in memory, but could fit into memory when some were stored in the compression cache, then the processor would never have to write a page to backing store (onto a local disk or over a network to another computer). Even if pages must be written to backing store, compressing them beforehand reduces the amount of data transferred.

Douglis 2

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 1.1 above	

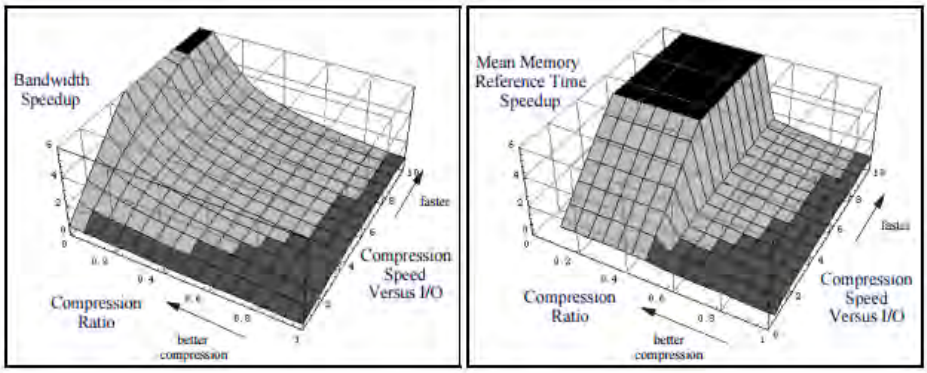
33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Douglis, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1 and 32 above.

The Compression Cache ...



(a) Transferring compressed pages to backing store. (b) Keeping compressed pages in memory.

Figure 1: Performance of compressing pages, modeled analytically. Speedups are shown as a function of the compression ratio (fraction of bytes left after compression) and the speed of compression relative to I/O. Decompression is assumed to be twice as fast as compression, as is roughly the case for algorithms such as LZRW1 [16]. There are three regions of speedup: the dark black areas at the top left show speedups that go off the top of the scale (6-fold improvement); the light areas show speedups of 1–6 relative to no compression, and the darker areas to the right show data points at which a slowdown would result.

[16] Ross N. Williams. An extremely fast ZIV-Lempel data compression algorithm. In *Data Compression Conference*, pages 362–371, April 1991.

Douglis 2

The initial prototype of the compression cache has been simplified in a number of ways. First of all, I use a fixed-size CCACHE, which is configured at kernel load time. The fixed-size cache is simpler but results in unneeded overhead for applications that would otherwise fit in memory; measurements of the effects of this are presented below. Secondly, compressed pages are written out using the same number of bytes as uncompressed pages: a full 4-Kbyte file block. In the future compressed pages will be combined into a smaller number of file blocks to save disk bandwidth. Third, I use only a single LZ-based compression algorithm [11] for all data. The compression algorithm reduces pages to 30–40% of their original size on average, depending on the application mix; I would expect application-specific compression algorithms to do much better. Finally, in order to estimate the effects of compression in a system with a large gap between processor and I/O speed, I configured the test system to page to a local disk running the original Sprite file system [6] rather than the faster Sprite LFS [9].

[11] Ross N. Williams. An extremely fast ZIV-Lempel data compression algorithm. In *Data Compression Conference*, pages 362–371, April 1991.

Douglis 1

Douglis **Claim 33**

"The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form."

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1 and 1.2 above

Compression is already widely used to reduce demand for secondary storage and networks. I suggest that it is now feasible to use compression to reduce the demand for memory as well. The basic idea is to take some memory that would normally be used directly by an application, and use it instead to hold a larger number of pages in compressed format. I call the area used for compressed data a *compression cache*. If the pages touched by a process could not normally fit in memory, but could fit into memory when some were stored in the compression cache, then the processor would never have to write a page to backing store (onto a local disk or over a network to another computer). Even if pages must be written to backing store, compressing them beforehand reduces the amount of data transferred.

Douglis 2

3 Design Considerations

Intuitively, the idea of trading processing (compression) for I/O is appealing: by and large, processors are improving in performance more quickly than I/O devices, especially disks.[2] If one can compress some pages so that they occupy little enough memory to permit all of a process's address space to reside in memory, it might be possible to avoid I/O to the backing store completely; the process would execute correspondingly faster. Note that this technique is fundamentally different from writing a dirty page into a file system that does compression, or a disk that does compression at the driver level, because compressed pages never have to go to backing store at all. Instead, compressed pages form an intermediate level in the storage hierarchy, between uncompressed pages and the backing store.

[2] Paging over a network rather than to a local disk is another issue. In some environments, it is more efficient to page over a 10-Mbps Ethernet to memory on a file server than to page to a local disk [9]. Some local-area networks, such as ATM networks (*e.g.*, Autonet [13]), provide bandwidth that is at least an order of magnitude greater than an Ethernet. However, for mobile computers on wireless networks, one can expect the disparity between processing and I/O to remain for some time.

Douglis 2

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claim 10 above.

2.2 Virtual Memory

The focus of the above systems has been disk space rather than performance. Other projects have considered ways not only to reduce disk space demands, but also to improve performance, particularly in the area of virtual memory.

Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth [14]. Because compression of the executables was performed off-line, an especially effective (but slower) compression algorithm was available. Bandwidth improved because the cost of decompression was offset by the reduction in data transferred from the disk. As a result, the performance of program invocation improved.

Atkinson *et al.* at Xerox PARC, considered the use of compression in order to reduce the cost of paging over wireless links [2]. Such paging might be needed in an environment with mobile computing devices that are too small to have local disks, such as the "tabs" advocated by Weiser [15]. The PARC researchers concentrated on read-only data, such as executables, because of the space and time overhead of performing on-line compression. Executables would be stored and transmitted in compressed format, and cached on a mobile computer in compressed format to increase the number of such executables that could be cached. Again, because compression would be performed off-line, an asymmetric compression algorithm could be used that would give very good compression ratios (with a correspondingly high overhead) while decompressing quickly. These researchers did consider on-line compression as well, resulting in a suggestion (reported by Appel and Li [1]) that pages be compressed and retained in memory. This idea, which they did not pursue extensively, is the primary theme of this paper.

Douglis 2

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

4.2 Variable Memory Allocation

Initially, the compression cache was implemented as a fixed-size region of physical memory. This was done partly for simplicity and partly because the need to vary its size was not yet apparent. In this version, the compression cache consisted of a number of pages, each divided into N fragments (In my experiments, N was defined to be 8, meaning blocks of 512 bytes with a pagesize of 4 Kbytes). When a page was compressed, the system allocated enough fragments to hold the compressed data. The fragments did not need to be allocated contiguously; instead, the compression was performed into a contiguous buffer and the compressed data was then scattered into multiple fragments. To satisfy a page fault, the fragments for a page were copied into a contiguous buffer and then decompressed.

Douglis 2

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Douglis, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Douglis, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Douglis, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 32, 33 and 49 above.	

Douglis **Claim 81**

"The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form."

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 16 and 23 above.	

Douglis **Claim 88**
"The method of claim 13, wherein the operating system comprises: a plurality of files."

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 15, 17 and 24 above.	

Douglis **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 32, 33 and 49 above.	

Douglis **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Douglis, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claim 32, 33, and 49 above.	

Douglis **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Douglis, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Douglis discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

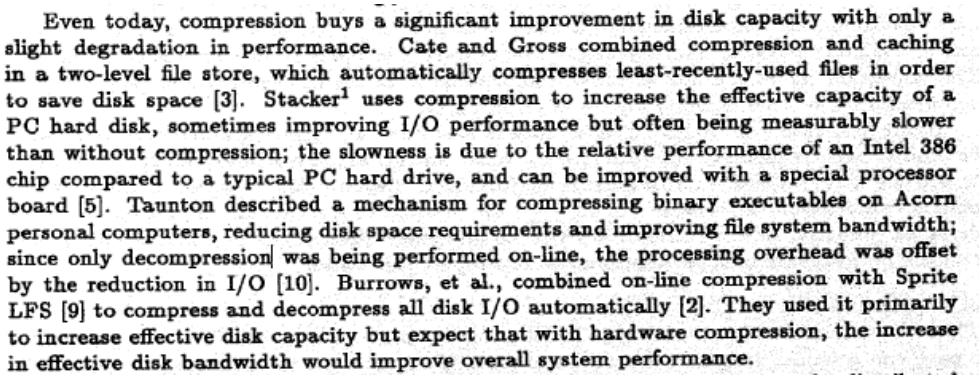
Douglis **Claim 97.2**
"and wherein the updating comprises: associating the additional compressed boot data with the boot data list."

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Douglis, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

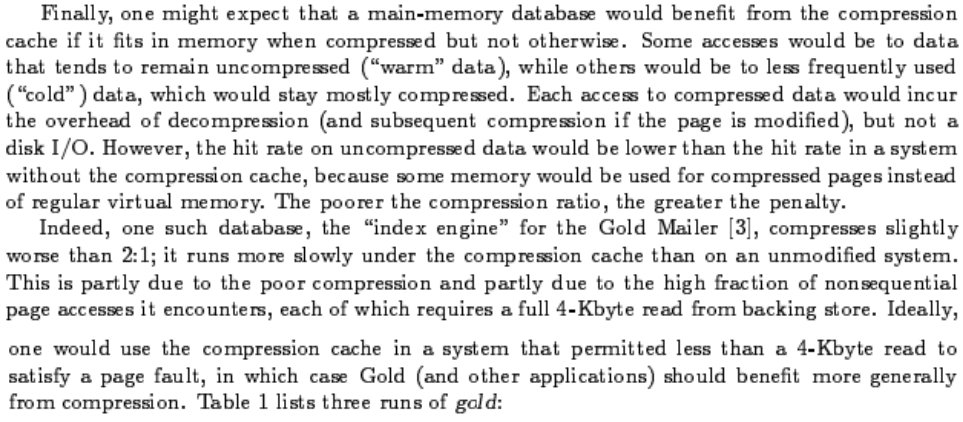
Douglis discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.



Even today, compression buys a significant improvement in disk capacity with only a slight degradation in performance. Cate and Gross combined compression and caching in a two-level file store, which automatically compresses least-recently-used files in order to save disk space [3]. Stacker[1] uses compression to increase the effective capacity of a PC hard disk, sometimes improving I/O performance but often being measurably slower than without compression; the slowness is due to the relative performance of an Intel 386 chip compared to a typical PC hard drive, and can be improved with a special processor board [5]. Taunton described a mechanism for compressing binary executables on Acorn personal computers, reducing disk space requirements and improving file system bandwidth; since only decompression was being performed on-line, the processing overhead was offset by the reduction in I/O [10]. Burrows, et al., combined on-line compression with Sprite LFS [9] to compress and decompress all disk I/O automatically [2]. They used it primarily to increase effective disk capacity but expect that with hardware compression, the increase in effective disk bandwidth would improve overall system performance.

Douglis 1



Finally, one might expect that a main-memory database would benefit from the compression cache if it fits in memory when compressed but not otherwise. Some accesses would be to data that tends to remain uncompressed ("warm" data), while others would be to less frequently used ("cold") data, which would stay mostly compressed. Each access to compressed data would incur the overhead of decompression (and subsequent compression if the page is modified), but not a disk I/O. However, the hit rate on uncompressed data would be lower than the hit rate in a system without the compression cache, because some memory would be used for compressed pages instead of regular virtual memory. The poorer the compression ratio, the greater the penalty.

Indeed, one such database, the "index engine" for the Gold Mailer [3], compresses slightly worse than 2:1; it runs more slowly under the compression cache than on an unmodified system. This is partly due to the poor compression and partly due to the high fraction of nonsequential page accesses it encounters, each of which requires a full 4-Kbyte read from backing store. Ideally, one would use the compression cache in a system that permitted less than a 4-Kbyte read to satisfy a page fault, in which case Gold (and other applications) should benefit more generally from compression. Table 1 lists three runs of *gold*:

Douglis 2

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Douglis, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.4.1, 5.6, and 8.6 above.

Compression is already widely used to reduce demand for secondary storage and networks. I suggest that it is now feasible to use compression to reduce the demand for memory as well. The basic idea is to take some memory that would normally be used directly by an application, and use it instead to hold a larger number of pages in compressed format. I call the area used for compressed data a *compression cache*. If the pages touched by a process could not normally fit in memory, but could fit into memory when some were stored in the compression cache, then the processor would never have to write a page to backing store (onto a local disk or over a network to another computer). Even if pages must be written to backing store, compressing them beforehand reduces the amount of data transferred.

The potential benefits of the compression cache depend on the relationship between the speed of compression and the I/O bandwidth of the system, as well as the compression ratio (anywhere from barely over 1:1 to about 4:1 in the experiments reported below). If the cost of compressing and copying a page were negligible, and pages compressed well, the compression cache could be used to give a computer the appearance of having additional physical memory. In practice, compressing and copying have costs associated with them, and the benefit of reducing traffic to the backing store is offset by the overhead of the compression cache. Overhead comes not only from the compression itself but from the additional page faults an application will experience when some memory is used for compressed pages (as well as the data structures used to support compressed pages). Furthermore, as mentioned above, not all applications compress well: for poorly suited applications, the effort to compress memory will be wasted and degrade rather than improve performance. Thus, depending on the application and the hardware environment, the benefits of reduced I/O may outweigh the costs of compression and additional faults, or vice-versa. Configuring the compression cache to improve performance in the first case while staying out of the way in the second case is an interesting, and difficult, problem.

Douglis 2

3 Design Considerations

Intuitively, the idea of trading processing (compression) for I/O is appealing: by and large, processors are improving in performance more quickly than I/O devices, especially disks.[2] If one can compress some pages so that they occupy little enough memory to permit all of a process's address space to reside in memory, it might be possible to avoid I/O to the backing store completely; the process would execute correspondingly faster. Note that this technique is fundamentally different from writing a dirty page into a file system that does compression, or a disk that does compression at the driver level, because compressed pages never have to go to backing store at all. Instead, compressed pages form an intermediate level in the storage hierarchy, between uncompressed pages and the backing store.

Keeping compressed pages in memory does not obviate the need for a backing store, however. It is possible for the collective address space of all running processes not to fit in memory even after compression. And even if they fit, it might be desirable to move some "old" pages to backing store in order to have more memory available for actively-used pages. In either case, pages could be transferred to backing store in compressed format, reducing the demand for bandwidth. This technique would be similar to paging into a file system or disk that does its own compression. The differences are:

Reduced I/O. With the compression cache, some pages might be faulted upon before being written to backing store. Those pages would no longer need to be written.

Variable memory allocation. By making compressed pages an explicit part of the memory hierarchy, the system can dynamically vary the amount of memory used for uncompressed pages, compressed pages, and file blocks. This is necessary to avoid impacting applications that do not need to compress pages, as discussed below in Section 4.2.

Douglis 2

Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

Douglis **Claim 107**

"The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Douglis, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 1.1, 5, 9.1 and 8 above.

> Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Douglis, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Douglis discloses this limitation:

See Claims 5.1, 8.1 and 9.1 above.

> Instead, memory for the compression cache is now treated as a variable-sized circular buffer. Physical pages are mapped into the kernel's virtual address space, one after another, eventually wrapping around to the start of the range of addresses for the compression cache. There is a notion of the *oldest* physical page—the one added to the cache the longest time ago—and new pages, which have been added most recently and may not contain data yet. Physical pages are added to one end of the queue and normally removed from the other end. (They may be removed from the middle if no clean pages are availabled at the oldest end.) When VM pages are compressed, they are compressed directly into the first unused region within the compression cache, following the last page that had been added to the cache. Before each page there is a small header that describes the page, the size it compressed to, whether it contains dirty data, a link to the next page in the cache, and other information.

Douglis 2

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

The publication Grove "System Administration," LINUX, Mar 1998. ("Grove") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Grove, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this claim limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like "`cp zImage /dev/fd0`") or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Grove, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this claim limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page _ for more information.

Grove, 4.9.1

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Grove, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this claim limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0''") or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Grove, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this claim limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

1.4.1 updating the boot data list,	Grove, as evidenced by the example citations below, discloses "updating the boot data list,"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this claim limitation:

4.9.1 Upgrading the kernel

Upgrading the kernel is a matter of obtaining the kernel sources and compiling them. This is generally a painless procedure, but you can run into problems if you try to upgrade to a development kernel, or upgrade to a new kernel version. The version of a kernel has two parts, the kernel version and patchlevel. As of the time of this writing, the latest stable kernel is version `2.0.33`. The `2.0` is the kernel version and `33` is the patch level. Odd-numbered kernel versions like `2.1` are development kernels. Stay away from development kernels unless you want to live dangerously! As a general rule, you should be able to upgrade easily to another patch level, but upgrading to a new version requires the upgrade of system utilities which interact closely with the kernel.

The Linux kernel sources may be retrieved from any of the Linux FTP sites (see page _ for a list). On `sunsite.unc.edu`, for instance, the kernel sources are found in `/pub/Linux/kernel`, organized into subdirectories by version number.

Kernel sources are released as a gzipped tar file. For example, the file containing the 2.0.33 kernel sources is `linux-2.0.33.tar.gz`.

Kernel sources are unpacked in the `/usr/src` directory, creating the directory `/usr/src/linux`. It is common practice for `/usr/src/linux` to be a soft link to another directory which contains the version number, like `/usr/src/linux-2.0.33`. This way, you can install new kernel sources and test them out before removing the old kernel sources. The commands to create the kernel directory link are

```
# cd /usr/src
# mkdir linux-2.0.33
# rm -r linux
# ln -s linux-2.0.33 linux
# cat xzf linux-2.0.33.tar.gz
```

Grove, 4.9.1

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like "`cp zImage /dev/fd0`") or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Grove, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this claim limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like "`cp zImage /dev/fd0`") or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Grove, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Grove, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Grove, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.4.1, and 2 above.	

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

5 (Preamble) A method for booting a computer system, the method comprising:	Grove, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1-1.4.2 above.	

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Grove, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Grove, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Grove, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Grove, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Grove, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

5.6 updating the boot data list;	Grove, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Grove, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1-1.3 above.	

6 (Preamble) A system comprising: a processor;	Grove, as evidenced by the example citations below, discloses "a system comprising: a processor:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0``) or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

6.1 a memory; and	Grove, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Grove, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.3 wherein the processor is configured:	Grove, as evidenced by the example citations below, discloses "wherein the processor is configured:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Grove, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Grove, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Grove, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Grove, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

Grove Claim 6.7
"to update the boot data list."

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Grove, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Grove, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Grove, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Grove, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Grove, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Grove, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.6 updating the boot data list,	Grove, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

Grove Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Grove, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Grove, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

9.2 storing the additional portion of the operating system in the first memory, and	Grove, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 8.1 above.	

Grove Claim 9.2
"storing the additional portion of the operating system in the first memory"

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Grove, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Grove, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 9.1 above.	

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Grove, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Grove, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

11.2 accessing the loaded boot data in compressed form from the memory;	Grove, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Grove, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Grove, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page ⌴ for more information.

Grove, 4.9.1

11.5 updating the boot data list.	Grove, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Grove, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Grove, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Grove, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Grove, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Grove, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C31
Invalidity of U.S. Patent 8,880,862 based on Grove

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Grove, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Grove, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.1 and 1.2 above.	

14.2 loading the boot data into a memory; and	Grove, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Grove, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

See Claims 1.2 and 1.3 above.

14.4 updating the boot data list.	Grove, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 1.4.1 above.	

15. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system.	Grove, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1 (Preamble) and 1.1 above.	

16. The method of claim 14, wherein the operating system comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 14, wherein the operating system comprises: a plurality of files."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

17. The method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program.	Grove, as evidenced by the example citations below, discloses "the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 15 above.	

19.1 The method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:	Grove, as evidenced by the example citations below, discloses "the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 14, wherein the request for the boot data comprises: a request to access boot data that is not associated with the boot data list, and wherein the updating comprises:), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0``) or install the image so LILO will boot from your hard drive. See page ␣ for more information.

Grove, 4.9.1

19.2 associating the accessed boot data that is not associated with the boot data list to the boot data list.	Grove, as evidenced by the example citations below, discloses "associating the accessed boot data that is not associated with the boot data list to the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, associating the accessed boot data that is not associated with the boot data list to the boot data list.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 2 above.	

23. The method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 16 above.	

24. The method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system.
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the portion of the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 15 above.	

27. The method of claim 1, wherein the memory comprises: a physical memory.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein the memory comprises: a physical memory."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page ⌴ for more information.

Grove, 4.9.1

28. The method of claim 1, wherein the operating system comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 16 above.	

29. The method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the boot data comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15 and 17 above.	

31. The method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein the accessing comprises: accessing the loaded portion of the boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page _ for more information.

Grove, 4.9.1

32. The method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein a form of dictionary encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like "`cp zImage /dev/fd0`") or install the image so LILO will boot from your hard drive. See page ⌟ for more information.

Grove, 4.9.1

33. The method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form.	Grove, as evidenced by the example citations below, discloses "the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 1, wherein Lempel-Ziv encoding was utilized to encode the portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 32 above.	

35. The method of claim 5, wherein the compressed boot data represents a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

36. The method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the compressed boot data comprises: a program code associated with an operating system of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15 and 17 above.	

39. The method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the loading comprises: loading the stored compressed boot data from the first memory to a second memory, and wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0``) or install the image so LILO will boot from your hard drive. See page ⌴ for more information.

Grove, 4.9.1

40. The method of claim 36, wherein the operating system comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 36, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 36, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

43. The method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the accessing comprises: accessing the loaded compressed boot data via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 31 above.	

44. The method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 32 above.	

45. The method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 32 and 33 above.	

47. The system of claim 6, wherein the boot data in the compressed form represents a plurality of files.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

48. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

49. The system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: an encoder configured to compress the boot data to provide the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 10 above.	

50. The system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, further comprising: a decoder configured to decompress the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Grove discloses this limitation:

In order to compile the kernel, you must have the `gcc` C compiler installed on your system. `gcc` version 2.6.3 or a more recent version is required to compile the 2.0 kernel.

First `cd` to `/usr/src/linux`. The command `make config` prompts you for a number of configuration options. This is the step where you select the hardware that your kernel will support. The biggest mistake to avoid is not including support for your hard disk controller. Without the correct hard disk support in the kernel, the system won't even boot. If you are unsure about what a kernel option means, a short description is available by pressing ? and Enter.

Next, run the command `make dep` to update all of the source dependencies. This is an important step. `make clean` removes old binary files from the kernel source tree.

The command `make zImage` compiles the kernel and writes it to `/usr/src/linux/arch/i386/boot/zImage`. Linux kernels on Intel systems are always compressed. Sometimes the kernel you want to compile is too large to be compressed with the compression system that `make zImage` uses. A kernel which is too large will exit the kernel compile with the error message: `Kernel Image Too Large`. If this happens, try the command `make bzImage`, which uses a compression system that supports larger kernels. The kernel is written to `/usr/src/linux/arch/i386/boot/bzImage`.

Once you have the kernel compiled, you need to either copy it to a boot floppy (with a command like ``cp zImage /dev/fd0'') or install the image so LILO will boot from your hard drive. See page _ for more information.

Grove, 4.9.1

51. The system of claim 6, wherein the first memory comprises: a physical memory.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, wherein the first memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the first memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 27 and 39 above.	

52. The system of claim 6, wherein the boot data the compressed form comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data the compressed form comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

53. The system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program.	Grove, as evidenced by the example citations below, discloses "the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the system of claim 6, wherein the boot data in the compressed form comprises: a program code associated with an operating system of the system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

59. The method of claim 8, wherein the operating system in the compressed form represents a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

60. The method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system.	Grove, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

63. The method of claim 8, wherein the second memory comprises: a physical memory.	Grove, as evidenced by the example citations below, discloses "the method of claim 8, wherein the second memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the second memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 27 and 38 above.	

64. The method of claim 8, wherein the operating system comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

65. The method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program.	Grove, as evidenced by the example citations below, discloses "the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the operating system in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

67. The method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access.	Grove, as evidenced by the example citations below, discloses "the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 8, wherein the accessing comprises: accessing the loaded first portion from the second memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 31 above.	

71. The method of claim 11, wherein the boot data in the compressed form represents a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

72. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

75. The method of claim 11, wherein the memory comprises: a physical memory.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 27 above.	

76. The method of claim 11, wherein the operating system comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

77. The method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the boot data in the compressed form comprises: a program code associated with the operating system and an application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

79. The method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein the accessing comprises: accessing the boot data in the compressed form from the memory via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 31 above.	

80. The method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein a form of dictionary encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 32, 33 and 49 above.	

81. The method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form.	Grove, as evidenced by the example citations below, discloses "the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 11, wherein Lempel-Ziv encoding was utilized to encode the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 32, 33 and 49 above.	

83. The method of claim 13, wherein the boot data in the compressed form represents a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form represents a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form represents a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

84. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

87. The method of claim 13, wherein the memory comprises: a physical memory.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein the memory comprises: a physical memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the memory comprises: a physical memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 27 above.	

88. The method of claim 13, wherein the operating system comprises: a plurality of files.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein the operating system comprises: a plurality of files"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the operating system comprises: a plurality of files), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 16 and 23 above.	

89. The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 15, 17 and 24 above.	

Grove **Claim 89**

"The method of claim 13, wherein the boot data in the compressed form comprises: a program code associated with the operating system and application program."

91. The method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein the accessing comprises: accessing the loaded boot data in the compressed form via direct memory access), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 31 above.	

92. The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 32, 33 and 49 above.	

Grove **Claim 92**

"The method of claim 13, wherein a form of dictionary encoding was utilized to encode the compressed boot data."

93. The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data.	Grove, as evidenced by the example citations below, discloses "the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claim 32, 33, and 49 above.	

Grove **Claim 93**

"The method of claim 13, wherein Lempel-Ziv encoding was utilized to encode the compressed boot data."

97.1 The method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list,	Grove, as evidenced by the example citations below, discloses "the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, further comprising: accessing additional compressed boot data that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 9.1-9.3 and 19 above.	

97.2 and wherein the updating comprises: associating the additional compressed boot data with the boot data list.	Grove, as evidenced by the example citations below, discloses "and wherein the updating comprises: associating the additional compressed boot data with the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, and wherein the updating comprises: associating the additional compressed boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.4.1, 2, 4, 5.6, and 8.6 above.	

98. The method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list.	Grove, as evidenced by the example citations below, discloses "the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 5, wherein the updating comprises: disassociating non accessed boot data from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

107. The method of claim 2, further comprising: storing the updated boot list in a non-volatile memory.	Grove, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: storing the updated boot list in a non-volatile memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.4.1, 5.6, and 8.6 above.	

108. The method of claim 2, further comprising: compressing at least a portion of the additional boot data.	Grove, as evidenced by the example citations below, discloses "the method of claim 2, further comprising: compressing at least a portion of the additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 2, further comprising: compressing at least a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 1.1, 5, 9.1 and 8 above.	

109. The method of claim 108, further comprising: storing the compressed additional boot data.	Grove, as evidenced by the example citations below, discloses "the method of claim 108, further comprising: storing the compressed additional boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, the method of claim 108, further comprising: storing the compressed additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Grove discloses this limitation: *See* Claims 5.1, 8.1 and 9.1 above.	

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

The publication Jones, The Microsoft Interactive TV system: An Experience Report, July 1997 ("Jones") invalidates claims 1-6, 8-11, 13-17, 19, 23-24, 27-29, 31-33, 35-36, 39-40, 43-45, 47-53, 59-60, 63-65, 67, 71-72, 75-77, 79-81, 83-84, 87-89, 91-93, 97-98, and 107-109 of United States Patent No. 8,880,862 ("the '862 Patent") pursuant to 35 U.S.C. § 102 and/or 35 U.S.C. § 103 either alone or in combination with other prior art references, and/or in combination with the knowledge of a person of ordinary skill.

The analysis provided in this chart may in some instances uses Realtime's proposed (or implied) claim constructions, and Apple reserves all rights to challenge these proposed (or implied) constructions. To the extent any of the charted prior art should fail to disclose an element of any claims of the '862 Patent, Apple reserves the right to rely upon the knowledge of one skilled in the art, or any other disclosed prior art, alone or in combination, whether produced by Apple or by Realtime, to show the element and thereby invalidate those claims. Citations given in the chart below are merely representative of the respective elements and are not meant to be exhaustive.

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

1 (Preamble) A method for providing accelerated loading of an operating system in a computer system, the method comprising:	Jones, as evidenced by the exemplary citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accelerated loading of an operating system in a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

> "The MMOSA real-time kernel was developed as a cooperative effort between members of the operating systems research group within Microsoft Research and a team of development staff committed to building a small, embedded-systems kernel meeting the needs of interactive television. MMOSA is the internal name for the ITV kernel: it stands for MultiMedia Operating System Architecture."

Jones, §4.

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, § 10.

1.1 loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;	Jones, as evidenced by the example citations below, discloses "loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading a portion of boot data in a compressed form that is associated with a portion of a boot data list for booting the computer system into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

> "The MMOSA real-time kernel was developed as a cooperative effort between members of the operating systems research group within Microsoft Research and a team of development staff committed to building a small, embedded-systems kernel meeting the needs of interactive television. MMOSA is the internal name for the ITV kernel: it stands for MultiMedia Operating System Architecture."

Jones, §4.

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, § 10.

> "Kanji font not in ROM. The set-top boxes we built for the NTT trial contained 1/2 MB of EPROM. This was sufficient for storing the compressed boot image but wasn't large enough to hold a Kanji font."

Jones, § 15.

1.2 accessing the loaded portion of the boot data in the compressed form from memory;	Jones, as evidenced by the example citations below, discloses "accessing the loaded portion of the boot data in the compressed form from memory;"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the boot data in the compressed form from memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, 14 at § 10.

> "Kanji font not in ROM. The set-top boxes we built for the NTT trial contained 1/2 MB of EPROM. This was sufficient for storing the compressed boot image but wasn't large enough to hold a Kanji font."

Jones, § 15.

1.3 decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and	Jones, as evidenced by the example citations below, discloses "decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form; and"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, 14 at § 10.

1.4.1 updating the boot data list,	Jones, as evidenced by the example citations below, discloses "updating the boot data list,"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

1.4.2 wherein the decompressed portion of boot data comprises a portion of the operating system.	Jones, as evidenced by the example citations below, discloses "wherein the decompressed portion of boot data comprises a portion of the operating system."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the decompressed portion of boot data comprises a portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

> "The MMOSA real-time kernel was developed as a cooperative effort between members of the operating systems research group within Microsoft Research and a team of development staff committed to building a small, embedded-systems kernel meeting the needs of interactive television. MMOSA is the internal name for the ITV kernel: it stands for MultiMedia Operating System Architecture."

Jones, §4.

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, 14 at § 10.

2. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list.	Jones, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

3. The method of claim 1, wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list.	Jones, as evidenced by the example citations below, discloses "wherein the updating comprises: removing an association of additional boot data that is associated with the boot data list from the boot data list."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein removing an association of additional boot data that is associated with the boot data list from the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

4. The method of claim 1, wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data.	Jones, as evidenced by the example citations below, discloses "wherein the updating comprises: associating additional boot data with the boot data list; and compressing a portion of the additional boot data."
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating wherein associating additional boot data with the boot data list; and compressing a portion of the additional boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claims 1.4.1, and 2 above.	

5 (Preamble) A method for booting a computer system, the method comprising:	Jones, as evidenced by the example citations below, discloses "a method for booting a computer system, the method comprising:"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, booting a computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claims 1-1.4.2 above.

See also

> "The MMOSA real-time kernel was developed as a cooperative effort between members of the operating systems research group within Microsoft Research and a team of development staff committed to building a small, embedded-systems kernel meeting the needs of interactive television. MMOSA is the internal name for the ITV kernel: it stands for MultiMedia Operating System Architecture."

Jones, §4.

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, 14 at § 10.

5.1 storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory;	Jones, as evidenced by the example citations below, discloses "storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing boot data in a compressed form that is associated with a portion of a boot data list in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

5.2 loading the stored compressed boot data from the first memory;	Jones, as evidenced by the example citations below, discloses "loading the stored compressed boot data from the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the stored compressed boot data from the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

5.3 accessing the loaded compressed boot data;	Jones, as evidenced by the example citations below, discloses "accessing the loaded compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.2 above.	

5.4 decompressing the accessed compressed boot data;	Jones, as evidenced by the example citations below, discloses "decompressing the accessed compressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed compressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.3 above.	

5.5 utilizing the decompressed boot data to at least partially boot the computer system;	Jones, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, § 10.

5.6 updating the boot data list;	Jones, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

5.7 wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.	Jones, as evidenced by the example citations below, discloses "wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the loading, the accessing, and the decompressing occur within a period of time which is less than a time to access the boot data from the first memory if the boot data was stored in the first memory in an uncompressed form.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claims 1-1.3 above.	

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

6 (Preamble) A system comprising: a processor;	Jones, as evidenced by the example citations below, discloses "a system comprising: a processor:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a system comprising: a processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.1 a memory; and	Jones, as evidenced by the example citations below, discloses "a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a memory.), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.2 a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor	Jones, as evidenced by the example citations below, discloses "a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a second memory configured to store boot data in a compressed form for booting the system and a logic code associated with the processor), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claims 1.1 and 1.2 above.	

6.3 wherein the processor is configured:	Jones, as evidenced by the example citations below, discloses "wherein the processor is configured:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the processor is configured), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.	

6.4 to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory,	Jones, as evidenced by the example citations below, discloses "to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to load a portion of the boot data in the compressed form that is associated with a boot data list used for booting the system into the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

6.5 to access the loaded portion of the boot data in the compressed form,	Jones, as evidenced by the example citations below, discloses "to access the loaded portion of the boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to access the loaded portion of the boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.2 above.	

6.6 to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data, and	Jones, as evidenced by the example citations below, discloses "to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to decompress the accessed portion of the boot data in the compressed form at a rate that decreases a boot time of the system relative to booting the system with uncompressed boot data), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.3 above.	

6.7 to update the boot data list.	Jones, as evidenced by the example citations below, discloses "to update the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, to update the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

Jones Claim 6.7
"to update the boot data list."

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

8 (Preamble) A method of loading an operating system for booting a computer system, comprising:	Jones, as evidenced by the example citations below, discloses "A method of loading an operating system for booting a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method of loading an operating system for booting a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1 (Preamble) above.	

8.1 storing a portion of the operating system in a compressed form in a first memory;	Jones, as evidenced by the example citations below, discloses "storing a portion of the operating system in a compressed form in a first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing a portion of the operating system in a compressed form in a first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 6.2 above.	

8.2 loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list;	Jones, as evidenced by the example citations below, discloses "loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the portion of the operating system from the first memory to a second memory, the portion of the operating system being associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

8.3 accessing the loaded portion of the operating system from the second memory in the compressed form;	Jones, as evidenced by the example citations below, discloses "accessing the loaded portion of the operating system from the second memory in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded portion of the operating system from the second memory in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.2 above.	

8.4 decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system;	Jones, as evidenced by the example citations below, discloses "decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed portion of the operating system to provide a decompressed portion of the operating system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claims 1.3 and 1.4.2 above.	

8.5 utilizing the decompressed portion of the operating system to at least partially boot the computer system; and	Jones, as evidenced by the example citations below, discloses "utilizing the decompressed portion of the operating system to at least partially boot the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed portion of the operating system to at least partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claims 1.3 and 1.4.2 above.

See also

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, § 10.

8.6 updating the boot data list,	Jones, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

Jones Claim 8.6
"updating the boot data list"

8.7 wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form.	Jones, as evidenced by the example citations below, discloses "wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the portion of the operating system is accessed and decompressed at a rate that is faster than accessing the loaded portion of the operating system from the first memory if the portion of the operating system was to be stored in the first memory in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claims 1-1.3 and 5.7 above.	

9.1 The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list; and	Jones, as evidenced by the example citations below, discloses "the method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, compressing an additional portion of the operating system that is not associated with the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claim 1.1 above.

See also

> "The MMOSA real-time kernel was developed as a cooperative effort between members of the operating systems research group within Microsoft Research and a team of development staff committed to building a small, embedded-systems kernel meeting the needs of interactive television. MMOSA is the internal name for the ITV kernel: it stands for MultiMedia Operating System Architecture."

Jones, §4.

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, § 10.

> "Kanji font not in ROM. The set-top boxes we built for the NTT trial contained 1/2 MB of EPROM. This was sufficient for storing the compressed boot image but wasn't large enough to hold a Kanji font."

Jones, § 15.

Jones Claim 9.1

"The method of claim 8, further comprising: compressing an additional portion of the operating system that is not associated with the boot data list"

9.2 storing the additional portion of the operating system in the first memory, and	Jones, as evidenced by the example citations below, discloses "storing the additional portion of the operating system in the first memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, storing the additional portion of the operating system in the first memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 8.1 above.	

9.3 wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system.	Jones, as evidenced by the example citations below, discloses "wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the utilizing comprises: utilizing the stored additional portion of the operating system to at least further partially boot the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 8.5 above.	

10. The method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder.	Jones, as evidenced by the example citations below, discloses "the method of claim 9, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder."

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, wherein the compressing comprises: compressing the additional portion of the operating system with a data compression encoder), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claim 9.1 above.

> "Custom video and audio hardware for the set-top box contains a MPEG-2 decoder, NTSC (the U.S. and Japanese analog television encoding standard) encoders & decoders, a tuner, and an audio mixer."

Jones, §3.

> "Everything from server machines and ATM switches to real-time MPEG-2 encoders and OS software was precisely duplicated."

Jones, §11.

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

11 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Jones, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising:"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1 (Preamble) above.	

11.1 loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system;	Jones, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device into a memory upon initialization of the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

11.2 accessing the loaded boot data in compressed form from the memory;	Jones, as evidenced by the example citations below, discloses "accessing the loaded boot data in compressed form from the memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in compressed form from the memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.2 above.	

11.3 decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Jones, as evidenced by the example citations below, discloses "decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.3 above.	

11.4 utilizing the decompressed boot data to load at least a portion of the operating system for the computer system; and	Jones, as evidenced by the example citations below, discloses "utilizing the decompressed boot data to load at least a portion of the operating system for the computer system"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, utilizing the decompressed boot data to load at least a portion of the operating system for the computer system), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, 14 at § 10.

11.5 updating the boot data list.	Jones, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

13 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Jones, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1 (Preamble) above.	

13.1 loading boot data in a compressed form that is associated with a boot data list from a boot device;	Jones, as evidenced by the example citations below, discloses "loading boot data in a compressed form that is associated with a boot data list from a boot device"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading boot data in a compressed form that is associated with a boot data list from a boot device), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

13.2 accessing the loaded boot data in the compressed form;	Jones, as evidenced by the example citations below, discloses "accessing the loaded boot data in the compressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing the loaded boot data in the compressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.2 above.	

13.3 decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form;	Jones, as evidenced by the example citations below, discloses "decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, decompressing the accessed boot data in the compressed form at a rate that decreases a time to load the operating system relative to loading the operating system with the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.3 above.	

13.4 updating the boot data list.	Jones, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	

Appendix C32
Invalidity of U.S. Patent 8,880,862 based on Jones

14 (Preamble) A method for providing accelerated loading of an operating system in a computer system, comprising:	Jones, as evidenced by the example citations below, discloses "a method for providing accelerated loading of an operating system in a computer system, comprising"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, a method for providing accelerated loading of an operating system in a computer system, comprising), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1 (Preamble) above.	

14.1 accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list;	Jones, as evidenced by the example citations below, discloses "accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, accessing boot data for booting the computer system, wherein a portion of the boot data is in a compressed form and is associated with a boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claims 1.1 and 1.2 above.

See also

> "The MMOSA real-time kernel was developed as a cooperative effort between members of the operating systems research group within Microsoft Research and a team of development staff committed to building a small, embedded-systems kernel meeting the needs of interactive television. MMOSA is the internal name for the ITV kernel: it stands for MultiMedia Operating System Architecture."

Jones, §4.

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, §10.

14.2 loading the boot data into a memory; and	Jones, as evidenced by the example citations below, discloses "loading the boot data into a memory"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, loading the boot data into a memory), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.1 above.	

14.3 servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form; and	Jones, as evidenced by the example citations below, discloses "servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form"

To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, servicing a request for the boot data from the computer system to access the loaded compressed boot data and to decompress the accessed compressed boot data at a rate that decreases a boot time of the operating system relative to loading the operating system utilizing the boot data in an uncompressed form), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions.

Jones discloses this limitation:

See Claims 1.2 and 1.3 above.

See also

> "Set-top boxes boot as follows. First, a small boot loader in EPROM is run. This decompresses a boot image also in the ROM and jumps to its entry point. The ROM boot image contains the MMOSA kernel, the network protocol stack and drivers, and a network boot program. The network boot program sends a DHCP request. It waits for a DHCP reply containing not only the usual IP address, but also some MITV-specific extended results such as sets of IP addresses of Class Store servers and a boot filename to use. It then loads a new boot image from a Class Store via TFTP and jumps to it."

Jones, 14 at § 10.

14.4 updating the boot data list.	Jones, as evidenced by the example citations below, discloses "updating the boot data list"
To the extent that Realtime contends this reference does not explicitly disclose this element of this claim (for example, updating the boot data list), Apple contends that one of skill in the art would understand the operation of booting a computer system to include the element that is missing similar to the manner in which the patentee relied upon such knowledge of skill in the art during prosecution. See Sections VI. and VII. of Apple's Invalidity Contentions. Jones discloses this limitation: *See* Claim 1.4.1 above.	