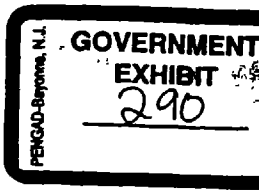

From: Bill Gates
Sent: Thursday, February 20, 1997 5:03 PM
To: Jim Allchin (Exchange); Paul Maritz
Subject: RE: AMD 3DX thoughts

I agree with all of your thoughts here.

If Intel has a real problem with us supporting this then they will have to stop supporting Java Multimedia the way they are. I would gladly give up supporting this if they would back off from their work on JAVA which is terrible for Intel. I have a call with Andy on this topic coming up on Monday.

-----Original Message-----
From: **Jim Allchin (Exchange)**
Sent: Thursday, February 20, 1997 2:00 PM
To: Paul Maritz; Bill Gates
Subject: FW: AMD 3DX thoughts

AMD was here a week or so ago. They described their progress on the K6. It was an excellent meeting and they are making great progress. I will make a copy of the slides and get them to both of you.

During the meeting we discussed some new instructions that AMD wants us to support called "AMD 3DX". The instructions (about 24 new opcodes) are very focused to make games fast. They use single precision floating point vectors similar in form to the MMX instructions. They are very fast due to the vector nature, saturating operations (no weird overflow/underflows), and no exceptions or conditions codes. There is zero overhead mixing 3DX and MMX. Below is Mark's opinion from the meeting. I concur with everything he says.

Intel will want to stomp on whatever instruction they use to hide the 24 new opcodes. They are going to take a currently illegal instruction and encode all 24 instructions within it. And even though AMD will pick an instruction away from the MMX codes giving Intel expansion room (assuming Intel does MMX 2), Intel still might decide to "accidentally" use whatever instruction area they pick to hammer them. This is AMD's biggest fear. It might take a conversation with Andy at some point if Intel does decide to do this. By the way, AMD says they will give everything to Intel without strings so that Intel could support the instructions also.

I want to support them. Do you have an opinion?

thanks,
jim

From: Mark Kenworthy
Sent: Thursday, February 13, 1997 2:19 PM
To: Carl Stork; Jim Allchin (Exchange); Moshe Dunie
Cc: Jay Torborg
Subject: AMD 3DX thoughts

As we look to the future of the non-server PC, it is clear that multimedia processing will dominate the processor usage, at least for the average end user. Technologies which provide better price/performance for PC multimedia are therefor a good thing that we should support. The 3DX instructions that AMD disclosed look like just the right thing to accelerate 3D graphics transform and lighting operations, although a detailed review by some of the Talisman graphics pipeline experts is in order.

I understand Carl's concern about incompatible instruction sets between competing CPUs, and believe we need to consider the possibility of this very carefully. If the average application is directly using these special instructions, I believe there is a significant risk to platform stability, since we have no control over the processor instructions, nor can we expect the average application writer to include processor specific checks. As such, I'm not thrilled to see AMD's plans to work with many companies to get these instructions into their applications, nor their request to include the instructions in MASM and VC.

Using these special multimedia instructions inside a driver, is also a bad thing. This will cause us (the OS) to lose control of the CPU for extended periods of time, especially with the future multimedia dominated applications. Also, the drivers they are supporting (3Dfx and Rendition), are outside of our DirectX and OpenGL APIs.

Using these special multimedia instructions inside our OS multimedia components, however, is a good thing. It will allow us to provide significantly better platform performance. We can regulate the CPU usage by these multimedia components, so we can ensure that the multimedia applications do not adversely affect the overall system operation or performance. DirectX 5 is partially structured to accept such processor specific optimizations in key routines. DirectX 6 will be more so. Our basic

plan is to ship companies like Intel and AMD source to these key inner loop routines, for them to optimize. We then surround these optimized code segments with processor specific checks, so even instruction collisions will not affect us. I would expect that the OpenGL folks are on a similar course. So, this seems like a good and safe path.

The architectural issues also concern me. We are rapidly moving to a point where we can compute a heck of a lot more than we can move over PCI or even AGP to some graphics card. AMD clearly understands this, and I'm sure Intel does as well. The only viable solutions I can see is either what AMD described at the end of their talk today, where graphics moves inside the northbridge chip, or moving graphics inside the processor itself (which is where we suspect Intel is headed). Making either of these strategies work well requires a memory bandwidth efficient graphics architecture, which uses at least some of the concepts embodied in Talisman. Without such an architecture, the system will starve on memory bandwidth.

This is probably how Intel plans to achieve dominance in the graphics space. Once graphics moves inside the processor or the northbridge, it will be next to impossible for the small graphics chip companies to compete, since it is all motherboard down and highly integrated. Cards hanging off a bus will not have the CPU<-> graphics bandwidth available to compete, so they lose, even if they have superior graphics technology. AMD could be a valuable ally in keeping competition open in the graphics arena, besides the processor arena.

So, what I'd like to see us do is:

1) Discourage application and driver use of these special instructions.
2) Aggressively support these special instructions in our OS multimedia components (same with Intel's).
3) Work closely with AMD to get alternative graphics technology into their northbridge chip (e.g., Talisman).

This strategy maximizes our system multimedia performance, binds more applications to our OS multimedia components, and keeps healthy competition in place for both the processor and graphics components.