Apple Inc. et al. (Petitioners)
v.
Memory Integrity, LLC (Patent Owner)

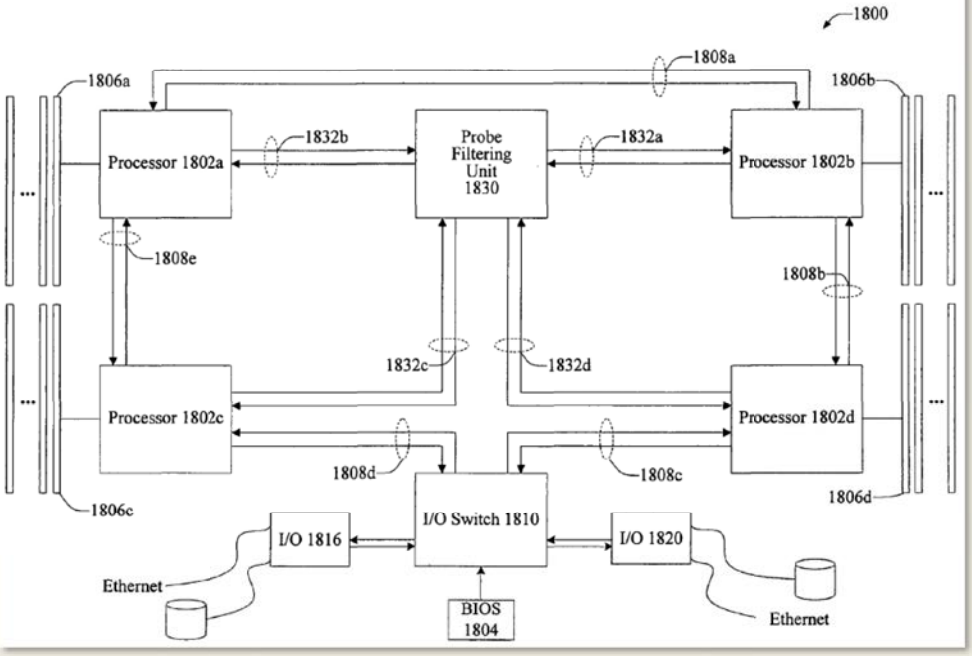
Original Claims Demonstratives
Trial Nos. IPR2015-00159 and -00163
U.S. Patent No. 7,296,121

Before Hon. J. S. BISK, N. T. POWELL, and K. BEGLEY,
Administrative Patent Judges

The ’121 Patent

The present invention generally relates to accessing data in a multiple processor system. More specifically, the present invention provides techniques for reducing memory transaction traffic in a multiple processor system.

Figure 18



’121 Patent, FIG 18.
IPR2015-00159, Paper No. 6, Petition at 19.

3

The ’121 Patent: Claim 1

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, the computer system further comprising a probe filtering unit which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories.

’121 Patent, Claim 1.

IPR2015-00159, Paper No. 6,
Petition at 23.

The Claimed "States"

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, the computer system further comprising a probe filtering unit which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of **states** associated with selected ones of the cache memories.

'121 Patent, Claim 1.

IPR2015-00159, Paper No. 6, Petition at 23.

The Claimed "States"

. . . probe filtering information representative of **states** associated with selected ones of the cache memories.

'121 Patent, Claim 1.

Applied by Petitioner (From Institution Decision)	Patent Owner Construction
"the term is not limited to cache coherence protocol states and is broad enough to include the condition of presence—i.e., what is stored in cache memory"	"cache coherency states, and . . . mere presence is not a 'state.'"

IPR2015-00159, Paper No. 12, Institution Decision at 10.

IPR2015-00159, Paper No. 25, PO Response at 11.

The Claimed "States"

- **Intrinsic Evidence**
  - No intrinsic evidence to support MI's construction
  - No express disclaimer. *See* IPR2015-00159, Institution Decision at 9 (citing '121 Patent at 14:30–36).
- **Extrinsic Evidence**
  - Microsoft Computer Dictionary cited in Institution Decision. *See* Institution Decision at 10.
  - Chaiken cited in Petition. *See* Petition at 10.
  - Webster's Dictionary cited in Petition. *See* Petition at 9.

The Claimed States "States": Intrinsic Evidence – Does not Support MI's Construction

States of "Memory Lines"

associated with a selected set of memory lines. In one example, the coherence directory **701** includes state information **713**, dirty data owner information **715**, and an occupancy vector **717** associated with the memory lines **711**. In some embodiments, the memory line states are modified, owned, shared, and invalid.

'121 Patent (Ex. 1001), 13:54-59.

ciated with them. By contrast, because the cache coherence directory provides information about where memory lines are cached as well as their states, probes only need be directed toward the clusters in which the requested memory line is cached. The state of a particular cached line will

'121 Patent (Ex. 1001), 19:36-40.

VS.

States of "Cache Memories"

transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories.

'121 Patent (Ex. 1001), Claim 1.

IPR2015-00159, Paper No. 35, Petitioner Reply at 2-3.

The Claimed "States"
Intrinsic Evidence – Does not Support MI's Construction

Intrinsic Evidence Identified in POPR (FIGS. 7 & 8)

"Invalid," "Shared," "Owned," "Modified." Ex. 1001, Figs. 7, 8. Additionally, the description of Figures 7 and 8 further demonstrate that the relevant "states" are cache coherence protocol states. Ex. 1001 at 13:44-15:19. Notably, Petitioners

IPR2015-00159, Paper No. 11, Preliminary Response at 15.

Figures 7 and 8 are strongly illustrative that the '121 Patent uses "state" to mean cache coherence protocol states. In particular, in describing Figure 7, the

IPR2015-00159, Paper No. 25, Patent Owner Response at 6.

FIG. 7 & 8 Not Limiting

any particular cache coherence protocol's set of states)." *Id.* at 15. Patent Owner also points to Figures 7 and 8, which show similar states in diagram form. *Id.* The '121 patent, however, sets these examples within broad language stating that "particular implementations may use a different set of states" and "[t]he techniques of the present invention can be used with a variety of different possible memory line states." Ex. 1001, 14:30–36. We,

IPR2015-00159, Paper No. 18, Institution Decision at 9.

The Claimed "States": Intrinsic Evidence - No Express Disclaimer

"[T]he PTO should only limit the claim based on the specification or prosecution history when those sources **expressly disclaim** the broader definition."

In re Bigio, 381 F.3d 1320, 1325 (Fed. Cir. 2004)
IPR2015-00159, Paper No. 35, Petitioner Reply at 2.

11

The Claimed "States": Extrinsic - Supports Board's Construction

Construction of "States" Applied by Petitioner (From Institution Decision)

"the term is not limited to cache coherence protocol states and is broad enough to include the condition of presence—i.e., what is stored in cache memory"

IPR2015-00159, Paper No. 12, Institution Decision at 10.

Microsoft Computer Dictionary (Ex. 3001) at 497–98.

state *n.* *See* status.

status *n.* The condition at a particular time of any of numerous elements of computing—a device, a communications channel, a network station, a program, a bit, or other element—used to report on or to control computer operations.

Microsoft Computer Dictionary (Ex. 3001) at 497–98.

Full-map directories. The full-map protocol uses directory entries with one bit per processor and a dirty bit. Each bit represents the status of the block in the corresponding processor's cache (present or absent). If the dirty bit is set, then one

Chaiken (Ex. 1004) at 50.

IPR2015-00159, Paper No. 35, Petitioner Reply at 4-5.

The Claimed "states": Extrinsic – Even MI's Evidence Supports Presence as State

Cornerstone of MI's construction of "state" is their argument that "presence in a cache is distinct from and a pre-condition to the existence of state for that cache line." PO Response at 7-8. Their own evidence undermines this point:

but the current state of a block in different caches is different. As before, if a block is not present in a cache we can assume it to be in a special "not present" state or even in the invalid state.

Culler Book (Ex. 2002) at 280.

- I(nvalid): The block is invalid. The cache either does not contain the block or it contains a potentially stale copy that it may not read or write. In this primer, we do not distinguish between these two situations, although sometimes the former situation may be denoted as the "Not Present" state.

Sorin Book (Ex. 2010) at 89.

in Table 5.1. The data is presented as the number of state transitions of a particular type per 1,000 references issued by the processors. Note in the table that a new state, NP (not present), is introduced. This addition helps clarify transitions where, on a

Culler Book (Ex. 2011) at 307-10.

IPR2015-00159, Paper No. 35, Petitioner Reply at 4-5.

The Claimed "States": Extrinsic - Supports Board's Construction

"[T]he fact that [MI] can point to definitions or usages that conform to their interpretation does not make the PTO's definition unreasonable when the **PTO can point to other sources that support its interpretation**."

In re Morris, 127 F.3d 1048, 1056 (Fed. Cir. 1997)
IPR2015-00159, Paper No. 35, Petitioner Reply at 4.

The Claimed "States": Board's Preliminary Findings

ones of the cache memories." Instead, for purposes of this decision, we are persuaded only that, on this record, the term is not limited to cache coherence protocol states and is broad enough to include the condition of presence—i.e., what is stored in cache memory.

IPR2015-00159, Paper No. 12, Institution Decision at 9-10.

16

"Programmed"

11. The computer system of claim 1 wherein each of the processing nodes is **programmed** to complete a memory transaction after receiving a first number of responses to a first probe, the first number being fewer than the number of processing nodes.

'121 Patent, Claim 11.

IPR2015-00159, Paper No. 25, Patent Owner Response at 11.

"Programmed"
Board's Preliminary Findings

Moreover, Petitioners' expert's declaration is conclusory and does not rise to the level necessary to demonstrate inherency—for example, by excluding alternative ways that Koster could operate without the recited "programm[ing]" of the "processing nodes." Ex. 1014 at D-17 to D-18. In contrast, the '121 Patent specifically describes a system in which the processing nodes are configured using programming to specify conditions for completion of a memory transaction. *See*

IPR2015-00163, Paper No. 13, Preliminary PO Response at 36.

required by claim 11." *Id.* at 35. Patent Owner seems to suggest that Koster leaves open that the microprocessor could be configured to complete memory transactions using something *other* than programming, but Patent Owner does not hint at what this alternative method might be. *Id.* at 36.

IPR2015-00163, Paper No. 12, Institution Decision at 21.

"Programmed"
MI Uses Construction to Manufacture Alternative

MI submits that the term "programmed" should be construed to refer to a device that has been "configured by a sequence of instructions." This construction

IPR2015-00159, Paper No. 25, PO Response at 13.

Finally, even if the Board chooses not to adopt an explicit construction for "programmed," it should at least determine that the broadest reasonable interpretation of "programmed" is not broad enough to encompass hardwired logic.

IPR2015-00159, Paper No. 25, PO Response at 17.

"Programmed"
MI's "Alternative" Inconsistent with Specification

with I/O devices. In one embodiment, the cache coherence controller **230** is a specially configured programmable chip such as a programmable logic device or a field programmable gate array.

'121 Patent (Ex. 1001), 7:49-52.
IPR2015-00159, Paper No. 35, Petitioner Reply at 6.

"Programmed"
MI's "Alternative" Undermined By MI's Expert

Q Are you aware of any programmable system that doesn't use a sequence of instructions?

MR. SAUNDERS: Objection; form, foundation.

THE WITNESS: It would be a play on words, you know. What does "programmable" mean? For example, you can say you have a field programmable logic, which is a structure that you can configure, and it doesn't use sequence of instructions.

And you can use the term "programmable," but it's really not programmable in a sense of executing a sequence of instructions. It is probably more field programmable logic. Probably

Dr. Oklobdzija Depo. Trans. (Ex. 1026), 123:13-24.
IPR2015-00159, Paper No. 35, Petitioner Reply at 6.

"Programmed"
MI's Own Evidence Elucidates Proper Construction

²program *also* **programme** *vt* **-grammed** *or* **-gramed; -gram·ming** *or* **-gram·ing** (1896) **1 a :** to arrange or furnish a program of or for **: BILL b :** to enter in a program **2 :** to work out a sequence of operations to be performed by (a mechanism) **:** provide with a program **3 a :** to insert a program for (a particular action) into or as if into a mechanism **b :** to control by or as if by a program **c** (1) **:** to code in

Merriam Webster's Dictionary Cited by MI (Ex. 2014) at 931.

mechanism)." Ex. 2014, p. 931. I believe this definition is consistent with the usage of the term "programmed" in the '121 Patent and the understanding of a person of ordinary skill in the art. Accordingly, I believe a reasonable interpretation of the term "programmed" is "designed to perform a sequence of operations," regardless of whether this design is in hardware or software.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 6.

IPR2015-00159, Paper No. 35, Petitioner Reply at 6.

Pong: Overview

Pong discloses a multiprocessor system implementing an asynchronous cache coherence protocol. Ex. 1003, [57], ¶ 12. In the asynchronous cache coherence protocol, each data block has associated state information, which "indicates whether a copy of the data block is valid or invalid." *Id.* ¶ 13. When a processor "propagates a read or write request"

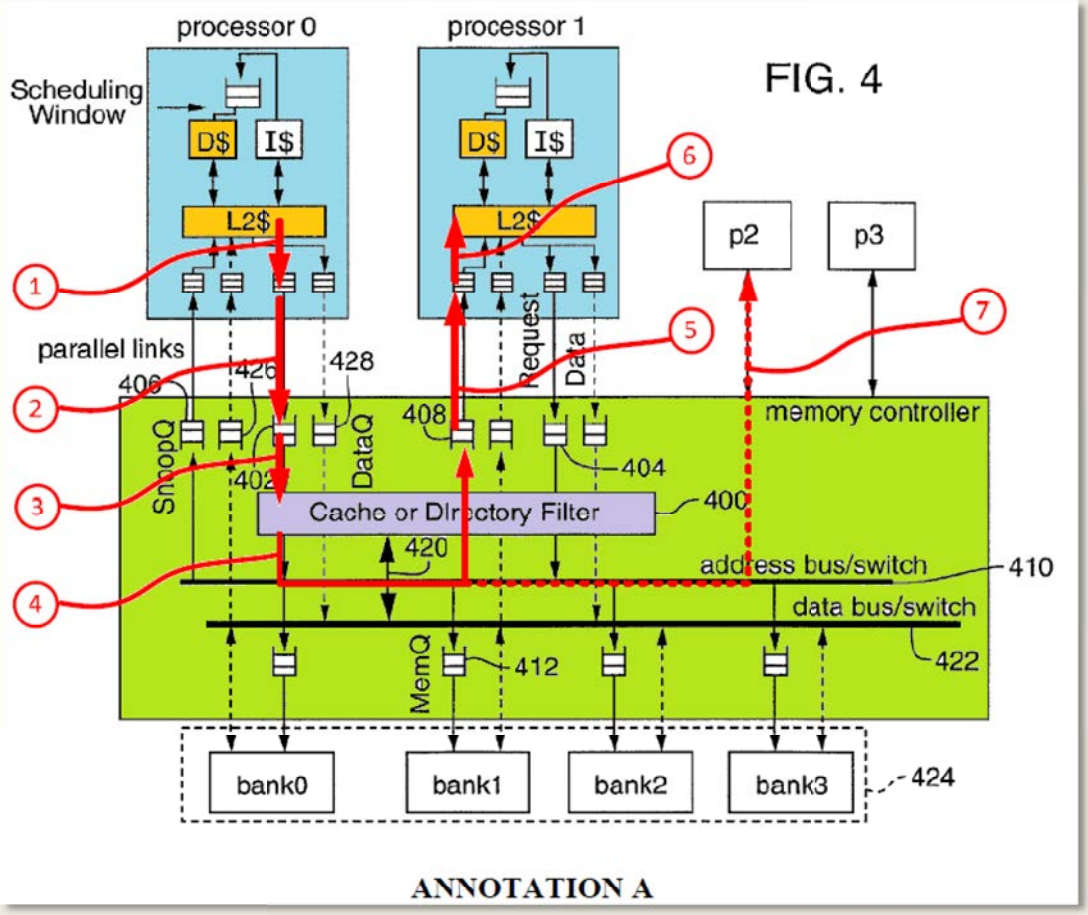
Pong discusses an implementation of the disclosed multiprocessor system using "point-to-point links to communicate memory requests." *Id.* ¶ 12; *see id.* ¶¶ 15, 29–30. Figure 2 of Pong is reproduced below.

Pong also discloses the use of a directory to "reduce traffic in the control path." *Id.* ¶ 51. "A directory, in this context, is a mechanism for identifying which processors have a copy of a data block." *Id.* The directory can be implemented with a "presence bit vector," with one bit per processor. *Id.* "When the bit corresponding to a processor is set in the bit vector, the processor has a copy of the data block." *Id.*

IPR2015-00159, Paper No. 12, Institution Decision at 14-16.

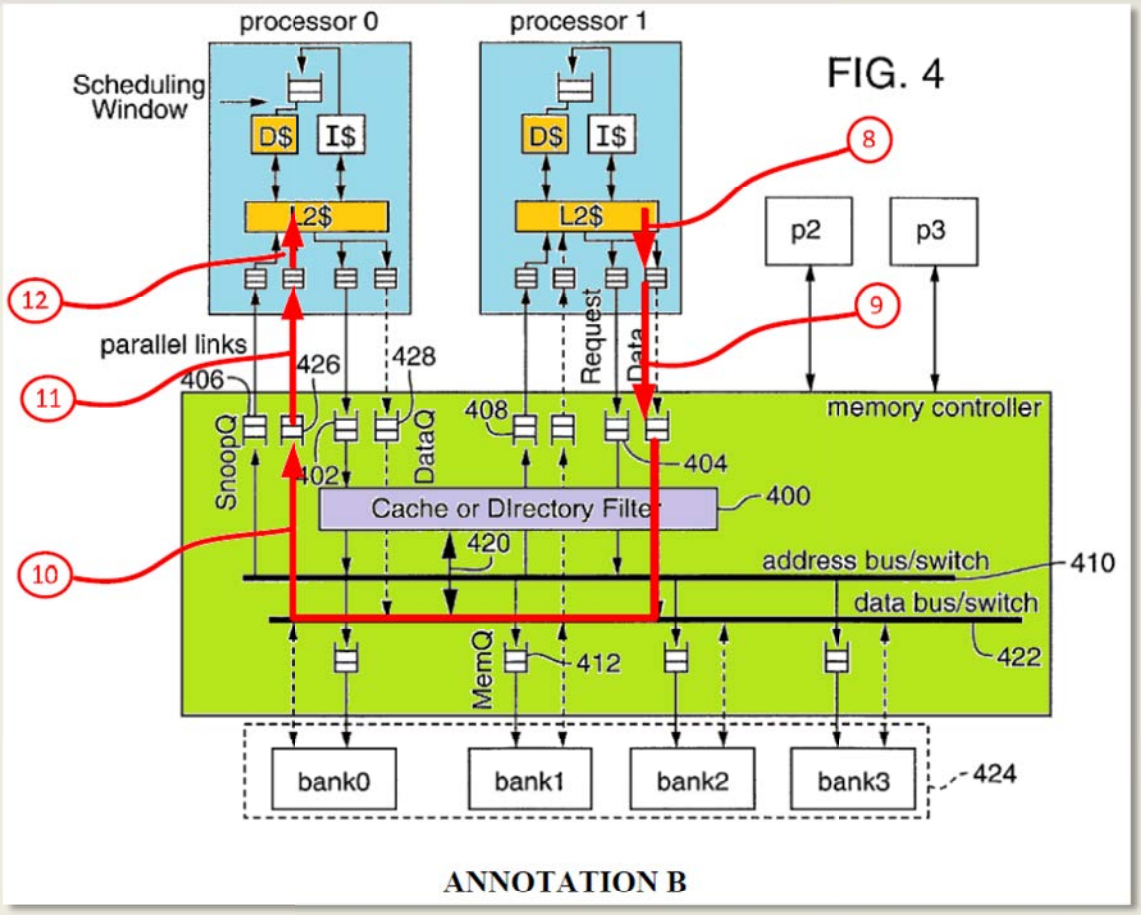
24

Pong: Overview



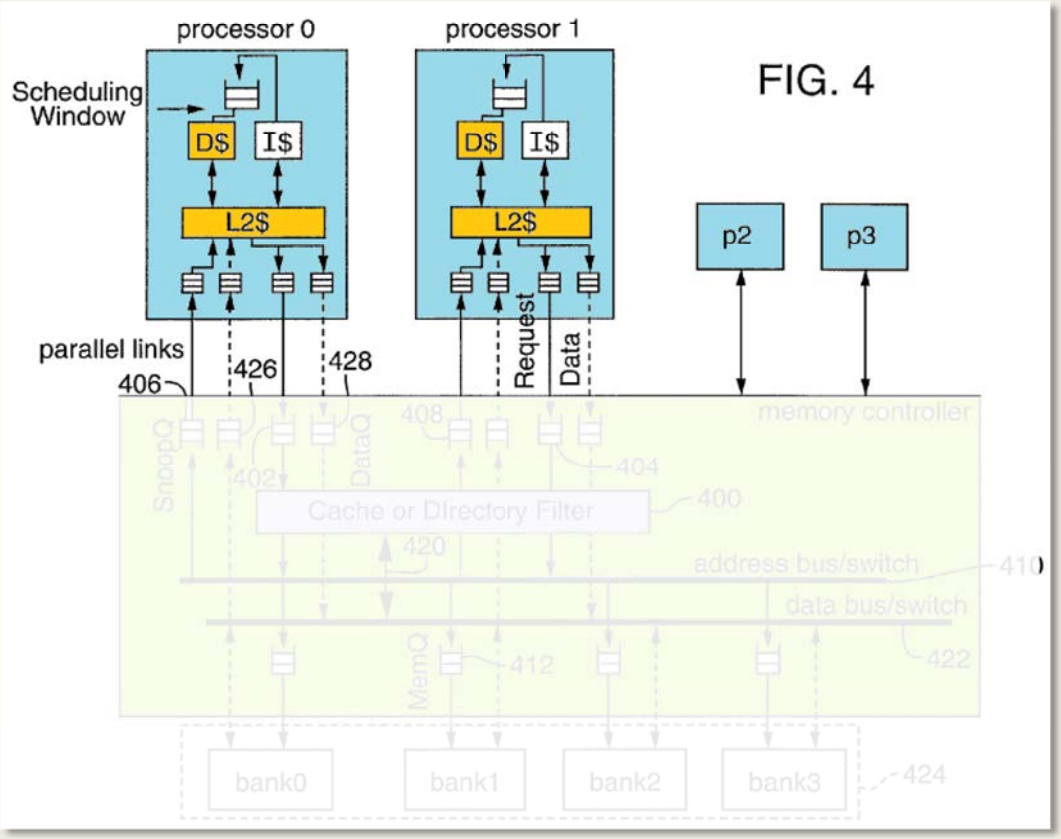
Dr. Horst Decl. (Ex. 1014), ¶¶ A-6 to A-7
IPR2015-00159, Paper No. 6, Petition at 25, 40.

Pong: Overview



Dr. Horst Decl. (Ex. 1014), ¶¶ A-14 to A-15
IPR2015-00159, Paper No. 6, Petition at 25, 40.

Pong
Mapping of Key Features of Claims



1. A computer system comprising **a plurality of processing nodes interconnected by a first point-to-point architecture**, **each processing node having a cache memory associated therewith**, the computer system further comprising a probe filtering unit which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories.

IPR2015-00159, Paper No. 6, Petition at 23-24.

Pong
Mapping of Key Features of Claims

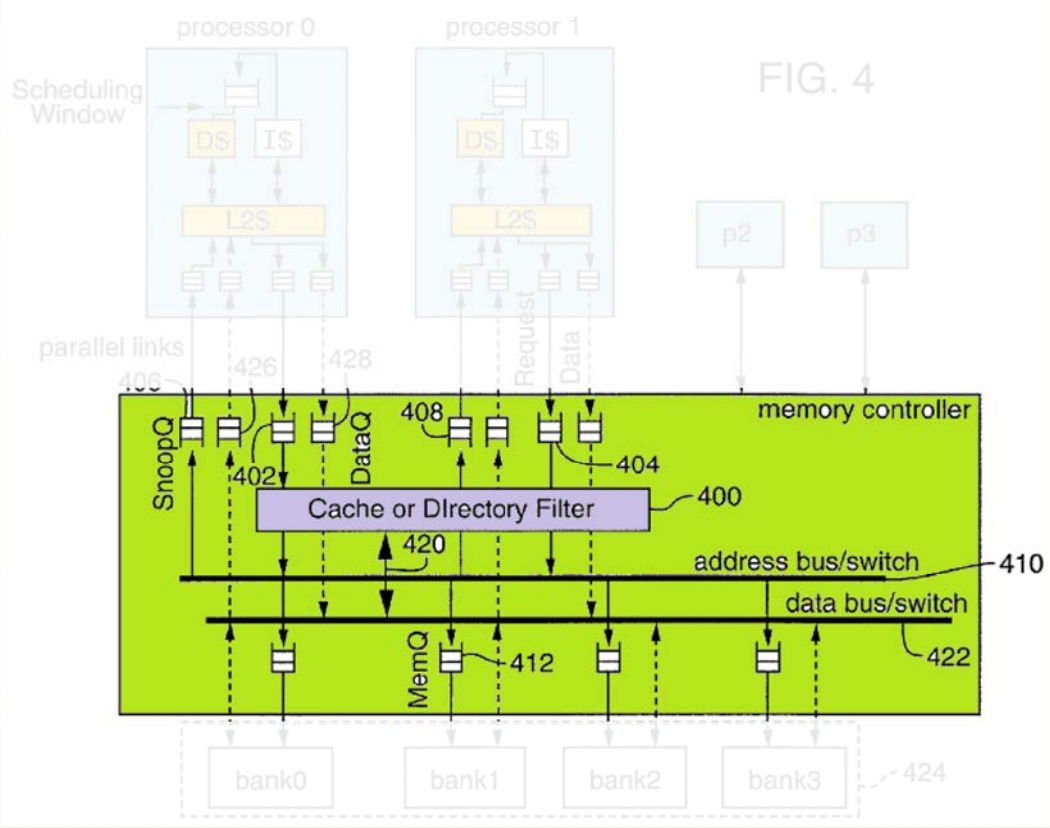


FIG. 4

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, **the computer system further comprising a probe filtering unit** which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes **with reference to probe filtering information representative of states associated with selected ones of the cache memories**.

IPR2015-00159, Paper No. 6, Petition at 24-27.

Pong
Mapping of Key Features of Claims

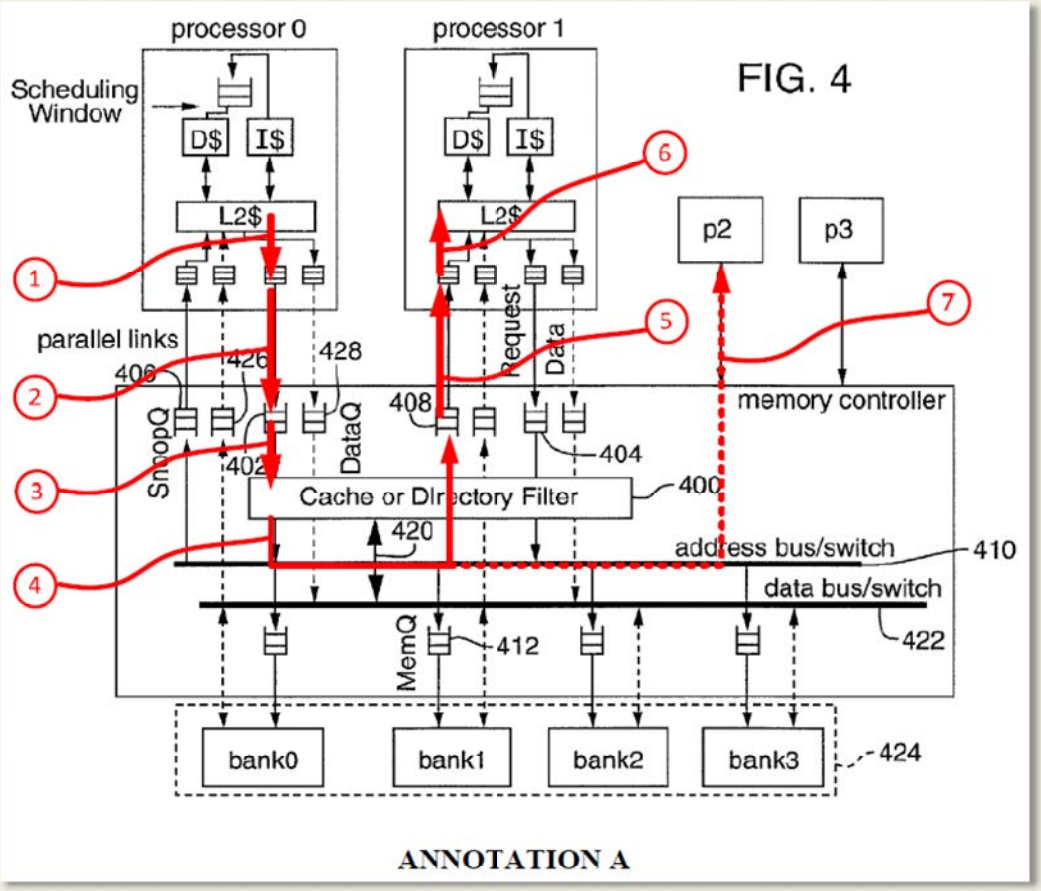


FIG. 4

ANNOTATION A

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, the computer system further comprising a probe filtering unit which is **operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes** with reference to probe filtering information representative of states associated with selected ones of the cache memories.

IPR2015-00159, Paper No. 6, Petition at 24-27.

Pong
Board's Preliminary Findings

32:7–10. In particular, on this record, we agree that these limitations are satisfied by Pong's shared memory multiprocessor, which employs "point-to-point links" and which includes multiple processors, each with "one or more caches." Ex. 1003 ¶¶ 30–31, Fig. 2. We, therefore, determine that Petitioner has shown a reasonable likelihood that it would prevail in establishing that Pong anticipates claims 1 and 16 of the '121 patent.

See Prelim. Resp. 25–48. Based on our review of Petitioner's arguments and evidence, we are persuaded, on this record, that Pong discloses the additional limitations of claim 25. Pet. 37–44. Accordingly, Petitioner has shown a reasonable likelihood that it would prevail in showing that Pong anticipates claim 25.

IPR2015-00159, Paper No. 12, Institution Decision at 22-23.

- Challenged '121 Patent.....3
- Claim Constructions.....6
- **Pong.....24**
  - Overview
  - **States (Claims 1–3, 8, 11, and 15–25)**
  - Probes (Claims 1–3, 8, 11, and 15–25)
  - Accumulating (Claims 15, 25)
  - Programmed (Claim 11)
- Koster (Claims 4–6, 11, 12, and 19–24).....56

"States" Disputes

1. A computer system comprising . . . a probe filtering unit which is operable . . .to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of **states** associated with selected ones of the cache memories.

'121 Patent, Claim 1.

Two issues:

1) MI contends that proper construction is not what the Board found in the Institution Decision
2) MI contends that Pong's bit vector fails to meet its unreasonably narrow construction

IPR2015-00159, Paper No. 35, Petitioner Reply at 1-4, 15-19.

(1) Pong's Presence Bit Vector Meets Previously Established Construction of Claimed "States"

- **Applied Construction** (addressed above): " . . .broad enough to include the condition of presence . . ."

- Undisputed that Pong meets applied construction:

> Petitioners appear to make two arguments regarding the "states" limitation and the Pong reference: **(1) that Pong's "presence bit vector" is "probe filtering information" representative of the "state" of mere "presence"**; and (2) that the "presence bit vector" also conveys whether a line is in a "valid" state.
>
> **As to Petitioners' first argument, for the reasons discussed above, that is foreclosed by [MI's] claim construction of state as a cache coherence protocol state, which does not include mere presence.**

IPR2015-00159, Paper No. 12, PO Response at 25.

33

(2) Pong's Presence Bit Vector Represents Two-State Protocol Based on Validity

cessors have a copy of a data block. One way to implement the directory is with a presence bit vector. Each processor has a bit in the presence bit vector for a data block. When the bit corresponding to a processor is set in the bit vector, the processor has a copy of the data block.

Pong (Ex. 1004) at ¶ 0051.

[0069] The discussion above refers to two types of cache coherence protocols: write invalidate and write update. Either of these protocols may be used to implement the invention. Also, while the above discussion refers to a

Pong (Ex. 1004) at ¶ 0069.

A-11. Specifically, in an update protocol, when the directory filter receives notice of a write request, all other caches with a copy are updated. In other words, every time a processor writes an update to a memory line, the directory filter receives notice of the update and sends a copy of the updated memory line to each of the other processors it determines are storing a copy of the memory line. *See* Ex. 1012, p. 282. Thus, from the perspective of the directory filter, every processor indicated as storing a copy of the memory line in its cache is storing a valid copy, because the directory filter updates those copies any time the memory line is updated.

Dr. Horst's Original Dec (Ex. 1014) at ¶ A-11.

IPR2015-00159, Paper No. 6, Petition at 40-41.

(2) MI's Expert and Board Agree That Pong's Presence Bit Vector Represents Validity

states associated with selected ones of the cache memories." The memory controller of Pong cannot selectively filter and forward "probes" to various processors based on validity in a write update protocol because every processor that has the cache line will have the line in a valid state.

Dr. Oklobdzija's Decl. (Ex. 2016) at ¶ 90.

Moreover, we are persuaded that Petitioner has made a sufficient showing—based on Pong's disclosures and the testimony of Dr. Horst—that Pong's presence bit vector also indicates validity where the write update protocol is implemented. Pong explains that the write update protocol "updates all of the cached copies of a data block when it is modified in a write operation." Ex. 1003 ¶ 48; *see e.g.*, Pet. 26 (citing Ex. 1003 ¶ 48); Ex. 1014 ¶ A-10 (citing Ex. 1003 ¶ 48). We credit and are persuaded by Dr. Horst's testimony that as a result, "every processor indicated as storing a copy of the memory line in its cache is storing a valid copy." Ex. 1014 ¶ A-11. We do not agree with Patent Owner that this testimony relies

IPR2015-00159, Paper No. 18, Institution Decision at 20.

35

(2) MI's Suggestion that Pong's Presence Bit Vector Does Not Indicate Validity Is Illogical

After all, it makes no sense for the memory controller to forward requests to processor caches that it believes have an invalid copy, as such processor caches are already known to be unable to respond to the requests and, thus, forwarding requests to these caches does nothing more than needlessly increase traffic in Pong's control path. Such a needless increase in traffic contradicts the very reason why the presence bit vector is used in Pong's system – i.e., to optimize performance by "limiting traffic in the control path." Ex. 1003, ¶ 0045.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 22.

IPR2015-00159, Paper No. 35, Petitioner Reply at 17.

(2) Construction of "States"
MI's Own Evidence (Sorin) Teaches that "Valid" is a State

In a system with only one actor (e.g., a single core processor without coherent DMA), the state of a cache block is either valid or invalid. There might be two possible valid states for a cache block if there is a need to distinguish blocks that are *dirty*. A dirty block has a value that has been written more recently than other copies of this block. For example, in a two-level cache hierarchy with a write-back L1 cache, the block in the L1 may be dirty with respect to the stale copy in the L2 cache.

A system with multiple actors can also use just these two or three states, as in Section 6.3, but

Sorin Book (Ex. 2010) at 89.

IPR2015-00159, Paper No. 35, Petitioner Reply at 17-18.

Pong Further Teaches Related "Evaluating" Limitation

evaluating the probe with the probe filtering unit to determine whether **a valid copy** of the memory line is in any of the cache memories, the evaluating being done with reference to probe filtering information associated with the probe filtering unit and representative of states associated with selected ones of the cache memories;

'121 Patent, Claim 25.

IPR2015-00159, Paper No. 6, Petition at 40-41.

- Challenged '121 Patent.....3
- Claim Constructions.....6
- **Pong.....24**
  - Overview
  - States (Claims 1–3, 8, 11, and 15–25)
  - **Probes (Claims 1–3, 8, 11, and 15–25)**
  - Accumulating (Claims 15, 25)
  - Programmed (Claim 11)
- Koster (Claims 4–6, 11, 12, and 19–24).....56

Pong Teaches Claimed "Probes"

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, the computer system further comprising a probe filtering unit which is operable to receive **probes** corresponding to memory lines from the processing nodes and to transmit the **probes** only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories.

'121 Patent, Claim 1.

IPR2015-00159, Paper No. 6, Petition at 23.

"Probes" Dispute

- **Undisputed**: Construction of "probes"

Construction of "Probes"
"[a] mechanism for eliciting a response from a node to maintain cache coherency in a system."

- **Disputed**: Pong's read requests satisfy construction of "probes"

IPR2015-00159, Paper No. 35, Petitioner Reply at 19-21.

Pong Teaches Claimed "Probes": Cache Coherent System

[0001] The invention relates to shared memory, multiprocessor systems, and in particular, cache coherence protocols.

Pong (Ex. 1003) at ¶ 0001.

cessors have a copy of a data block. One way to implement the directory is with a presence bit vector. Each processor has a bit in the presence bit vector for a data block. When the bit corresponding to a processor is set in the bit vector, the processor has a copy of the data block.

Pong (Ex. 1003) at ¶ 0051.

integrated into the memory controller. The directory filter **400** receives requests from the request queues (e.g., **402**, **404**) in the memory controller, determines which processors have a copy of the data block of interest, and forwards the request to the snoopQ(s) (e.g., **406**, **408**) corresponding to these processors via the address bus **410**. In addition, the

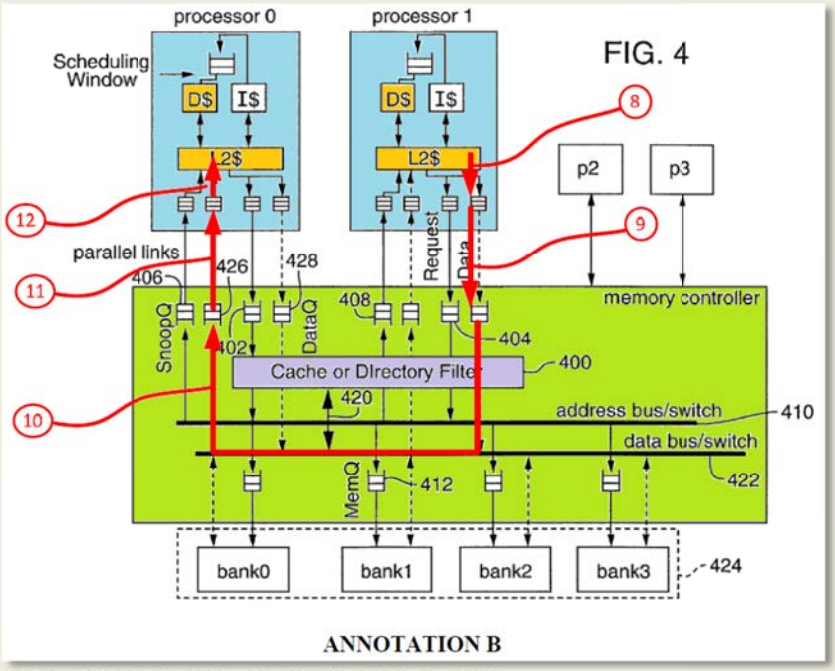
Pong (Ex. 1003) at ¶ 0057.

IPR2015-00159, Paper No. 35, Petitioner Reply at 19-20.

Pong Teaches Claimed "Probes": Request Elicits Two Responses To Maintain Coherence (1/2)

coherence protocols."). As such, the read requests are designed to elicit cache coherent copies of the requested memory line. *See, e.g.*, Ex. 1028, pp. 276-77.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 28.
IPR2015-00159, Petitioner Reply (Paper No. 35) at 20.



Dr. Horst Decl. (Ex. 1014), ¶¶ A-14 to A-15
IPR2015-00159, Paper No. 6, Petition at 25, 40.

Pong Teaches Claimed "Probes": Request Elicits Two Responses To Maintain Coherence (2/2)

The read requests also elicit a response (to maintain cache coherency) from the memory controller node, which forwards the request to other processors and updates its presence bit vector directory. Ex. 1003, ¶ 0057. Thus, Pong's cache-

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 28.

the control path as explained in the next section." Ex. 1003, ¶ 0049. In other words, Pong describes the directory playing a central role in the cache coherence protocol, as it is used, for example, to direct write invalidations to the processors storing cached copies of an invalidated memory line. Thus, an update to Pong's directory filter 400 maintains cache coherency within the system by, for example, ensuring that write invalidations are forwarded to the proper processors. Because

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 30.

IPR2015-00159, Paper No. 35, Petitioner Reply at 20.

45

Claimed Accumulation

15. The computer system of claim 1 wherein the probe filtering unit is operable to **accumulate** responses to each probe, and respond to requesting nodes in accordance with the accumulated responses.

'121 Patent, Claim 15.

25. . . . **accumulating** probe responses from the selected processing nodes with the probe filtering unit; and . . .

'121 Patent, Claim 25.

IPR2015-00159, Paper No. 6, Petition at 34-35.

"Accumulate" Dispute

1) **Disputed**: Pong Teaches Multiple Responses to a Single Probe

2) **Disputed**: Pong Teaches "Gathering" Responses

3) **Disputed**: Pong Teaches Storing Multiple Responses "*at the same time*"

IPR2015-00159, Paper No. 12, Institution Decision at 22-23.

Accumulating:
(1) Pong Teaches Multiple Responses to a Single Probe

integrated into the memory controller. The directory filter **400** receives requests from the request queues (e.g., **402**, **404**) in the memory controller, determines which processors have a copy of the data block of interest, and forwards the request to the snoopQ(s) (e.g., **406**, **408**) corresponding to these processors via the address bus **410**. In addition, the

Pong (Ex. 1003) at ¶ 57.

of the disclosure. *See, e.g.*, Ex. 1003, ¶¶ 0024, 0048. Thus, in the implementation shown in FIG. 4, Pong describes that the directory filter 400 forwards individual requests (either read or write) to those processors that "have a copy of the data block of interest." *See* Ex. 1003, ¶ 0057.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 32.

IPR2015-00163, Paper No. 35, Petitioner Reply at 22-23.

Accumulating:
(1) Pong Teaches Multiple Responses to a Single Probe

[0024] This approach avoids the need for processors to report snoop results. Each processor processes requests for a block of data independently. In particular, each processor propagates a read or write request through its cache hierarchy independently. When a processor probes its local cache and discovers that it does not have a data block requested by another processor, it simply drops the request without responding. Conversely, if the processor has the requested block, it proceeds to provide it to the requesting processor.

Pong (Ex. 1003) at ¶ 24.

FIG. 4. In other words, Pong describes that when a processor receives a request and it has the requested data block, it provides a copy of the data block to the requesting processor. Pong provides no alternative description in which a processor receives a request, has the requested data block, and does not respond with the data block.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 33.

IPR2015-00163, Paper No. 35, Petitioner Reply at 22-23.

Accumulating:
(2) Pong Teaches "Gathering"

For purposes of this proceeding, the Board should construe these terms as "gather two or more responses to a probe." This construction is supported by the

IPR2015-00159, Prelim. PO Response (Paper No. 11) at 24.

0024. When the processor is responding to a request for a data block, it "transfers the data block to an internal data queue 374," which then "processes data blocks in FIFO order, and transfers it to the corresponding data queue 352 in the memory controller." Ex. 1003, ¶ 0043. In other words, the memory controller of Pong gathers, in its data queues, each of the responses from each of the processors to which a request was sent.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 35.

IPR2015-00163, Paper No. 35, Petitioner Reply at 24.

Accumulating:
(3a) Storing "At the Same Time" Not Required

MI's Implicit Construction From PO Response

storing multiple responses "at the same time"

IPR2015-00159, Paper No. 25, PO Response at 36.

Unsupported By Record Evidence:

ac·cu·mu·late \ə-'kyü-m(y)ə-ˌlāt\ *vb* **-lat·ed; -lat·ing** [L *accumulatus*, pp. of *accumulare*, fr. *ad-* + *cumulare* to heap up — more at CUMULATE] *vt* (15c) : to gather or pile up esp. little by little : AMASS ⟨~ a fortune⟩ ~ *vi* : to increase gradually in quantity or number

MI's Dictionary (Ex. 2004) at 8.

Q My question was a little different.

My question was: Outside of the actual disclosure that's in the '121 patent, does the phrase "accumulating responses" have a special meaning in the field of cache coherency?

A I haven't heard it outside of '121.

Dr. Oklobdzija's Depo. Trans. (Ex. 1026) at 142:7-12.

IPR2015-00159, Paper No. 35, Petitioner Reply at 21-24.

Accumulating:
(3b) Pong Teaches Storing "At the Same Time"

which many millions are load instructions. *See* Ex. 2024, p. 12. Under such common, rigorous operational loads, a person of ordinary skill in the art would have understood that the request and response buffers described by Pong would almost always be queuing multiple requests and responses at any given time. As a result, for at least a subset of the read requests, the responses to each read request would be stored simultaneously in the queues, as the queues would be unable to pass the individual responses through to the requesting processor without some delay.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 37.

IPR2015-00163, Paper No. 35, Petitioner Reply at 24.

- Challenged '121 Patent.....3
- Claim Constructions.....6
- **Pong.....24**
  - Overview
  - States (Claims 1–3, 8, 11, and 15–25)
  - Probes (Claims 1–3, 8, 11, and 15–25)
  - Accumulating (Claims 15, 25)
  - **Programmed (Claim 11)**
- Koster (Claims 4–6, 11, 12, and 19–24).....56

Pong Teaches "Programmed" Microprocessors: MI Admits that Pong "Completes" According to Claim

11. The computer system of claim 1 wherein each of the processing nodes is **programmed** to complete a memory transaction . . .

'121 Patent, Claim 11.

mechanism)." Ex. 2014, p. 931. I believe this definition is consistent with the usage of the term "programmed" in the '121 Patent and the understanding of a person of ordinary skill in the art. Accordingly, I believe a reasonable interpretation of the term "programmed" is "designed to perform a sequence of operations," regardless of whether this design is in hardware or software.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 6.

transaction after receiving a first number of responses. *Id.* Rather, such a configuration must be either hardwired into the processor or set via some sort of specific programming targeted at the internal configuration of the processor itself. *Id.* Thus, the mere fact that a processor executes a set of instructions does not

IPR2015-00159, Paper No. 25, PO Response at 33.

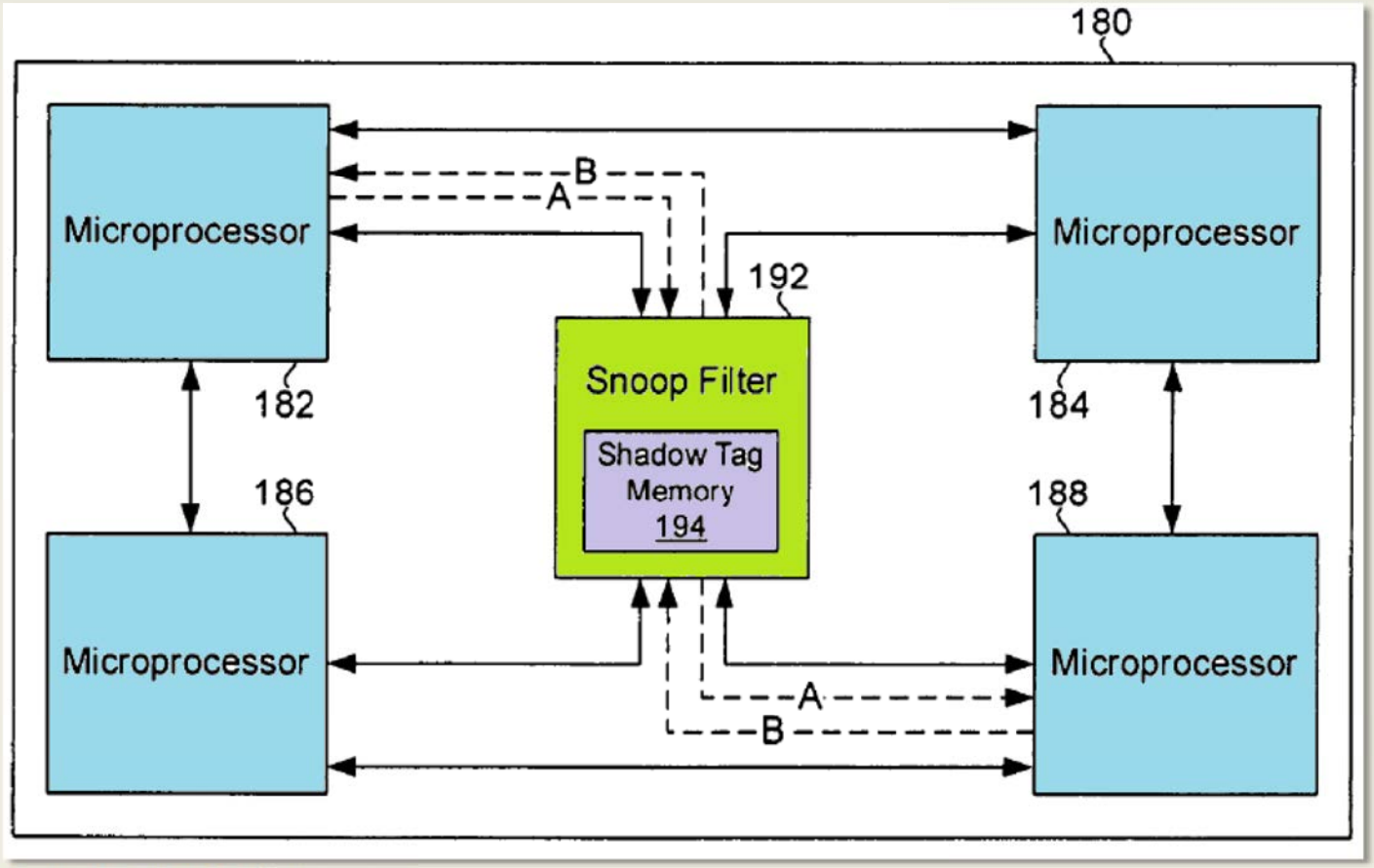
- Challenged '121 Patent……...3
- Claim Constructions……6
- Pong (Claims 1–3, 8, 11, and 15–25)……24
- **Koster……56**
  - **Overview**
  - Programmed (Claim 11)
  - States (Claims 4–6, 11, 12, and 19–24)
  - Temporary Storage (Claim 12)

Koster: Overview

Koster discloses a "snooping-based cache-coherence filter for a point-to-point connected multiprocessing node." Ex. 1009, title. In Koster, when a microprocessor requests data that is not available in its local cache, it sends a request for that data to a snoop filter. *Id.* at abs. The snoop filter stores a copy of the tags of data stored in the local cache memories of each of the microprocessors. *Id.* When the snoop filter receives a request for data, it can determine which microprocessors have copies of the requested data and relay the data request only to those microprocessors. *Id.*

IPR2015-00163, Paper No. 18, Institution Decision at 17.

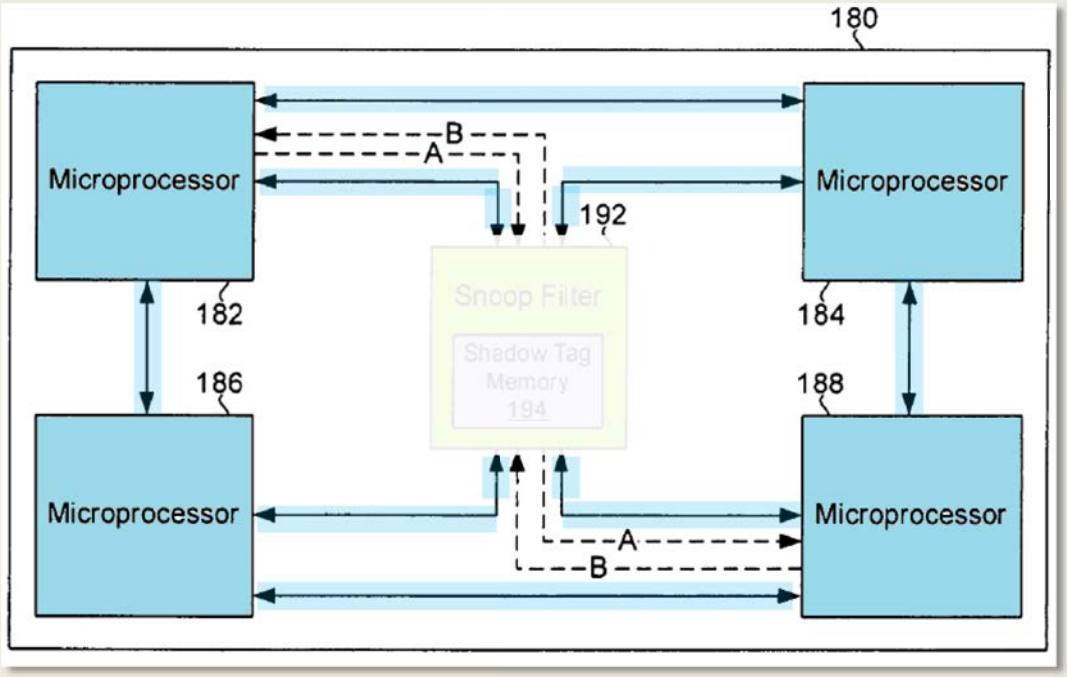
Koster: Overview



Koster (Ex. 1009), FIG. 9.

IPR2015-00163, Paper No. 18, Institution Decision at 18.

Koster
Mapping of Key Features of Claims

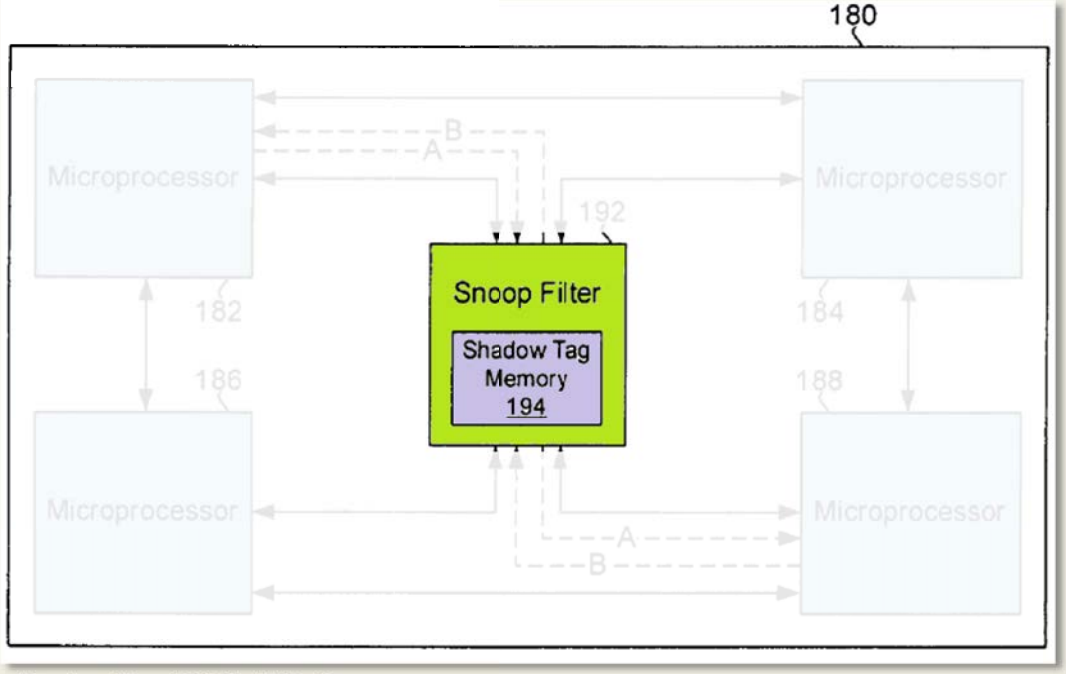


Koster (Ex. 1009), FIG. 9.

1. A computer system comprising **a plurality of processing nodes interconnected by a first point-to-point architecture**, **each processing node having a cache memory associated therewith**, the computer system further comprising a probe filtering unit which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories.

IPR2015-00163, Paper No. 1, Petition at 26-27.

Koster
Mapping of Key Features of Claims

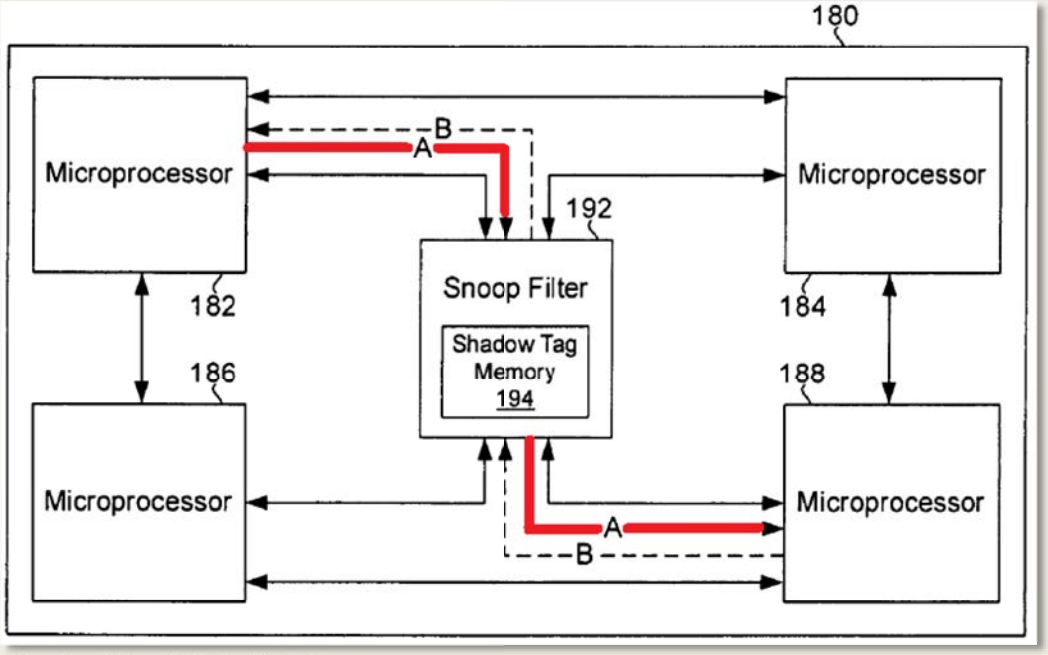


Koster (Ex. 1009), FIG. 9.

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, **the computer system further comprising a probe filtering unit** which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes **with reference to probe filtering information representative of states associated with selected ones of the cache memories**.

IPR2015-00163, Paper No. 1, Petition at 27-29.

Koster
Mapping of Key Features of Claims



Koster (Ex. 1009), FIG. 9.

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, the computer system further comprising a probe filtering unit which is **operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes** with reference to probe filtering information representative of states associated with selected ones of the cache memories.

IPR2015-00163, Paper No. 1, Petition at 27-29.

- Challenged '121 Patent.....3
- Claim Constructions.....6
- Pong (Claims 1–3, 8, 11, and 15–25).....24
- **Koster.....56**
  - Overview
  - **Programmed (Claim 11)**
  - States (Claims 4–6, 11, 12, and 19–24)
  - Temporary Storage (Claim 12)

Koster Teaches "Programmed" Microprocessors: MI Admits that Koster "Completes" According to Claim

11. The computer system of claim 1 wherein each of the processing nodes is **programmed** to complete a memory transaction . . .

'121 Patent, Claim 11.

mechanism)." Ex. 2014, p. 931. I believe this definition is consistent with the usage of the term "programmed" in the '121 Patent and the understanding of a person of ordinary skill in the art. Accordingly, I believe a reasonable interpretation of the term "programmed" is "designed to perform a sequence of operations," regardless of whether this design is in hardware or software.

Dr. Horst's Reply Declaration (Ex. 1025) at ¶ 6.

a memory transaction after receiving a first number of responses. *Id.* Rather, such a configuration must be either hardwired into the processor or set via some sort of specific programming targeted at the internal configuration of the processor itself. *Id.* Thus, the mere fact that a processor executes a set of instructions does not

IPR2015-00163, Paper No. 31, PO Response at 29.

- Challenged '121 Patent.....3
- Claim Constructions.....6
- Pong (Claims 1–3, 8, 11, and 15–25).....25
- **Koster.....57**
  - Overview
  - Programmed (Claim 11)
  - **States (Claims 4–6, 11, 12, and 19–24)**
  - Temporary Storage (Claim 12)

"States" Disputes

1. A computer system comprising . . . a probe filtering unit which is operable . . .to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of **states** associated with selected ones of the cache memories.

'121 Patent, Claim 1.

Two issues:

1) MI contends that proper construction is not what the Board found in the Institution Decision
2) MI contends that Koster's "local state memory" fails to meet its unreasonably narrow construction

IPR2015-00163, Paper No. 40, Petitioner Reply at 2-5, 8-13.

(1) Koster's "Local State Memory" Meets Previously Established Construction of Claimed "States"

- **Applied Construction** (addressed above): " . . .broad enough to include the condition of presence . . ."
- Undisputed that Koster meets applied construction:

> As explained above, this term should be construed to refer to "cache coherency states." **Under [MI's] construction**, Koster's tags do not satisfy this limitation because they are not representative of cache coherency states.

IPR2015-00163, Paper No. 31, PO Response at 21.

65

(2a) Koster's Tags Meet Claimed "States" For Same Rationale as Pong's Bit Vector

a

thereto. At least partly in order to determine whether to forward or cancel snooping-based cache-coherence broadcasts, the snoop filter **162** has local state memory (referred to and shown in FIG. **7** as "shadow tag memory") **164** that stores the tags of data stored in the local cache memories (e.g., "L**2**" cache memories) (not shown) of each of the microprocessors **152**, **154**, **156**, **158**. In other words, the shadow tag memory **164** has a copy of the tags of data stored in the local cache memories of the microprocessors **152**, **154**, **156**, **158**.

Koster (Ex. 1009) at ¶ 40.

40. In the second case, Koster's tags do not merely provide the snoop filter with addresses identifying locations of copies of requested data. Instead, they provide the snoop filter with addresses identifying locations of copies of requested data that are in a valid cache coherency state. After all, a person of ordinary skill in the art would understand that Koster's snoop filter 162 uses the tags to forward data requests only to those processors known by the snoop filter 162 to have valid copies, i.e., only those processors able to productively respond to the data request.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 40.

IPR2015-00163, Paper No. 40, Petitioner Reply at 12.

(2b) Koster Teaches Claimed "States" With Explicit Disclosure of "MOESI Cache-Coherency Protocol"

b

In one or more embodiments of the present invention, a shadow tag memory may be optimistically maintained as a set-associative cache. Further, in one or more embodiments of the present invention, the set-associative cache may use a MOESI (Modified Owner Exclusive Shared Invalid) cache-coherency protocol.

Koster (Ex. 1009), 6:33-38.

where to send a probe." Ex. 2016, ¶ 50. However, a person of ordinary skill in the art would understand that, in order to "use a MOESI . . . cache-coherency protocol," the shadow tag memory 164 would store information about the protocol (i.e., "MOESI information"). Ex. 1009, 6:33-38 (emphasis added). Moreover,

Dr. Horst's Reply Dec (Ex. 1025) at ¶ 41.

IPR2015-00163, Paper No. 40, Petitioner Reply at 12.

(2c) Koster Teaches Claimed States By MI's Own Logic

c

thereto. At least partly in order to determine whether to forward or cancel snooping-based cache-coherence broadcasts, the snoop filter **162** has local state memory (referred to and shown in FIG. **7** as "shadow tag memory") **164** that stores the tags of data stored in the local cache memories (e.g., "L**2**" cache memories) (not shown) of each of the microprocessors **152**, **154**, **156**, **158**. In other words, the shadow tag memory

Koster (Ex. 1009), 6:9-15.

As Dr. Oklobdzija opines, although "state" may have many broad and different meanings in both general English usage, as well as in the general field of computers, the term "state" connotes a specific meaning in the field of cache coherency—a cache coherency state. Oklobdzija Decl. ¶ 15. As an example of

IPR2015-00163, Paper No. 31, PO Response at 4.

IPR2015-00163, Paper No. 40, Petitioner Reply at 10.

(2c) Koster Teaches Claimed States By MI's Own Logic

THE WITNESS: In the claim -- the claim, you're correct, says using MOESI cache-coherency protocol and MOESI cache-coherency protocol -- I mean, MOESI cache-coherency states, modified on exclusive shared invalid.

BY MR. RUECKHEIM:

Q And Claim 5 at least says that these states are being used to maintain cache coherency, correct?

A That is what it says, that's correct.

Q Is Koster written in the field of cache coherency?

A Yes.

C

Dr. Oklobdzija Depo. Trans. (Ex. 1026) at 148:3-15.

IPR2015-00163, Paper No. 40, Petitioner Reply at 11.

- Challenged '121 Patent.....3
- Claim Constructions.....6
- Pong (Claims 1–3, 8, 11, and 15–25).....24
- **Koster.....56**
  - Overview
  - Programmed (Claim 11)
  - States (Claims 4–6, 11, 12, and 19–24)
  - **Temporary Storage (Claim 12)**

Koster Teaches Claimed "Temporary Storage"

12. The computer system of claim 11 wherein the probe filtering unit has **temporary storage** associated therewith for holding read response data from one of the cache memories, and the first number is one.

'121 Patent, Claim 12.

IPR2015-00163, Paper No. 1, Petition at 36-37.

Summary of "Temporary Storage" Dispute

- Parties agree that "temporary storage" encompasses a buffer
- Parties dispute whether Koster's snoop filtering unit temporarily stores read response data

IPR2015-00163, Paper No. 40, Petitioner Reply to PO Response at 15-19.

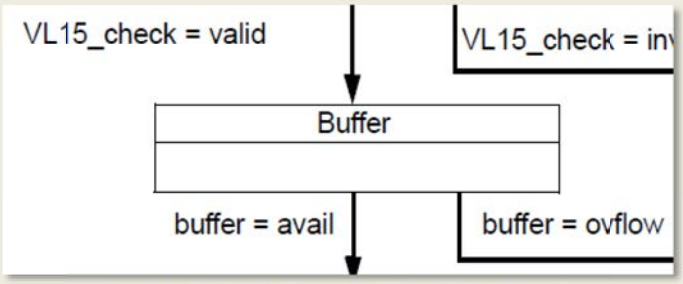
Koster Teaches Claimed "Temporary Storage" Because Infiniband Explicitly Buffers

In one or more embodiments of the present invention, high-bandwidth interconnect for point-to-point connected multiprocessing nodes may be implemented using interconnect technologies such as, for example, Infiniband or PCI Express. In one or more other embodiments of the present

Koster (Ex. 1009) at 5:40-44.

C7-8: Data packets shall be checked as specified by Figure 52 Data Packet Check machine on page 149 and Section 7.4, "Data Packet Check," on page 148. The order of checks within this state machine indi-

Infiniband Spec (Ex. 1029) at 148.



Infiniband Spec (Ex. 1029) at 149.

IPR2015-00163, Paper No. 40, Petitioner Reply at 18-19.

REFERENCE SLIDES

Construction of "States"
Amendment by Construction

Each of proposed substitute claims 19-34 were drafted by first converting the respective original claim 19-24 to independent form, and then adding the same proposed new limitations: "wherein said states comprise cache coherency states of a cache coherence protocol, and wherein said cache coherence protocol includes at least a modified state, an exclusive state, a shared state, and an invalid state, and

IPR2015-00163, Paper No. 32,
Patent Owner Motion to Amend at 2.

Construction of "States"
Dr. Oklobdzija's Citation to Dragon Not a Problem

23. Dr. Oklobdzija's citation of the Dragon protocol does not contradict my understanding of the presence bit vector in Pong. *See* Ex. 2016, ¶ 90. In the Dragon protocol, all states are "valid" states, because "the protocol always keeps the blocks in the cache up-to-date, so it is always okay to use the data present in the cache if the tag match succeeds." Ex. 2011 at 302. This is confirmatory of my interpretation of a presence bit vector in write update protocols: when a cache line is present it is necessarily valid, because a write update protocol always updates caches to keep them valid.

Dr. Horst's Reply Decl. (Ex. 1025) at ¶ 23.

IPR2015-00163, Paper No. 32,
Patent Owner Motion to Amend at 2.

Pong is Enabled: MI Fails to Establish *Prima Facie* Case

"Also, **prior art publications and patents are presumed to be enabled**. *In re Antor Media Corp.*, 689 F.3d 1282, 1287-88 (Fed. Cir. 2012); *Amgen Inc. v. Hoechst Marion Roussel, Inc.*, 314 F.3d 1313, 1355 (Fed. Cir. 2003). . . . The test for determining enabling disclosure under 35 U.S.C. § 112, first paragraph, is **whether one of ordinary skill in the art could make or use the claimed invention from the disclosed subject matter together with information in the art without undue experimentation**. *United States v. Telectronics, Inc.*, 857 F.2d 778, 785 (Fed. Cir. 1988). A disclosure can be enabling even though some experimentation is necessary. *Hybritech Inc. v. Monoclonal Antibodies, Inc.*, 802 F.2d 1367, 1384 (Fed. Cir. 1986). **The issue is whether the amount of required experimentation is undue, not whether any experimentation is necessary.** *In re Vaeck*, 947 F.2d 488, 495 (Fed. Cir. 1991). *In re Angstadt*, 537 F.2d 498, 504 (CCPA 1976)."

***Liberty Mutual*, CBM2013-00009, Paper 68, p. 40 (P.T.A.B. Feb. 11, 2014).**
IPR2015-00159, Paper No. 35, Petitioner Reply at 7-8.

Pong is Enabled:
MI Fails to Establish *Prima Facie* Case

"The factors suitable for consideration include (1) the quantity of experimentation necessary, (2) the amount of direction or guidance presented, (3) the presence or absence of working examples, (4) the nature of the invention, (5) the state of the prior art, (6) the relative skill of those in the art, (7) the predictability or unpredictability of the art, and (8) the breadth of the claims. *See In re Wands*, 858 F.2d 731, 737 (Fed. Cir. 1988)."

***Liberty Mutual*, CBM2013-00009, Paper 68, p. 40 (P.T.A.B. Feb. 11, 2014).**
IPR2015-00159, Paper No. 35, Petitioner Reply at 8.

Pong is Enabled:
MI Fails to Establish *Prima Facie* Case

The testimony of Mr. Miller and Mr. Ehsani do not explain, with sufficient specificity, what and how much experimentation would have been required by one with ordinary skill, and why that amount of experimentation should be regarded as "undue." Note that even if the required experimentation may be complex, that does not make it undue, if the art typically engages in such experimentation. *In re Wands*, 858 F.2d at 737.

***Liberty Mutual*, CBM2013-00009, Paper 68, p. 40 (P.T.A.B. Feb. 11, 2014).**
IPR2015-00159, Paper No. 35, Petitioner Reply at 8.

83. In light of the above problems, it is my opinion that Pong does not enable one of skill in the art implement a cache-coherent, point-to-point architecture without undue experimentation. Given the lack of disclosures, it is my opinion that, based on the disclosures of Pong and the background knowledge of one skilled in the art, it would take in excess of two years to design, verify, and implement a computer system with a point-to-point architecture and supporting cache coherency. Further, the end result may be a scheme entirely different from Pong's.

Dr. Oklobdzija Decl. (Ex. 2014) at ¶ 83.

IPR2015-00159, Paper No. 35, Petitioner Reply at 10.

79

Pong is Enabled:
MI's "Problems" Are Simply Alternative Embodiments

Clear Headings

[0022] Introduction

[0028] Point-to-point Links In the Memory Control Path

[0038] The Data Path

[0044] Further Optimizations

[0046] Directory Based Filter for Read Misses

[0050] Directory Based Filter for All Traffic

[0053] Implementation of the Memory Directory

[0055] Separate Memory Depository

Explicit Distinction Between Embodiments

possibly memory devices). **FIGS. 4 and 5** show alternative implementations of the multiprocessor system depicted in **FIG. 2**. Since these Figures contain similar components as those depicted in **FIGS. 2 and 3**, only components of interest to the following discussion are labeled with reference numbers. Unless otherwise noted, the description of the components is the same as provided above.

Ex. 1003 (Pong) at ¶ 0056.

IPR2015-00159, Paper No. 35, Petitioner Reply at 10-13.

80

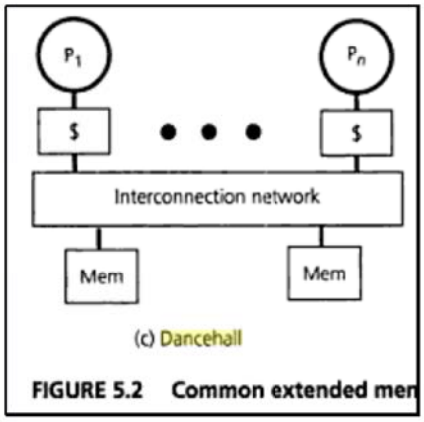
Pong is Enabled:
MI's Prime Example of "Confusion," Clear to POSITA

them. For example, in paragraph 47, Pong describes an embodiment in which a requesting processor uses information contained in a directory to specifically address requests to a processor storing a valid copy. Ex. 1003, ¶ 0047. The Culler textbook also discusses protocols which involve "finding the source of the directory information upon a miss and determining the location of the relevant copies." *See* Ex. 1028, p. 565. Pong describes an alternative approach in paragraph 56, in which the directory itself "filters the request and addresses it to the appropriate processors." Ex. 1003, ¶ 0056. Again, the Culler textbook describes this as a well understood alternative to the embodiment of paragraph 47 that reduces transactions on the interconnect, referring to it as "intervention forwarding." *See* Ex. 1028, p. 585. In other words, the Culler textbook demonstrates that a person of ordinary skill in the art would have understood Pong's description in paragraphs 47 and 56 as well-known alternative embodiments, and would not have viewed them as conflicting.

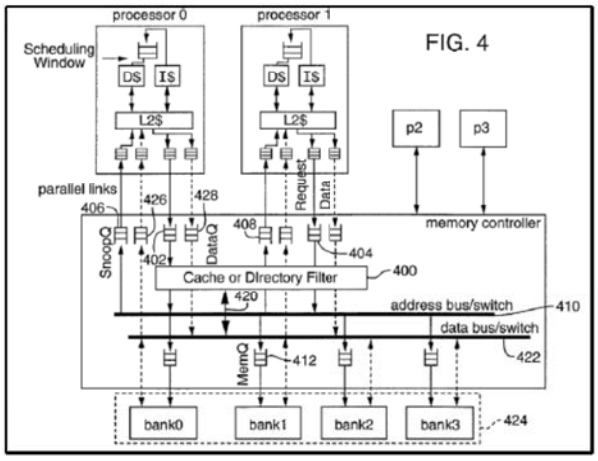
Horst Reply Decl. (Ex. 1025) at ¶ 13.

IPR2015-00159, Paper No. 35, Petitioner Reply at 12-13.

Pong is Enabled:
POSITAs Knew How to Implement Pong's Architecture



Culler Book
Cited by PO



Pong

271. Pong's architecture, known as "dancehall," is shown in FIG. 5.2(c) of the Culler textbook, and applies to systems in which each processor has its own private cache and that is connected to main memory by a point-to-point interconnection network. *See* Ex. 1028, pp. 271. Chapter 8 of the Culler details various cache

Dr. Horst Reply Decl. (Ex. 1025) at ¶ 11.

IPR2015-00159, Paper No. 35, Petitioner Reply at 11-12.

Pong is Enabled:
MI's Expert Testified to Knowledge of POSITA

THE WITNESS: No. I mentioned, you know, there are figures that show the structure interconnection, et cetera. This line gives kind of suggestion to -- that this can be implemented as a programmable chip. As I said, the rest is up to the engineers to -- to figure out further.

Dr. Oklobdzija's Depo. Trans. (Ex. 1026) at 101:15-20.

Compare IPR2015-00159, Paper No. 35, Reply in Support of MTA at 7-8 (citing Dr. Oklobdzija's Reply Decl. (Ex. 2042) at ¶ 7).

Koster Teaches Claimed "Temporary Storage": Described Functionality Impractical Without Buffering

wards broadcast A to microprocessor **188**. In response to broadcast A being forwarded to microprocessor **188**, microprocessor **188** issues a response B (having a copy of the requested data) that is forwarded back to the requesting microprocessor **182** through the snoop filter **192**.

By forwarding response B through the snoop filter **192**, the snoop filter **192** is able to update its shadow tag memory **194**.

Koster (Ex. 1009) at 7:10-16.

IPR2015-00163, Paper No. 40, Petitioner Reply at 15-16, 18.