Page 1

UNITED STATES INTERNATIONAL TRADE COMMISSION
WASHINGTON, D.C.
Before the Honorable Thomas B. Pender
Administrative Law Judge

- - - - - - - - - - - - - - - - - - x

In the Matter of

CERTAIN CONSUMER ELECTRONICS	Investigation
AND DISPLAY DEVICES WITH	No. 337-TA-932
GRAPHICS PROCESSING AND GRAPHICS	
PROCESSING UNITS THEREIN	

- - - - - - - - - - - - - - - - - - x

CONTAINS ADVANCED MICRO DEVICES, INC.
CONFIDENTIAL BUSINESS INFORMATION SUBJECT TO
PROTECTIVE ORDER - VIDEOTAPED DEPOSITION OF
ANDREW E. GRUBER, on behalf of Advanced Micro
Devices, Inc. and Individually

Tuesday, March 17, 2015, 9:14 a.m.
Robins, Kaplan, Miller & Ciresi L.L.P.
800 Boylston Street
Boston, Massachusetts

Reported by:
MICHAEL O'CONNOR, RPR, CRR, CBC, CCP
Job No. 91486

Page 2

Tuesday, March 17, 2015

9:14 a.m.

Videotaped Deposition of ANDREW E. GRUBER, individually, and as Corporate Designee on behalf of Advanced Micro Devices, Inc. before Michael D. O'Connor, a Registered Professional Reporter and Certified Realtime Reporter and Notary Public in the Commonwealth of Massachusetts.

Page 3

APPEARANCES:

ROBINS KAPLAN MILLER & CIRESI
Attorney for Advanced Micro Devices:
800 LaSalle Avenue
Minneapolis, Minnesota 55402
BY: AARON FAHRENKROG, ESQ.

- AND -

STERNE KESSLER GOLDSTEIN & FOX
Attorney for Advanced Micro Devices:
1100 New York Avenue, N.W.
Washington, DC, 20005
BY: JONATHAN TUMINARO, ESQ.

ORRICK HERRINGTON & SUTCLIFFE
Attorney for NVIDIA Corporation:
1152 15th Street, N.W.
Washington, DC, 20005
BY: CHRISTOPHER HIGGINS, ESQ.
I. NEEL CHATTERJEE, ESQ.

Page 4

APPEARANCES (Cont'd):

KEKER & VAN NEST
Attorney for Qualcomm Incorporated:
633 Battery Street
San Francisco, California 94111
BY: EUGENE PAIGE, ESQ.

Also Present: Leann McKeon, Videographer

Also Present via speakerphone:
Daniel Leventhal, Esq.,
Norton Rose Fulbright (US), LLP

Page 5

PROCEEDINGS

(Gruber Exhibit 1, Document entitled "Complainant NVIDIA Corporation's Notice of Deposition of Andrew Gruber," marked for identification)

(Gruber Exhibit 2, Document entitled "Application For Issuance of Subpoena Duces Tecum and Subpoena Ad Testificandum to Advanced Micro Devices, Inc.," marked for identification)

09:13

VIDEOGRAPHER: This is the start of 09:13
tape labeled number one of the videotaped 09:13
deposition of Andrew Gruber in the Matter of 09:13
Certain Consumer Electronics and Display Devices 09:13
With Graphics Processing and Graphics Processing 09:13
Units Therein, Investigation No. 337-TA-932 in 09:13
the United States International Trade Commission, 09:13
Washington, D.C., before the honorable Thomas B. 09:13
Pender. 09:14

This deposition is being held at 800 09:14
Boylston Street, Boston, Massachusetts, on March 09:14
17, 2015, at approximately 9:14 a.m. My name is 09:14
Leann McKeon. I'm the legal video specialist 09:14
from TSG Reporting, Inc. headquartered at 747 09:14
Third Avenue, New York, New York. The court 09:14

Page 6

reporter is Michael O'Connor in association with 09:14
TSG Reporting. Will counsel please introduce 09:14
yourselves. 09:14

MR. HIGGINS: Chris Higgins from 09:14
Orrick, Herrington & Sutcliffe, on behalf of 09:14
Complainant, NVIDIA Corporation. And with me is 09:14
Neel Chatterjee, also from Orrick. 09:14

MR. FAHRENKROG: Aaron Fahrenkrog 09:14
from Robins Kaplan, LLP on behalf of non-party 09:14
Advanced Micro Devices, Inc. 09:14

MR. TUMINARO: Jonathan Tuminaro from 09:14
Sterne, Kessler, Goldstein & Fox on behalf of 09:14
non-party AMD. 09:14

MR. PAIGE: Eugene Paige of Keker & 09:14
Van Nest on behalf of Respondent Qualcomm and the 09:14
witness in his personal capacity. 09:14

ANDREW E. GRUBER,

called as a witness, having been duly
sworn by a Notary Public, was examined and
testified as follows:

EXAMINATION BY

BY MR. HIGGINS: 09:15

Q. Good morning, Mr. Gruber. 09:15

A. Good morning. 09:15

Q. You have been handed what's been 09:15

Page 7

marked as Exhibits 1 and 2. Exhibit 1 is 09:15
"Complainant NVIDIA Corporation's Notice of 09:15
Deposition of Andrew Gruber." Exhibit 2 is 09:15
"Application For Issuance of Subpoena Duces Tecum 09:15
and Subpoena Ad Testificandum to Advanced Micro 09:15
Devices, Inc." 09:15
Do you see those? 09:15
A. Yes. 09:15
Q. Are you here to testify on behalf of 09:15
both the deposition notice and the subpoena to 09:15
AMD? 09:15
A. Yes. 09:15
Q. When was the R400 built? 09:15
A. The R400 project was remained the 09:15
R500 project, and eventually involved into the 09:15
R600 project, which was the chip that finally 09:15
made it to market. The same database also 09:15
evolved into the internal code name Xenos chip, 09:15
which was in the Microsoft Xbox 360 project. 09:16
That Xbox project shipped in late 09:16
2005. The R600 shipped after that. I believe it 09:16
was in 2007. 09:16
Q. Was a chip ever built that was based 09:16
solely on the R400 project? 09:16
A. No. By the time that the Xbox 09:16

Page 8

shipped, there were certain evolution of the 09:16
database. So there were changes. But the 09:16
internals were largely based on the R400 09:16
unchanged. 09:16

Q. Was a chip ever built that was based 09:16
solely on the R500 project? 09:16

A. No. The R500 project evolved into 09:16
the R600 project. There were substantial changes 09:17
from the R500 to the R600. But still a lot of 09:17
the engineering in the R400 showed up in the 09:17
R600. 09:17

Q. What were the substantial changes 09:17
from the R500 to the R600? 09:17

A. The R600 still had the same unified 09:17
shader idea as the R500. In the R600, the 09:17
instruction set changed from a pure vector model, 09:17
what was called a Vec 4 model, into four parallel 09:17
scaler instructions. 09:17

The data flow in the R500 to R600 was 09:17
very similar. The main change was in the, as I 09:18
say, the instruction set and the approach to 09:18
instruction execution. 09:18

Q. Did the R600 use the same sequencer 09:18
as the R500? 09:18

MR. PAIGE: Vague and ambiguous. 09:18

Page 9

A. The sequencer between the R500 and the R600 had substantial changes. The R500 team was primarily from the Marlboro office, and that was the same team that had just continued on as the R400 changed into the R500. 09:18 09:18 09:18 09:18 09:18

The R600 was more of a combined effort in between the East Coast office and the West Coast office. So the sequencer aspects changed substantially between the R500 and the R600. 09:18 09:18 09:19 09:19 09:19

Q. Why did the sequencer change substantially between the R500 and the R600? 09:19 09:19

A. There were just different people working on it. I was not involved as much in the R600 as in the R500. So I can't talk about a lot of the specific changes in the sequencer, but I do believe there were changes from the R500 to the R600. 09:19 09:19 09:19 09:19 09:19 09:19

Q. When was the R400 project started? 09:19

A. The R400 project started taking shape in late 2000. That's when the initial concepts were kind of first come up with and discussions started about it within ATI. 09:19 09:19 09:20 09:20

Q. Why was the R400 project started? 09:20

A. Well, we needed a follow-on chip. 09:20

Page 10

The way that ATI worked at that time, we had a 09:20
leapfrogging design system. So it was a group on 09:20
the East Coast that was doing the odd sequence -- 09:20
sorry -- the even sequence of chips. So the East 09:20
Coast team did the R200 and then the West Coast 09:20
team would do the R300, and the East Coast team 09:20
would do the R400. 09:20

This is because the market needs were 09:20
such that the need for a new chip was shorter 09:21
than the design time needed for a new chip. So 09:21
we had these leapfrogging teams to be able to 09:21
meet the market windows. 09:21

So the R400 was intended as our new 09:21
chip after the -- well, after the West Coast 09:21
R300. But, from the East Coast point of view, 09:21
after the R200. We needed both increased 09:21
performance as well as we were looking at 09:21
supporting next-generation APIs. 09:21

Q. Did the West Coast team work on the 09:21
next odd numbered R500? 09:21

A. That was originally the goal. As 09:21
schedules changed and the R400 took longer than 09:22
anticipated to meet the development effort, what 09:22
happened was that the -- we met that marketing 09:22
window rather than with the R400 with a revision 09:22

Page 11

of the R300, which we called the R420. 09:22

The R400 was renamed the R500, and 09:22 aimed at the next market window. As it turned 09:22 out, the R500 slipped as well. And so the West 09:22 Coast team did wind up producing a chip that went 09:22 into the R500 marketing window, and that was 09:22 called the R580. 09:22

Q. When was the 420 chip released? 09:22

A. I am not sure. I'd have to go back 09:22 into records and get back to you on that. 09:23

Q. Do you have a general idea of the 09:23 time frame when it may have been released? 09:23

A. The R420. It was probably released 09:23 in the 2004 time frame. I couldn't... 09:23

Q. Do you know when the 580 was 09:23 released? 09:23

A. It probably would have been released 09:23 about a year later, maybe 18 months later. 09:23

Q. Why did the R400 take longer than 09:23 anticipated to meet the development effort? 09:23

A. The R400 was a very aggressive chip 09:23 in a number of ways. The unified shader idea was 09:23 certainly new, and there was a lot of engineering 09:24 associated with that. But there were a lot of 09:24 other changes involved in the R400 as well. 09:24

Page 12

There were compression changes in how 09:24
we dealt with memory that took a lot longer than 09:24
we thought. There was negotiations with 09:24
Microsoft over how the API would shape up. 09:24
So there were back-and-forth 09:24
discussions about that. So sometimes engineering 09:24
just takes longer than you anticipate. 09:24
Q. Where was the work on the R400 done, 09:24
in what location? 09:25
A. The primary work on the R400 was done 09:25
in the Marlboro office of ATI. There was some 09:25
work done as well in the Orlando office of ATI. 09:25
The Orlando team came on board with the R400 a 09:25
little later than the Marlboro team did, because 09:25
the Orlando office was also involved in the R300 09:25
series of products. 09:25
So while we had a leapfrogging 09:25
effort, the Orlando team was actually on both of 09:25
those efforts. 09:25
Q. Which parts of the R400 did the 09:25
Marlboro office work on? 09:25
A. As I said, the Marlboro office worked 09:25
on virtually all of it. It worked on the 09:25
sequencer, on the shader pipe, on what we called 09:26
the render back-end, which NVIDIA called the ROP 09:26

Page 13

unit, the texture system, on the command 09:26
processor, although some of the command processor 09:26
was done in Orlando. 09:26
Most of the vertex processing after 09:26
shading was done in the Orlando office, that it's 09:26
a clipping and viewport transform. 09:26
Q. What aspects of the engineering 09:26
associated with the unified shader contributed to 09:26
the delay in the R400 project? 09:26
MR. FAHRENKROG: Object to the form 09:26
of the question. 09:26
MR. PAIGE: Join in the objection. 09:26
A. A lot of the data flow was 09:27
challenging. How do you get the data from the 09:27
vertex shader back into the pixel shader. 09:27
Dealing with some of the shared resources for 09:27
both vertices and pixels in a manner such that 09:27
you didn't get deadlock and that you got adequate 09:27
performance, that one didn't overwhelm the other 09:27
was also challenging. 09:27
Q. In the R400, what were the shared 09:27
resources between vertices and pixels? 09:27
A. The shader itself, that is the 09:27
execution unit and the sequencer logic, were 09:27
shared. The general purpose registers were 09:28

Page 14

shared also on a slowly moving dynamic basis 09:28
where there were two different pools, but the 09:28
pool sizes could change. 09:28

Q. What were the issues with the general 09:28
purpose register in the unified shader design? 09:28

MR. PAIGE: Object to the form of the 09:28
question. 09:28

MR. FAHRENKROG: Objection. 09:28

A. Allowing the shader that had the most 09:28
work to do to get more of the resources than the 09:29
other shader, while at the same time having a 09:29
reasonable allocation mechanism for those. We 09:29
like to allocate in kind of a FIFO order, because 09:29
that's easy to control, but it's difficult to do 09:29
that when you have two different shader programs 09:29
that take radically different amounts of time, 09:29
that take unpredictable amounts of time. 09:29

Q. Were these issues with the general 09:29
purpose registers ever resolved in the R400 09:29
project? 09:29

MR. PAIGE: Vague and ambiguous. 09:29

A. Yes. The R400 had a register 09:29
allocation scheme that was functional and that 09:29
worked well, and that was actually carried 09:30
forward into the Xbox chip. I don't think there 09:30

Page 15

were -- I can't think of a difference between the 09:30
R400 design and what was produced in the Xbox, in 09:30
that area. 09:30

Q. When was the design of the general 09:30
purpose registers in the R400 finalized? 09:30

A. The general purpose registers were 09:30
finalized fairly early on. I would say prior to 09:30
the end of 2001, the overall scheme was 09:30
established and really didn't change much after 09:30
that. 09:31

Q. What was your role in the R400 09:31
project? 09:31

A. I was the head architect of the 09:31
Marlboro site. Steve Ryan was under me, and he 09:31
was nominally specifically in charge of the R400. 09:31
As the project evolved, I took over more of that 09:31
role from Steve. 09:31

Q. Who else worked on the R400 project? 09:31

A. Well, there was a very large team. I 09:31
mean, by the end, it was certainly over 100 09:31
people. 09:31

Q. Who was on the initial team that 09:31
started work on the R400 project? 09:31

A. Initially, there were just a small 09:31
group of architects working. Steve Morein, 09:31

Page 16

Laurent Lefebvre, Andy Skende were people who worked on it. 09:32 09:32

Q. When were additional people added to the R400 team? 09:32 09:32

A. Throughout 2001, as people came off of the final work on the R200, there were more and more people involved. I would say during the first half of 2001, it was a fairly small architectural team. 09:32 09:32 09:32 09:32 09:32

Q. When did the R400 project end? 09:32

A. I believe it ended -- well, it didn't end. We evolved it into the R500. That is, we realized we wouldn't make that market cycle in early 2003. 09:32 09:32 09:33 09:33

Q. When did work on the R500 project begin? 09:33 09:33

A. Immediately after. There was no work stoppage. It was just kind of a redirection. We were late on the R400, and we didn't want to be late on the R500 as well. So the nature of the work did not change at all between R400 and R500. 09:33 09:33 09:33 09:33 09:33

The API it was aimed at was the same API. 09:33 09:33

Q. When did work on the Xenos chip begin? 09:33 09:33

Page 17

A. That also began in early 2003. I 09:33
think that the Xenos work began shortly after the 09:33
transition to R500. 09:33

Q. Did the same teamwork on the Xenos 09:33
chip and the R500 project? 09:33

A. There was a lot of overlap, but there 09:34
were also separate people working on the Xenos 09:34
chip that were working on the R500. Most of the 09:34
Xenos work wound up being focused in the Orlando 09:34
site, while most of the Marlboro site continued 09:34
work on the R500. 09:34

But the databases were so similar, 09:34
that there was a lot of back and forth between 09:34
the two. 09:34

Q. Which project did you work on? 09:34

A. I was primarily on the R500. 09:34

Q. Why was a different team assigned to 09:34
work on the Xenos? 09:34

A. There were aspects of the Xenos that 09:34
were unique to Xenos that weren't going to be 09:34
part of the anticipated R500 chip. Xenos had a 09:34
very different memory interface, for instance, 09:35
there was a dedicated memory chip as opposed to a 09:35
standard off-the-shelf memory chips in the Xenos. 09:35

There were also interfaces to the 09:35

Page 18

host processor in the Xenos chip that were not 09:35
anticipated to be part of the R500. And there 09:35
were also just the general Microsoft interaction, 09:35
as Microsoft was the project owner, that weren't 09:35
a factor in the R500. 09:35

Q. Was Microsoft the project owner of 09:35
the Xenos chip? 09:35

A. They were the overall project owner, 09:35
yeah. 09:35

Q. What did they do as the overall 09:35
project owner of the Xenos chip? 09:35

MR. FAHRENKROG: I'll insert an 09:35
objection here. I caution you not to reveal any 09:35
Microsoft confidential information. We've 09:36
objected to that in response to your subpoena, 09:36
and he's not here to talk about anything 09:36
confidential to Microsoft here today. You're 09:36
welcome to ask your questions about Xenos, and so 09:36
forth. 09:36

Q. What did Microsoft do as the overall 09:36
project owner of the Xenos? 09:36

A. They specified some of the interfaces 09:36
and some of the scenarios that we had to support. 09:36

Q. Which interfaces did they specify? 09:36

A. There was an interface in between the 09:36

Page 19

host and the Xenos chip that was Microsoft 09:36
specific that would allow the host to directly 09:36
write into Xenos memory. 09:36

Q. What were the specific scenarios that 09:37
you had to support for Microsoft? 09:37

MR. FAHRENKROG: Object to the form 09:37
of the question. 09:37

A. Well, for instance, there was -- I 09:37
mentioned this dedicated memory associated with 09:37
the part, and that had a certain size associated 09:37
with it. 09:37

Microsoft was involved in specifying 09:37
the size of that memory, and the size of that 09:37
memory defined what screen resolutions would be 09:37
supported in one pass versus multipass. 09:37

So, you know, whether given say 1280, 09:38
or whatever standard high definition is, whether 09:38
that would be a multipass. And that was 09:38
primarily a cost issue. So that's why it fell to 09:38
Microsoft. 09:38

The engineering of the chip was ATI, 09:38
but a number of the specifications of the chip 09:38
were Microsoft. 09:38

Q. So did Microsoft specify certain 09:38
aspects of the chip design for the Xenos? 09:38

Page 20

A. They specified certain capabilities 09:38
of the chip. I wouldn't say Microsoft specified 09:38
design aspects. 09:38

Q. What capabilities of the chip did 09:38
Microsoft specify? 09:38

A. Well, as I mentioned, the ability for 09:38
the host to directly write into the chip. The 09:38
size of the external memory. They might have -- 09:39
as I said, I wasn't on the Xenos project, but I 09:39
expect that they had certain performance goals as 09:39
well that had to be met. 09:39

Q. Who is the person or people at 09:39
Microsoft that were involved with the Xenos 09:39
project? 09:39

MR. FAHRENKROG: Object to the 09:39
question as asking for confidential Microsoft 09:39
information. 09:39

A. I don't recall offhand. 09:39

Q. Was the Xenos chip a single chip GPU 09:39
implementation? 09:39

MR. FAHRENKROG: Object to the form 09:39
of the question. 09:40

A. The GPU was a single chip. There was 09:40
an additional dedicated memory part that held a 09:40
tile buffer that was on the same -- was in the 09:40

Page 21

same package, but it was not part of the same 09:40
die. 09:40

Q. Were any rendering functions 09:40
performed on a separate chip that had a tile 09:40
buffer? 09:40

A. I believe it was purely a eDRAM part, 09:40
an embedded DRAM part. I would have to go back 09:40
to actual notes to check to see that there wasn't 09:41
any logic functionality on that part. 09:41

I remember very early on there was 09:41
some discussion about that, but I do not believe 09:41
that was in the part. 09:41

Q. What notes would you have to go back 09:41
and check? 09:41

A. Oh, I'd have to go through ATI 09:41
proprietary information. It wouldn't be my 09:41
personal notes, but I'd have to get proprietary 09:41
information from ATI about Xenos, like the Xenos 09:41
overall specification. 09:41

Q. Was the render back-end module on a 09:41
separate chip? 09:41

A. Again, certainly the entire render 09:41
back-end module would not have been on a separate 09:41
chip. There may have been some aspect on that 09:41
separate chip, but I can't testify one way or the 09:41

Page 22

other. 09:42

Q. Why would certain aspects of the 09:42
render back-end module be placed on a separate 09:42
chip? 09:42

MR. FAHRENKROG: Object to the form 09:42
of the question. Vague. 09:42

A. The reason why you would do that was 09:42
to convert read-modify-write operations into pure 09:42
write operations to preserve bandwidth. 09:42

Q. In general, what type of operations 09:42
does the render back-end module perform? 09:42

MR. FAHRENKROG: Object to the form. 09:42
Vague. 09:42

MR. PAIGE: Object to the form. 09:42

A. It generally performs blending 09:42
operations and logic operations like and/or. 09:42

Q. What reason would blending operations 09:43
be moved to a separate chip? 09:43

A. For the same reason to preserve 09:43
memory bandwidth. If you were doing 09:43
read-modify-write operations, could be converted 09:43
to just a write-only operation. 09:43

Q. Did Microsoft request the use of a 09:43
separate chip in the Xenos design? 09:43

A. I don't recall whether that aspect of 09:43

Page 23

the design was originated with Microsoft or was 09:43
suggested by ATI. 09:43

Q. Why didn't Microsoft use the R400 09:43
design? 09:44

MR. FAHRENKROG: Object to form. 09:44
Foundation. 09:44

MR. PAIGE: Join in the objection. 09:44

A. We used the R400 as the basis for the 09:44
Microsoft proposal. As we understood more about 09:44
what Microsoft was looking for, there was, you 09:44
know, design changes proposed to meet 09:44
Microsoft-specific needs. 09:44

I don't know whether the specific 09:44
change of the use of an embedded DRAM separate 09:44
chip came from Microsoft or ATI. 09:44

Q. What parts of the R400 design did not 09:44
meet Microsoft-specific needs? 09:44

MR. PAIGE: Object to the form. 09:45

A. Well, for one thing, Microsoft didn't 09:45
want the graphics DRAM aspect of it, that is, as 09:45
a separate memory pool. Microsoft wanted things 09:45
to work out of a single system pool. 09:45

Once we lost the separate memory pool 09:45
for graphics, that's when the eDRAM showed up as 09:45
a way of at least providing some additional 09:45

Page 24

memory bandwidth to meet GPU needs. 09:45
The actual shader system from R400 to 09:45
Xenos was pretty much identical. 09:46
Q. How did the R400 design use the 09:46
graphics DRAM? 09:46
A. The R400 generally rendered into 09:46
graphics DRAM as a separate memory pool, and it 09:46
had a high-speed bus into the graphics DRAM. It 09:46
used it purely as a memory. 09:46
There was a memory hub as well that 09:46
could direct rendering into a system memory at a 09:46
slower rate. 09:46
Q. Did the R400 have memory in addition 09:46
to graphics DRAM? 09:47
A. Again, the graphics DRAM was not part 09:47
of the R400. It was a separate chip, external. 09:47
It had internal memories and, as well, it had 09:47
access to system memory. 09:47
Q. What internal memory did the R400 09:47
use? 09:47
A. The R400 had various caches as well 09:47
as FIFOs and buffers that would join different 09:47
blocks. There were fairly small pieces of 09:47
memory. 09:47
Q. Were general purpose registers within 09:47

Page 25

the R400 design? 09:47
A. Yes. 09:48
Q. Were those part of the various caches 09:48
and -- 09:48
A. Yes. 09:48
Q. -- buffers? 09:48
A. Yes. 09:48
Q. Was the R400 ever taped out? 09:48
A. It was never taped out as the R400. 09:48
The design evolved and was eventually taped out 09:48
as either Xenos as well as the R600. 09:48
Q. Was the R500 ever taped out? 09:48
A. The R500 was never taped out. It 09:48
evolved into the R600, which was taped out. 09:48
Q. Was the R600 based off the R500 or 09:48
Xenos? 09:49
A. It was based off the R500. 09:49
Q. When did the R600 tape out? 09:49
A. Again, I'm not entirely sure. Tape 09:49
out was probably 2005/2006 time frame, but it 09:49
could have even been later than that. 09:49
Q. When was Xenos taped out? 09:49
A. Xenos shipped the end of 2005. 09:49
Tapeout was probably late 2004, although it could 09:49
have stretched into 2005. 09:49

Page 26

Q. Was there ever any attempt to tape 09:49
out the R400? 09:49

MR. PAIGE: Object to the form. 09:49

A. Not as the R400. Again, only in the 09:49
evolved design that was called something else. 09:50

Q. Was the R600 a single-chip GPU 09:50
implementation? 09:50

A. Yes. 09:50

Q. Was the R600 ATI's first single-chip 09:50
GPU implementation with a unified shader? 09:50

A. Regardless of the capabilities of the 09:50
memory in Xenos, I regard the Xenos chip as a 09:50
single-chip GPU that had the unified shader. 09:50

Q. Did ATI tape out any single-chip GPUs 09:50
with unified shaders prior to Xenos in late 2004 09:51
or early 2005? 09:51

MR. FAHRENKROG: Object to the form. 09:51

A. No. 09:51

(Gruber Exhibit 3, Article entitled 09:52
"ATI R400 GPU Cancelled," marked for 09:52
identification) 09:52

Q. You have been handed what's been 09:52
marked as Exhibit 3. It's an article posted on 09:52
neoseeker.com, entitled "ATI R400 GPU Cancelled." 09:52

A. Yes. 09:52

Page 27

Q. About midway down on the page there's 09:52
a box that says, "The R500 is the R400 which was 09:52
canceled earlier this year." 09:52
Do you see that? 09:52

A. Yes. 09:52

Q. Is that an accurate statement? 09:52

MR. FAHRENKROG: Object to the form 09:52
of the question. 09:52

A. The first sentence, yes, I believe 09:52
that. It was just a name change. From an 09:52
engineering point of view, it's pretty much the 09:52
same. It took us a while to get all the 09:53
documentation, even up to date. We continued on 09:53
with a large mixture of things that said R400 and 09:53
things that said R500. 09:53

Q. What do you mean by, It took a while 09:53
to get all the documentation up -- 09:53

A. Well, there were various places in 09:53
the documentation and in the code where you would 09:53
have R400 in the same documentation and code that 09:53
was being used in the R500 project. And it took 09:53
a while before all of those name changes and 09:53
comments were made up to date to reflect the fact 09:53
that the name change occurred. 09:53

Q. Other than changing 400 to 500, were 09:53

Page 28

there other changes made to bring the documents 09:53
up to date? 09:53

MR. PAIGE: Object to the form. 09:54

A. As any design continues, there are 09:54
small changes made in the design. Fewer as you 09:54
get closer to tapeout, because you just have to 09:54
go with what you have. 09:54

So there were probably small changes 09:54
between, you know, in the R500 that weren't in 09:54
the R400, but they were quite small. 09:54

Q. What were the changes between the 09:54
R500 and the R400? 09:54

A. I'm not aware of them off the top of 09:54
my head. It is possible that we added a 09:54
dedicated vertex cache as opposed to using the 09:54
texture cache, but I'm not entirely sure of that. 09:54
It might have been aimed as a different process 09:55
technology, for instance, but I don't think that 09:55
would affect the design that much. 09:55

Q. Why would a separate dedicated vertex 09:55
cache have been added? 09:55

A. It's part of the unified shading 09:55
system, and we want to avoid, for instance, 09:55
textures holding up -- that is pixel processing 09:55
holding up vertex processing and vertex 09:55

Page 29

processing holding up pixel processing. 09:55
So there was a concern that, you 09:55
know, we have a long latency texture, but 09:55
vertices would tend to hit more. So you might 09:55
have a situation where, you know, vertices are 09:55
waiting -- vertex processing is waiting on pixel 09:55
processing unnecessarily, because there were 09:56
certain order enforced in the data flow that was 09:56
not logically required. 09:56

Q. This concern with latency you just 09:56
mentioned, was that present in the R400? 09:56

A. The concern about latency? 09:56

Q. Yes. 09:56

A. Latency is always a concern in the 09:56
R400. One of the challenges of the unified 09:56
shader and one of the advantages of the unified 09:56
shader is it allowed us to do a more effective 09:56
job of latency hiding, especially for vertex 09:56
processing. 09:56
One feature that Microsoft was 09:56
pushing in their new API was the idea of being 09:56
able to fetch data from the vertex engine, like 09:56
being able to fetch displacements, if you were 09:57
doing a displacement map. 09:57
And normal vertex shaders have very 09:57

Page 30

small number of threads and don't do a good job 09:57
of latency hiding by unifying the vertex shader 09:57
into the pixel shader. It could take advantage 09:57
of the latency-hiding characteristics that pixel 09:57
shaders naturally have. 09:57

Q. In order for a unified shader to take 09:57
advantage of the latency-hiding characteristics, 09:57
would it need a dedicated vertex cache? 09:57

A. No, that is not a requirement. 09:57

Q. Then why was a vertex cache added to 09:57
the R500 to address latency? 09:57

MR. FAHRENKROG: Object to the form 09:57
of the question. 09:57

A. I'm not testifying that it definitely 09:57
was added. I have a memory of it being added, 09:58
but between the R500 and the R600. I don't want 09:58
to say for sure that it was in the R500. 09:58

I'm sorry, could you repeat the 09:58
question? 09:58

Q. If a vertex cache is not required to 09:58
hide latency in a unified shader, why was it 09:58
added at some time after the R400 project ended? 09:58

A. There was a concern that the pixels 09:58
were unnecessarily interfering with vertex 09:58
fetching. It was just a fear and efficiency 09:58

Page 31

concern at that point. And so it was more of a 09:58
tweak in the design than something that was 09:59
required. 09:59
It was just, you know, one of those 09:59
engineering changes that you do because you have 09:59
the opportunity to do so. 09:59
Q. In the R400, how were pixels 09:59
unnecessarily interfering with vertex fetching? 09:59
A. The texture system was strictly 09:59
ordered in the sense that you got data back in 09:59
the order that you requested. When I say "you," 09:59
you got data back, I mean the shader ALUs got 09:59
data back in that order. 09:59
But vertices and pixels have no 09:59
inherent order. However, once a vertex does a 10:00
texture fetch, it will come back after the last 10:00
texture fetch from the pixel shader. So that 10:00
imposed an order that was otherwise not required. 10:00
So if you could have the vertices 10:00
come back out of order, either with a separate 10:00
cache or by changing the requirements of the 10:00
texture system, you could break that dependency 10:00
that was otherwise not required. 10:00
10:01

Page 32

(Gruber Exhibit 4, Document 10:01
entitled "Respondents' Invalidity Contentions," 10:01
marked for identification) 10:01

Q. You have been handed what's been 10:01
marked as Exhibit 4. 10:01

A. Yes. 10:01

Q. It's excerpts from a document called, 10:01
"Respondents' Invalidity Contentions." Have you 10:01
seen the Respondents' Invalidity Contentions? 10:01

A. No. 10:01

Q. If you would turn to the page at the 10:01
bottom labeled 252. 10:01

When was the R400 offered for sale to 10:01
Microsoft? 10:01

MR. PAIGE: Objection. Foundation. 10:01

A. I do not know if the R400 per se was 10:01
offered for sale to Microsoft. 10:02

Q. Do you know what the basis for the 10:02
statement in Respondents' Invalidity Contentions 10:02
stating that the R400 was offered for sale to 10:02
Microsoft in the spring of 2002 was? 10:02

MR. PAIGE: Let me interpose an 10:02
objection here. If there's any information that 10:02
you've learned from communicating with attorneys, 10:02
I would instruct you not to include that in your 10:02

Page 33

answer. But, otherwise, you may answer the 10:02
question. 10:02

A. What was the question again? 10:02

Q. Do you know what the basis for this 10:02
statement in Respondents' Invalidity Contentions 10:02
is? 10:02

A. Could you show me where on the page 10:02
the statement is? 10:02

Q. Bottom of Page 252. 10:02

A. "Upon information and belief offered 10:02
for sale in the spring of 2002." I don't know. 10:02
I can't answer to that. 10:02

MR. HIGGINS: Let's take a quick 10:03
break. 10:03

VIDEOGRAPHER: The time is 10:03 a.m. 10:03
At this time we'll pause recording and go off the 10:03
record. 10:03

(Recess) 10:13

VIDEOGRAPHER: The time is 10:13 a.m. 10:13
We are on the record and recording. 10:13

MR. FAHRENKROG: If I could jump in 10:13
real quick and just designate the transcript 10:13
Confidential Business Information of AMD. 10:13

BY MR. HIGGINS: 10:13

Q. If you could turn to Exhibit 2, 10:13

Page 34

Page 10. So as AMD's 30(b)(6) witness, you have 10:13
an obligation to investigate -- perform an 10:13
investigation as to Topic No. 4 as to whether 10:13
there was a sale of the R400 to Microsoft, and I 10:13
just want to confirm that it is AMD's position 10:13
that it does not know if an offer for sale of the 10:14
R400 to Microsoft occurred? 10:14

MR. FAHRENKROG: Let me jump in on 10:14
that. That seems more of a question to me. It's 10:14
AMD's position that we have objected to Topic 10:14
No. 4 and the scope of it and him testifying 10:14
about information related to Microsoft. So he's 10:14
not here to testify about that. It's not AMD's 10:14
position that AMD does not know when it made an 10:14
offer for sale to Microsoft. 10:14

MR. HIGGINS: The objection to the 10:14
depo topic didn't cover -- didn't state that you 10:14
weren't going to produce the witness to state 10:14
whether or not an offer for sale to Microsoft 10:14
occurred. Microsoft information, whether it's 10:14
confidential or not, is not the subject of that 10:14
question. 10:14

MR. FAHRENKROG: Well, you can ask 10:14
him questions around the knowledge he has. And I 10:14
would let you know that it's not AMD's position 10:14

Page 35

that we don't know whether an offer for sale was 10:14
made. 10:14
MR. CHATTERJEE: He's a 30(b)(6) 10:14
witness. Either they have a position or they 10:14
don't. Are you saying that AMD has no position? 10:15
MR. FAHRENKROG: I am not taking a 10:15
position one way or the other. I'm telling you 10:15
that your statement of our position is incorrect. 10:15
What I'm saying also is that we object to this 10:15
topic -- 10:15
MR. CHATTERJEE: If that's the case, 10:15
the witness needs to answer it, not you. 10:15
MR. FAHRENKROG: I'm stating our 10:15
position around our objection. We've objected to 10:15
this topic. We don't think it's an appropriate 10:15
topic here. He's not here to testify on it. 10:15
That's our objection. You can ask him the 10:15
question. 10:15
MR. CHATTERJEE: Okay. Go ahead and 10:15
ask the question. 10:15
BY MR. HIGGINS: 10:15
Q. As AMD's 30(b)(6) witness, was the 10:15
R400 offered for sale to Microsoft? 10:15
A. I don't have personal knowledge of 10:15
that, but I believe that within ATI there is such 10:15

Page 36

knowledge. 10:15
Q. Prior to today's depo, did you 10:15
perform any investigation as to whether that sale 10:15
of the R400 occurred? 10:15
A. No, I do not. 10:15
Q. What information would you have to 10:15
review in order to make the determination of 10:16
whether an offer for sale of the R400 was made to 10:16
Microsoft? 10:16
A. I think we'd have to go back into 10:16
records, perhaps e-mails, both internal and 10:16
between ATI and Microsoft to come up with the 10:16
date and the details of the offer for sale. 10:16
Q. So there was an offer for sale to 10:16
Microsoft? 10:16
A. Oh, there was definitely an offer for 10:16
sale. I'm not sure if it was of the R400 in 10:16
particular or some derivative of the R400. 10:16
I expect that there was an offer for 10:16
sale of the R400, but I don't know when that 10:16
occurred either. 10:16
Q. And when you say that you expect 10:16
there was an offer for sale for the R400, do you 10:17
mean to Microsoft or to another party? 10:17
A. No, to Microsoft. Again, I was not 10:17

Page 37

-- I can't talk from personal knowledge here, but 10:17
my experience in these type of things, because I 10:17
was involved in an earlier attempt to sell IP to 10:17
Microsoft, is that we tried to sell what we have. 10:17
What typically happens is that 10:17
there's a negotiation back and forth, and there 10:17
are requests for differences. 10:17
Q. Did this back and forth occur between 10:17
ATI and Microsoft? 10:17
A. You know, I can't really say. As I 10:17
said, I think that I'd have to do some 10:17
investigation into the matter that I haven't done 10:17
at this point to provide a definitive answer. 10:18
Q. Was the R400 offered for sale to any 10:18
other party? 10:18
A. Not that I know of. But again, I 10:18
haven't done an investigation into that topic. 10:18
Q. Did Microsoft reject ATI's offer to 10:18
sell the R400? 10:18
MR. FAHRENKROG: Object to the form. 10:18
A. I really can't answer. 10:18
Q. Do you not know whether Microsoft 10:18
rejected an offer for sale of the R400? 10:18
A. I do not know at this time. I 10:18
presume that there is documentation within ATI on 10:18

Page 38

this negotiation, but I have not seen it. 10:19

Q. If there was a discussion with Microsoft with respect to an offer for sale of the R400, as the head architect, would you have been involved in that discussion? 10:19 10:19 10:19 10:19

A. Not necessarily. I mean, it's possible that I was, but it's also possible that it was done more at a business level. 10:19 10:19 10:19

Q. When did negotiations with Microsoft for the Xenos chip begin? 10:19 10:19

A. I don't have any recollection of that. I may not have been directly involved with it. Again, I'm sure there are documentation within ATI, but I haven't seen it. 10:19 10:19 10:19 10:19

Q. Did the Xenos project begin after the R400 was offered for sale to Microsoft? 10:20 10:20

MR. PAIGE: Object to form. 10:20

A. I can't answer that, because I do not know the date of when or if the R400 was offered for sale to Microsoft. 10:20 10:20 10:20

Q. Would the R400 have been offered for sale to Microsoft after the Xenos project had already started? 10:20 10:20 10:20

MR. PAIGE: Object to the form. 10:20

MR. FAHRENKROG: Object to the form. 10:20

Page 39

A. No. I think that the Xenos chip 10:20
would have been started in response to any offer 10:20
or negotiation with the R400, because the Xenos 10:20
chip would have simply been a response to the 10:21
offer, and whatever modifications Microsoft 10:21
wanted in the database. 10:21

Q. And when did work on the Xenos chip 10:21
begin? 10:21

A. Early 2003. 10:21

Q. When did discussions with Microsoft 10:21
regarding the Xenos chip begin? 10:21

MR. FAHRENKROG: Object to the form. 10:21

A. I don't know. I can't answer that. 10:21

Q. Would those discussions have happened 10:21
prior to the Xenos project beginning? 10:21

MR. PAIGE: Object to the form. 10:21

A. Yes, they would. 10:21

Q. Do you have an idea how much earlier 10:21
in time those negotiations would have started 10:22
before ATI would have begun the project? 10:22

MR. FAHRENKROG: Object to the form. 10:22

A. No. It's purely a function of how 10:22
well the negotiations would have gone. It would 10:22
clearly have been measured in months, though, 10:22
because nothing would have gone faster than that. 10:22

Page 40

Q. Why would nothing have gone faster 10:22
than that? 10:22

A. Well, to agree to, you know, make a 10:22
special chip for Microsoft is a substantial 10:22
decision. That simply takes a certain period of 10:22
time. You also have to check the technical 10:22
feasibility of doing whatever those changes were. 10:22

So that isn't a short time period. 10:22

(Gruber Exhibit 5, Document 10:23
entitled "Non-Disclosure Agreement," Bates 10:23
AMDITC-NVIDEA_0000157, marked for identification) 10:23

Q. You have been handed what's been 10:23
marked as Exhibit 5, a document entitled 10:23
"Non-Disclosure Agreement," with a Bates label 10:23
AMDITC-NVIDIA_0000157. 10:23

Have you seen this document before? 10:23

A. Not to my knowledge. 10:23

Q. Do you see at the top where it says, 10:23
"To protect confidential information, ATI 10:24
Technologies, Inc., and its subsidiaries ('ATI'), 10:24
and University of Waterloo"? 10:24

A. Yes. 10:24

Q. What was ATI working on with the 10:24
University of Waterloo? 10:24

MR. FAHRENKROG: Object to the form. 10:24

Page 41

Foundation. 10:24

A. My recollection is Dr. McCool had 10:24
some technology involved with shading languages. 10:24

Q. What do you mean by "technology 10:24
involved with shading languages"? 10:24

MR. FAHRENKROG: Objection. Outside 10:24
the scope of the 30(b)(6) notice. 10:24

A. I don't recall much of the details of 10:24
this. I just remember it had something to do 10:24
with shading languages. 10:24

Q. Do you know what information about 10:24
the R400 was disclosed to the University of 10:24
Waterloo? 10:25

A. I just don't have any recollection 10:25
after how many years it has been, 13 years. I 10:25
remember the name Michael McCool. 10:25

Q. What types of shading languages were 10:25
used with the R400? 10:25

A. Well, generally we were aimed at 10:25
implementing the Microsoft DX9 assembly language, 10:25
shader model 3.0. 10:25

Q. Did the R400 ever implement Microsoft 10:26
DX9? 10:26

A. Under simulation, it did. I mean it 10:26
was never produced, but we demonstrated working 10:26

Page 42

shaders. 10:26

Q. Were working shaders demonstrated to 10:26
the University of Waterloo? 10:26

A. I don't know. 10:26

Q. What was Michael Doggett's role on 10:26
the R400? 10:26

A. Michael Doggett was on the 10:26
architecture team. I think he did a number of 10:26
things. I specifically remember him working in 10:26
the texture system, but I think that he did other 10:26
things as well. 10:26

Q. Did he do any work on the shading 10:26
language? 10:26

A. I don't recall. Michael also acted 10:26
as kind of a representative to academia. So even 10:27
if it wasn't his particular area, he would have 10:27
been involved in this. 10:27

Q. Did he have agreements with other 10:27
academic institutions with respect to work being 10:27
performed on the R400? 10:27

MR. FAHRENKROG: Object to form. 10:27
Outside the scope. 10:28

MR. PAIGE: Lacks foundation. 10:27

A. I don't recall. It's possible. 10:27

Q. Did you do any work with academic 10:27

Page 43

institutions while on the R400 project? 10:27

A. I remember a visit to Stanford, and 10:27
talking to a professor there who later went to 10:28
work for NVIDIA, but I forgot his name. 10:28
Professor Hanrahan and his team. But nothing of 10:28
substance came from that. 10:28

Q. What did you talk to the professor at 10:28
Stanford about? 10:28

MR. FAHRENKROG: Object to form. 10:28
Outside the scope. 10:28

A. I don't recall. It had something to 10:28
do with shaders and GPUs. But beyond that, I 10:28
don't recall. Stream processors. I remember 10:28
that. 10:28

Q. What was the date that you spoke with 10:28
the professor at Stanford? 10:28

MR. FAHRENKROG: Object to form. 10:28
Outside the scope. 10:28

A. I just don't remember. 10:28

Q. Do you know if you met with the 10:29
professor at Stanford before or after the R400 10:29
project started? 10:29

MR. FAHRENKROG: Object to form. 10:29
Outside the scope. 10:29

A. I can't even give you an answer on 10:29

Page 44

that I'm afraid. 10:29

MR. FAHRENKROG: Please give me a chance to make my objection. 10:29 10:29

THE WITNESS: I'm sorry. 10:29

MR. FAHRENKROG: Thank you. 10:29

Q. What about your discussions with the professor at Stanford with respect to stream processors, do you remember? 10:29 10:29 10:29

MR. FAHRENKROG: Object to form. Outside the scope. 10:29 10:29

A. I just don't have a strong memory of that at all. I remember the trip. I remember talking to a couple of his grad students as well, but I don't recall the nature of the conversation. 10:29 10:29 10:29 10:29 10:29

Q. Do you remember the names of those grad students that you spoke with? 10:29 10:29

A. I believe one's last name was Proudfoot. 10:29 10:30

Q. Proudfoot? 10:30

A. (Witness nods). 10:30

Q. Do you remember any other names? 10:30

A. It shouldn't be hard to track that person down. 10:30 10:30

No. Oh, I remember they were working 10:30

Page 45

on a shading language called Brook. 10:30

Q. B-r-o-o-k? 10:30

A. Yes. 10:30

Q. What was the Brook shading language? 10:30

A. I don't recall anymore. It's amazing 10:30
I came up with that. 10:30

Q. Was any of the technology from the 10:30
Stanford stream processors incorporated in the 10:31
R400? 10:31

MR. FAHRENKROG: Object to the form. 10:31

MR. PAIGE: Join in the objection. 10:31

A. No. 10:31

Q. Did you say no? 10:31

A. Yes, no. 10:31

Q. Is the stream processor that Stanford 10:31
developed different than the unified shader in 10:31
the R400? 10:31

MR. FAHRENKROG: Object on form. 10:31
Lacks foundation. Outside the scope of the 10:31
notice. 10:31

A. I'm not familiar enough with the 10:31
Stanford design to say one way or the other. 10:31

Q. What is your understanding of the 10:31
stream processor? 10:32

MR. FAHRENKROG: Object to form. 10:32

Page 46

Outside the scope. Lacks foundation. 10:32
A. A stream processor is generally something that works on a small amount of incoming data, processes -- does a lot of processing, and then outputs a small amount of output data. 10:32 10:32 10:32 10:32 10:32
Q. Was the Stanford stream processor a GPU? 10:32 10:32
MR. PAIGE: Object to the form. 10:32
MR. FAHRENKROG: Object to the form. 10:32
A. I believe that the goal was trying to see how well GPUs act as stream processors, and whether there are modifications that could make them into better stream processors. 10:32 10:33 10:33 10:33
Q. Did you begin working at ATI in 1994? 10:33
A. I did. 10:33
Q. What initial position did you hold? 10:33
A. I was the lead architect on the ATI Rage product. Lead 3D architect. 10:33 10:33
Q. How long did you work on the ATI Rage product? 10:34 10:34
A. The Rage were a series of products. It probably went on for five years. 10:34 10:34
Q. Do you recall what you worked on after the ATI Rage products? 10:34 10:34

Page 47

A. Then we switched on to the next 10:34
family, which were called Radeons, Radeon 100, 10:34
Radeon 200, called the R100 and R200 as well 10:34
internally. 10:34

Q. Did you work on the Radeon R100? 10:34

A. I did. 10:34

Q. Did you work on the R200? 10:34

A. I did. 10:34

Q. Did you work on the R300? 10:34

A. I did not. 10:34

Q. Why did you not work on the R300? 10:34

A. We acquired a group on the West Coast 10:34
from a company called ARTX. I guess we acquired 10:34
the company ARTX. And at that point we went to 10:34
the leapfrogging design teams, where the R300 10:35
would be done by the West Coast team, and the 10:35
R200 and R400 would be done by the East Coast 10:35
team. 10:35
Again, with Orlando -- at some point 10:35
in here we moved vertex processing into the GPU. 10:35
I'm not sure whether that was in the R100 or the 10:35
R200. The vertex processor was done by the 10:35
Orlando group, and that was done for both the 10:35
R200 and the R400 by the Orlando group. 10:35

Q. While you were at ATI, did they have 10:35

Page 48

any formal system to track the status of 10:35
projects? 10:35

A. Yeah. There was status tracking. 10:35
I'm not sure if it was formal in, you know, a 10:35
status tracker tool or via just, you know, 10:36
Microsoft Project, Microsoft spreadsheets. But 10:36
those certainly existed. 10:36

Q. Was there any status tracking done 10:36
for the R400? 10:36

A. Yes. 10:36

Q. Do you know what form the status 10:36
tracking of the R400 was in? 10:36

A. I don't recall. I believe it was 10:36
Microsoft Project, but I could not say 100 10:36
percent. 10:36

Q. What types of things go into the 10:36
status tracker that was done in Microsoft 10:36
Project? 10:36

A. The various milestones, how far we 10:36
were from the milestones, the resources 10:36
associated with each of the blocks, the number of 10:36
tests written and passed, both in the emulator 10:37
and in the RTL code. 10:37

Q. If a milestone was not met, was it 10:37
rescheduled to a later point in time in the 10:37

Page 49

tracking system? 10:37
MR. PAIGE: Object to the form. 10:37
A. Yes. The milestone was adjusted. 10:37
Q. Was there a milestone for the tapeout 10:37
of the R400? 10:37
A. I can't speak to direct knowledge, 10:37
but I expect that there was. 10:37
Q. Do you know how many times the 10:37
milestone of the tapeout for the R400 was 10:37
changed? 10:37
A. I do not. 10:37
Q. What is the best source of 10:37
information that would have been at ATI to 10:38
determine what the status of a given project was 10:38
at a point in time? 10:38
MR. FAHRENKROG: Object to the form. 10:38
A. It would have been this status 10:38
tracker, as I say, I believe in Microsoft 10:38
Project. I'm not sure if those were saved or 10:38
under revision control. So I can't say whether 10:38
that's available now. 10:38
Q. If someone viewed the status tracker 10:38
of the R400, how would they confirm what stage of 10:38
the development the R400 was in at a point in 10:38
time? 10:38

Page 50

MR. PAIGE: Object to the form. 10:38

A. I think they would be able to see achievements against milestones. 10:38 10:38

Q. Would there be any kind of documentation they could go to to confirm the entry in the milestones? 10:38 10:39 10:39

MR. FAHRENKROG: Object to the form. 10:39

A. I don't know. 10:39

Q. What is the most accurate source of information that describes the function of the R400? 10:39 10:39 10:39

A. I would say the top level and block-level specs are a good starting point for understanding. The most accurate would be the RTL code, which would be the definitive answer on how something works. 10:39 10:39 10:39 10:39 10:39

Q. Are there differences between the top level and block-level specs for the R400 and the RTL code for the R400? 10:39 10:39 10:40

MR. PAIGE: Object to the form. Lacks foundation. 10:40 10:40

A. Certainly not intentionally. But in some cases, I think there are differences. 10:40 10:40

Q. Are you aware of any differences between the top level and block-level specs for 10:40 10:40

Page 51

the R400 and the RTL code for the R400? 10:40

A. I'm not aware of any -- well, it 10:40
depends on what version of the spec you are 10:40
looking at. I certainly looked at back-rev 10:40
versions of specs that I could see have like 10:40
known typos in them that are clearly not the same 10:40
as what's in the RTL. 10:40

But I don't, in the most up-to-date 10:40
specs, I don't know of any difference that was 10:40
present, but I suspect that there were still 10:41
differences. 10:41

Q. How can you tell what the most 10:41
up-to-date ATI specification is for the R400? 10:41

MR. FAHRENKROG: Object to the form. 10:41

A. I would just look at the last one in 10:41
revision control. 10:41

Q. Is the revision control included in 10:41
each document? 10:41

A. There is some revision control, but 10:41
you had to, you know, manually add it into the, 10:41
like, the front page of the document. A better 10:41
source is the edit date or what comes out of the 10:41
revision control when it was checked in, which 10:41
will automatically assign, you know, order 10:41
things. 10:42

Page 52

MR. HIGGINS: We have to change tape. 10:42
So take a break. 10:42

VIDEOGRAPHER: The time is 10:42 a.m. 10:42
At this time we will end cassette number one. 10:42
We'll stop recording and go off the record. 10:42

(Recess) 10:49

VIDEOGRAPHER: The time is 10:50 a.m. 10:50
At this time we are beginning cassette number 10:50
two. We are on the record. We are recording. 10:50

BY MR. HIGGINS: 10:50

Q. Are the R400 specification documents 10:50
or the R400 RTL a better indication of the status 10:50
and development of the R400 project? 10:50

MR. FAHRENKROG: Object to the form. 10:50
Vague. Overly broad. 10:50

A. The RTL is more accurate in terms of 10:50
functionality, that is functionality at that 10:51
time. The documentation gives you information 10:51
beyond that, that in some ways goes to status as 10:51
well. 10:51

But I'd say that the accuracy of the 10:51
RTL is probably the best guide to where you are 10:51
in terms of full implementation. 10:51

Q. If a feature is not included in the 10:51
RTL code, could it be implemented in the GPU? 10:51

Page 53

MR. FAHRENKROG: Object to the form. 10:51

A. Anything not in the RTL code is not 10:51
in the GPU. 10:52

Q. Did the R400 include a primitive 10:52
assembly block? 10:52

A. Yes, it did. 10:52

Q. What components were included in the 10:52
primitive assembly block? 10:52

A. The primitive assembly block managed 10:52
the parameter cache. It also performed setup 10:52
operations for the triangle. It did clipping. 10:52
It did viewport transform. 10:52

Q. Did it do anything else? 10:52

A. It did setup for the interpolators. 10:52
So it created the barycentric values and the Z, 10:52
that is the depth interpolation as well. 10:53

Q. What inputs was the primitive 10:53
assembly block designed to have? 10:53

A. The primitive assembly block got 10:53
indices to initiate, to kind of track whether the 10:53
vertex was already in the parameter cache or not. 10:53

It also got input from positions that 10:53
it would use as input to the whole setup function 10:53
and the clipping and the viewport transform. 10:53

Q. From what units or modules would the 10:53

Page 54

primitive assembly block receive these inputs 10:53
from? 10:54

A. So the indices would be fetched from 10:54
memory. The positions would come via the shader 10:54
through some kind of position output buffer. 10:54

Q. What units or modules would the 10:54
primitive assembly block output data? 10:54

A. It would output to the raster engine 10:54
the clipped triangles, as well as the Z block 10:54
information. It would also talk to the position 10:55
cache to retrieve the particular positions as 10:55
input it was interested in. Although, I'm not 10:55
sure the position cache may actually have been 10:55
inside the primitive assembly block, so it 10:55
wouldn't have to talk to an external block. 10:55

Q. Did the primitive assembly block 10:55
output to any other module other than the 10:55
rasterizer? 10:55

A. Any other than the rasterizer? Oh, 10:55
well, sure, because it would have to output 10:55
basically the indices that would have to be 10:55
transformed into the shader. So if it missed in 10:56
the parameter cache, it would create a vector of 10:56
indices that would be loaded through the 10:56
interpolator block. 10:56

Page 55

It was just a data path into the 10:56
shader, and it would talk to the sequencer to 10:56
tell it, okay, I have a vector of vertices ready 10:56
to be transformed. 10:56

Q. What is a vector of vertices? 10:56

A. The shader works in a SIMD fashion. 10:56
That's single instruction multiple data. So by a 10:56
"vector," I mean the group of vertices that will 10:56
be processed as a unit. Some people might call 10:56
that a thread once it gets into the shader. 10:56

In this case, each vertex is 10:57
represented by an index into a vertex buffer so 10:57
the rest of the vertex can be fetched based on 10:57
that index. 10:57

Q. Is there any difference between 10:57
vector of vertices in a vertex thread? 10:57

A. Under some definitions, no. But 10:57
"thread" can be used in a lot of different ways. 10:57
So I just use the term "vector," because that's 10:57
what most of the R400 documentation uses it. 10:57

I think "thread" is, at least for 10:57
some meanings of a thread, that's reasonable. 10:57

Q. With respect to the R400, is a 10:57
"vertex thread" the same thing as a "vector of 10:58
vertices"? 10:58

Page 56

MR. FAHRENKROG: Object to the form. 10:58
Asked and answered. 10:58

A. The only distinction that I'd make is 10:58
that something is generally called a thread once 10:58
it acquires control flow information about it and 10:58
is in the process of being executed. 10:58

In the case of a vector of vertices, 10:58
these may be queued up for input into the shader 10:58
system, and may not have acquired any kind of 10:58
control flow information or temporary storage 10:58
available, attached to them, that we normally 10:58
think of when we say a thread, meaning like a 10:58
thread of execution. 10:58

There is no, you know, these 10:58
vertices, as they come out of the PA, they don't 10:58
have any, like, program counter associated with 10:59
them. 10:59

Q. Did the vector of vertices get a 10:59
program counter when they enter the shader 10:59
processor? 10:59

A. Yes, they do. 10:59

Q. When did the development of the 10:59
primitive assembly block begin? 10:59

A. Well, that was part of the initial 10:59
R400 plan. So I don't think RTL work on the 10:59

Page 57

block began in late 2000, early 2001. The 10:59
initial design of the block did begin in that 11:00
time frame. 11:00

Q. Was the final design of the primitive 11:00
assembly block completed as part of the R400 11:00
project or as part of the Xenos project? 11:00

MR. PAIGE: Object to the form. 11:00

A. Well, I'm sure the primitive assembly 11:00
block evolved as any design evolves through time. 11:00
But it was largely completed in the R400 time 11:00
frame. I don't think that was one of the blocks 11:00
that caused the R400 to be delayed. 11:00

Q. What were the blocks that caused the 11:00
R400 to be delayed? 11:00

A. The primary block that had an issue 11:00
was the RB, the render back-end. 11:00

Q. What was the issue with the render 11:01
back-end? 11:01

A. The caching system for the 11:01
decompressed logic just took a long time for us 11:01
to actually get working. 11:01

Q. Why did it take a long time to get 11:01
working? 11:01

A. Sometimes you have too ambitious a 11:01
design. Sometimes you have less competent 11:01

Page 58

engineers than you thought you had. 11:01
Q. Were there any other blocks in the 11:01
R400 that caused it to be delayed? 11:01
A. The RB sticks out in my mind as the 11:01
long pole block. I don't recall any issues with 11:01
other blocks, whether there were, you know, other 11:01
poles that also meant that we couldn't make the 11:01
particular market window. 11:02
Q. Did the R400 have a rasterizer? 11:02
A. Yes, it did. 11:02
Q. Where was the rasterizer in the R400? 11:02
A. The rasterizer took the output of the 11:02
primitive assembly. This is not the indices, but 11:02
the triangles that are the result of the output 11:02
of the clipper and the setup unit within the PA. 11:02
And then it fed the sequencer with the resulting 11:02
pixels. It would produce what we called 11:02
barycentric coordinates per pixel. So what we 11:02
called an I and a J were a representative pixel, 11:03
as well as an X/Y location. 11:03
Q. What components were included within 11:03
the rasterizer? 11:03
A. Well, there were edge functions that 11:03
determined whether a pixel was inside or out. So 11:03
some kind of triangle walker. 11:03

Page 59

There was also an interface to the RB 11:03
and back, which would do the Z processing, the 11:03
depth processing logic, to determine whether a 11:03
pixel should be culled or not rather than go to 11:03
the shader. 11:04

Q. When did the development of the 11:04
rasterizer in the R400 begin? 11:04

A. Again, the design took place or the 11:04
initial architecture in 2000, in late 2000 and 11:04
early 2001. My recollection is the rasterizer 11:04
itself was largely based off of previous versions 11:04
of rasterizers. So it was not -- that wasn't 11:04
that challenging of a block. 11:04

Q. Which previous versions was the R400 11:04
rasterizer based on? 11:04

A. It would have come from the R200 11:04
rasterizer. 11:05

Q. Did the R400 include a scan 11:05
converter? 11:05

A. Yeah, by "rasterizer" and "scan 11:05
converter," we mean very similar things. 11:05

Q. Are there different blocks in the 11:05
R400 for a rasterizer and scan converter? 11:05

A. No. 11:05

Q. Did the R400 include a sequencer? 11:05

Page 60

A. Yes, it did. 11:05

Q. And where was the sequencer in the R400? 11:05 11:05

A. So the sequencer was very tightly tied to the shader system, that it took input from the rasterizer and from the primitive assembly and setup blocks. The primitive assembly really. 11:05 11:05 11:05 11:05 11:06

Q. What inputs did the sequencer receive from primitive assembly? 11:06 11:06

A. So it received this vector of indices that represented triangles -- sorry, represented vertices, that hadn't been transformed yet. 11:06 11:06 11:06

Q. What did it do with the vector of vertices when it receives them? 11:06 11:06

A. So it would try to allocate a thread for them. That is, it would try to allocate GPR space and it would try to allocate a control storage space associated with that vector of vertices so it could be submitted for processing to the GPU -- to the ALUs. Excuse me. 11:06 11:06 11:06 11:06 11:06 11:07

Q. How did the sequencer allocate storage space to the vector of vertices? 11:07 11:07

A. So we had two different types of storage space, at least two different types 11:07 11:07

Page 61

within the sequencer it was put further. There 11:07
was space for the control information, which are 11:07
things like the program counter, predicates, a 11:07
return stack. And we had space for temporaries 11:07
in the GPRs. 11:07

In the control flow, we had a pool of 11:08
control flow resources that were allocated on a 11:08
FIFO basis dedicated to the vertices. 11:08

In the GPRs, there was a pool of 11:08
registers, and over time the registers could go 11:08
either to vertices or pixels. But at any given 11:08
period of time, there was a circular buffer 11:08
associated with vertices and a separate circular 11:08
buffer associated with pixels, and we'd allocate 11:08
space out of the circular buffer, again, in 11:08
order, based on following this ring buffer for 11:08
that particular vector of vertices based on state 11:08
information that told us how many registers this 11:09
particular program -- the particular vertex 11:09
shader program that was loaded at the time 11:09
required. 11:09

Q. Did the sequencer in the R400 assign 11:09
a program counter to a vector of vertices? 11:09

A. Yes, it's based on state. Right. 11:09
There was some state that said, Here is where the 11:09

Page 62

starting program counter is for vertex shader, 11:09
for the vertex shader. 11:09

So when a vector was loaded into the 11:09
thread buffer, that would be the initial program 11:09
counter associated with that vector. 11:09

Q. Is the sequencer part of the shader 11:09
processor? 11:10

A. It's intimately tied to the shader 11:10
processor. We, just in terms of organization in 11:10
the R400, we broke it up into two blocks; a 11:10
sequencer and a processor. Where the sequencer 11:10
would have all of the control flow information 11:10
and do more sophisticated things, while the 11:10
shader processor was the -- the SP block was more 11:10
of a pure data path. 11:10

But if you were talking in a more 11:10
generic level, I would include the sequencer and 11:10
the shader processor. 11:10

Q. Which unit, the sequencer or the 11:10
shader processor, assigned a program counter to 11:10
the vector of vertices? 11:10

A. The sequencer would assign the 11:10
program counter, and would manage how that 11:11
program counter got incremented over time. 11:11

Q. Where was the program counter stored 11:11

Page 63

in the R400? 11:11
A. So the program counter was part of 11:11
this thread pool state that was inside of the 11:11
sequencer. 11:11
Q. And where was the thread pool state 11:11
stored? 11:11
A. It was in some local memories within 11:11
the sequencer. 11:11
Q. Was there a name for the local 11:11
memories within the sequencer where the thread 11:11
pool state was stored? 11:11
A. I don't recall whether we had a 11:11
particular name for it. We had two different 11:11
pools of memories in the sequencer associated 11:11
with thread state or I really should say that a 11:11
single thread was split over two different types 11:12
of memories, and one of the memories was only 11:12
single ported and one of the memories was 11:12
multiported, because the information associated 11:12
with the thread in the multiported memory was 11:12
needed more frequently than the information in 11:12
the single pool of memory. 11:12
So we might have had a different name 11:12
for them. Like we might have called one state 11:12
memory and one register memory, but I'd have to 11:12

Page 64

go back to the actual documentation. I mean, I 11:12
could look at -- I could probably -- I don't have 11:12
it here -- to come up with the actual names that 11:12
we referred each of those pools of memory by. 11:12

Q. When did development of the R400 11:12
sequencer begin? 11:12

A. Again, the initial development began 11:12
with the initial R400 in late 2000, continuing 11:13
into 2001. Because this was new, that is the 11:13
unified shader was new and the sequencer was the 11:13
key part of it, that was certainly a focus early 11:13
on. 11:13

Q. When was the development of the 11:13
sequencer finished? 11:13

A. Well, again, it evolves over time. 11:13
But I think it reached something close to its 11:13
final state in the spring of 2002. 11:13

Q. Were changes made to the sequencer in 11:13
the R400 after the spring of 2002? 11:13

A. I'm sure changes were made, yeah. 11:13
There was a continuing, you know, evolution of a 11:13
design where, you know, things generally change, 11:13
even, you know, up to a few months prior to 11:14
tapeout. 11:14

The overall architecture and the 11:14

Page 65

basic functionality was established by the spring 11:14
of 2002. 11:14

Q. Were there changes made to the 11:14
sequencer during the Xenos project? 11:14

A. I don't -- well, are you asking as 11:14
part of the R400 to Xenos transition or just 11:14
throughout the Xenos project? 11:14

Q. Both. 11:14

A. I don't think there was any change 11:14
made as part of the transition, and I don't know 11:14
of any changes made through the Xenos project. 11:14

But knowing that we run tests and 11:14
sometimes bugs crop up, I wouldn't be surprised 11:14
if there were changes made. The RTL check-in 11:15
code under source control would be the best way 11:15
of assessing when a given change was made. 11:15

Q. Were there any bugs in the sequencer 11:15
code while the Xenos was in development? 11:15

MR. PAIGE: Object to the form. 11:15

A. I think my answer is similar to the 11:15
last one. I don't know offhand of bugs in the 11:15
sequencer code, but it wouldn't surprise me if 11:15
they were. And the way to establish that would 11:15
be to look at the history of check-in and the 11:15
history of changes, which generally the check-in 11:15

Page 66

comments would say, you know, oh, I'm fixing a 11:15
bug at this point. 11:15

Q. Was there a particular person in 11:15
charge of the development of the sequencer? 11:16

A. Laurent Lefebvre was the architect in 11:16
charge of it. There was probably a block lead on 11:16
it as well. I'm not sure whether Laurent did the 11:16
RTL code for it or whether somebody else was 11:16
doing the RTL code. So I'm not sure about that. 11:16
I could tell, again, by looking at the check-in. 11:16

It's probable that Laurent did a lot 11:16
of the RTL code as well. 11:16

Q. Was anyone else working on the 11:16
sequencer with Laurent? 11:16

A. I don't recall, but I could tell by 11:16
whatever the history of check-ins were. 11:16

Q. Did Laurent work on the R500 project? 11:16

A. Yes, he did. 11:17

Q. Did Laurent work on the Xenos 11:17
project? 11:17

A. I don't think so. That is, everybody 11:17
on the R400 did it at times, you know, help out 11:17
on the Xenos project, but I don't think he was 11:17
really on the Xenos project. But I can't testify 11:17
definitively one way or the other. 11:17

Page 67

(Gruber Exhibit 6, Document 11:18
entitled "Xenos Sequencer Specification, Version 11:18
3.0," Bates AMD-ITC-0020788 through 11:18
AMD-ITC-0020843, marked for identification) 11:18

Q. You have been handed what's been 11:18
marked as Exhibit 6, a document entitled "Xenos 11:18
Sequencer Specification, Version 3.0," Bates 11:18
label AMD-ITC-0020788 through 20843. 11:18
Have you seen this document before? 11:18

A. I have seen versions of this 11:18
document. I'm not sure if I've seen this 11:18
particular version. 11:18

Q. When is the last time you viewed a 11:18
version of this document? 11:18

A. Yesterday, as a matter of fact. 11:18

Q. Which version did you view yesterday? 11:18

A. I believe it was 2.02. 11:19

Q. This Version 3.0, what date does it 11:19
reflect the development of the sequencer? 11:19

A. The date is June 27, 2005. 11:19

Q. Turn to Page 6 in the revision 11:19
history. 11:19

A. Okay. 11:19

Q. Between May 1, 2003 and September 24, 11:19
2004, what work was done on the development of 11:19

Page 68

the sequencer? 11:19

A. It doesn't show any additional 11:19
changes in the specification. That doesn't mean 11:20
work wasn't continuing to be done on the RTL 11:20
database. 11:20

Q. Was work done on the sequencer RTL 11:20
database between May 1, 2003 and September 4, 11:20
2004? 11:20

A. I can't say whether the database -- 11:20
whether the RTL database changed or not. I'd 11:20
have to go into the CVS. Certainly work was 11:20
continuing on the chip. Whether the sequencer 11:20
needed additional work or not, I couldn't say. 11:20

Q. What does it mean when it says 11:20
"Updated export address table to reflect hardware 11:21
implementation"? 11:21

A. Sometimes that the hardware will find 11:21
an easier way of doing things that doesn't 11:21
actually change the functionality. And so we'll 11:21
change the spec to allow the hardware to do the 11:21
easier path, because it really doesn't matter one 11:21
way or the other. 11:21

But as it turns out, once you do the 11:21
hardware implementation, you find out that one 11:21
way is easier than the other. So the idea was 11:21

Page 69

the hardware guys went back and said, oh, can we 11:21
change the spec, because this will be easier, so 11:21
the spec was updated. 11:21

Q. Then in September 2004 did the 11:21
hardware guys go back and change the spec? 11:21

A. The hardware guys asked Laurent to 11:21
change the spec. 11:22

Q. Do you know what changes were made in 11:22
2004? 11:22

A. The detail of the changes, no. I'd 11:22
have to get into the details of the RTL code to 11:22
really understand this or to, you know, go into 11:22
the spec if -- I'd have to compare the two revs 11:22
of the spec to see what actually changed. 11:22

It looks like it's something fairly 11:22
minor. 11:22

Q. What work was done on the development 11:22
of the sequencer between September 24, 2004 and 11:22
June 27, 2005? 11:22

A. Again, it appears that there wasn't a 11:22
lot of specification changes, but there was 11:22
continued development on the chip itself, and, if 11:23
so, there might have been fixes in the RTL. 11:23

So, you know, it's not that the 11:23
architecture changed, but part of designing a 11:23

Page 70

chip is to implement the architecture. So that's 11:23
what was going on. And the spec changes become 11:23
fewer, because the changes associated with the 11:23
development of the block are mainly 11:23
implementation changes. 11:23

Q. What is meant by "Update of spec to 11:23
reflect bug fixes and features"? 11:23

A. Sometimes you will fix a bug by 11:23
changing a spec as opposed to actually making the 11:23
hardware meet the spec. Other times, a fix in 11:23
the hardware or an implementation in the hardware 11:24
will result in a small change in the 11:24
specification that, you know, nobody really cares 11:24
much about. 11:24

I don't know what this particular 11:24
refers to. Features, a lot of times, a 11:24
late-adding thing is like a debug feature, where 11:24
you'd add some additional visibility, and you'd 11:24
want that updated in the spec, but you're not 11:24
really sure at the time that you're doing the 11:24
initial architecture what those debug features 11:24
would be. 11:24

It also looks like here that, you 11:24
know, a lot of the time was simply that the spec 11:24
was fairly stable, and that Laurent added all of 11:24

Page 71

the implementation-related changes at one time. 11:25

Q. At what point did the spec become 11:25
stable? 11:25

MR. PAIGE: Object to the form. 11:25

A. As I said, the last major change that 11:25
I know about was in the spring of 2002. 11:25

Q. April 8, 2003, what were the R500 11:25
modifications that were added? 11:25

A. I don't know what this refers to, how 11:25
much of it -- how many changes were associated 11:25
with the R500. I'd have to go into the details 11:25
of this. 11:25

Q. Are you aware of any changes made to 11:25
the sequencer for the R500? 11:25

A. As I said, I have a memory that we 11:25
added a vertex cache. So fetching from the 11:26
vertex cache would certainly be something that 11:26
the sequencer would be -- would have to be aware 11:26
of. 11:26

Q. How does the sequencer interact with 11:26
the vertex cache in the R500? 11:26

A. So the vertex cache is now a second 11:26
input into the shader, and the sequencer has to 11:26
be aware of whether the vertex data has returned 11:26
or not to be able to know whether a given thread 11:26

Page 72

is ready to be launched if it's waiting for its 11:26
vertex data to come back. 11:26

Q. When would the vertex cache have been 11:26
added to the R500? 11:27

A. I am not sure. I'd have to go back 11:27
and specifically look for the change list in 11:27
various blocks to see the updated vertex cache. 11:27

Q. Does the sequencer receive any inputs 11:27
from the vertex cache? 11:27

A. Yes. It would receive a bit saying 11:27
that data is back to be able to know that that 11:27
thread has received its data. 11:27

Q. What do you mean by the "thread has 11:27
received its data"? 11:28

A. I mean, it will do a fetch 11:28
instruction. This is the same mechanism that is 11:28
used for fetching textures, and in the R400, the 11:28
same signal was used for both textures and 11:28
vertices. 11:28

In the R500, as we switched -- as we, 11:28
you know, assuming that we did move the vertex 11:28
cache functionality into a separate path, then 11:28
you'd have a separate wire saying "vertex data is 11:28
back" versus a wire saying "texture data is 11:28
back." 11:28

Page 73

In both cases, the shader program is capable of saying "I need to stall until data is back." So any shader program that did that would be marked as not ready for execution until the control bit in the sequencer winds up being set the same, based on the incoming data. 11:28 11:28 11:28 11:28 11:28 11:29

Q. Did the Xenos chip have a vertex cache? 11:29 11:29

A. I'd have to look. I'm not sure whether we added a vertex cache. I don't want to guess. 11:29 11:29 11:29

Q. Where would you have to look to determine whether the Xenos chip had a vertex cache? 11:29 11:29 11:29

A. We would have to look in the top-level Xenos document with a block diagram in it. I'm not sure that I have this available right here. It wouldn't -- well, I may be able -- it's possible that it is in the sequencer document, if it would be in the interface. 11:29 11:29 11:29 11:29 11:29 11:29

MR. FAHRENKROG: Is there a current question pending other than where would you have to look? I'm just trying to follow the track here. 11:30 11:30 11:30 11:30

MR. HIGGINS: I'm just waiting for 11:30

Page 74

him to look. 11:30
MR. FAHRENKROG: I didn't hear you 11:30
ask him. Okay. 11:30
A. Yeah, there is a vertex cache, at 11:30
least in this spec, because it does have an SQ to 11:30
VC control bus. 11:31
Q. Does this document tell you whether 11:31
the vertex cache was implemented in the Xenos or 11:31
would you have to look to the RTL to determine 11:31
that? 11:31
A. No. I think we can rely on this that 11:31
there is a vertex -- well, this is the Xenos 11:31
spec. Let me just see the dates. No, that's a 11:31
large enough change that this can be relied upon. 11:31
Q. How much of a change is large enough 11:31
where you can rely on the documents and not have 11:31
to resort to the RTL? 11:31
MR. PAIGE: Object to the form. 11:31
A. At least in this case, where there's 11:31
a substantial interface consisting of multiple 11:32
signals, tells me that while I can't say that 11:32
every detail is correct, the whole interface with 11:32
multiple signals is more than a detail. So I can 11:32
rely on that. 11:32
There are multiple references to 11:32

Page 75

vertex cache in this document. 11:32
Q. Could a feature be written in the 11:32
Xenos specification that is not ultimately 11:32
implemented in the RTL for the Xenos? 11:32
MR. FAHRENKROG: Object to the form. 11:32
MR. PAIGE: Join in the objection. 11:32
A. While it's theoretically possible, I 11:32
don't see how our documentation would be that far 11:33
wrong that that's a realistic possibility. 11:33
Q. What do you mean by "that far wrong"? 11:33
A. I mean at least in the case of the 11:33
vertex cache, if you're talking about a very 11:33
substantial feature. If you're talking about a 11:33
minor feature, it is possible that there's 11:33
something in the documentation that isn't 11:33
reflected in the chip. 11:33
But if you're talking about major 11:33
blocks, like a vertex cache, there is no way that 11:33
the documentation would have something that the 11:33
hardware would not have. 11:33
Q. When a new feature is being 11:33
implemented on a project, is it first put into a 11:33
specification document or in RTL? 11:33
MR. FAHRENKROG: Object to the form. 11:33
Overly broad. 11:33

Page 76

MR. PAIGE: Join in the objection. 11:33

A. It's first put into the specification 11:33
document generally. I will not say that that's 11:34
an absolute rule. 11:34

Q. How would you confirm whether a 11:34
feature that's put in the specification document 11:34
is ultimately implemented in the RTL? 11:34

A. To get 100 percent confirmation, 11:34
you'd have to validate that the RTL has this same 11:34
interface as it has in the underlying logic. 11:34

Q. What does it mean at the top of the 11:34
document where it says, "Originate date 9 July 11:34
2003"? 11:34

A. It's put in automatically by Word. 11:34
So it means either that the document started from 11:34
scratch then, or a copy of something was made and 11:34
then renamed at that time. 11:34

Q. Would changes such as that be 11:34
reflected in the revision history of the 11:35
document? 11:35

A. Yeah, they certainly might. For 11:35
instance, this, you know, has a revision date of 11:35
Revision 01 of 2001, even though the origin date 11:35
has 2003. 11:35

What that tells me is that even 11:35

Page 77

though a new physical file was created, it 11:35
started with the same data that was further back. 11:35

So I imagine that this has to do with 11:35
some major change, maybe when the R500 -- when 11:35
they name changed from R400 to R500, rather than 11:35
simply renaming the document, maybe it was 11:35
copied. 11:36

Q. If you look at revision history, the 11:36
R500 modifications were added on April 8, 2003, 11:36
correct? 11:36

A. Right. 11:36

Q. So why would the document be changed 11:36
on July 9, 2003 to reflect something that was 11:36
already changed on April 8, 2003? 11:36

MR. FAHRENKROG: Object to the form. 11:36

MR. PAIGE: Object to the form. 11:36

A. I would just be guessing. It's 11:36
possible that that's when we -- it was -- a copy 11:36
was made to reflect the Xenos database, you know, 11:36
and sort of have a separate copy of the spec for 11:36
Xenos versus R500. It's possible that that was 11:36
just when the forking of the Xenos spec database 11:36
occurred. 11:37

Q. Would a major change to the 11:37
specification be adding a vertex cache? 11:37

Page 78

MR. FAHRENKROG: Object to the form. 11:37
A. I don't think that simply adding a 11:37
vertex cache to the specification would result in 11:37
copying of the specification. I think it more 11:37
has to do with program management type changes as 11:37
opposed to technical changes that would have led 11:37
to this. 11:37
Q. When was the vertex cache added to 11:37
the sequencer specification? 11:37
MR. PAIGE: Object to the form. 11:37
A. Well, all I can do is go by this, 11:37
where it says -- oh, it doesn't actually say 11:38
here. I don't know. I expect it had to do with 11:38
when we added a vertex cache to the design as 11:38
part of the R500. 11:38
Q. Would adding a vertex cache generate 11:38
a new revision of the specification? 11:38
A. No, not automatically. If we had 11:38
chosen to add that at some other point in the 11:38
design cycle, it would not necessarily have led 11:38
-- oh, to a new revision or to a new origin date? 11:38
Q. A new revision in the document. 11:38
A. Oh, yeah, I expect that it would 11:38
have. 11:38
Q. Was the vertex cache added on July 9, 11:38

Page 79

2003 in this revision? 11:38

A. I don't know. There's really no way 11:38
for me to know here. 11:39

I mean, it would be more likely it 11:39
was added when Laurent says on April 8th, "adding 11:39
R500 modifications." But that's just a guess. 11:39

Q. Turn to Page 24 of the document, 11:39
Bates number 20811. Look at Section 12.1 it 11:39
says, "SP Stall Conditions." Do you see that? 11:39

A. Yes. 11:39

Q. What is meant by PS and PV? 11:39

A. Those are bypass registers. What 11:40
that means is that PS means previous scaler and 11:40
PV means previous vector, and it means the result 11:40
of the previous operation. 11:40

The bypass logic is done in the SQ 11:40
rather than in the SP. So the idea is that if an 11:40
instruction is dependent on the previous 11:40
operation, and the data has not made it to the 11:40
register file yet but is kind of in flight to the 11:40
register file -- this is only within the shader 11:40
-- and so you may do an add operation and then 11:40
another add operation, with the output of one add 11:40
operation is used as the input for the second add 11:41
operation, if those two instructions follow each 11:41

Page 80

other, the data bypasses the register file and 11:41
is, instead, sent directly from the output of one 11:41
instruction to the input of the following 11:41
instruction. 11:41

Q. What does it mean when it says "none" 11:41
in both of those entries? 11:41

A. It means that we don't have to stall 11:41
based on that use, because of the bypass logic. 11:41

Q. So on the R400 sequencer, there were 11:41
no stall conditions within -- strike that. 11:41

Within the SP of the R400, there were 11:42
no stall conditions; is that correct? 11:42

A. No. I think this is an improvement 11:42
from what was -- what at least was in early 11:42
versions of the R400. I can't say when exactly 11:42
all the stalls were taken out. 11:42

But there was a stall condition, at 11:42
least in early versions of the R400, where if you 11:42
were using only a part of, let's say, the 11:42
previous vector, you wanted a mixture of the GPR 11:42
data and the previous result. 11:42

These are all vector instructions. 11:42
So you use, say, call them the X, Y, Z and W 11:43
components or call them, ARGB, and maybe the 11:43
previous instruction updated the B component. 11:43

Page 81

And then the next instruction uses all of them, 11:43
the ARGB. 11:43

So the ARG needs to come from the 11:43
register file. The old data of the register file 11:43
is perfectly good data. It's sitting there. It 11:43
wasn't modified by the previous instruction. The 11:43
B data was modified by the previous instruction, 11:43
and has to come from the previous vector 11:43
register. 11:43

I believe the previous version of the 11:43
-- at some point in the R400, there was a stall 11:43
that was detected by the sequencer for that, and 11:43
we would read from the GPR. So we would wait for 11:43
the data to go into the GPRs, and then pull it 11:43
out. 11:44

This says, no, you don't have to 11:44
stall. You can mix the two. You can take some 11:44
data from the GPRs and some data from the 11:44
previous vector. 11:44

Q. About when was that change 11:44
implemented? 11:44

A. I don't know. I'd have to go through 11:44
it. That's a very specific change, and I don't 11:44
know when it would have occurred. 11:44

Q. Did you have to look in the 11:44

Page 82

specification documents or the RTL to figure out 11:44
when that change was made? 11:44
A. The best answer would be from the 11:44
RTL. 11:44
Q. Do you know what module of the R400 11:44
RTL you would look to? 11:44
A. It would be in the sequencer module. 11:44
Q. Do you know which module -- 11:44
A. Within the sequencer? No, I'd have 11:44
to bury into that to really figure it out. 11:44
Q. Did you write any of the R400 RTL? 11:44
A. I wrote a little of the RTL in the 11:44
shader pipe itself, in the shader pipe data path. 11:45
Q. You're able to read and write RTL? 11:45
A. Yes. 11:45
(Gruber Exhibit 7, Document 11:45
entitled "R400 Architecture Proposal Version 11:45
0.1," Bates AMD-ITC-0005407 through 11:45
AMD-ITC-0005422, marked for identification) 11:46
Q. You have been handed what's been 11:46
marked as Exhibit 7, ATI document entitled "R400 11:46
Architecture Proposal Version 0.1," Bates label 11:45
AMD-ITC-0005407 through 5422. 11:46
Have you seen this document before? 11:46
A. Yes, I have. I've seen this 11:46

Page 83

document. 11:46

Q. When is the last time you've seen 11:47
this document? 11:47

A. I saw this document yesterday. 11:47

Q. Turn to Page 6 of the document. 11:47
Under Section 1.2, what does it mean by "Memory 11:47
is the most open issue on the R400"? 11:47

A. Well, I think what he refers to here 11:47
is that a lot of the design of a chip be built 11:47
around trying to get all of the memory bandwidth 11:47
that is available out of the memory. 11:47

So when he says "It's the most open 11:47
issue," I think that you can think about that not 11:47
as saying, you know, what is the technology or 11:48
what is the design, but merely how much memory 11:48
bandwidth are we going to have available to the 11:48
R400, because that tells you how big a chip you 11:48
have to build. 11:48

Q. Was that issue ever resolved for the 11:48
R400? 11:48

A. Yes. We had a target memory 11:48
technology and a target memory bandwidth. 11:48

Q. What was the target memory 11:48
technology? 11:48

A. I would have to look at some overall 11:48

Page 84

specification of the chip that would tell me what 11:48
our bandwidth was. 11:48

Q. What do you mean by "target memory 11:48
technology"? 11:48

A. I mean, you know, there are various 11:48
generations of DDR and various speeds. So maybe 11:48
in this thing, maybe you wanted, you know, 1,600 11:48
megahertz PC DDR3. And especially -- that's more 11:49
of a system memory. In graphics, we had 11:49
specialized graphics memory, so there is like 11:49
GDDR3. 11:49

And it was a matter of timing the 11:49
release of the chip to the release of available 11:49
memory, technologies and speeds. 11:49

Q. Turn to Page 8 of the document. 11:49
Halfway down the page under heading 1.10, 11:49
"Performance," there's a sentence -- 11:49

A. I'm sorry, I may be on the wrong 11:49
page. What page am I looking for? 11:49

Q. Page 8. Under the heading 1.10 11:49
"Performance." 11:50

A. Yes. 11:50

Q. Where were the R400 team considering 11:50
a dual chip solution for the very high end? 11:50

MR. FAHRENKROG: Object to the form 11:50

Page 85

of the question. 11:50

A. Certainly the market looks for 11:50
scalability. You want to be able to win in the 11:50
high end, but you don't sell most of your chips 11:50
in the high end. So there's a business challenge 11:50
in terms of how you -- how do you win the high 11:50
end without doing a specialized chip for the high 11:50
end. 11:50

Q. How would the single-chip design be 11:50
split up into the dual-chip design? 11:51

MR. PAIGE: Object to the form. 11:51

A. It depends. Sometimes you might have 11:51
some dedicated bus connecting the two chips. In 11:51
other cases, you don't really do much special at 11:51
all. 11:51

Certainly when ATI finally came out 11:51
with multiple chip solutions, there was very 11:51
little -- I mean, I can't think of anything in 11:51
the chips that meant that the high end that was 11:51
present purely to support the dual chip solution. 11:51

So you just took one of your, you 11:51
know, medium high end chips and you put two on a 11:52
board for your very high end high-end chips. 11:52

There may be some on various register 11:52
settings that were added to the chip. And as I 11:52

Page 86

say, sometimes there would be like a bus connecting kind of the memory, the back-end memory interfaces of the two chips in, like, some possible designs. 11:52 11:52 11:52 11:52

But there was very little done in the core of the chip to support dual chip. 11:52 11:52

Q. The dual chip solution they're referring to, is that two GPUs or just implementing a single GPU across two chips? 11:52 11:52 11:52

A. What I think we finally wound up with was the dual GPU solution. It wasn't like that one of the chips wasn't a GPU. Let me just read this so that I get the context of this particular statement correct. 11:52 11:52 11:52 11:53 11:53

Yeah, I think in this case it also meant dual GPUs. 11:53 11:53

Q. Did it also include taking a single GPU and implementing it on multiple chips? 11:53 11:53

MR. FAHRENKROG: Object to the form. 11:53

A. At this point in the design, that may have been a possibility as well. That's hard to say. 11:53 11:53 11:53

Q. Was the dual chip implementation pursued at all after November of 2000? 11:53 11:53

A. I don't recall any pursuance of this 11:53

Page 87

for the R400, because any solution would also 11:54
have to have a single-chip solution as well. So 11:54
I can't recall anything like that. 11:54

There may have been some, you know, 11:54
pursuit of specialized memories, but I think that 11:54
will still be viewed as GPU memory rather than, 11:54
you know, split up a GPU. 11:54

Q. What do you mean by "specialized 11:54
memory"? Are you talking about placing some form 11:54
of memory on a chip other than the GPU? 11:54

A. No. I'm talking about what was used 11:54
as your memory pool. There was some talk about 11:55
memory chips that have some logic built into 11:55
them, you know, similar to what might have been 11:55
done in Xenos. But there's talk about standard 11:55
chips doing that as well. Whether they would 11:55
ever occur or not, who knows. 11:55

I think the fact that we really had 11:55
to go for standard chips focused the design 11:55
fairly early on on not pursuing any of those 11:55
paths. 11:55

Q. When you say "standard chips," are 11:55
you referring to a single-chip design? 11:55

A. I'm talking about memories that are 11:55
available from multiple vendors as opposed to 11:55

Page 88

contracting with a memory vendor to produce a 11:55
design specifically for your system. 11:55

Q. Was the Xenos a standard chip design? 11:56

MR. FAHRENKROG: Object to the form. 11:56

A. Yeah, the Xenos was a standard chip. 11:56
I mean, the Xenos memories were specialized 11:56
memories, the tile memory. 11:56

Q. Was the Xenos tile memory available 11:56
from multiple vendors? 11:56

A. No. 11:56

Q. So was the Xenos tile memory a 11:56
standard chip design? 11:56

MR. FAHRENKROG: Object to the form. 11:56

A. No. Well, it wasn't a standard 11:56
memory design. It might have used standard chip 11:56
design techniques to produce it. 11:56

Q. Was this Xenos tile memory on a 11:56
separate chip than the GPU? 11:57

A. Yes. It's a single package, but a 11:57
separate chip. 11:57

Q. Was this separate tile memory on 11:57
another chip considered during the R400 11:57
development? 11:57

A. I think it was probably given some 11:57
thought at some time. But the economics of it in 11:57

Page 89

the R400 market meant that we couldn't really 11:57
consider it. 11:57

Q. What was it about the economics of 11:57
the R400 market that meant you couldn't consider 11:57
it? 11:57

A. The cost of the memory was too high 11:57
for a chip that had to compete against, you know, 11:58
in the market with other GPUs, and it was 11:58
especially being tied to one memory vendor meant 11:58
that you couldn't take advantage of the 11:58
competition for memories. 11:58

So you might be stuck, even if your 11:58
GPU chip was competitive, if your memory chip 11:58
cost was not competitive, then you'd have a 11:58
business problem. 11:58

Q. Who designed the eDRAM in the Xenos 11:58
chip? 11:58

A. I don't remember. I'd have to look 11:58
that up. 11:58

Q. Was the design of the eDRAM in the 11:58
Xenos chip specified by ATI or Microsoft? 11:58

A. I think that that was an ATI design. 11:58

Q. Did ATI choose to implement an eDRAM 11:59
on a separate chip in the Xenos or was that a 11:59
feature requested by Microsoft? 11:59

Page 90

A. I have no specific knowledge of 11:59
whether this was an ATI proposal or a Microsoft 11:59
proposal. 11:59

Q. Turn back to Page 8, towards the 11:59
bottom of the page, Section 3 that says 11:59
"Schedule." Do you see that? 11:59

A. Yes. 11:59

Q. Was this schedule met? 11:59

A. No. 11:59

(Gruber Exhibit 8, Document 12:00
entitled "R400 Sequencer Specification SQ, 12:00
Version 2.02," Bates AMD-ITC-0015547 through 12:00
AMD-ITC-0015599, marked for identification) 12:01

Q. You have been handed what's been 12:01
marked as Exhibit 8, an ATI document entitled 12:01
"R400 Sequencer Specification, Version 2.02," 12:01
Bates number AMD-ITC-0015547 through 15599. 12:01

Have you seen this document before? 12:01

A. Yes. 12:01

Q. When is the last time you saw this 12:01
document? 12:01

A. Yesterday. 12:01

Q. Where in the R400 sequencer 12:01
specification document is there an algorithm or 12:01
process described related to the creation of 12:01

Page 91

threads? 12:02
MR. FAHRENKROG: Object to the form 12:02
of the question. 12:02
MR. PAIGE: Join in the objection. 12:02
A. 6.3 talks about the state and the 12:02
creation of threads. 6.3, "Implementation." 12:02
Q. What in Section 6.3 describes the 12:03
process of creating a thread in the R400? 12:03
A. Well, it talks about the buffers that 12:03
the threads would live in once they're created. 12:03
I can look elsewhere to see if there's any talk 12:03
specifically to the creation of a thread. 12:03
Q. Are threads created in two buffers? 12:03
A. Yes. I would say that when a thread 12:03
is put into that buffer, it is created; and when 12:03
it leaves the buffer, it no longer exists. 12:03
Q. How is a thread put into one of the 12:04
buffers? 12:04
A. So a thread comes from either a set 12:04
of vertices from the PA or pixels from the 12:04
rasterizer, and one of those vectors is selected 12:04
and assign a space in the thread buffer, either 12:04
in the vertex thread buffer or in the pixel 12:04
thread buffer, and all of its state bits are 12:04
initialized at that point. 12:04

Page 92

It's assigned a GPR base, which means 12:04
it has to allocate from the GPR register file 12:04
that tells it where it can keep its temporary 12:05
values. 12:05

Q. How is it determined which vector is 12:05
assigned a space in the thread buffer? 12:05

A. I'm not sure what you're asking. 12:05
You're saying if I have a pixel vector and a 12:05
vertex vector, or if you're saying which of those 12:05
do I choose, or are you asking how does it know 12:05
where it goes in the thread buffer? 12:05

Q. How do you choose? There's multiple 12:05
vectors to choose from. How do you choose which 12:05
vector goes into the thread buffer? 12:05

A. Well, the inputs are FIFO'd. So I'm 12:05
only going to choose in between a pixel vector or 12:06
the vertex vector. In other words, the pixel 12:06
vectors are ordered and the vertex vectors are 12:06
separately ordered. 12:06

As far as selection in between vertex 12:06
vectors and pixel vectors, there's an arbitration 12:06
mechanism. It's typically giving vertex priority 12:06
to handle which of them go in. But that's only 12:06
-- that's only due to the fact that there's kind 12:06
of a single-ported load into the thread buffer. 12:06

Page 93

The thread buffer resources 12:06
themselves are separate, such that you don't need 12:06
to arbitrate for, say, the state bits for pixels 12:06
versus the state bits for vertices, because 12:07
they're not -- they're separate pools of 12:07
resources. 12:07

The GPRs are shared over time. And 12:07
so when somebody is selected, he has to first 12:07
make sure that he has space in the GPR pool to be 12:07
able to be allocated. 12:07

Q. In what way are vertex vectors given 12:07
priority over the single-ported load into the 12:07
thread buffers? 12:07

MR. FAHRENKROG: Object to the form. 12:07

A. You know, it's just if I have both of 12:07
them -- as I said, if you have pixel vectors and 12:07
vertex vectors currently wanting to load, and I 12:07
have space for both of them, I'll just do the 12:08
vertex vector first. 12:08

Q. What happens if you don't have space 12:08
for both of them? 12:08

A. Then I can't select that. For 12:08
instance, I can't start a vertex thread if I 12:08
don't have any GPR space for that vertex vector 12:08
thread, because part of the vertex vector -- 12:08

Page 94

sorry -- part of the thread state is a GPR base 12:08
that has to be initialized. And if I can't 12:08
initialize that, then I can't start the thread. 12:08

MR. HIGGINS: We need to change 12:08
tapes. So why don't we take a break. 12:08

VIDEOGRAPHER: The time is 12:08 p.m. 12:08
At this time we'll end cassette number two. 12:08
We'll stop recording and go off the record. 12:08

(Luncheon Recess) 12:31

VIDEOGRAPHER: The time is 1:11 p.m. 12:31
At this time we are beginning cassette number 01:11
three. We are on the record. We are recording. 01:11

BY MR. HIGGINS: 01:12

Q. I'm going to refer again to the R400 01:12
sequencer specification. I believe it's Exhibit 01:12
-- 01:12

A. Eight. 01:12

Q. -- 8. Look on Page 7 of the 01:12
document. Then at the first paragraph, it says, 01:12
"There are two separate reservation stations, one 01:12
for pixel vectors and one for vertices vectors. 01:12
This way a pixel can pass a vertex and a vertex 01:12
can pass a pixel." 01:12

Do you see that? 01:12

A. Yes. 01:12

Page 95

Q. What does that mean? 01:12

A. It means that the vertices are 01:12
ordered relative to vertices, and pixels are 01:12
ordered relative to pixels. A vertex thread 01:12
starts later than a pixel thread may finish 01:13
earlier than a pixel thread. 01:13

Q. In what ways would a vertex thread 01:13
that starts later than a pixel thread finish 01:13
earlier than a pixel thread? 01:13

A. They are running two different shader 01:13
programs. So how long it takes any one element 01:13
to execute the shader program. Shader programs 01:13
could be radically different. So there is no 01:13
relationship in terms of how long it would take. 01:13

By keeping them in separate, these 01:13
reservation stations are FIFO oriented, meaning 01:13
first in, first out. By keeping them in separate 01:13
reservation station pools, you're able to order 01:13
them independently. 01:14

Q. So when it says "a pixel can pass a 01:14
vertex and a vertex can pass a pixel," is this 01:14
really just referring to the length of time it 01:14
takes a thread to execute its instructions? 01:14

MR. FAHRENKROG: Object to the form. 01:14

A. Yes. This is not an individual 01:14

Page 96

pixel. This is really talking about the threads, 01:14
and that pixel -- the lifetime of a pixel thread 01:14
may be shorter than the lifetime of a vertex 01:14
thread or vice versa. 01:14

Q. How many entries are there in the 01:15
vertex reservation station for vectors of 01:15
vertices? 01:15

A. As I recall, there are 16. 01:15

Q. How many entries are there in the 01:15
pixel reservation station for vectors of pixels? 01:15

A. As I recall, there's 48. 01:15

Q. Did that number ever change during 01:15
the development of the R400? 01:15

MR. PAIGE: Object to the form. 01:15

A. I don't know off the top of my head. 01:15
There's nothing magic about those numbers. So 01:15
they certainly could have changed without 01:15
affecting much in the design. 01:15

Q. What was the reasoning behind the 16 01:15
entries for vertices and 48 entries for pixels? 01:15

A. We were trying to make it such that 01:15
the thread buffer capacity was not going to be 01:16
the limiting factor. So we wanted to have enough 01:16
such that we didn't think that we'd have idle ALU 01:16
cycles. 01:16

Page 97

The limiting factor in terms of the 01:16
number of threads that are active at any given 01:16
point we were hoping would be the general purpose 01:16
register files, which are more expensive than 01:16
these are. So we were hoping to optimize the use 01:16
of the general purpose register files. 01:16

I explained earlier how you can't 01:16
really allocate a thread unless you can also 01:16
allocate space in the general purpose register 01:17
file. 01:17

Vertices -- vertex shaders tend to 01:17
use more general purpose registers than pixel 01:17
shaders do. So you don't need as many active 01:17
threads before you become limited by the general 01:17
purpose register file. 01:17

Q. You said you were hoping the general 01:17
purpose registers would be the unit or components 01:17
that optimize the number of threads running. Is 01:17
that what ultimately is reflected in the RTL of 01:17
the R400? 01:17

MR. FAHRENKROG: Object to the form. 01:17

A. Whether you're limited because of the 01:17
GPRs or because of the reservation stations is 01:18
not really a function of the RTL. The RTL 01:18
provides a certain number of reservation station 01:18

Page 98

resources, a certain number of GPR resources. 01:18
But the shader program itself says 01:18
how many of those resources a given thread 01:18
actually needs. And so it's a function of the 01:18
programs that get run on the machine as well as 01:18
the compiler for those programs. 01:18
Q. Did the functions of the general 01:18
purpose registers at the end of the R400 project 01:19
function as you had hoped they did, as you 01:19
mentioned earlier? 01:19
MR. FAHRENKROG: Object to the form. 01:19
MR. PAIGE: Object to the form. 01:19
A. I think we typically were register 01:19
file limited as opposed to reservation station 01:19
limited. Again, there could be oddball cases 01:19
where that was not true. 01:19
Q. If you turn to the last page of 01:19
Exhibit 8. Under the heading "Open Issues," it 01:19
reads, "Need to do some testing on the size of 01:19
the register file as well as on the register file 01:19
allocation method (dynamic versus static)." 01:19
Do you see that? 01:19
A. Yes. 01:20
Q. What is that referring to? 01:20
A. That is referring to whether we have 01:20

Page 99

enough general purpose registers to run the 01:20
number of threads that we were hoping to run. 01:20

It's not so much as to whether you're 01:20
limited by the static resources -- by the 01:20
reservation station or not. It's that you might 01:20
be limited at the GPR by having too few threads 01:20
to hide your latency such that we have the 01:20
reservation station room, but we don't have the 01:20
GPR room. 01:20

And so the reason why we need to do 01:20
testing here is that latency is one of those 01:20
things that's very hard to test up front without 01:20
a very good model. And it's the number of 01:21
threads that you're using to hide latency. 01:21

Q. Was testing performed on the size of 01:21
the register files? 01:21

A. We did performance testing and we, 01:21
you know, saw, you know, how well we were doing 01:21
against plan. 01:21

Q. When was that performance testing 01:21
done? 01:21

A. I can't say for sure whether we were 01:21
at that point where we could run a full 01:21
performance test on the R400 database, that is a 01:21
systems test. 01:21

Page 100

We certainly ran some block-level 01:21
testing, you know, throughout the design 01:22
development stages in 2001 and 2002. But whether 01:22
we were doing system testing at the time when we 01:22
changed over to the R5000, I'd have to go back 01:22
and look at the testing logs to see what tests we 01:22
could actually run. 01:22

Q. In order to determine which register 01:22
file size was required, would you have had to do 01:22
the full system level performance test? 01:22

MR. PAIGE: Object to the form. 01:22

A. I think it would be best to do that, 01:22
but it would have been possible to do some, at 01:22
least, preliminary testing based on fixed latency 01:22
models earlier in the design. 01:22

Q. During the R400 project, was the size 01:23
of the register file ever determined? 01:23

A. Yes, it was. I mean, we had a stake 01:23
in the ground for the size of the register file. 01:23

Q. When was the size of the register 01:23
file in the R400 determined after the testing was 01:23
performed? 01:23

MR. PAIGE: Object to the form. 01:23

A. Again, I'm not sure when the testing 01:23
was performed. But the testing was mainly, you 01:23

Page 101

know, to say, you know, do we have a serious 01:23
problem here, and the lack of that we would have 01:23
just gone with our current register file size. 01:23

Q. Did the size of the register files 01:23
ever change during the R400 project? 01:23

A. Not that I recall. 01:23

Q. Did the size of the register files 01:23
change during the R500 project? 01:24

A. Not that I recall. 01:24

Q. Is the size of the register file in 01:24
the R500 the same as in the R400? 01:24

A. I do not recall the size of the 01:24
register file in the R500 or whether it changed. 01:24
So you would have to look at the RTL to figure 01:24
out whether it changed or not. 01:24

Q. Is the size of the register file in 01:24
the Xenos the same as the size of the register 01:24
file in the R400? 01:24

A. This is one of those parameters that 01:24
is very easy to change. So I do not know whether 01:24
it was changed or not. 01:24

My guess is no. I think that they 01:24
were built of 128-by-128 register pieces and did 01:24
not change. But I cannot answer that for sure. 01:25

Q. What would you have to refer to in 01:25

Page 102

order to answer that for sure? 01:25

A. For sure, it would have to be the RTL. 01:25 01:25

Q. For the Xenos? 01:25

A. Well, if you're asking whether it changed, then I'd have to look at both the Xenos and the reference that you're asking whether it changed from. 01:25 01:25 01:25 01:25

Q. During the R400 project, was there any testing done on the register file allocation method? 01:25 01:25 01:25

A. Yes. 01:25

Q. When was that testing performed? 01:25

A. I can't say for sure. I'd have to look at the testing log to see. But, I mean, the basic register file allocation mechanism was basic to the operation of the part. 01:25 01:25 01:25 01:25

So I'm sure that it was -- that it was tested in the design process, both at the block -- certainly at the block level. 01:25 01:25 01:26

Q. As of May, 2002, what was the issue with the register file allocation method? 01:26 01:26

MR. FAHRENKROG: Object to the form. 01:26

MR. PAIGE: Object to the form. 01:26

A. I think this refers to how well the 01:26

Page 103

dynamic allocation method works. The static was 01:26
certainly functional, and it was an issue of how 01:26
much better performance do we get using the 01:26
dynamic mode of the register file. 01:26

Q. Was the dynamic mode of the register 01:26
file allocation method ever functional in the 01:26
R400? 01:26

A. I believe it had some bugs associated 01:26
with it. I mean, it was basically functional, 01:27
but I'm not sure it was completely functional. 01:27

Q. What bugs were associated with the 01:27
dynamic register file allocation method in the 01:27
R400? 01:27

A. I'd have to look at the bug log to 01:27
tell you. I recall that there were bugs, but I 01:27
don't recall the details of the bugs. 01:27

Q. Were the issues related to these bugs 01:27
ever resolved? 01:27

MR. FAHRENKROG: Object to the form. 01:27

A. Yes. The basic design was proven to 01:27
work. 01:27

Q. When was the basic design proven to 01:27
work? 01:27

A. Well, I don't -- the nature of these 01:27
bugs were subtle enough that they would not come 01:27

Page 104

up until you had silicon. So it wouldn't have 01:28
been until the R600 or possibly -- oh, no, I 01:28
guess Xenos would have had the same mechanism. 01:28

I don't know whether this worked in 01:28
the Xenos silicon or not. There may have been an 01:28
issue with the Xenos silicon with this. 01:28

Q. When you say "this," are you 01:28
referring to the dynamic register file 01:28
allocation? 01:28

A. I'm talking about the dynamic 01:28
register file allocation, yes. 01:28

Q. What were the issues with the Xenos 01:28
silicon -- 01:28

A. Again, I don't recall the details. I 01:28
just know that things didn't work when -- again, 01:28
I'm not 100 percent sure there was a problem with 01:28
the Xenos, but I seem to have a recollection that 01:28
there was a problem found in silicon, you know, 01:28
the first silicon with this. 01:28

Q. So when the first silicon was made 01:28
for the Xenos chip, the dynamic register file 01:29
allocation method did not function properly, 01:29
correct? 01:29

MR. PAIGE: Object to the form. 01:29

A. Correct. 01:29

Page 105

Q. Was the problem with the dynamic 01:29
register file allocation method, the Xenos chip, 01:29
ever corrected? 01:29

A. I can't say without looking at the 01:29
revision history whether there was a version of 01:29
Xenos that did have corrected. 01:29

Q. When you say "looking at a version of 01:29
the revision history," which document or RTL 01:29
coder you're referring to? 01:29

A. ATI had some bug-tracking software, 01:29
you know, system. I can't recall what the 01:29
details are, but it would be in the bug tracking. 01:30
It wouldn't be -- or possibly in the RTL as well. 01:30
If I looked in the RTL revision history, that 01:30
would probably have the fix associated with this 01:30
in it. 01:30

Q. How did the fixed static that's 01:30
referred to here register file allocation method 01:30
work? 01:30

A. Register pools, the size of the 01:30
vertex pool versus the pixel pool was set by a 01:30
software register, and vertices allocated out of 01:30
the vertex portion of the pool, and pixels 01:30
allocated out of the pixel portion of the pool. 01:30

Q. Was there any register file 01:30

Page 106

allocation set by the hardware? 01:31
A. Allocation was done by the hardware 01:31
every time that a thread was launched. The 01:31
source of the allocation depended on whether it 01:31
was a vertex thread or a pixel thread. 01:31
Q. How did the software set the division 01:31
of the register file? 01:31
A. It was purely heuristic based 01:31
probably on experimentation. Ideally you would 01:31
want to set it based on the individual shaders 01:31
that were loaded. I'm not sure if the driver 01:31
actually did that or just set it up once and left 01:31
it alone. 01:31
Q. Was there any default hardware 01:31
setting of the division of the registers between 01:31
vertices and pixels in the R400? 01:32
A. There's a register that defines how 01:32
much space is used for vertices and how much 01:32
space is used for pixels. I don't know if that 01:32
gets initialized if nobody touches it, because 01:32
typically, you know, the driver, when it comes 01:32
up, would set it to some setting. 01:32
I'm not sure what setting the driver 01:32
used. It probably started out at 50/50, but then 01:32
as a result of experimentation, it was set to 01:32

Page 107

something else. 01:32

Q. When you say "result of experimentation," are you referring to experimentation by the application writer? 01:32 01:32 01:32

A. No. I'm referring to experimentation by our driver team on typical or important applications to see, you know, what is a setting that would run well across a range of applications. 01:32 01:32 01:32 01:32 01:32

Q. What was that setting for the register file division in the R400? 01:32 01:33

A. Well, the R400 never got to the point where that experimentation was done. I think in our testing, we probably just used, you know, half of the register file for the pixels and half of the register files for the vertex. 01:33 01:33 01:33 01:33 01:33

Q. Do you know on what project this experimentation was first done? 01:33 01:33

A. It would have to be Xenos. 01:33

Q. Do you know when this experimentation was done on Xenos? 01:33 01:33

A. I do not. 01:33

(Gruber Exhibit 9, Declaration under 37 C.F.R. 1.131, marked for identification) 01:34 01:34

01:34

Page 108

(Gruber Exhibit 10, Declaration under 37 C.F.R. 1.131, marked for identification) 01:35 01:35

Q. Mr. Gruber, you have been handed what's been marked as Exhibits 9 and 10. They are both entitled "Declaration Under 37 C.F.R. Section 1.131." 01:35 01:35 01:35 01:35

A. Yes. 01:35

Q. Exhibit 9 was filed during the prosecution of Application No. 11/746,446, and Exhibit 10 was filed during the prosecution of Application No. 13/109,738. 01:35 01:35 01:35 01:35

Have you seen either of these documents before? 01:35 01:35

A. Yes, I have. I saw them when I signed them, and I saw them yesterday. 01:35 01:35

Q. Starting with Exhibit 9, do you know which patent this relates to? 01:35 01:36

A. No. I can read where it is entitled "Multithread Graphics Processing System." 01:36 01:36

(Gruber Exhibit 11, U.S. Patent No. 8,400,459, for identification) 01:36 01:36

(Gruber Exhibit 12, U.S. Patent No. 8,760,454, for identification) 01:37 01:37

Q. You have been handed what's been marked as Exhibits 11 and 12. Exhibit 11 is a 01:37 01:37

Page 109

copy of U.S. Patent 8,400,459. Exhibit 12 is a 01:37
copy of U.S. Patent No. 8,760,454. 01:37
You've seen these two documents 01:38
before? 01:38

A. Yes. 01:38

Q. Are you listed as an inventor on both 01:38
Exhibits 11 and 12? 01:38

A. I am. 01:38

Q. If you look at Exhibits 9 and 10, the 01:38
two declarations you submitted with respect to 01:38
these patents, attached to both declarations are 01:38
portions of RTL code? 01:38

A. Yes. 01:38

Q. Is the RTL code that's attached to 01:38
Declaration 9 RTL code from the R400? 01:38

A. I can't say for sure, but it does 01:38
look like R400 code. 01:38

Q. Same question for Exhibit 10? 01:39

A. Let me look at Exhibit 10. Again, 01:39
Exhibit 10 references structures and instructions 01:39
that I'm familiar with from R400. So I think 01:39
it's probably that it is -- that these do 01:39
represent R400 codes. 01:39

Q. Do you believe that the declarations 01:39
you submitted to the patent office in Exhibits 9 01:39

Page 110

and 10 reflect a full disclosure of the R400 01:39
project? 01:39

MR. FAHRENKROG: Object to the form. 01:40
Vague. 01:40

MR. PAIGE: Join in the objection. 01:40

A. What disclosure are you referring to? 01:40
Where it says A, B, C, D here or are you talking 01:40
about the disclosure of the patents themselves? 01:40

Q. Not the patents. Just what's in 01:40
Exhibits 9 and 10, these two declarations. 01:40

A. Okay. 01:40

Q. There's the statements A through D, 01:40
and then there's code attached to each 01:40
declaration. Is what's attached to these 01:40
declarations the full set of code for the R400? 01:40

A. No, this isn't the full set of code 01:40
for the R400. This is the code for one portion 01:40
of the R400. 01:40

Q. What portion of the R400 -- 01:40

A. This is a module from within the 01:41
sequencing. 01:41

MR. FAHRENKROG: Which exhibit were 01:41
you referring to? I apologize, just to make sure 01:41
it's clear. 01:41

MR. HIGGINS: Both. 01:41

MR. FAHRENKROG: When you answer, 01:41
please reference certain exhibits. 01:41

A. Okay. So I'm talking about Exhibit B 01:41
looks to be a portion of the R400 sequencer code. 01:41

Q. Are you referring to Exhibit B in 01:41
Exhibit 9? 01:41

A. Yes. Let me just make sure I'm 01:41
looking at -- I'm not sure, actually, that this 01:41
is Exhibit A or Exhibit B. It's not labeled. 01:41
I'll look, figuring one is Exhibit A and one is 01:41
Exhibit B. 01:41

Q. What is your understanding of 01:42
reduction to practice? 01:42

MR. FAHRENKROG: Object to the form. 01:42
Calls for a legal conclusion. 01:42

MR. PAIGE: Join in the objection. 01:42

A. Yeah, I can't really explain a legal 01:42
term. I can tell you what my engineering 01:42
understanding is. 01:42

Q. What was your understanding of what 01:42
the purpose of the declaration in Exhibit 9 and 01:42
Exhibit 10 was to do -- 01:42

MR. FAHRENKROG: Objection to the 01:42
extent it calls for -- 01:42

MR. HIGGINS: Can I finish my 01:42

Page 112

question? 01:42

MR. FAHRENKROG: I'm sorry, I thought 01:42
you were done. A couple pauses there. 01:42

Q. When you submitted Exhibits 9 and 10, 01:42
the declarations to the patent office, and you 01:42
signed them, what was your belief as to the 01:42
purpose of these declarations? 01:43

MR. FAHRENKROG: Is your question 01:43
finished? Can I object? 01:43

MR. HIGGINS: Go ahead. 01:43

MR. FAHRENKROG: Object to the extent 01:43
it calls for any attorney-client privileged 01:43
information, and to the extent your knowledge is 01:43
based on information communicated to you by your 01:43
attorneys, I instruct you not to answer. 01:43

A. I'm sorry, my attorney has instructed 01:43
me not to answer. 01:43

Q. Did you understand why you were 01:43
signing declarations reflected in Exhibits 9 and 01:43
10 when you signed them? 01:43

MR. FAHRENKROG: You can answer that 01:43
one yes or no. 01:43

A. Yes. 01:43

Q. What was your understanding of why 01:43
you signed declarations in Exhibits 9 and 10? 01:43

MR. FAHRENKROG: So again, to the 01:43
extent that that understanding is based on what 01:43
you were told or directed to do by an attorney at 01:43
the time, I instruct you not to answer. 01:43

A. Sorry, my attorney instructs me not 01:43
to answer. 01:43

Q. Did you have any understanding of 01:43
what you were signing when you signed the 01:44
declarations in Exhibits 9 and 10? 01:44

MR. PAIGE: Asked and answered. 01:44

A. Yes, I did. 01:44

Q. What was your understanding? 01:44

MR. FAHRENKROG: Same thing. Same 01:44
instruction. Please do not communicate anything 01:44
that was communicated to you by an attorney. 01:44

A. Based on my attorney's advice, I 01:44
decline to answer. 01:44

Q. Did you read the declaration before 01:44
you signed it? 01:44

A. Yes, I did. 01:44

Q. Did you understand what the 01:44
declaration said before you signed it? 01:44

A. Yes, I did. 01:44

Q. What was your understanding of what 01:44
was written in the declaration prior to you 01:44

Page 114

signing it? 01:44

A. That I was attesting to the fact that these exhibits reflected the state of the R400 at a certain point in time. 01:44 01:44 01:44

Q. When you mean it reflected the state of the R400 at a certain point in time, are you referring to the entirety of the R400 project? 01:44 01:44 01:44

MR. FAHRENKROG: Object to the form. 01:45

A. I was referring to the particular pieces of the R400 that were embodied by these design codes. 01:45 01:45 01:45

Q. What particular pieces of the R400 were embodied by these design codes attached as exhibits to declarations in Exhibits 9 and 10? 01:45 01:45 01:45

A. In one case, this is the part of the sequencer. And in the other case, it appears to be a wider block that includes the shader execution unit as well. 01:45 01:45 01:45 01:45

Q. If you look at Exhibit 10 in Paragraph 2, what do you mean by "We conceived the invention prior to June 30, 2003"? Paragraph 2 on the bottom of the first page. 01:46 01:46 01:46 01:46

A. In this case, I believe the invention is the shader subsystem that was included in the R400. 01:46 01:46 01:46

Page 115

Q. What is the shader subsystem in the 01:46
R400? 01:47
A. The unified shader and sequencer. 01:47
Q. And what did you mean by you 01:47
conceived of the invention of the unified shader 01:47
and sequencer prior to June 30, 2003? 01:47
A. We had a design and a specification 01:47
that we believed worked. We also had simulation 01:47
and implementation at the RTL level. 01:47
Q. What date did you conceive of the 01:47
invention you're referring to here? 01:47
MR. FAHRENKROG: Objection to the 01:47
extent the term "conceived" is being used in a 01:47
legal context outside of what's in the 01:47
declaration. 01:47
A. Early aspects of it were conceived in 01:47
late 2000, and the ideas were refined, I would 01:48
say, over the next six to nine months. 01:48
Q. When was the entirety of the 01:48
invention that you reference in Exhibit 2 in your 01:48
declaration conceived? 01:48
MR. FAHRENKROG: Object to the form. 01:48
Calls for a legal conclusion. 01:48
MR. PAIGE: Join in the objection. 01:48
A. There were certainly modifications as 01:48

Page 116

the design evolved, but the fundamental aspects 01:48
were designed prior to 2002 certainly. 01:48

Q. Do you know specifically what date 01:48
you conceived of the entirety of the invention as 01:49
you state in Paragraph 2 of your declaration in 01:49
Exhibit 10? 01:49

MR. FAHRENKROG: Object to the form. 01:49

A. No, I cannot provide a date. 01:49

Q. What in Exhibit 10, in your 01:49
declaration, shows that you conceived the 01:49
invention prior to June 30, 2003? 01:49

MR. FAHRENKROG: Object to the form. 01:49

MR. PAIGE: Object to the form. 01:49

A. Well, if these code snippets 01:49
adequately date from prior to -- this is not 01:49
obviously the full code of the R400, but it's the 01:49
shader code, as well as the RTL code here of a 01:49
portion of the SQ. The fact that these date from 01:49
prior to the date shown here shows that the 01:50
invention must have been conceived prior to this 01:50
date. 01:50

Q. In the RTL code that's attached to 01:50
Exhibit 10, where are the dates shown? 01:50

A. I don't see any dates in this code 01:50
snippet. Though this code snippet is certainly 01:50

Page 117

compatible with my belief of the state of the 01:50
development prior to 2003. 01:51

Q. What is your belief as to the state 01:51
of the development prior to 2003? 01:51

A. That we had a fully functional shader 01:51
system. "Functional," I mean that we were able 01:51
to do RTL level simulation of the shader system 01:51
successfully and pass tests showing a mixture of 01:51
vertex and pixel processing. 01:51

Q. On what date did you have a fully 01:51
functional shader system of the R400? 01:51

A. I can't give you a date off the top 01:51
of my head. I'd have to go back and look at our 01:51
status reports to see, you know, what tests were 01:51
running and what tests weren't running in any 01:51
given point. 01:51

Q. When you signed this declaration 01:51
stating you had a fully functional shader system 01:52
with the R400 prior to 2003, what was that 01:52
determination based upon? 01:52

MR. FAHRENKROG: Object to the form. 01:52

A. It was based on my memory and my 01:52
knowledge that we did have such a shader system 01:52
running. 01:52

Q. Was your memory and knowledge in 2011 01:52

Page 118

of the R400 shader system different than your 01:52
memory and knowledge of the R400 shader system 01:52
today? 01:52
MR. FAHRENKROG: Object to the form. 01:52
A. I suppose it's probably changed in 01:52
some respects, but not in that one. 01:52
Q. So if this was based off of your 01:52
memory of the R400, and your memory is not 01:52
changed in that respect from the time of filing 01:52
this declaration until today, how did you sign 01:53
the declaration saying you had this fully 01:53
functional shader system prior to June 30, 2003 01:53
if you can't give me a date today prior to June 01:53
30, 2003 when you had a fully functional shader 01:53
system? 01:53
MR. FAHRENKROG: Object to the form. 01:53
MR. PAIGE: Object to the form. 01:53
MR. FAHRENKROG: Incomplete 01:53
hypothetical. 01:53
A. I can't give you a specific date, but 01:53
I can tell you that it was prior to June 30, 01:53
2003. 01:53
Q. How can you tell me it was prior to 01:53
June 30, 2003? 01:53
A. Because I know when we were working 01:53

on this, and that the amount of time from the 01:53
time that we conceived it until the time we had, 01:53
you know, working tests through the shader system 01:53
was much less than the time from conception to 01:53
June 30, 2003. 01:53

Q. How much less than the time? 01:53

A. Again, I can't give you a specific 01:54
date, but I can for sure say it was before the 01:54
end of 2002. 01:54

Q. Why did the declaration not provide a 01:54
specific date prior to June 30, 2003? 01:54

MR. PAIGE: Object to the form. 01:54

A. Are you asking me? 01:54

MR. FAHRENKROG: Objection. To the 01:54
extent it was communicated by an attorney, don't 01:54
answer. 01:54

Q. When you say you had a fully 01:54
functional shader system prior to June 30, 2003, 01:54
which of points A, B, C and D in Exhibit 10 are 01:55
you referring to? Exhibit 10 is the larger one. 01:55

A. Okay. All of these are true. 01:55

Q. So is the fact that you say you had a 01:56
fully functional shader system based on the model 01:56
code that's attached to this declaration? 01:56

MR. FAHRENKROG: Object to the form. 01:56

Page 120

A. Well, I know we had both model code 01:56
and RTL code functional prior to this. 01:56

Q. When did you have both model code and 01:56
RTL code functional prior to this declaration? 01:56

A. Again, I can't give a specific date. 01:56

Q. If you look at Exhibit A in your 01:57
declaration that's Exhibit 10. The top says 01:57
"Exhibit A - Model Code." Is this the computer 01:57
simulation model code that's referenced in the 01:57
declaration Part A? 01:57

A. Yes. 01:57

Q. Can you point to where threads are 01:57
created in the model code in Exhibit A? 01:58

MR. PAIGE: Object to the form. 01:58

MR. FAHRENKROG: Object to the form. 01:58

A. This is very complicated code. I'm 01:58
not sure that I'm going to be able to point to 01:59
this as I sit here. 01:59

I see there's a lot of allocation 01:59
being done here, but it's mostly allocation 01:59
associated with execution. 01:59

Q. Is this computer emulation in Exhibit 02:00
A, is it multithreaded? 02:00

MR. FAHRENKROG: Object to the form. 02:00

MR. PAIGE: Join in the objection. 02:00

Page 121

A. I don't believe the code itself is 02:00
multithreaded, that is the C++ code. I really 02:00
haven't had enough exposure to the details of the 02:00
C model code to give a definitive answer. 02:00

Q. Can C model emulation code be 02:00
multithreaded? 02:00

MR. FAHRENKROG: Object to the form. 02:00

A. Yes, it can be multithreaded. 02:00

Q. Can a C model emulation execute 02:00
things simultaneously? 02:01

MR. PAIGE: Object to the form. 02:01

MR. FAHRENKROG: Object to the form. 02:01

A. Yes, it can. 02:01

Q. How can it do that? 02:01

A. If you have -- I'm not necessarily 02:01
saying this code. I thought you were asking me 02:01
theoretically whether model code that is 02:01
multithreaded could execute multiple threads 02:01
simultaneously. 02:01

Yes, you can run a thread on two 02:01
different processors that are executing 02:01
simultaneously. 02:01

Q. Can the model code in Exhibit A 02:01
perform two functions simultaneously? 02:01

A. I don't have enough experience with 02:01

Page 122

this code to say whether it can or it can't. 02:01

Q. Is the model code in Exhibit A of 02:01
Exhibit 10 a cycle-accurate model? 02:01

MR. PAIGE: Object to the form. 02:02

A. No. Our C models were never intended 02:02
to be cycle accurate. 02:02

Q. Refer back to the beginning of the 02:02
declaration, Exhibit 10 on Page 2. The bullet 02:02
point labeled D, why was that part of your 02:02
declaration made, information and belief? 02:02

MR. FAHRENKROG: Objection to the 02:02
extent that your answer would reveal any 02:02
information communicated by an attorney, I 02:02
instruct you not to answer. 02:02

A. I'm sorry, I can't answer on the 02:02
advice of my attorney. 02:02

Q. When you signed this declaration, why 02:02
did you only state that what is contained in 02:02
Paragraph D only based upon your information and 02:02
belief? 02:02

MR. FAHRENKROG: Same instruction. 02:03

A. I can't answer on the advice of my 02:03
attorney. 02:03

Q. You had no understanding of what this 02:03
paragraph meant when you signed your declaration; 02:03

Page 123

is that correct? 02:03

MR. FAHRENKROG: Objection. Same instruction and asked and answered. 02:03 02:03

A. I'm sorry, I can't answer on the advice of my attorney. 02:03 02:03

Q. When you signed the declaration, did you understand what was written in the declaration? 02:03 02:03 02:03

MR. PAIGE: Object to the form. 02:03

MR. FAHRENKROG: You can answer that yes or no. 02:03 02:03

A. Yes. 02:03

Q. What did you understand Paragraph D to mean when it says that what you're stating is only on information and belief? 02:03 02:03 02:03

MR. FAHRENKROG: You can answer as to your personal understanding, except to the extent it would reveal communications from an attorney, then I instruct you not to answer. 02:03 02:03 02:03 02:03

A. I'm sorry, my attorney instructs me not to answer. 02:03 02:03

Q. You have no personal understanding of what Paragraph D meant when you signed this declaration? 02:03 02:03 02:03

MR. FAHRENKROG: Objection. Asked 02:03

Page 124

and answered. 02:03

MR. HIGGINS: Actually, you 02:03
instructed him not to answer. 02:04

MR. FAHRENKROG: He gave his answer 02:04
pursuant to the instruction. 02:04

MR. CHATTERJEE: Counsel, your 02:04
instruction is actually wrong. He's not asking 02:04
about the communications. He should be answering 02:04
this whatever he understood when he signed 02:04
something under oath. He's not asking about 02:04
communications with lawyers. There's absolutely 02:04
no basis to say that his understanding of what he 02:04
said to the patent office is a privileged 02:04
communication. He's not asking about the 02:04
communication. 02:04

MR. FAHRENKROG: If you listen 02:04
closely to my instruction, it's not to reveal 02:04
anything being communicated to him by an 02:04
attorney. I said he can answer the question as 02:04
to his personal understanding. 02:04

MR. CHATTERJEE: Okay. 02:04

A. My personal understanding was that 02:04
the model code and the chip design code 02:04
demonstrated a functional embodiment of the 02:04
patent. 02:04

Page 125

Q. When you signed the declaration, what 02:04
facts did you have that supported your 02:05
understanding that the model code and chip design 02:05
demonstrated a functional embodiment of the 02:05
patent? 02:05

A. That the model code and the RTL code 02:05
were codes from the R400 design effort prior to 02:05
June, 2003, and that -- and I knew that we had 02:05
such codes running functionally at that time. 02:05

Q. Is the code that's attached to your 02:05
declaration the same code that was incorporated 02:05
into the Xenos chip? 02:05

A. I don't know whether it was the same 02:06
code that was incorporated into the Xenos chip. 02:06
I mean, it looks similar to a lot of R400 code 02:06
that we had during the R400 design effort. 02:06

I don't know whether exactly this 02:06
code was what was incorporated into the Xenos 02:06
chip, but this code certainly performed the 02:06
functions of whatever code was in the Xenos chip. 02:06
There may have been slight differences. 02:06

MR. HIGGINS: Off the record for a 02:07
second. 02:07

VIDEOGRAPHER: The time is 2:07. At 02:07
this time we will stop recording and go off the 02:07

Page 126

record. 02:07
(Recess) 02:07
(Gruber Exhibit 13, Source Code, 02:12
marked for identification) 02:18
VIDEOGRAPHER: The time is 2:18 p.m. 02:18
At this time we're beginning cassette number 02:18
four. We're on the record. We are recording. 02:18
BY MR. HIGGINS: 02:18
Q. What is an RTL simulator? 02:18
A. RTL was the chip design language. A 02:18
simulator does a cycle-by-cycle simulation of 02:18
what will finally be built in the chip. The chip 02:18
is produced from the RTL. 02:18
Q. Does an RTL simulation run on 02:18
hardware? 02:18
A. Well, an RTL simulation runs on a 02:18
computer. So in that sense it runs on hardware. 02:18
Q. If you look in Paragraph C of your 02:19
declaration, Exhibit 10. It says, "The chip 02:19
design code was used by an RTL simulator system 02:19
to validate the operation of an integrated 02:19
circuit version of the claimed graphics 02:19
processing system." 02:19
Do you see that? 02:19
A. Yes. 02:19

Page 127

Q. When was an RTL simulation done on a complete integrated circuit version of the R400? 02:19 02:19

MR. FAHRENKROG: Object to the form. 02:19

A. We were doing simulation all through the design effort of the R400 as the blocks were built, they were simulated. As to when the complete GPU was simulated, I can't point to a specific date. 02:20 02:20 02:20 02:20 02:20

But, for instance, the shader subsystem, which this code encompasses, was certainly simulated fairly early in the design as it was built. 02:20 02:20 02:20 02:20

Q. What shader subsystem are you referring to? 02:20 02:20

A. I'm talking about the shader and sequencer blocks. Oftentimes we would simulate a block by putting it into an environment where it would be passed vectors into the block and out of the block. 02:20 02:21 02:21 02:21 02:21

Q. What RTL simulator program did you use when simulating the R400 RTL? 02:21 02:21

A. I don't recall. I seem to remember one from Mentor Graphics, but it has been too long, and I've used too many different ones for me to say for sure. 02:21 02:21 02:21 02:21

Page 128

Q. Was an RTL simulator a software 02:22
program? 02:22

MR. FAHRENKROG: Object to the form. 02:22

A. It generally is. There are hardware 02:22
simulators as well. 02:22

Q. Which simulator was used on the R400 02:22
RTL? 02:22

A. Well, we definitely used a software 02:22
program. We might have used a hardware program 02:22
as well -- a hardware system as well. 02:22

Q. When using a software RTL simulator 02:22
program, does it configure the hardware on the 02:22
computer to perform graphics operations? 02:22

MR. FAHRENKROG: Object to the form. 02:22
Vague. 02:22

A. I don't know what you mean by 02:22
"configure the hardware"? It runs software, 02:22
software code that simulates the RTL. 02:23

Q. Does an RTL simulation allocate 02:23
actual register file space? 02:23

MR. FAHRENKROG: Object to the form. 02:23

MR. PAIGE: Join in the objection. 02:23

A. It allocates space in a structure 02:23
managed by the software that represents register 02:23
file space. 02:23

Page 129

Q. Does it actually allocate any physical register space for a thread during the RTL simulation? 02:23 02:23 02:23

MR. PAIGE: Vague and ambiguous. 02:23

MR. FAHRENKROG: Object to the form. 02:23

A. It will allocate space in a memory structure. Whether you call that a register file, I mean, it's a simulated register file. It's simulated by having it just be in a structure understood by the software code. 02:23 02:24 02:24 02:24 02:24

Q. Is the code attached to Exhibit B in your declaration Exhibit 10 sufficient to perform an RTL simulation with the entire integrated circuit design of the R400? 02:24 02:24 02:24 02:24

MR. PAIGE: Object to the form. 02:24

MR. FAHRENKROG: Object to the form. 02:24

A. This is not a -- we're talking about Exhibit 10, correct? 02:24 02:24

Q. Exhibit B to Exhibit 10, yes. 02:24

A. This code, this software simulator code, does not simulate the RTL. It is -- it simulates the functions of the RTL. 02:24 02:25 02:25

Q. Go towards the back. It's part of the last 20 pages or so of Exhibit B, the chip design code. 02:25 02:25 02:25

Page 130

A. I'm sorry. I'm sorry. 02:25
Q. It looks like it's after Page 93 of 02:25
Exhibit A. 02:25
A. Okay. So this is the chip design 02:25
code for a portion of the SQ. I'm sorry, what 02:25
was the question? 02:25
Q. The code in Exhibit B -- 02:25
A. Yes. 02:26
Q. -- that we're looking at -- 02:26
A. Yes. 02:26
Q. -- is this sufficient to simulate 02:26
the entire integrated circuit chip design of the 02:26
R400? 02:26
MR. FAHRENKROG: Object to the form. 02:26
MR. PAIGE: Object to the form. 02:26
A. It will not simulate the entire R400. 02:26
It would simulate important aspects of the 02:26
unified shader. 02:26
Q. What other code would be needed to 02:26
perform an RTL simulation of the entire 02:26
integrated circuit design of the R400? 02:26
MR. FAHRENKROG: Object to the form. 02:26
Overly broad. 02:26
MR. PAIGE: Join in the objection. 02:26
A. You would need the entire RTL code 02:26

Page 131

for the R400. This is not all the code in the 02:26
R400. 02:26

Q. If you move on to the small stack of 02:26
documents to your right in yellow. 02:27

MR. FAHRENKROG: Objection to the 02:27
characterization as small. I understand it was 02:27
in gest. 02:27

Q. Exhibit 13 in front of you is AMD 02:28
confidential source code, Bates label 02:28
AMD932SC000001 through 2110. 02:28

Have you viewed these source code 02:28
files before? 02:28

MR. CHATTERJEE: Let the record 02:28
reflect a big, heavy sigh. 02:28

A. It is possible I've viewed some of 02:28
them before. I'm quite sure I haven't viewed 02:28
them in their entirety. 02:28

Q. Let me specify. Have you viewed this 02:29
printed stack of source code before? 02:29

A. No. 02:29

Q. Do you know whether the source code 02:29
printed in front of you between the Bates ranges 02:29
1 and 2110, represents the entirety of the R400 02:29
RTL code? 02:29

A. I do not, no. 02:29

Page 132

Q. What would you have to do to make 02:29
that determination? 02:29
A. I would have to compare this code 02:29
with all the modules that we use to produce the 02:29
-- or to simulate the R400. 02:29
Q. If you go to the page Bates labeled 02:29
319 through 338. Let me know when you have it. 02:29
A. Okay. 02:30
Q. 319 to 338 is a directory of all the 02:30
source code files that Respondents had printed 02:30
off the AMD source code computers that's sitting 02:30
in the small stack of papers in front of you. 02:30
A. Yes. 02:30
Q. If you looked through this directory 02:30
listing of the files that were printed, would you 02:30
be able to determine if it represents the 02:30
complete R400 RTL code? 02:30
A. I'm not sure that I'd be able to 02:30
attest to that myself. I think what I would 02:30
probably need is the build script that is used 02:31
for building R400 and check the reference of 02:31
these against that build script. 02:31
I mean, I'm perfectly willing to 02:31
believe you that these are all the ones. But if 02:31
you're asking for my personal knowledge, I 02:31

Page 133

certainly don't have all of those -- all the 02:31
blocks involved committed to memory. 02:31

MR. PAIGE: Your representation is 02:31
that this block is what Respondents had printed 02:31
from the source code? 02:31

MR. HIGGINS: Yes. 02:31

MR. PAIGE: Okay. I believe up to 02:31
Bates range 1450. 1451 through 2110 was a 02:32
separate production. 02:32

Q. What is the build script for the R400 02:32
RTL? 02:32

A. So when you wanted to -- it's a 02:32
mechanism for running simulation that would grab 02:32
all the particular files involved in the 02:32
simulation. 02:32

Q. Did a build script for the R400 02:32
exist? 02:32

A. I believe that's how we did it, yeah, 02:32
that there was a script or an execution. 02:32

Q. When was the build script for the 02:33
R400 created? 02:33

A. Well, it evolved with the R400 as 02:33
modules were added. There would be various 02:33
versions of it. So it would evolve with the 02:33
R400. 02:33

Page 134

Q. Was there a build script for the R400 02:33
that related to the complete integrated circuit 02:33
chip design for the R400? 02:33

MR. FAHRENKROG: Object to the form. 02:33

MR. PAIGE: Object to the form. 02:33

A. I believe so. I cannot answer for 02:33
sure. I don't see why there wouldn't be. 02:33

Q. When would the build script have been 02:33
created? 02:33

A. Again, it would have -- I mean, you 02:33
know, things would evolve. So as you add blocks, 02:33
they get added to the build script. 02:33

You would also use it, for instance, 02:33
you might use a version of one if you wanted some 02:33
of the blocks from RTL, and then use other blocks 02:33
modeled from the C model. 02:34

Q. When was the last block added to the 02:34
R400 design? 02:34

MR. PAIGE: Object to the form. 02:34

A. I don't know. I'd have to look on 02:34
the check-ins to see when that happened prior to 02:34
the conversion to the R500. 02:34

Q. Were blocks added to R500? 02:34

A. I expect that there may -- that there 02:34
were blocks added to R500. For instance, as we 02:34

Page 135

added the vertex cache, that would be a block 02:34
that would be added. 02:34

Q. If you look at the code in Bates 02:34
range 1 through 64. 02:35

A. I'm sorry, did you say 1 through 64? 02:35

Q. 1 through 64. 02:35

A. Okay. This is the clause.arb. 02:35

Q. If you see at the top in the file 02:35
directory, it has the date March 27, 2002? 02:36

A. Yes. 02:36

Q. It's the same date through all 64 02:36
pages. 02:36

A. Okay. 02:36

MR. PAIGE: Are you asking or 02:36
representing that? 02:36

MR. HIGGINS: It's kind of a little 02:36
of both. Let's go back. 02:36

Q. Is all of the code in Bates ranges 1 02:36
to 64 labeled with the date March 27, 2002? 02:36

MR. PAIGE: The document speaks for 02:36
itself. He can look at the 64 pages if you want 02:36
him to. 02:36

A. Yes. 02:37

Q. Does this collection of code, Bates 02:37
ranges 1 to 64, represent the entirety of the RTL 02:38

Page 136

code for the sequencer of the R400 as of March 02:38
27, 2002? 02:38

A. I can't be 100 percent certain, but 02:38
it appears most of the functionality is here. 02:39

Q. Are you aware of any modules of RTL 02:39
code for the R400 sequencer that are not in Bates 02:39
ranges 1 to 64? 02:39

A. I'm not aware of any. 02:39

Q. If you go to Bates ranges 65 through 02:39
278. 02:40

A. All right. 02:40

Q. The second part, Bates ranges 339 02:40
through 537. 02:40

A. I'm sorry, what am I looking for? 02:41

Q. 339 through 537. Do these two ranges 02:41
of code represent the entirety of R400 sequencer 02:41
RTL code as of May 29, 2002? 02:41

A. I'm just not going to be able to give 02:41
you a definitive answer. I would have to, again, 02:42
look to see how the SQ was built in that time 02:42
frame. I'm not familiar enough with the 02:42
internals of the RTL code of the SQ, especially 02:42
after 13 years, to tell you whether these are all 02:42
of the codes. 02:42

They certainly do have a lot of the 02:42

Page 137

modules that I'm familiar with. 02:42

Q. Did you do any investigation prior to today's deposition to determine whether the RTL code produced by AMD is the complete set of RTL code for the R400? 02:42 02:42 02:42 02:42

A. I did not. 02:42

Q. Did you understand that you had an obligation to be familiar with the documents produced by AMD and to testify on those today? 02:42 02:43 02:43

MR. FAHRENKROG: Object to the form. 02:43

A. I understood that I had an obligation to represent AMD. 02:43 02:43

Q. But you did not perform any investigation as to the RTL code that AMD produced; is that correct? 02:43 02:43 02:43

MR. FAHRENKROG: Object to the form. 02:43

MR. PAIGE: Misstates prior testimony. 02:43 02:43

A. I did look over some ATI-produced RTL code. I did not look over every piece of ATI RTL code or assure that it represented the complete SQ source. 02:43 02:43 02:43 02:43

Q. Which ATI-produced RTL code did you review? 02:43 02:43

A. Well, I produced -- I certainly 02:43

Page 138

looked at some of the code associated with those 02:44
patents that we looked at earlier. 02:44

Q. Did you look at any code that was not 02:44
included in the declarations of Exhibits 9 and 02:44
10? 02:44

A. No, I did not. 02:44

Q. So prior to today's deposition, did 02:44
you review any source code produced by AMD in 02:44
this investigation? 02:44

MR. PAIGE: Vague and ambiguous. 02:44

A. No, I did not produce -- I did not 02:44
look at the ATI source code that was represented 02:44
in these... 02:44

Q. Who wrote the RTL source code for the 02:44
sequencer module, the R400? 02:44

A. I believe it was Laurent Lefebvre, 02:44
but I couldn't say for sure that Laurent was the 02:45
only author. 02:45

Q. Who else wrote RTL code for the R400? 02:45

MR. FAHRENKROG: Object to the form. 02:45

A. For the R400? There was a large team 02:45
of people working on the source code for the 02:45
R400. I could only, you know, come up with a 02:45
handful of names. 02:45

Q. What are the names you can come up 02:45

Page 139

with? 02:45

A. I remember Jay Wilkinson working on 02:45
the render back end. Andi Skende working on the 02:45
shader system. Jocelyn Houle working on the 02:45
texture pipe. That's all that comes to mind now. 02:46

Q. Which portions of the RTL code did 02:46
you write? 02:46

A. I rewrote some of the shader code. 02:46
But other than that, I didn't write any RTL code 02:46
in the R400. 02:46

Q. When did you rewrite some of the 02:46
shader code? 02:46

A. I couldn't give you an exact date, 02:46
but it was during some of the Xenos development 02:46
effort. 02:46

Q. Why did you rewrite some of the 02:46
shader code during the Xenos development? 02:46

A. We were trying to save area. It was 02:46
not functionally different. 02:46

Q. Which modules of RTL code did you 02:46
rewrite during the Xenos development? 02:47

A. I don't remember the names of the 02:47
modules, but it would be the shader pipe itself, 02:47
the data path. 02:47

Q. What about the data path of the 02:47

Page 140

shader pipe did you rewrite? 02:47

A. The adder and the normalizer logic. 02:47

Q. Why did you rewrite those two 02:47
modules? 02:47

A. We were just -- we would compile them 02:47
and try to see where the timing was tight. The 02:47
idea is if you can reduce the timing, you can 02:47
reduce the area. 02:47

Q. So if we were to walk through the 02:47
rest of the RTL code that's sitting here as 02:48
grouped by date, would you be able to tell me if 02:48
any of those groups of RTL code represent the 02:48
complete code for the sequencer in the R400? 02:48

MR. FAHRENKROG: Object to the form. 02:48
Overly broad. 02:48

A. I wouldn't be able to say whether 02:48
there was a missing module or not without having 02:48
a list of modules, you know, based on the script 02:48
that was used to compile the simulation. 02:48

Q. Does AMD -- did AMD keep a record of 02:48
all the simulations performed on the RTL of the 02:48
R400? 02:49

A. I don't think we kept a record of all 02:49
the simulations performed. I know that we 02:49
certainly kept a record of tests, whether a test 02:49

Page 141

was passing or not passing, at any given time. 02:49

Whether that still exists, I think we would have to go into the -- go into AMD's files to really know. 02:49 02:49 02:49

Q. Did an RTL simulation performed on the entire integrated circuit design of the R400 ever pass the simulation test? 02:49 02:49 02:49

MR. FAHRENKROG: Object to the form. 02:49

MR. PAIGE: Vague and ambiguous. 02:49

A. I can't answer definitively. I expect that it did, but, you know, we had lots of tests, and I'm sure that some of them passed. But I can't, sitting here without reference, say that an entire chip silicon with all the modules being RTL as opposed to the C simulator. 02:49 02:49 02:50 02:50 02:50 02:50

I would have to go back and really see what we could dig up as to what the state of the chip was at any given date. 02:50 02:50 02:50

Q. If an RTL simulation performed on an entire chip silicon with all the modules as RTL passes the test, what is the usual next step in the development process? 02:50 02:50 02:50 02:50

MR. FAHRENKROG: Object to the form. 02:50

MR. PAIGE: Object to the form. 02:50

A. Well, you have a lot of tests, and 02:50

Page 142

you have both directed tests and you have random 02:51
tests. So a lot of times, you know, we would 02:51
focus on getting the random test passing, and 02:51
then you would try to get -- sorry, you would try 02:51
to get the directed test passing, because it 02:51
would be easier to debug, and then you would run 02:51
the random tests. 02:51

And only when you got to a suitable 02:51
level of time where, you know, you weren't seeing 02:51
bugs or the bugs you were seeing were ones you 02:51
really didn't care about would you say, okay, 02:51
well I guess we can tape it out. 02:51

Q. Was that point ever reached during 02:51
the development of the R400? 02:51

A. No. We never got enough of the test 02:51
passing in all the blocks to be able to tape out 02:51
the R400. 02:52

Q. Go to Bates range 1451 through 1694. 02:52

A. Wait a second. Let me just make sure 02:52
that I'm -- this is 278. Okay. I'm sorry, I 02:52
need the numbers again. 02:52

Q. 1451. 02:52

A. Okay. Through? 02:52

Q. 1694. Let me know when you have it 02:52
in front of you. 02:53

Page 143

A. I'm just trying to get a little more organized here. All right. 1451. 02:53 02:53

Q. At the top of the page in the directory it says "R400_sp_snapshots." Do you see that? 02:53 02:53 02:54

A. Yes. 02:54

Q. What does that represent? 02:54

A. The R400 snapshots within the source code? I don't know. I'd have to -- I'm not sure how I could find out. I presume this was -- represents times when a backup was made and dated and perhaps kept in a more secure space. 02:54 02:54 02:54 02:54 02:54

Q. What does snapshot refer to? 02:54

A. With a view of the database as it existed at the time. 02:54 02:54

Q. Is it a view of the complete SP database as it existed on March 27, 2002? 02:54 02:54

MR. FAHRENKROG: Object to the form. 02:54

MR. PAIGE: Lack of foundation. 02:54

A. I don't know, but that would be my assumption. 02:54 02:54

Q. What is SP? 02:54

A. Shader pipe. 02:54

Q. Similar to the sequencer code that we were just referring to, if we walked through this 02:55 02:55

Page 144

code in groups by dates, would you be able to determine whether this is the complete set of RTL code for the R400 shader processor? 02:55 02:55 02:55

A. No, I could not tell you that it was the complete set without reference to the scripts that were used to build it. 02:55 02:55 02:55

Q. Was RTL code ever created for the R400 primitive assembly? 02:55 02:55

A. I believe that it was, but I cannot say for sure. 02:55 02:56

Q. Was any RTL code ever created for the R400 scan converter? 02:56 02:56

A. Yes, I'm sure that that was -- I'd like to look at what tests were running to say absolutely for sure, but I believe that, yes, there was scan converter code. 02:56 02:56 02:56 02:56

Q. Was the RTL code for sequencer module ever integrated with the RTL code for the SP module? 02:56 02:56 02:56

A. Again, I'd like to see what tests we run, but I'm pretty sure the answer is yes. 02:56 02:57

Q. When was the RTL code for the sequencing and shader processor modules integrated? 02:57 02:57 02:57

A. Again, it would have to have been 02:57

Page 145

prior to the end of 2002 certainly, and, you 02:57
know, perhaps even before the end of 2001. 02:57

Q. What does it mean for the sequencer 02:57
code to be integrated with the shader processor? 02:57

MR. FAHRENKROG: Object to the form. 02:57

A. We were able to run a simulation 02:57
where both of those are RTL simulators and they 02:58
are talking to one another. 02:58

Q. In order for an RTL simulation to be 02:58
run, have the sequencer and the SP talking to 02:58
each other, would that mean that the inputs into 02:58
the shader processor and the outputs of the 02:58
sequencer to the shader processor would have to 02:58
match up? 02:58

MR. FAHRENKROG: Object to the form. 02:58

MR. PAIGE: Join in the objection. 02:58

A. Yes. You would have to -- you would 02:58
be driving the inputs of one with the outputs of 02:58
the other. 02:58

Q. What would happen if an RTL 02:58
simulation was run and the inputs and outputs of 02:58
the sequencer and shader processor did not 02:58
correctly match up? 02:58

MR. PAIGE: Objection. Hypothetical. 02:58

A. It simply wouldn't run correctly. 02:58

Page 146

You'd have unknowns on certain of the inputs, 02:59
which would propagate fairly quickly. 02:59
Q. Could you look at the bottom stack of 02:59
code that starts at 1695 and runs through the 02:59
entirety down to 2110. 02:59
A. Okay. 02:59
Q. What does this collection of code 02:59
relate to? 03:00
MR. FAHRENKROG: Objection to the 03:00
form. 03:00
MR. PAIGE: 500 pages of code, what 03:00
it relates to is being asked? 03:00
A. So at least that the first page is 03:00
related to the sequencer. It's the arbiter 03:00
within the sequencer. 03:00
Q. Is this RTL code? 03:00
MR. PAIGE: When you say this, do you 03:01
mean all 500 pages of RTL code or are you asking 03:01
some part of it? 03:01
A. Well, it appears to be a simulator 03:01
code. 03:01
Q. Does this code relate to an emulator 03:01
for the R400? 03:01
A. Yes. 03:01
Q. Sorry, I just want to be clear. Is 03:01

Page 147

this code for the emulator R400 -- 03:02

A. Yes. 03:02

Q. -- or for the simulator -- 03:02

A. Oh, okay. Well, we often call the 03:02
emulator and simulator the same thing. But if 03:02
you mean the code that runs the RTL code, by 03:02
simulator -- if you mean the code that runs the 03:02
RTL code, no. This is not the code that runs the 03:02
RTL code. This is the C model emulator. 03:02

Q. Did a fully functional C model 03:02
emulator exist for the entire chip design of the 03:03
R400? 03:03

MR. FAHRENKROG: Object to the form. 03:03

A. I believe that it did, but I cannot 03:03
answer you 100 percent without doing some 03:03
additional research. 03:03

Certainly the emulator existed for 03:03
the shader system. 03:03

Q. The R400 emulator, did it operate in 03:03
a multithreaded environment or single-threaded 03:03
environment? 03:04

MR. PAIGE: Vague and ambiguous. 03:04

A. I couldn't answer you without doing 03:04
further research. 03:04

Q. Could the R400 emulator process two 03:04

Page 148

threads in parallel? 03:04

MR. FAHRENKROG: Object to the form. 03:04

MR. PAIGE: Join in the objection. 03:04

A. I can't answer you without doing 03:04
further research whether we did multithread the 03:04
emulator or not. 03:04

(Gruber Exhibit 14, Document 03:05
entitled "R400 Emulator Version 0.0 January 15, 03:05
2002," marked for identification) 03:05

Q. You have been handed what's been 03:05
marked as Exhibit 14, document entitled "R400 03:05
Emulator," dated January 15, 2002, Bates label 03:05
AMD-ITC-0019672 through 19678. 03:05

Have you seen this document before? 03:05

A. Not to my recollection. 03:05

Q. Is this the same R400 emulator that 03:05
we have just been discussing with respect to the 03:05
code? 03:05

MR. FAHRENKROG: Object to the form. 03:05

A. Yes. 03:05

Q. If you look on Page 4 of the 03:05
document, Bates number 19675, if you look at 03:05
point 3 under Section 2. 03:06

A. Okay. 03:06

Q. Does this refresh your recollection 03:06

as to whether the emulator operated in a single 03:06
thread or multithread? 03:06

A. Well, my recollection was that it did 03:06
run in a single thread. My uncertainty was based 03:06
on whether in a later revision of the emulator we 03:06
tried to multithread it to increase performance. 03:06

Q. Under point 1 in Section 2, what does 03:06
it mean by "The emulator contains no knowledge 03:06
about the parallelism used in the hardware"? 03:06

A. The emulator is a functional 03:06
emulator. It does not match the hardware cycle 03:06
by cycle. It does match the hardware as certain 03:06
checkpoints, generally block boundaries. 03:07

Q. I'm going to have you turn back to 03:07
Exhibit 8, and at the same time -- 03:08

A. Let me put things back together. 03:08

Q. Do you still have them in order or do 03:08
we need to take a break. 03:08

A. They're in order. I kept them in 03:08
order. 03:08

MR. FAHRENKROG: Very sophisticated. 03:08

MR. HIGGINS: That's impressive. 03:08

Q. So if you go to the section of code 03:08
with the date June 27, 2003. It's going to be in 03:08
Bates ranges 279 through 318. 03:08

Page 150

A. They're not in order. 279 to 318? 03:09
Q. Yes. 03:09
A. It's the 279 part that's missing. 03:09 All right. 03:09
Q. 279 through 318; do you have that? 03:09
A. I do. 03:09
Q. The next sections -- why don't we go 03:09 off the record for a second. 03:10
VIDEOGRAPHER: The time is 3:10 p.m. 03:10 At this time we'll pause recording and go off the 03:10 record. 03:10
(Recess) 03:15
VIDEOGRAPHER: The time is 3:19 p.m. 03:19 At this time we're on the record. We are 03:19 recording. 03:19
BY MR. HIGGINS: 03:19
Q. If you pull out Exhibit 8, the R400 03:19 sequencer spec. 03:19
A. Okay. 03:19
Q. If you turn to Page 9 of the 03:19 document, Bates number 1555. 03:19
A. Yes. 03:20
Q. I'm going to be referring to this 03:20 figure -- 03:20
A. Okay. 03:20

Page 151

Q. -- and if you turn to the source code Page 303. It should be the module sq_input_arb.v. 03:20 03:20 03:20

A. Did you say 303? 03:20

Q. Yes. 03:20

A. Okay. 03:20

Q. Does this module, the sq_input_arb, correspond to the block labeled input arbiter on the diagram Exhibit 8? 03:20 03:20 03:20

MR. FAHRENKROG: Object to the form. 03:20

MR. PAIGE: Object to the form. 03:20

A. No. No, it doesn't. The input arbiter shown here is talking about input arbitration to the reservation stations. This particular module is arbitrating access to the GPRs when there's data to be written. So these are not the same. 03:21 03:21 03:21 03:21 03:21 03:21

Q. What RTL module corresponds to the input arbiter shown on Page 9 of Exhibit 8? 03:21 03:21

MR. FAHRENKROG: Object to the form. Are you asking about the RTL you've given him or something else? 03:21 03:21 03:21

MR. PAIGE: Foundation. 03:21

MR. HIGGINS: RTL in general. 03:21

Q. What RTL module of the R400 03:21

Page 152

corresponds to the input arbiter shown on Page 9 03:21
of Exhibit 8? 03:22
MR. FAHRENKROG: Object to the form. 03:22
MR. PAIGE: Assumes facts not in 03:22
evidence. 03:22
A. It would take me a certain amount of 03:22
time. I don't know off the top of my head. I 03:22
could -- I could try and go through this, and see 03:22
if I can find it, but I do not know. 03:22
Q. Do you know the name of the unit or 03:23
component that performs the arbitration into 03:22
reservation stations? 03:22
MR. FAHRENKROG: Vague and ambiguous. 03:22
A. You're talking about the code 03:22
component; is that what you're saying? The name 03:22
of the module; is that what you're asking for? 03:22
Q. Either one at this point. 03:23
A. No, I don't. I don't -- you know, I 03:23
see it called input arbiter here, and that's -- 03:23
and I'm hoping it will be called that, although 03:23
it isn't this one; I can tell you that. Let me 03:23
see what this says. 03:23
The pages that you've pulled out 03:24
appear to be the sequencer -- let me know when 03:24
you want to go on to something else. 03:24

Page 153

Q. While we have these sections pulled that you were just referring to, the one that starts on 756, sq_pix_thread_buff.v. 03:24 03:24 03:24

A. You're talking about some other section. 756. Okay. 03:24 03:24

Q. Does this module correspond to the pixel reservation station? 03:24 03:25

MR. FAHRENKROG: Object to the form. 03:25

MR. PAIGE: Vague and ambiguous. 03:25

A. Pix thread buffer. Yes, I believe that this corresponds to the pixel reservation station. 03:25 03:25 03:25

Q. If you look at the last section of code that you pulled out that begins Page 908, sq_vtx_thread_buff, is this RTL module the vertex reservation station? 03:25 03:25 03:25 03:25

MR. FAHRENKROG: Object to the form. 03:25

MR. PAIGE: Vague and ambiguous. 03:25

A. Yes, I think it is. 03:26

Q. From where do these two reservation stations receive inputs from? 03:26 03:26

A. According to this, it's from the ISM, which is probably the input state machine. 03:26 03:26

Q. What is the input state machine? 03:26

A. I don't know. I'm just reading from 03:26

Page 154

the code. I don't think I was ever familiar in 03:26
that detail with the sequencer. I certainly 03:27
haven't remembered anything after the extent of 03:27
time we're talking about. 03:27

I'm just looking for it here. So if 03:27
I can find it, I don't know if I can. I think I 03:27
have to analyze this code in fairly high detail 03:27
to really track it back. 03:27

Here is something called the input 03:27
state memory. 03:27

Q. What is the input state memory? 03:28

A. I'm just looking on Page 934. I 03:28
don't know. I'd be looking for the input data. 03:28
Let me look at the top level -- I mean, I think 03:28
the way I would do this is if I had this code on 03:28
a -- in a computer as opposed to printed out, is 03:28
that I would search, starting with the inputs, 03:28
and trace them through. 03:28

It's a lot harder based on paper to 03:28
really say how data gets into those buffers. 03:28

Q. Can you tell by looking at these two 03:28
reservation stations where they receive inputs 03:29
from? 03:29

MR. FAHRENKROG: Object to the form. 03:29
Vague. 03:29

Page 155

A. Again, all I see is event and control packet input, initial values for state and status. It's from something called the ISM. So it's hard to go backwards like that. 03:29 03:29 03:29 03:29

Q. Is the input from the ISM the only input that the vertex and pixel reservation stations receive? 03:30 03:30 03:30

A. No. It receives -- there are other inputs here, where it's receiving control from the vertex cache and from the texture cache, as well as from the ALUs. 03:30 03:30 03:30 03:30

It also gets input from the execution arbiters and the return data from the -- there are a lot of inputs here. But those are more once the thread is loaded. 03:30 03:30 03:30 03:31

Q. Do either of the pixel or vertex reservation stations receive an input from an input arbiter? 03:31 03:31 03:31

MR. FAHRENKROG: Vague and ambiguous. 03:31

A. I cannot tell from this source code without further analysis whether this is an arbiter -- whether this output comes from an arbiter or not. 03:31 03:31 03:31 03:31

Obviously at one point there was an input arbiter based on this picture of May, 2002. 03:31 03:31

Page 156

So I expect the answer is, yes, that they do come 03:31
from an input arbiter. 03:31

And my recollection, which I cannot 03:31
attest to with 100 percent, is that vertices were 03:32
simply given priority on the input, but that was 03:32
just a temporary thing, and that there was 03:32
certainly a check to see whether space was 03:32
available in the GPRs. 03:32

Q. You say vertices were given priority 03:32
on the input. What input are you referring to? 03:32

A. I'm talking about -- I'm looking at 03:32
this picture here on that input arbiter. You 03:32
know, I'd have to go through this code in great 03:32
detail to find where that arbitration is done. 03:32

Q. The vertex reservation station, for 03:33
example -- 03:33

A. Yes. 03:33

Q. -- Page 908. 03:33

A. Okay. 03:33

Q. How many inputs are there to the 03:33
vertex reservation station? 03:33

A. Again, there are all of these signals 03:33
listed as inputs. I mean, some of these are 03:33
outputs as well. 03:33

Q. Are you referring to lines 241 03:33

Page 157

through 419? 03:34

A. Here are some inputs. I was actually just looking at the wire definitions in the inputs here. 03:34 03:34 03:34

So there are the parameters. Yeah, so if you look here, there's all of those test inputs. Then these ISM inputs, those are the input state machines that I'm talking about. 03:34 03:34 03:34 03:34

But there are a whole bunch of other inputs as well that I think are not part of loading the thread, but just part of having a thread in the buffer as it executes, and it gets updated. 03:34 03:34 03:34 03:34 03:35

Q. Other than the inputs on lines 262 through 268, that begin with ISM, are there any other inputs that relate to loading a thread in the buffers? 03:35 03:35 03:35 03:35

A. I don't believe so. I think those are the inputs. 03:35 03:35

Q. Do the ISM -- let me start over. Lines 262 to 268, within those ISM modules, do any of those contain a priority for vertices over pixels? 03:35 03:35 03:35 03:35

MR. FAHRENKROG: Object to the form. 03:35

MR. PAIGE: Objection. Vague and 03:35

Page 158

ambiguous. 03:35

A. These things would be after any priority in code, priority selection or priority arbitration would be done. So this would be the output of an arbiter. 03:35 03:35 03:35 03:36

Q. Is the ISM the arbiter? 03:36

A. The ISM might be the arbiter. I'd have to look at the ISM block, if there is a block labeled ISM. 03:36 03:36 03:36

Q. Do you know if there's a module labeled or titled ISM -- 03:36 03:36

A. I don't know. That's why I would have to analyze the code to figure out. 03:36 03:36

I mean, what I would do is I would take these ISM wires, these inputs, and trace them back to see what module produces them. 03:36 03:36 03:36

Q. If you look at pages 319 through 338. 03:36

A. Hang on. 03:36

Q. It's the directory of code we were looking at earlier. 03:36 03:36

A. Did you say 309? 03:37

Q. 319 through 338. 03:37

A. Okay. 319 through 338. 03:37

Q. Let me when you get it. 03:37

A. I'm there. 03:37

Page 159

MR. PAIGE: Just to be clear, your 03:37
representation is what Respondents printed off on 03:37
the computer and not everything found on the 03:37
computer, correct? 03:37

MR. HIGGINS: Yes, this represents 03:37
the prints that Respondents made up through Bates 03:37
number 1450. 03:37

MR. PAIGE: So it's not on the 03:37
computer; you're not saying that? 03:37

MR. HIGGINS: No. This is just the 03:37
-- actually, this is everything that's on the 03:37
computer. This is not what was printed. This is 03:37
everything that's on the computer. This is the 03:37
directory listing of what's on the computer. 03:37

MR. PAIGE: Okay. So what you said 03:38
before about Respondents printed is incorrect? 03:38

MR. HIGGINS: Respondents printed 03:38
from this collection of code in the computer. 03:38
This is a collection of code for the sequencer 03:38
that is on the computer. 03:38

MR. PAIGE: Okay. So what you're 03:38
saying is this is everything on the computer. 03:38
The stuff we have been looking at here and here 03:38
and here, this stuff that comes from the 03:38
computer, but is not necessarily everything on 03:38

Page 160

the computer? 03:38

MR. HIGGINS: Yes. The stuff we have been looking at is what Respondents chose to print. 03:38 03:38 03:38

MR. PAIGE: Okay. 03:38

Q. Mr. Gruber, in this directory listing of RTL code for the sequencer, where is the IMS? 03:38 03:38

A. ISM. 03:38

Q. ISM. 03:38

MR. FAHRENKROG: Object to the form. 03:38

MR. PAIGE: Join in the objection. 03:38

A. I'm looking. 03:38

MR. FAHRENKROG: I'd also like to state a general objection with respect to the questions that ask him to search through the source code. I think that's outside the scope of the subpoena on AMD to know every line of the source code set which has been set in front of him, which is over 2,000 pages or however many modules listed in this directory. 03:39 03:39 03:39 03:39 03:39 03:39 03:39 03:39

So I think you've had a lot of latitude with that questioning, but I do think it's outside the scope. 03:39 03:39 03:39

MR. HIGGINS: Well, we would disagree. 03:39 03:39

Page 161

A. I do see one module called sq_pism.v. 03:39
and another module called sq_vism.v. That would 03:39
be my guess is where those things are. 03:40

Q. Where are you referring to? 03:40

A. I'm on Page 323 and 324. 03:40

Q. You said 324? Do you know what 03:40
sq_pism refers to? 03:40

A. My guess is pixel input state 03:40
machine, and "v" is vertex input state machine. 03:40
As I said, I really never had strong familiarity 03:40
with the details of this code as opposed to the 03:41
functionality of it. 03:41

Q. Is sq_pism and sq_vism two separate 03:41
input arbiters? 03:41

MR. FAHRENKROG: Object to the form. 03:41

A. I don't know. 03:41

Q. If you go to Page 439. Pull out 439 03:41
through 537. In the module sq_pism, is there an 03:41
output labeled ism_rts? 03:42

A. I'm sorry, what are you looking for? 03:42

Q. The ISM outputs we were talking 03:42
about. Does the pism module receive -- 03:42

A. I'm looking. I don't actually see 03:43
them here. 03:43

Q. Does the pism include any arbitration 03:43

Page 162

method between pixels and vertices? 03:44
MR. FAHRENKROG: Object to the form. 03:44
MR. PAIGE: Vague and ambiguous. 03:44
A. All I can say is I don't see any 03:44
arbitration between pixels and vertices in this. 03:44
Q. Would the pixel input state machine 03:44
receive vertices as an input? 03:44
A. I don't see any vertices received as 03:44
an input. 03:44
Q. Would the vertex input state machine 03:44
receive pixels as an input? 03:44
MR. FAHRENKROG: Objection. 03:44
Foundation. 03:44
A. I'd have to see that code. 03:44
Q. It's on Page 530. 03:44
A. I don't see any pixel-related input 03:45
into the vism. 03:45
Q. If the vism doesn't receive pixel 03:45
input, can it arbitrate between vertices and 03:46
pixels? 03:46
MR. FAHRENKROG: Object to the form. 03:46
MR. PAIGE: Join in the objection. 03:46
A. No. I don't see how. 03:46
Q. Does either the vertex input state 03:46
machine or the pixel input state machine perform 03:46

Page 163

any arbitration between vertices and pixels? 03:46

MR. FAHRENKROG: Object to the form. 03:46

A. I don't believe so, based on my 03:46
current knowledge of this code, which is five 03:46
minutes old. 03:46

VIDEOGRAPHER: The time is 3:46 p.m. 03:46
At this time we'll end cassette number four. 03:46
We'll stop recording and go off the record. 03:46

(Recess) 03:56

VIDEOGRAPHER: The time is 3:57. At 03:57
this time we'll begin cassette number five. We 03:57
are on the record. We are recording. 03:57

BY MR. HIGGINS: 03:57

Q. Mr. Gruber, if you go back to the 03:57
page that begins on 319. It's the directory 03:57
listing. 03:57

A. Okay. 03:57

Q. In this directory listing of RTL 03:57
modules in the sequencer, can you identify any 03:57
module that performs arbitration between vertex 03:57
and pixel threads to the reservation stations? 03:57

MR. FAHRENKROG: Object to the form. 03:58
And again, outside the scope of the topics, 03:58
asking him to search through source code for an 03:58
answer to your question. 03:58

Page 164

MR. PAIGE: Actually, searching the 03:58
directory listings, not even the source code 03:58
itself. 03:58

A. No. I'd have to analyze the code, 03:58
probably backwards, from the thread buffer or the 03:58
pixel buffer to say whether such arbitration 03:58
exists or not, and if so, what module it appears 03:58
in. 03:58

Q. Sitting here today, can you identify 03:58
any RTL module in the R400 that arbitrates 03:58
between vertex and pixel threads as an input to 03:58
the reservation stations? 03:58

MR. FAHRENKROG: Same objection. 03:59

A. Sitting here today, I cannot. I can 03:59
tell you how I would go about finding whether 03:59
such a module does exist, and it would involve 03:59
tracing the inputs and outputs back. But it's 03:59
very hard to do with a lot of paper. 03:59

Q. If you go to Page 286. It's going to 03:59
be in that big stack to your right most likely. 03:59

A. Why is it not in one of these? 286 03:59
is gone. Where did it go? 04:00

Q. You may have pulled a stack earlier 04:00
that started at 279. 04:00

MR. FAHRENKROG: I'll also state an 04:00

Page 165

objection that requiring the witness to navigate over 2,000 pages of code in random places, nine or more places, is very burdensome for the witness. 04:00

A. This is 900, 756, 319, 430 -- I'm sorry, what page? 04:01

Q. 286. 04:01

A. Let me see if I for some reason put it way down here. 04:01

MR. HIGGINS: Do you want to go off the record to help him find it? 04:01

MR. FAHRENKROG: Sure. 04:01

VIDEOGRAPHER: The time is 4:01. At this time we'll pause recording and go off the record. 04:01

(Discussion off the record) 04:02

VIDEOGRAPHER: The time is 4:02. We are on the record. We are recording. 04:02

BY MR. HIGGINS: 04:02

Q. Mr. Gruber, do you have the source code pages 286 through 302? 04:02

A. Yes, I do. 04:02

Q. What is the function of sq_gpr_alloc module? 04:02

MR. PAIGE: Object to the form. 04:02

Page 166

MR. FAHRENKROG: Object to the form. 04:02
A. This allocates space in the general purpose register file as threads are allocated. 04:03 04:03
Q. Does the GPR alloc module include any priority? 04:03 04:03
MR. FAHRENKROG: Object to the form. 04:03
MR. PAIGE: Vague and ambiguous. 04:03
A. There is some priority associated with this dynamic mode in the sense that based on the dynamic mode, it chooses how to move the head and tail pointer boundary. 04:03 04:03 04:03 04:03
If you look on Page 293, there's where that code starts. 04:03 04:03
Q. How does the dynamic mode utilize a priority? 04:04 04:04
A. Well, it's determining whether to allocate more space for pixels or space for vertices via this pointer control state machine. So it does that. 04:04 04:04 04:04 04:04
I believe the inputs are based on how busy the output buffers are associated with either vertices or pixels. 04:04 04:04 04:04
Q. Which inputs are based on how busy the output buffers are? 04:04 04:04
A. The inputs into the boundary control 04:04

Page 167

mechanism, that is, the boundary between pixels 04:05
and vertices in the GPR register file. 04:05

Q. How is it determined whether to 04:05
allocate more space for pixels or vertices? 04:05

A. I'm just trying to find it in the 04:05
codes. Since I have the code here, I don't want 04:05
to -- well, it uses the amount of space in the 04:05
output buffers to decide whether we're more pixel 04:06
limited or more vertex limited. 04:06

I know that from the high-level 04:06
documentation. I'd have to analyze this code to 04:06
really point out where in the code that is done. 04:06

Q. If the GPR alloc module is not 04:06
operating in the dynamic mode, is it operating in 04:06
static mode? 04:06

A. Yes. 04:06

Q. When it operates in static mode, does 04:06
it utilize any priority? 04:06

MR. PAIGE: Object to the form. 04:06
Vague and ambiguous. 04:06

A. No, it does not. 04:06

MR. FAHRENKROG: Counsel, is it okay 04:07
for the witness to move the yellow paper out of 04:07
the way to clear space for the next exhibit? 04:07

MR. HIGGINS: Yes. Your choice. 04:07

Page 168

(Gruber Exhibit 15, U.S. Patent No. 04:07
6,897,871, marked for identification) 04:08

(Gruber Exhibit 16, U.S. Patent No. 04:07
7,239,322, marked for identification) 04:08

Q. You have been handed what's been 04:08
marked as Exhibits 15 and 16. Exhibit 15 is U.S. 04:08
Patent No. 6,897,871. Exhibit 16 is 7,239,322. 04:08
Have you seen these exhibits before? 04:08

A. Yes. 04:08

Q. Are you listed as an inventor on both 04:08
of these patents? 04:08

A. Yes, I am. 04:08

Q. The register file allocation we were 04:08
just discussing in the GPR and alloc module, is 04:09
that functionality disclosed in these patents? 04:09

MR. FAHRENKROG: Object to the form. 04:09
Calls for a legal conclusion. Also overly broad. 04:09

A. Which functionality were you talking 04:09
about? 04:09

Q. The register allocation. In 04:09
particular, the dynamic register allocation. 04:09

A. No, these don't cover the dynamic 04:09
register allocation. 04:10

Q. Do either of these patents describe 04:10
how threads are created? 04:10

Page 169

MR. FAHRENKROG: Object to the form. 04:10
Calls for a legal conclusion. Overly broad. 04:10

MR. PAIGE: Vague and ambiguous. 04:10

A. Well, this patent does talk about 04:10
thread creation. I'm looking at arbiter here 101 04:11
for loading threads. 04:11

Q. Do either patent -- does either 04:11
patent discuss a priority related to thread 04:11
creation? 04:11

MR. FAHRENKROG: Same objection as to 04:11
form. Legal conclusion. Overly broad. 04:11

MR. PAIGE: Join in the objections. 04:11

A. I don't see anything in here about 04:13
priority on thread creation. Although, I do see 04:13
lots about priority on thread execution. No, I 04:13
take that back. 04:13

If I look at Line 65, Column 3, on 04:13
the '322 patent, "one embodiment an input arbiter 04:13
provides to the command threads to each of the 04:13
first reservation station, and the second 04:14
reservation station 304 based on whether the 04:14
command thread is a pixel command thread or 04:14
vertex command thread." 04:14

Q. Is that a priority or is it just 04:14
sending vertex threads to the vertex reservation 04:14

Page 170

station and pixel threads to the pixel 04:14
reservation station? 04:14

MR. FAHRENKROG: Object to the form. 04:14
Calls for a legal conclusion. 04:14

MR. PAIGE: Argumentative. 04:14

A. I'm not sure what the legal 04:14
interpretation of this would be. 04:14

Q. What is your interpretation of this? 04:14

A. Well, the fact that it's an arbiter 04:14
suggests to me that there is some arbitration. I 04:14
expect that the arbitration is such that even 04:14
though a pixel thread has to go to a pixel 04:14
station and a vertex thread has to go to a vertex 04:14
station, there's an arbitration simply due to 04:14
time when that happens. 04:15

In other words, the reason why you 04:15
would call it an arbiter here is because there's 04:15
only one set of wires that come out, and that's 04:15
either loaded into the pixel station or the 04:15
vertex station. 04:15

So you may have done that either to 04:15
save wires or to save control logic where you can 04:15
just -- you can just load one thread in a given 04:15
cycle, and that's why you would do an input 04:15
arbiter as opposed to just having two entirely 04:15

Page 171

separate paths. 04:15
Q. Is there any disclosure in the patent 04:16
of how this arbiter receives the command threads? 04:16
MR. FAHRENKROG: Object to the form. 04:16
Calls for a legal conclusion. Also this line of 04:16
questioning about the disclosure of these patents 04:16
is outside the scope of the deposition topics. 04:17
MR. PAIGE: Join in the objections. 04:17
A. Let me just look at the picture that 04:17
this refers to. It says "not illustrated in 04:17
Figure 4." Well, that's helpful. 04:17
I don't see anything further 04:17
referenced in this paragraph. 04:18
Q. The arbiter that you're referring to 04:18
in Figure 4, is that arbiter 306? 04:18
A. No, it's not, because Line 66, I 04:18
guess, says "although not illustrated in Figure 04:18
4." 04:18
Q. So there's no input arbiter described 04:18
in this specification? 04:18
MR. FAHRENKROG: Object to the form. 04:18
Asked and answered. Legal conclusion and outside 04:18
the scope of the topics. 04:18
MR. PAIGE: And misstates prior 04:18
testimony. 04:18

Page 172

A. I pointed to the language on page -- 04:18
on Column 3, Line 67 that does talk about an 04:18
input arbiter. 04:19

Q. Does that input arbiter that you're 04:19
referring to utilize any priority with respect to 04:19
vertex and pixel threads? 04:19

MR. FAHRENKROG: Same objection. 04:19

A. Well, it's an arbiter, and it has to 04:19
arbitrate on some priority basis. As I say, this 04:19
patent does not seem to expand on how that 04:19
arbiter works. 04:19

Q. Does an arbiter have to use a 04:19
priority? 04:19

MR. FAHRENKROG: Object to the form. 04:19

A. Well, it doesn't have to use a fixed 04:19
priority, but it has to arbitrate. So it has to 04:19
select one, and the one who he selects has higher 04:19
priority. At least that's how I interpret the 04:19
word "arbiter." 04:19

Q. What do you mean by "the one who he 04:19
selects has higher priority"? 04:19

A. I mean, an arbiter selects in between 04:19
one or more inputs, and I would interpret the one 04:20
that he selects as being the one who had higher 04:20
priority, at least for that cycle, even if he's 04:20

Page 173

selecting on a random basis, then on that cycle, 04:20
whoever won that spin of the dice, would have 04:20
higher priority. 04:20

Q. If the arbiter just goes back and 04:20
forth between two inputs, is that a priority? 04:20

MR. FAHRENKROG: Object to the form. 04:20
Vague. Incomplete hypothetical. 04:20

A. I would say on any given cycle, 04:20
someone has priority, and if you didn't have 04:20
priority in the last cycle, then you'll have 04:20
priority on this cycle. 04:20

Often, like a round robin arbiter, 04:20
you would talk about who has priority on this 04:20
cycle. 04:20

Q. Exhibit 15, the '871 patent. Does 04:20
the '871 patent describe a multithreaded graphics 04:21
processor or a single-threaded graphics 04:21
processor? 04:21

MR. FAHRENKROG: Object to the form. 04:21
Vague. Calls for a legal conclusion. 04:21

MR. PAIGE: Join in the objections. 04:21

A. It's a multithreaded processor. 04:21

Q. What leads you to believe that it is 04:22
a multithreaded processor? 04:22

A. Well, I'm looking at Column 5, Line 04:22

Page 174

14. It says, "The unified shader 62 has the 04:22
ability to simultaneously perform vertex 04:22
manipulation operations and pixel manipulation 04:22
operations at various degrees of completion by 04:22
being able to freely switch between such programs 04:22
or instructions, maintained in the instruction 04:22
store very quickly." 04:22

Q. Can it perform those operations 04:22
without operating on threads? 04:22

MR. FAHRENKROG: Object to the form. 04:22
Vague. Also outside the scope of this 04:22
deposition. 04:22

MR. PAIGE: Join in the objections. 04:23

A. Can it do it without threads? Well, 04:23
no, because the operations are in various degrees 04:23
of completion. So you have to track where you 04:23
are. 04:23

So that implies that you have -- that 04:23
you're multithreading; at least it does to my 04:23
mind. 04:23

Q. Does the '871 patent mention threads 04:23
anywhere? 04:23

MR. FAHRENKROG: Object to the form. 04:23

A. I can't give you a definitive answer 04:23
without spending time looking at every single 04:23

Page 175

point. I don't see the word "thread" anywhere in the -- off the top of my head or in my quick glance. 04:23 04:23 04:23

Q. Can a unified shader operate without threads? 04:24 04:24

MR. FAHRENKROG: Object to the form. Overly broad. 04:24 04:24

MR. PAIGE: Vague and ambiguous. 04:24

A. I think you could design such a unified shader. It wouldn't be a good unified shader. 04:24 04:24 04:24

Q. What did you do to prepare for today's deposition? 04:24 04:24

A. I went over numerous versions of top-level specs and block-level specs. I clarified in my mind the sequence of chips that ATI produced. 04:24 04:25 04:25 04:25

I looked over the patent -- the patent applications to the follow-on patent application that I had signed. I looked over the code attached to those, as well as obviously meeting with my lawyers. 04:25 04:25 04:25 04:25 04:25

Q. Which top-level and block-level specs did you review? 04:25 04:25

A. I looked at the sequencer 04:25

Page 176

specification, I looked at the shader 04:25
specification. I looked at the R400 -- I don't 04:26
know whether it was called the overview spec or 04:26
it was the initial spec produced by Steve Morein. 04:26
I looked at various versions of that. 04:26
Q. You said you met with your lawyers. 04:26
Which lawyers are you referring to? 04:26
A. Both Qualcomm lawyers and AMD 04:26
lawyers. 04:26
Q. Did you meet with Qualcomm and AMD 04:26
lawyers at the same time? 04:26
A. No, I did not. 04:26
Q. Did you talk to anyone at AMD to 04:26
prepare for today's deposition? 04:27
A. No. 04:27
Q. Did you talk to anyone at Qualcomm to 04:27
prepare for today's deposition? 04:27
A. No one beyond the legal department. 04:27
I mean, I spoke to both in-house Qualcomm counsel 04:27
as well as legal support. But that was more in 04:27
just knowing where to go. 04:27
Q. Since you left AMD, do you still 04:27
communicate with any employees of AMD? 04:27
MR. FAHRENKROG: Objection. Outside 04:27
the scope of the AMD topics. 04:27

Page 177

A. Occasionally I will have lunch or an 04:27
occasional dinner with AMD and ex-AMD employees. 04:28

Q. Which ex-AMD employees do you meet 04:28
with? 04:28

A. I had dinner with Bob Feldstein a 04:28
couple months ago. I occasionally have lunch 04:28
with Paul Mitchell who's at NVIDIA now. 04:28

Q. Which current AMD employees have you 04:28
met with? 04:28

A. I occasionally have lunch with Mark 04:28
Fowler. 04:28

Q. Anyone else? 04:28

A. It has been over a year, but I had 04:28
lunch with Phil Rogers about a year ago. 04:28

Q. Who is Phil Rogers? 04:29

A. Phil is CTO, I believe, at AMD 04:29
software CTO. 04:29

Q. Did you speak with any of these 04:29
people about the topics of your deposition today? 04:29

A. The last time I spoke to Bob 04:29
Feldstein, I asked him whether he thought -- I 04:29
think I said something like, Can you guys come up 04:29
with better patents to advocate than this, and he 04:29
said, Yeah, maybe they're not that good. That 04:29
was the extent. 04:29

Q. Why did you ask that? 04:29
MR. FAHRENKROG: Objection. Outside 04:29
the scope of the AMD-related topics on this line 04:29
of questioning. 04:29
A. I was surprised that NVIDIA didn't 04:30
have stronger patents. 04:30
Q. Why is it your view that these are 04:30
not strong patents? 04:30
MR. FAHRENKROG: Same objection. The 04:30
witness is testifying as an individual, not on 04:30
behalf of AMD. 04:30
MR. PAIGE: And in giving an answer, 04:30
please do not reveal the content of any 04:30
communications you've had with Qualcomm counsel. 04:30
A. I just didn't think they were very 04:30
fundamental. 04:30
Q. Why did you not think they were very 04:30
fundamental? 04:30
MR. FAHRENKROG: Same objection. 04:30
MR. PAIGE: Same caution. 04:30
A. The particular -- the particular 04:30
aspects of the patents were having to do with 04:30
lighting and, you know, special lighting 04:30
processors, which I thought nobody used any more. 04:31
That was the main issue. 04:31

Page 179

Q. Which patents are you referring to? 04:31
A. I don't remember. I just glanced at 04:31
these patents to see what the general subject 04:31
was, and there was one patent involved in 04:31
prosecution that had to do with a specialized 04:31
lighting processor. 04:31
Q. What do you mean there was one patent 04:31
involved in the prosecution? 04:31
A. I believe there were five or six 04:31
patents asserted against Qualcomm and Qualcomm's 04:31
customers, and one of them had to do with the 04:31
lighting. 04:31
Q. When you said involved in 04:31
prosecution, were you referring to the patents 04:31
asserted in this case? 04:31
A. Yes. 04:31
Q. When you made that comment to Bob 04:31
Feldstein, was that just about this one patent or 04:31
was it about all seven asserted patents? 04:31
A. I said it that I didn't see any of 04:31
them that were particularly strong. 04:32
Q. Why, in your view, were they not 04:32
particularly strong? 04:32
MR. FAHRENKROG: Restate the 04:32
objection that the witness is testifying as an 04:32

Page 180

individual, not as a corporate representative of 04:32
AMD on this topic. 04:32

A. Just didn't -- I just didn't see 04:32
anything that would really apply to most existing 04:32
graphics processors, most modern ones. 04:32

Q. So who, in your view, has stronger 04:32
graphics patents than NVIDIA? 04:32

MR. FAHRENKROG: Same objection. 04:33

A. I don't know. I mean, you know, I 04:33
don't know that anyone does, but I thought that 04:33
NVIDIA's would be stronger. That's all I can 04:33
say. Maybe no one has a strong patent. 04:33

Although, I like this one, the one 04:33
with my name on it. 04:33

Q. Which one are you referring to? 04:33

A. The unified shader. 04:33

Q. Which one is that? 04:33

A. '871. 04:33

Q. Why do you like that one? 04:33

A. Because I think it covers a 04:33
fundamental aspect, because I think unified 04:33
shading is important, and I think this covers it. 04:33

Q. What fundamental aspect of unified 04:33
shading does the '871 -- 04:33

A. I said it covers unified shading, 04:33

Page 181

which I think is fundamental to modern 04:33
processors. 04:33

Q. When did you meet with your AMD 04:33
lawyers prior to today's deposition? 04:34

A. Yesterday. 04:34

Q. How long did you meet with AMD 04:34
lawyers? 04:34

A. Six hours. 04:34

Q. What did you discuss during those six 04:34
hours? 04:34

MR. FAHRENKROG: Objection. Calls 04:34
for a privileged communication between counsel 04:34
for the non-party AMD and former employee, which 04:34
is privileged. So I instruct you not to answer 04:34
that question. 04:34

A. I'm not going to answer that 04:34
question. I could have figured that one out by 04:35
myself, though. 04:35

Q. When did you leave AMD and go to 04:35
Qualcomm? 04:35

A. Early 2009. 04:35

Q. Why did you leave AMD and go to 04:35
Qualcomm? 04:35

A. I went to Qualcomm as part of the 04:35
mobile team, an effort that I had been involved 04:35

Page 182

with. 04:35

MR. HIGGINS: Let's take a quick 04:35
break. 04:35

VIDEOGRAPHER: The time is 4:35 p.m. 04:35
At this time we're stop recording and go off the 04:35
record. 04:35

(Recess) 04:36

VIDEOGRAPHER: The time is 4:40 p.m. 04:39
At this time we are on the record. We are 04:40
recording. 04:40

EXAMINATION BY 04:40
MR. CHATTERJEE: 04:40

Q. Hello, Mr. Gruber. I introduced 04:40
myself earlier. I'm Neel Chatterjee, and I 04:40
represent NVIDIA in this case. 04:40

I want to ask you some follow-up 04:40
questions from my colleague's questioning, and I 04:40
want to start with where he ended which was 04:40
talking about your review of NVIDIA's patents. 04:40
When did you do that? 04:40

A. I'm sorry, when did I look over 04:40
NVIDIA's patents? 04:40

Q. Yes. 04:40

A. The day that they became public about 04:40
this lawsuit. So I don't know when that was, but 04:40

Page 183

it's probably a couple months ago. 04:40

Q. So you looked at it the day that Qualcomm was sued? 04:40 04:40

A. I don't know if it was the day that Qualcomm was sued. The day it was publicly announced. 04:40 04:40 04:41

Q. Okay. Other than your counsel, have you talked with anyone else, other than Mr. Feldstein, about these patents? 04:41 04:41 04:41

A. No. As I say, I really didn't talk to Bob about the patent. I just kind of gave him my opinion of the value of the patent. We didn't talk about any technical aspect of the patent. 04:41 04:41 04:41 04:41

Q. Did you study the claims in the patents? 04:41 04:41

A. I just -- I wouldn't say that I studied them. I just looked at them. I read through the patents. 04:41 04:41 04:41

Q. And did you read the claims? 04:41

A. Yes. 04:41

Q. And formed opinions about them? 04:41

MR. PAIGE: Object to the form. 04:41

A. Did I form any opinions about them? I generally judge them on like, you know, whether I can think of areas where there's infringement 04:41 04:41 04:41

Page 184

and stuff that I've worked on. I didn't see 04:41
anything that I was overly worried about. 04:42
Q. Had you seen any of them before? 04:42
A. Any of those patents or any of those 04:42
claims? 04:42
Q. Yes, any of those patents or any of 04:42
those claims. 04:42
A. I don't think so. 04:42
Q. Were you aware that Exhibit 9 and 10, 04:42
the 131 declarations that you submitted, were 04:42
actually in response to an assertion by the 04:42
examiner that the patents you got were invalid, 04:42
and this was an attempt to swear behind the dates 04:42
of those patents? 04:42
MR. FAHRENKROG: Object to the form. 04:42
Calls for a legal conclusion. 04:42
A. No, I wasn't fully aware that was the 04:42
reason for this. 04:42
Q. Would it surprise you to learn that? 04:42
MR. FAHRENKROG: Object to the form. 04:42
A. Surprise me? Maybe a little bit. 04:42
Q. Why would it surprise you a little 04:42
bit? 04:42
MR. FAHRENKROG: Object to the form 04:42
of the question. 04:42

Page 185

A. Simply because I know -- my feeling 04:42
was that ATI was significantly in advance on the 04:43
development of a unified shader. 04:43

Q. Why is that your feeling? 04:43

A. Because I know that we had been 04:43
working on one for a long time, and then when 04:43
Xenos came out, you know, we were ahead of -- 04:43
that was the first unified shader on the market, 04:43
and at the time the Xenos came out, NVIDIA had 04:43
been poopooing the whole idea of a unified 04:43
shader. 04:43

Q. Who at in NVIDIA was poopooing the 04:43
idea of a unified shader? 04:43

A. I just know what I read in things 04:43
like AnandTech. 04:43

Q. So it was kind of by reading kind of 04:43
the marketing -- 04:43

A. Yeah, the marketing. I'm not talking 04:43
about talking to any individual. 04:44

Q. Kind of like the industry pundits? 04:44

A. Right. 04:44

Q. Is that right? 04:44

A. Yes. 04:44

Q. Are you proud of the unified shader 04:44
technology you helped create? 04:44

Page 186

A. Yes. 04:44

Q. And how many patents do you have? 04:44

A. I don't know the exact number. It's probably 40, 50. 04:44 04:44

Q. How many of them relate to this unified shader architecture? 04:44 04:44

A. A handful, I believe. Not more than that. 04:44 04:44

[redacted]

[redacted]

Q. Do you have any interest in the 04:45
outcome of this case? 04:45

MR. FAHRENKROG: Vague and ambiguous? 04:45

A. No direct interest. Kind of rooting 04:45
for Qualcomm, but I don't have any monetary 04:45
interest. 04:45

Q. You're rooting for Qualcomm because 04:45
you work there? 04:45

A. Yes. 04:45

Q. You made a comment when Mr. Higgins 04:45
was talking to you, you told Mr. Feldstein you 04:45
were surprised that NVIDIA did not have stronger 04:45
patents. Why was that surprising to you? 04:45

A. NVIDIA has been in the graphics field 04:45
a long time, and they were first in many things 04:45
in the graphics field. So I just would have 04:45
expected that, you know, maybe they had some 04:46
killer patents. 04:46

Q. Can you give me some examples of 04:46
things where they were first? 04:46

A. I think they were first in probably a 04:46
vertex shader. Other areas they were first? I 04:46
don't know. It's a long time. I'm trying to 04:46
think back. 04:46

Page 188

There's a long history of graphics 04:46
there. They would have been first at least in 04:46
the PC world in a lot of the floating point 04:47
aspects of graphics. 04:47

Q. Anything else you can think of? 04:47

A. Not off top of my head. 04:47

Q. Would you say NVIDIA is an innovative 04:47
company? 04:47

A. Yes. 04:47

Q. I want to go to the very beginning of 04:47
your deposition with Mr. Higgins. You had talked 04:47
about -- I think the term you used was evolution, 04:47
that you talked about going from the R400 to the 04:47
R600, and you said there was an evolution. Do 04:47
you recall that? 04:47

A. Yes. 04:47

Q. Why were the product numbers changed? 04:47

A. Well, because we generally decide on 04:47
product numbers aimed at market windows, and we 04:47
like to have a sequence of product numbers. 04:47

So, you know, when you come out with 04:47
a brand new architecture, you want to have a 04:47
brand new product number associated with it, and 04:48
it would have been awkward, for instance, to have 04:48
the West Coast team do an R300 and then do an 04:48

Page 189

R500 with a missing R400. 04:48

I'm just speaking for the marketing group. It's a guess on my part. But that was kind of how we did both marketing and engineering. 04:48 04:48 04:48 04:48

Q. Now, was the R420 part of this evolution? 04:48 04:48

A. It was part of an evolution of the R300 series. 04:48 04:48

Q. Was it part of the evolution from the R400 to the R600? 04:48 04:48

A. No, it was not. 04:48

Q. So do you know why it was released as the R420? 04:48 04:48

A. Well, because they wanted something slotted into something called the R400 series. They didn't want to use the word R400, because that would lead to internal confusion. So they picked something that was R4, but not R400. 04:48 04:48 04:48 04:48 04:49

Q. Who is the "they"? 04:49

A. I don't know. 04:49

Q. It was lore within the company? 04:49

A. Hmm? 04:49

Q. It was lore within the company? 04:49

A. Yes. I don't know where the name 04:49

Page 190

came from, but this is the reason. 04:49

Q. Now, this term "evolution," what do you mean by the term "evolution"? 04:49 04:49

A. Well, any design as you work on it, you learn things about the design. You learn things during the simulation process, what you find out, where there are bottlenecks that you didn't know about otherwise. 04:49 04:49 04:49 04:49 04:49

You also, as the time goes on, the requirements for the part have changed. Something that was fast enough if released at time end is not fast enough if released at time end plus six months. 04:49 04:49 04:49 04:49 04:49

The technology changes such that, you know, trade-off between how much memory costs and logic costs and the overall size of the part that you're trying to build. So those types of things. 04:49 04:50 04:50 04:50 04:50

Q. And with those requirements changing, does that cause alteration of the designs? 04:50 04:50

A. Yes. 04:50

Q. Now, how long a time period was this for the evolution of the R400 to the R600? 04:50 04:50

A. I don't recall off the top of my head when the R600 shipped, because I didn't review 04:50 04:50

Page 191

that as part of this -- my preparation here. But the R200 -- sorry, the R400, you know, changed to R500 in early 2003. So I think the R600 was about three years later in terms of when it finally shipped. 04:50 04:50 04:50 04:50 04:50

Q. So when was the R400, when did the development of it start? 04:50 04:51

A. The development -- the engineering -- the architectural effort started in late 2000, and then over 2001, it moved from a pure architecture effort to an architecture and implementation effort. 04:51 04:51 04:51 04:51 04:51

Q. So I just want to make sure I have the timing right. So the R400, the engineering effort, started around late 2000? 04:51 04:51 04:51

A. No -- well, if you include architecture effort, yeah, as opposed to, you know, people actually working coding RTL. People working coding RTL started later than that. I would say, you know, probably in mid-2001. 04:51 04:51 04:51 04:51 04:51

There's an architecture section of a schedule, and then there's, you know, implementation. It may overlap, but architecture leads. 04:51 04:51 04:51 04:51

Q. When was the R400 moved into the R500 04:51

Page 192

again? 04:51

A. Early 2003. 04:51

Q. Early 2003. Then did you say two or 04:51
three years after that the R600 development 04:52
started? 04:52

A. No, I didn't mean that's when it 04:52
started. I was thinking about when it was done. 04:52

Q. Okay. So it was done -- the R600 was 04:52
done two or three years later? 04:52

A. Yes. I would say probably about 04:52
three years later, because there was an R500 in 04:52
between. 04:52

Q. And was there an R500 that was 04:52
actually commercially released? 04:52

A. No -- well, again, similar to the 04:52
R420, there was an R500 part in that time frame 04:52
that was unrelated to this R500 effort. 04:52

Q. So was the first product that had the 04:52
unified shader architecture that was commercially 04:52
released the R600? 04:52

MR. PAIGE: Objection. Misstates 04:52
prior testimony. 04:52

A. No, because the Microsoft Xbox had 04:52
the unified shader architecture very closely 04:53
related to what we were working on in the R500. 04:53

Page 193

Q. And that was the Xenos chip? 04:53

A. The Xenos chip. 04:53

Q. When about was that? 04:53

A. That was like for the holiday season in 2005. 04:53 04:53

Q. So it would have been commercially released when, six months earlier? 04:53 04:53

A. Well, I don't know what you mean by -- it was, you know, a part for one customer. So it was never really offered for sale, except to that one customer. 04:53 04:53 04:53 04:53

Q. I understand that. But when you say for the holiday season, obviously AMD or ATI would have had to sell it before -- 04:53 04:53 04:53

A. Oh, you're saying when it went on sale? 04:53 04:53

Q. To Microsoft. 04:53

A. Oh, to Microsoft. Yeah. I wish I knew. It had to be at least six months prior to that. 04:53 04:53 04:53

Q. You're not certain exactly how long? 04:53

A. I'm not certain. 04:53

Q. Do you know why the R400 did not meet its anticipated release date? 04:54 04:54

A. The schedule just slipped. It was a 04:54

Page 194

more complicated part than we thought. We had 04:54
some -- we had some issues. I don't recall the 04:54
shader being the real issue, because, you know, 04:54
we started that early, and we did a lot of 04:54
engineering on that. 04:54

I know, as I said earlier, that the 04:54
long pull was the RB, the render back end. That 04:54
just was not coming together. 04:54

Q. So the render -- it's your 04:54
recollection that for the R400 the render back 04:54
end was the big obstacle to the release? I need 04:54
you to say with a yes or no. 04:54

A. Yes. I'm sorry. 04:54

Q. I always say there's two important 04:55
people in the room, the witness and the court 04:55
reporter. If you don't say it, the court 04:55
reporter can't get it down. And that's just not 04:55
because I'm married to a court reporter. 04:55

I want to talk -- let's see. Would 04:55
the RTL code for the R400 be the best indicator 04:55
of what had been done on the chip? 04:55

MR. FAHRENKROG: Object to the form. 04:55

A. Yes. The RTL code defines what was 04:55
done on the chip. 04:55

Q. A couple small questions. How long 04:55

Page 195

did you meet with AMD counsel? I don't want to know what was said, just how long was it? 04:55 04:55

MR. FAHRENKROG: Objection. Asked and answered. 04:55 04:55

A. About six hours. 04:55

Q. How about Qualcomm counsel? 04:56

A. Yesterday, because I had a deposition a couple weeks ago as well? 04:56 04:56

Q. Right, yesterday. 04:56

A. Maybe two hours. 04:56

Q. Do you know someone named Michael Doggett? 04:56 04:56

A. I do. 04:56

Q. Was Mr. Doggett involved in the R400? 04:56

A. Yes. 04:56

Q. What was his involvement? 04:56

MR. FAHRENKROG: Objection. Asked and answered. 04:56 04:56

A. I remember him -- Michael did a number of things, but I specifically remember him working on the texture pipe, as well as I mentioned that he was involved in some academic relations. 04:56 04:56 04:56 04:56 04:56

Q. You mentioned someone named Laurent earlier today, saying that he wrote much of the 04:56 04:56

Page 196

RTL code. Do you remember talking about that? 04:56
A. Yes. 04:56
Q. And what was his involvement in the 04:56
development of the R400? 04:56
A. He was involved in the architecture 04:56
fairly early on. He was hired in to work on the 04:56
architecture of the R400. 04:56
One of the things that happens in a 04:56
development cycle is that you have people who are 04:57
busy, because they're getting the current chip 04:57
out. So you have this kind of overlapping 04:57
development. And in the early part, it's hard 04:57
getting a lot of people working on it. So you 04:57
are working with a small group of architects. 04:57
Laurent had the advantage, he didn't 04:57
have a previous project to work on. 04:57
Q. Did you talk to him at all to prepare 04:57
for your deposition today? 04:57
A. I did not. 04:57
Q. Why not? 04:57
MR. FAHRENKROG: Object to the form. 04:57
A. He just simply wasn't offered to me 04:57
to talk to. 04:57
Q. You were talking earlier today about 04:57
a register file versus a preservation station. 04:57

Page 197

Do you remember talking about that -- 04:57

A. Yes. It was a reservation station. 04:57

Q. Reservation station. I'm sorry. So 04:57
I'm just going to ask you a really basic station. 04:57

What is a register file? 04:58

A. So what we meant by the register file 04:58
-- a register file is generically just a bunch of 04:58
storage. 04:58

But what we meant by the general 04:58
purpose register file was the storage associated 04:58
with a thread where the thread would keep all of 04:58
the temporary data that it calculated while it 04:58
was producing its final output. 04:58

Q. And a reservation station, what is 04:58
that? 04:58

A. So a reservation station is also 04:58
associated with a thread, but rather than holding 04:58
data that's produced by the shader, it holds all 04:58
the control information associated with the 04:58
thread; for instance, the current program counter 04:58
associated with the thread. 04:58

Q. Mr. Higgins asked you about a vertex 04:58
cache. Do you remember talking about that? 04:59

A. Yes. 04:59

Q. Do you remember the first time AMD or 04:59

Page 198

ATI came up with the idea of adding a vertex 04:59
cache? 04:59
MR. PAIGE: Object to the form. 04:59
Vague and ambiguous. 04:59
A. No. We had vertex caches prior to 04:59
the R400. The issue was whether to add a vertex 04:59
cache to the R400. 04:59
I know that that came up somewhere in 04:59
the R400 design phase, and we tried a number of 04:59
-- well, one, we didn't know whether we really 04:59
needed it. It wasn't needed for functionality. 04:59
And as a matter of fact, we later 05:00
produced parts -- I say "we," in this case, 05:00
Qualcomm has produced parts based on the R400 05:00
database. It doesn't have a vertex cache. It's 05:00
more of a performance issue, and whether we 05:00
needed it for performance. 05:00
So we kind of fought it in the R400, 05:00
and finally gave in at some point. 05:00
Q. Do you know if when you gave in, it 05:00
was at the R400 or the R500 stage of things? 05:00
A. My recollection, which may not be 05:00
perfect, was it was part of the R500 changes, 05:00
between the R400 and the R500. 05:00
Q. Now, you referred in the early part 05:00

Page 199

of the deposition to a C model emulator. What is a C model emulator? 05:00 05:00

A. So what I mean by that is a model of the chip that's in -- that's written in C or C++ that is functionally the same as the chip; that is, it produces the same outputs, and it matches at certain checkpoints within the chip. 05:00 05:01 05:01 05:01 05:01

Q. And the C model is something that operates on the simulation software, right? 05:01 05:01

A. It is simulation software. It just runs. You compile it and it's a program like anything else. It's not built on top of anything. 05:01 05:01 05:01 05:01

Q. Do you recall what simulation program you used? 05:01 05:01

A. Well, this is what I'm saying. The C model doesn't need a simulation program to run. It is a program, and it runs just like Microsoft Word runs. 05:01 05:01 05:01 05:01

Q. So you don't need something like VCS or anything like that? 05:01 05:02

A. There is something else like VCS that takes in the RTL as the input, and that simulates that. 05:02 05:02 05:02

MR. PAIGE: Could I just register an 05:02

objection here, Neel. I understand we had an 05:02
agreement in the case that we would have 05:02
different questioners asking questions based on 05:02
different patents, different areas, things like 05:02
that. 05:02

It appears what's happening now is 05:02
you're asking a bunch of follow-up questions to 05:02
questions Mr. Higgins asked. I'm not sure why we 05:02
need two questioners for that. I object to it, 05:02
but you can continue. 05:02

Q. In Exhibit 9 and 10 of the 05:02
deposition, you had these declarations that you 05:02
submitted and you referred to RTL simulation in 05:02
the declaration. Do you remember that? 05:02

A. Yes. 05:02

Q. Did you use a particular RTL 05:02
simulator? 05:02

MR. FAHRENKROG: Objection. Asked 05:02
and answered. 05:02

MR. PAIGE: Objection. Asked and 05:02
answered. 05:02

A. We did. I don't recall the ones that 05:02
we used, and we might have used more than one 05:02
during the course of the project. 05:02

Q. Did you use a hardware emulator? 05:02

A. We might have. I know that we were 05:02
doing things with those. I don't recall whether 05:03
the R400 was ever ported to a hardware emulator. 05:03

Q. Do you know if there was ever a 05:03
physical representation of a chip for the R400 05:03
ever made? 05:03

MR. FAHRENKROG: Objection. Vague. 05:03
Asked and answered. 05:03

A. There was no chip made on the 05:03
database called R400. There was chips made on 05:03
databases that were based on the R400, the R500, 05:03
and the Xenos came pretty soon after the change 05:03
from R400 to R500. 05:03

Q. Was there a way to assign vertex data 05:03
and pixel data to threads based on allocation 05:03
priority between the two? 05:04

MR. PAIGE: Objection. Vague and 05:04
ambiguous. 05:04

MR. FAHRENKROG: Object to the form. 05:04

A. The thread pools themselves, that is 05:04
the control mechanism, was separate. There was a 05:04
dynamic mechanism for sharing the GPRs that was 05:04
based on a priority that was kind of 05:04
heuristically developed in that part of the RTL, 05:04
based on looking at how busy the output buffers, 05:04

Page 202

be it the vertex and pixel buffers, was. 05:04

So if there weren't enough vertices 05:04
being generated, it would allocate more room for 05:04
the vertices. And at least in some version of 05:05
the code, there was an input thread arbiter, but 05:05
I'd have to go through the code to say whether 05:05
that was in -- whether that remained in the 05:05
design. 05:05

But that input thread arbiter would 05:05
obviously make a decision, and I think it made a 05:05
decision based on pixel versus vertex, but I'd 05:05
have to -- I can't say that for sure without 05:05
finding the code associated with that input 05:05
arbiter. 05:05

Q. And you don't know whether that was 05:05
for the 400, the 500 or the 600 version, right? 05:05

MR. FAHRENKROG: Object to the form. 05:05

A. Well, I am talking specifically about 05:05
the 400 version. 05:05

Q. Okay. Was that arbiter programmable? 05:05

MR. FAHRENKROG: Object to the form. 05:06

MR. PAIGE: Vague and ambiguous. 05:06

A. The dynamic mechanism of the GPRs was 05:06
a programmable mechanism. The input arbiter, I 05:06
can't say for sure, because I have to find the 05:06

Page 203

actual code. But my belief is that it was not 05:06
programmable. 05:06
Q. And for the one that was 05:06
programmable, do you know when that was first 05:06
conceived of? 05:06
A. That was present in the design very 05:06
early on. I see mention of it before 2002. 05:06
Q. Did the R400 have a loop instruction? 05:06
A. Yes. 05:06
Q. When was that first conceived? 05:06
A. I'd have to go back and look. The 05:06
loop instruction was very heavily based on 05:06
Microsoft DX assembly language, which also have a 05:06
loop instruction. 05:07
So I don't think that AMD can even 05:07
claim invention credit for that in that we were 05:07
simply implementing a requirement from Microsoft. 05:07
Q. Okay. And do you know when that was 05:07
done? 05:07
A. I don't. I'd have to -- I think you 05:07
could look it up in terms of Microsoft when they 05:07
-- when they released DX specs. 05:07
Q. Did R400 have a call instruction? 05:07
A. It did. 05:07
Q. And when was that first conceived? 05:07

Page 204

A. Again, that was mostly based on the Microsoft DX assembly language. So I wouldn't claim any great invention on the part of AMD. 05:07 05:07 05:07

Q. Did the R400 have a return instruction? 05:07 05:07

A. Yes, it did. 05:07

Q. And when was that first -- 05:07

A. Again, it's simply based on the API requirements. There was some innovation in terms of how to implement these things. But the implementation, you know, followed the requirements, and the requirements were set by Microsoft. 05:08 05:08 05:08 05:08 05:08 05:08

Q. Do you have any sense of what time frame that work was done? 05:08 05:08

A. I don't as I'm sitting here. I could find out. 05:08 05:08

Q. Did R400 have a branch instruction? 05:08

A. R400, it did have a form of a branch instruction. It was not a typical branch instruction. 05:08 05:08 05:08

Q. And why was it atypical? 05:08

A. Well, because R400 had really two types of instructions. There were control flow instructions, and the control flow instructions 05:08 05:09 05:09

Page 205

kind of kicked off execution or ALU instructions. 05:09
And so the ALU instructions were like 05:09
add, multiply, subtract, and the control flow 05:09
instructions said, okay, execute, you know, eight 05:09
ALU instructions, and then the following eight 05:09
instructions would be the add, multiply, 05:09
subtract. 05:09
So the branch instructions were part 05:09
of those control flow instructions. So you 05:09
couldn't just put a -- you know, it wasn't put in 05:09
the middle there. And the branch instructions 05:09
only worked -- what do I want to say here? It's 05:09
a SIMD machine. So that means there's one 05:10
instruction flow for multiple pieces of data. 05:10
There was not a branch instruction 05:10
that you could do like a test on an individual 05:10
piece of data and say if this thing is a true 05:10
branch. The branch only functioned on all of the 05:10
pieces of data. 05:10
So that's what I mean when I say it 05:10
wasn't a fully general branch instruction. 05:10
Q. When was that done, do you know? 05:10
A. That was also done early if you look 05:10
at the early SQ documentation. There's 05:10
documentation about how the flow instructions 05:10

Page 206

work. 05:10

Q. SQ, what is that? 05:10

A. That's the sequencer block. Most of 05:10
this pile. 05:10

Q. Did the R400 rasterizer convert 05:10
vertex data output into pixel data? 05:11

A. It would take in triangles, which 05:11
were based on vertex data output. But vertices 05:11
don't produce pixels. It's the grouping of them 05:11
into 2D primitive that produces pixels. 05:11
Obviously the triangles are a function of the 05:11
output vertices. 05:11

Q. So is that a yes or a no or kinda 05:11
yes, kinda no? 05:11

A. It's a kinda yes. It's not a real 05:11
yes, because we don't -- the rasterizer did not 05:11
directly take in vertices. It took in -- 05:11
vertices first had to go through a fair amount of 05:11
processing prior to getting to the rasterizer. 05:11

Q. Now, did your unified -- I'm 05:12
forgetting the word, because it's getting late -- 05:12

A. Shader. 05:12

Q. The unified shader, could it also 05:12
process primitives? Let me ask the question 05:12
again so the record is clear. 05:12

Page 207

Did the unified shader process 05:12
primitives? 05:12

A. Not in the R400 or R500. It would 05:12
process either vertices or it would process 05:12
pixels. 05:12

Q. What about the 600? 05:12

A. The 600 did support processing 05:12
primitives. 05:12

Q. So could the R400 assign threads to 05:12
vertex, primitive and pixels? 05:12

A. No. 05:12

Q. It could only do it for vertex and 05:12
pixels? 05:13

A. Correct. 05:13

Q. But the R600 could do it for all 05:13
three? 05:13

A. Yes. 05:13

Q. Give me just one second. This was a 05:13
question I had. Could pixels and vertex data 05:13
proceed through threads out of order in the R400? 05:13

MR. PAIGE: Object to the form. 05:13

A. Could pixels and vertex data proceed 05:13
through threads out of order? Not within a 05:13
thread, because it was an SIMD machine. So all 05:14
of the pixels in a given thread were pretty much 05:14

Page 208

lockstep. 05:14

But pixels could be processed out of order with pixels, although they had to finish in order. Similarly, vertices could be processed out of order with other -- vertex threads could be processed out of order with other vertex threads, but they had to finish in order. 05:14 05:14 05:14 05:14 05:14 05:14

There was no ordering requirement on pixels versus vertices. 05:14 05:14

Q. Did you perform any of the simulations on the R400? 05:14 05:14

A. In terms of actually running them and seeing how they did? 05:14 05:14

Q. Yes. 05:14

A. I don't think so. 05:14

MR. CHATTERJEE: I think that's all I got. Thank you very much. 05:15 05:15

MR. FAHRENKROG: Okay. I just want to restate that we'll designate the transcript AMD confidential business information and the witness will read and sign the transcript. 05:15 05:15 05:15 05:15

(Continued on next page to include jurat)

Page 209

VIDEOGRAPHER: The time is 5:15. At this time we'll end today's deposition of Andrew Gruber. We'll stop recording and go off the record. 05:15 05:15 05:15 05:15

(Time Noted: 5:15 p.m.)

-------------------

ANDREW E. GRUBER

Subscribed and sworn to before me

this   day of   2015.

--------------------------------------

Page 210

C E R T I F I C A T E

Commonwealth of Massachusetts)
) ss:
County of Suffolk)

I, Michael D. O'Connor, a Notary Public within and for the Commonwealth of Massachusetts, do hereby certify:

That ANDREW E. GRUBER, the witness whose deposition is hereinbefore set forth, was duly sworn before me and that such deposition is a true record of the testimony given by such witness.

I certify that I am not related to any of the parties to this action by blood or marriage; and that I am in no way interested in the outcome of this matter.

IN WITNESS WHEREOF, I have hereunto set my hand this 18th day of March, 2015.

____

Michael D. O'Connor, RPR, CRR, CBC, CCP

Page 211

I N D E X

------------ EXHIBITS (Cont'd) ------------

ANDREW E. GRUBER	EXHIBIT	PAGE
Exhibit 7	Document entitled "R400 Architecture Proposal Version 0.1"	82
Exhibit 8	Document entitled "R400 Sequencer Specification SQ, Version 2.02"	90
Exhibit 9	Declaration under 37 C.F.R. 1.131	107
Exhibit 10	Declaration under 37 C.F.R. 1.131	108
Exhibit 11	U.S. Patent No. 8,400,459	108
Exhibit 12	U.S. Patent No. 8,760,454	108
Exhibit 13	Source Code	126
Exhibit 14	Document entitled "R400 Emulator Version 0.0 January 15, 2002"	148
Exhibit 15	U.S. Patent No. 6,897,871	168
Exhibit 16	U.S. Patent No. 7,239,322	168

Page 213

ERRATA SHEET FOR THE TRANSCRIPT OF:

Case Name: In the Matter of Certain Consumer Electronics Investigation and Display Devices with Graphics Processing and Graphics Processing Units Therein

Dep. Date: March 17, 2015

Deponent: Andrew E. Gruber

CORRECTIONS:

Pg.	Ln.	Now Reads	Should Read	Reason

____

Signature of Deponent

SUBSCRIBED AND SWORN BEFORE ME

THIS ___ DAY OF ______________, 2015

____

(Notary Public) MY COMMISSION EXPIRES: ________