# *3DNow!™ Technology in Microsoft DirectX 6.0*

## *Delivering Leading-Edge 3D Graphics Performance for the New Era of Realistic Computing*

*ADVANCED MICRO DEVICES, INC.*
*One AMD Place*
*Sunnyvale, CA 94088*

*Contact:*
*Dale Weisman*
*Public Relations*
*(512) 602-5820*

*Anne Camden*
*Public Relations*
*(512) 602-2120*

### *Overview*

This white paper explains how 3DNow!™ technology improves upon the mature x86 instruction set (now more than 20 years old) to provide PC users with a richer, 3D-enhanced computing experience. The benefits of 3DNow! technology can be achieved with minimal impact on existing and emerging x86 software thanks to the support of an industry standard: the DirectX application programming interface (API). This paper provides an overview of DirectX, a review of the graphics pipeline and how it is enhanced by 3DNow! technology, and a summary of how the 3DNow! instruction set is used in different software layers, as well as the impact of the new 3D instructions on each layer.

### *Introduction: What is DirectX Technology?*

The DirectX API is a group of technologies designed by Microsoft to make Windows® based computers an ideal platform for running and displaying applications rich in multimedia elements, such as full-color graphics, video, 3D animation, and surround sound. Built directly into the Microsoft® Windows family of operating systems, DirectX is an integral part of the Windows 95/98 and Windows NT® 5.0 operating systems, as well as Internet Explorer 4.0. DirectX components may also be automatically installed on a PC by advanced multimedia games and applications written for the Windows 95 operating system.

Microsoft's goal in developing DirectX was to provide developers with a common set of interfaces and functions that would accomplish two things. First, DirectX would allow developers to be confident that their multimedia applications would run on any Windows-based PC, regardless of what hardware is used, and at the same time ensure that their software products take full advantage of high-performance hardware capabilities to achieve the best possible performance. Second, DirectX would make life easier for developers by giving them tools that simplify the creation and playback of multimedia content, while also making it easier to integrate a wide range of multimedia elements.

DirectX provides developers with new opportunities for creativity and innovation by allowing them to focus on building unique features for their applications without having to worry about which display adapter, sound card, 3D graphics controller, or microprocessor is installed in the PC. Because DirectX was designed to support future innovations in software and hardware, developers and consumers can be confident that they will continue to get the best possible performance from their applications as technology advances.

A DirectX API is a layer of software that provides special functionality to a program independent of the underlying hardware. With this common interface, software developers do not

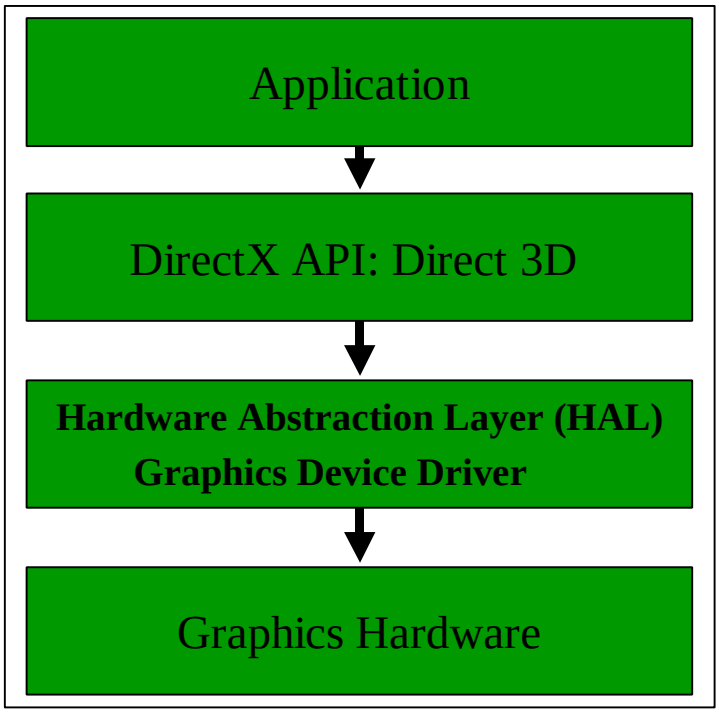
have to create unique functionality with their applications if that functionality exists in the API. Using code that already exists helps developers speed time to market. Less testing is also required since API code is tested not only by the creator but also by the many developers using the API.

***What is DirectX 6.0?***

DirectX 6.0 is the next generation of Microsoft's set of multimedia APIs. DirectX 6.0 will provide greater stability and reliability across all APIs and will augment current features of DirectX 5.0 with new leading-edge multimedia capabilities and enhancements, including faster performance and additional features in the Direct3D API, such as single-pass multitexturing, bump mapping, vertex buffers, stencil planes, and texture compression.

The DirectX 6.0 release, scheduled for July 1998, will be optimized for 3DNow! technology within Direct3D. This will enable enhanced 3D performance for applications that use the floating-point-intensive geometry functions that are performed within Direct3D. AMD has played a role in enabling faster 3D performance by working with Microsoft to speed up the graphics pipeline functions of transformation, lighting, and clipping.

The use of the DirectX API is illustrated below:

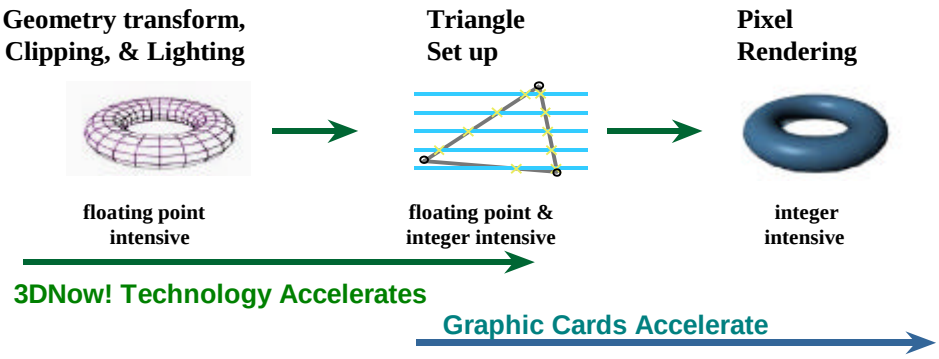



Microsoft released the developer's version of DirectX 6.0 in March 1998 and announced that a beta version would ship in May, to be followed with a final golden master release in July 1998. See **http://www.microsoft.com/directx/pavilion/general/whatisdx.htm** for more details about Microsoft's DirectX 6.0.

### *3DNow!™ Technology*

Because of the proliferation of much faster 3D graphics accelerator hardware in today's mainstream PC configurations, 3D graphics is becoming a standard feature in mainstream PCs. To take 3D graphics to the next level of performance, a major bottleneck in the graphics pipeline will have to be addressed. The cause of this bottleneck is the host processor's existing floating point capabilities. 3DNow! technology opens the bottleneck and hence accelerates multimedia and 3D graphic applications by increasing host processor floating point computation capabilities.

The front end of the 3D pipeline encompasses floating-point-intensive geometry. A game or 3D modeling package can also include mathematical modeling of physical objects, which places an added burden on the processor for floating point operations (detailed later in this paper). The greater the burden placed on the processor for floating point operations, the slower the frame rates, resulting in lower visual quality experienced by PC users. The chart below represents a simple flow of the graphics pipeline, showing where floating point instructions are used.




AMD has made tremendous strides in eliminating the floating point bottleneck. 3DNow! technology enables multiple single precision floating point instructions to be executed in a single clock cycle of the AMD-K6®-2 processor by using SIMD (Single Instruction Multiple Data) techniques. *(How this is accomplished is discussed later in this paper.)* Because the early stages of the graphics pipeline depend on host floating point processor bandwidth to perform physics and geometry, AMD believes that 3DNow! technology will result in more fluid, lifelike 3D objects and 3D simulations. Although this White Paper focuses on the enhancement of the 3D graphics pipeline through DirectX 6.0, the benefits of 3DNow! technology extend beyond 3D graphics. In fact, 3DNow! technology includes instructions that provide additional functionality for a broad range of multimedia applications over and above MMX™ instructions, including AC-3 audio and MPEG-2 video decode playback. *(See 3DNow! Technology White Paper for additional details.)*

### *Direct3D Graphics Rendering Pipeline*

Although the 3DNow! Technology White Paper presents a higher-level view of the 3D graphics pipeline, this white paper defines the steps that pertain to Direct3D—the key 3D performance technology within DirectX.

3D computer images are made up of multiple polygons (usually triangles) that are displayed on a computer screen or monitor. Some images can be very complex, containing tens of thousands of polygons. Today's typical gaming applications contain several thousand polygons per frame. This number is growing as hardware capabilities increase and is expected to top ten thousand polygons per frame by year-end for certain applications. To render these complex scenes, many millions of operations per second are required to generate triangles on the host CPU and then send them to the graphics card for conversion to the 2D display. The Direct3D graphics pipeline includes the following stages:

- **Transformation.** The computer conceptualizes an image based on mathematical models. The models of individual objects are said to be in *model space*. When the objects are placed in a 3D world, they are said to be in *world space*. Once an object is viewed from an *eye point* in this world, the viewer is then in *view space*. Finally, a perspective view is selected, and the image is transformed into *screen space* and then displayed. The transformation stage transforms the vertices from model space coordinates into screen coordinates. This process involves many floating point matrix multiplications among other operations. If the object is to be clipped, flags are set for each vertex indicating whether the vertex is inside or outside the viewing frustum (parts of objects or objects behind another are "clipped" from view). The 3DNow! instructions include special conditional operations that accelerate setting these flags. (Note: A vertex is a point in space. Vertices are connected with line segments to form polygons. Thus, a triangle requires three vertices.)
- **Lighting.** In this stage, a color (used for shading) is applied to each vertex based on the lighting model chosen. This floating-point-intensive operation includes reciprocal square roots and divides that are used to calculate complex lighting. The greater the number of lights, the more floating point intensive. For example, a room illuminated by several lights and a glowing fireplace, with light streaming in from another room, can create a very realistic (and computationally complex) setting.
- **Clipping.** This involves removing any triangles or pieces of triangles that are outside the viewing area (called the *viewing frustum*). Triangles that straddle the viewport must be clipped so that they are fully contained within the viewport. This involves interpolating

coordinates, colors, and texture coordinates and potentially generating new triangles and vertices. This is also a floating-point-intensive operation involving several floating point divides for each triangle that straddles the viewport, among other operations.

- **Rasterization (or Rendering)**. This is the process of taking transformed and lit vertices and rendering them onto the display. Some of the more common features applied here are Gouraud shading, texture mapping, and alpha blending. This is also the portion of the pipeline implemented by 3D graphics chips. The earliest task in rasterization, known as "setup," involves some floating point code. Most of this year's new high-performance graphics accelerator products (Rendition, Voodoo 2, Nvidia Riva 128, etc.) include dedicated hardware to improve setup performance. However, lower cost graphic chips still need some floating point assistance from the host processor (which is implemented by each independent hardware vendor or IHV in the software display driver, not in DirectX). If there is no 3D accelerator on the display adapter, this portion of the pipeline must be performed in software, which can also benefit from 3DNow! technology.

### *Optimization within Direct3D*

Direct3D has the ability to modify (via commands) much of the data that is passed through it. There are many commands (or API calls), as well as a few data types, within Direct3D. More details on these commands and data types can be found in Microsoft's DirectX developer web site. Suffice it to say, there are many commands and three data types that impact the most frequently used floating point instructions. The three data types are:

- **Vertex (Model Vertex):** This structure defines an untransformed and unlit vertex (model coordinates). Used by Direct3D's Retained Mode, this vertex type is often required when the programmer wants Direct3D to perform the transformation and the lighting prior to rasterization. This vertex type will gain the maximum benefit from 3DNow! technology since it requires the most floating point operations within Direct3D. Clipping can be done on model vertices for the DrawPrim and DrawIndexedPrim calls, as well.
- **Lit Vertex:** This structure defines an untransformed vertex (model coordinates with color). This vertex type is used when Direct3D will not be used to provide the lighting of these vertices. Direct3D transforms these vertices prior to rasterization. They are useful for pre-lit models and scenes with static light sources. This provides performance improvement by 3DNow! technology for the transformations that use floating point matrix multiply. Clipping can also be performed on lit vertices for the DrawPrim and DrawIndexedPrim calls.

- **Transformed and Lit Vertex:** This structure defines a transformed and lit vertex (screen coordinates with color). The most common form of vertex used by games in 1997, this vertex type is used when the transform and lighting is not being done through Direct3D. If this vertex format is used, the game will most likely not be accelerated within the Direct3D API. In the future, many games will move away from this type of vertex and towards the model vertex.

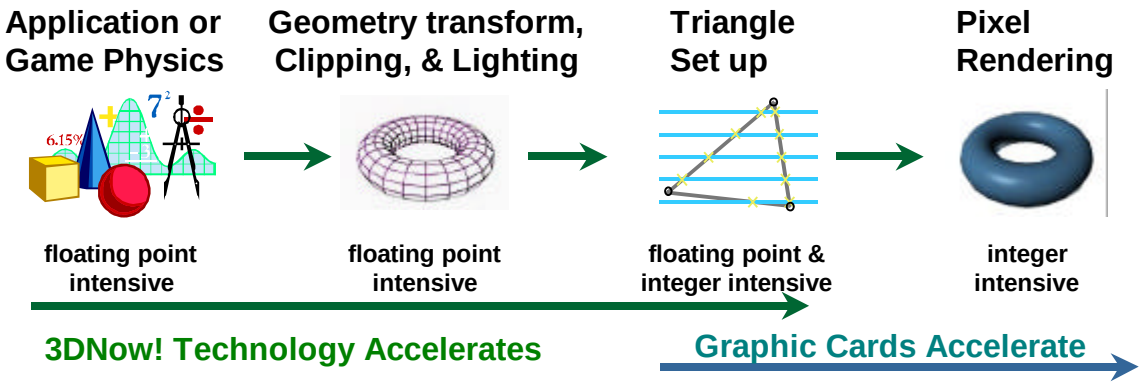
### *Immediate Mode Versus Retained Mode*

Programmers working in DirectX will be interested to know that there are actually two DirectX 3D APIs. The first, called "Direct3D Immediate Mode," is part of what is known as the "DirectX Foundation." This low-level API is most commonly used. It has no high-level knowledge of the applications database or processing and is presented with rendering lists contained in arrays. This is the API that is being directly accelerated by 3DNow! technology.

The second API is "Direct3D Retained Mode" and is part of what is called "DirectX Media." Retained Mode received its name because it "retains" knowledge of the applications database and helps manage it for the programmer. The key point to remember is: *Direct3D Retained Mode is built on top of Direct3D Immediate Mode. All optimizations made to Immediate Mode will accelerate all Retained Mode applications.*

### *Physics: Feeding the Graphics Pipeline*

The graphics-rendering pipeline has been discussed in detail. It is also important to discuss what feeds the pipeline: the modeling of real-world objects through physics. Physical modeling refers to the ability of application programmers to define their world with mathematics. For games, this means that the objects are defined with mathematics that would represent an object's real-life physical characteristics, including mass, force, buoyancy, friction, gravity, density, speed, elasticity, etc. This capability, combined with the interactions among physical characteristics, is very compelling. For example, to generate the seemingly simple act of skipping a rock across a pond, a complex series of static and dynamics interactions must take place to create the scene. Until now, only workstation-class programs have used complex math to create complex 3D worlds because there was insufficient processing power in the PC environment. 3DNow! technology will enable applications to take these new "physical" properties into mainstream PCs.




A prime example of physical interaction is the MaruBench benchmark, which uses optimized code based on the physics of gravity. MaruBench shows off the interaction of some 500 objects floating in outer space. The benchmark is based on the equation: Gravity = Mass (1) x Mass (2) divided by distance between the masses squared. This is a pair-wise equation, meaning the computer must generate the relationship between each set of pairs in this universe (~130,000 pairs). This generates numerous matrix multiples that are greatly accelerated by 3DNow! technology. Since a large number of floating point calculations must take place, a processor with the 3DNow! instruction set can feed a graphics pipeline much faster than a processor without the new instruction set. The amount of motion seen (measured in frames per second) in MaruBench can be up to 2.5 times faster on a processor with 3DNow! instructions than one without it. MaruBench is available on the AMD Developers Web Site.

### *Balancing the Graphics Pipeline*

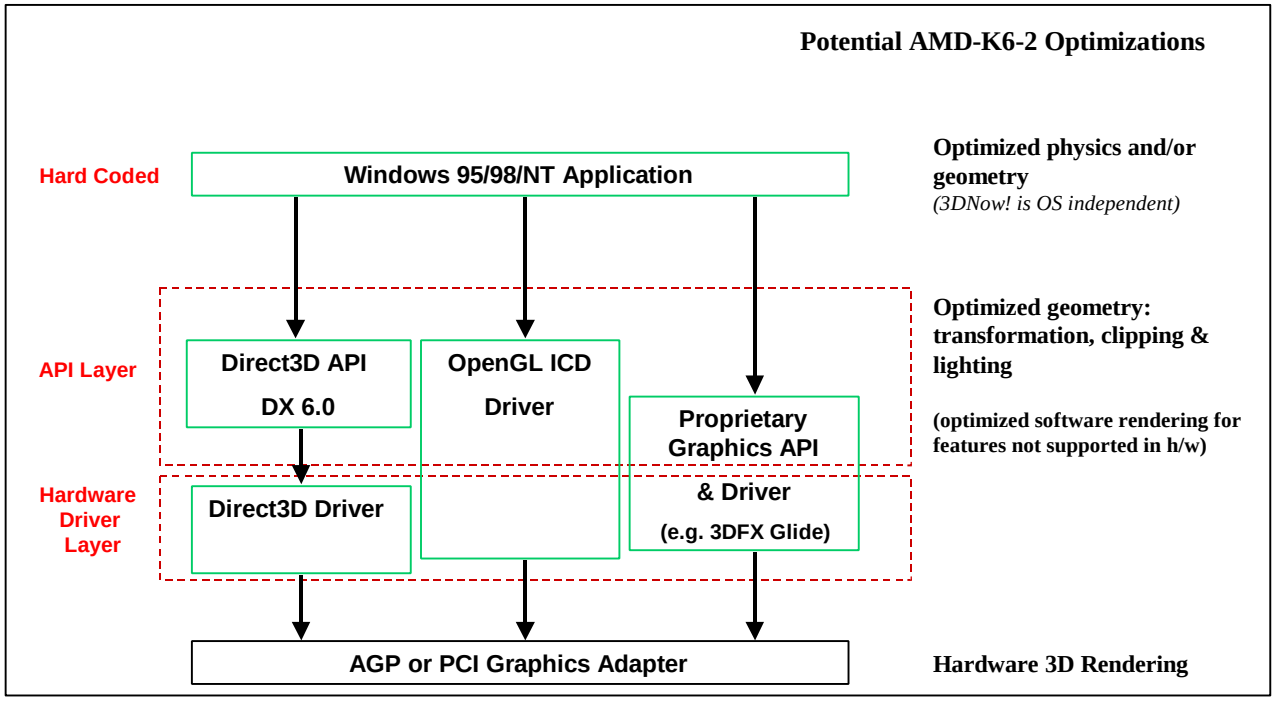
Overall, the 3D graphics pipeline is a very complex function as demonstrated by the fact that 3D graphics capabilities are just now migrating to mainstream personal computing. Granted that host CPU floating point capability is a bottleneck in the pipeline, many other factors affect the 3D performance ultimately experienced by the PC user. Many graphics chip vendors provide white papers that discuss the various factors that impact 3D performance, such as double buffering, texture mapping, mip mapping, Z-Buffer, antialiasing, etc. The structure of the data types within an application is also very important because it will often determine if the data can take advantage of SIMD-style instructions.

Without discussing all the variables, it is important to note that a balanced pipeline is necessary for the PC user to see improved results. Because the graphics pipeline is not based entirely on floating point operations, a four-to-one improvement will never be fully realized. Any increase in physics or geometry performance may not be realized by the end user if the pipeline is

not balanced (e.g., the bottleneck might have moved to another place). An example of this is a fast processor with a poor graphics chips set. Generally speaking "the mileage will vary" depending on the system configuration. However, Marumark certainly demonstrates what kind of 3D performance is possible and that the performance increase can be more than an incremental (e.g., 400 MHz vs 300 MHz clock speeds).

### *3DNow! Technology Performance Enhancements*

The benefits of 3DNow! technology must be realized through software modifications. To understand these benefits, it is helpful to look at the software architecture.




First, 3DNow! technology is operating system independent; that is, no modifications are required within Windows 9x or Windows NT to support this technology. Second, floating point operations are used throughout the various layers of software. There are three basic areas for software optimization involving 3DNow! technology that will yield performance improvements:

- Native 3DNow! optimization within an application or game
- Optimizations within an API, such as DirectX 6.0 or OpenGL, in which the application calls the API to perform the floating-point-intensive 3D geometry routines
- Graphic driver level optimizations.

Each of these methods alone yields performance improvements. Optimizing all three components will yield the maximum improvement possible. AMD and our ISV/IHV partners will

have optimized two common layers of software—the API software layer (DirectX and OpenGL), as well as the graphic card driver layer.

A natively optimized software title is one in which an ISV has hand-coded or used AMD's SDK tools to optimize their code. Native optimization can be done either as part of the initial ground-up design of a new application, or it can be done near the completion of an application. Although most performance optimization usually occurs near the end of development, it is important to consider the code structure as early as possible in the development process. Early adoption is encouraged because it enables new features and data structures that will lead to optimal performance.

AMD offers programming tools that assist developers in the optimization process. These tools allow developers to replace inefficient floating point (x87) instructions with SIMD-style instructions that support multiple instructions per clock cycle. Converting x87 instructions to 3DNow! instructions involves two 32-bit single precision floating point values grouped into a single 64-bit vector. Two vector computations can run through the dual fully pipelined register units within the AMD-K6-2 processor. This allows for potential peak performance of four 32-bit single precision floating point values being processed in one clock cycle, or a 4:1 processing advantage over standard x87 code implementation (used in Pentium® II). As a result, a 300-MHz AMD-K6-2 processor has a potential peak rating of 1.2 Gigaflops (single precision), while a competing processor using standard x87 would be rated at 300 Megaflops (0.3 Gigaflops). This significant increase in processing power will bring 3D into mainstream personal computing as ISVs begin to design applications with 1.2 Gigaflops of performance in mind.

*Note: AMD and many computer industry trade publications will publish actual 3DNow! technology performance data and benchmarks after the formal introduction of the AMD-K6-2 processor on May 28, 1998.*

### *AMD-K6®-2 Processor-Optimized Software in the Marketplace*

AMD is working to optimize all software layers to improve application performance. The initial optimized applications entering the market will be hard-coded games, applications, and drivers. Although many of these hard-coded games and applications can and do call the DirectX API (generally DirectX 5.0), they will not rely on AMD optimizations within the API. (Note: Any version of DirectX is backwards compatible, enabling DirectX 5.0 applications to run unmodified on DirectX 6.0.) The software optimizations will occur within the ISV's code, and the performance benefits will be derived from floating point routines performed within the game or application.

Growing numbers of applications are using geometry calls (transformation, lighting, and clipping) within the API. A current key technology that falls into this category is VRML plug-ins for Web browsers. This includes both Internet Explorer (based on DirectX technology) and Netscape Navigator (based on OpenGL technology). Microsoft has indicated that DirectX 6.0, plus Internet Explorer VRML, will improve performance. Benchmark applications, such as 3D WinBench™ 97 and 98, also use DirectX for much of the geometry. In addition, a few games use the Direct3D graphics pipeline through both Immediate Mode and Retained Mode.

AMD is also working with key graphics chips vendors to optimize their drivers to gain every bit of performance. Although AMD has found that in most cases there is some floating point computation within the graphics drivers, significant performance improvements can be obtained by optimizing the code for the AMD-K6-2 processor pipeline. AMD optimization on these and other routines will increase performance for all applications. Again, the resulting performance will vary depending upon the driver.

*Note: Lists of 3DNow! optimized applications, technologies, and drivers are available on the AMD Developers Web Site*.

### *Summary*

Ultimately, DirectX 6.0 technology will be a key tool for realizing the performance benefits of the AMD-K6-2 processor. Not only will the API provide performance boosts for AMD customers, partners, and end users as the AMD-K6-2 processor enters the marketplace, but it will also enable performance gains for many new ISV titles without requiring the title to be rewritten for these new technologies. DirectX 6.0 is a key facet of the AMD strategy because most software developers that use DirectX today will use the performance-minded floating point operations in DirectX 6.0 for enhanced performance and faster time to market.

Over time, greater numbers of software titles are expected to rely on Direct3D's geometry pipeline, and increasingly ISVs will want to use performance optimizations within Direct3D to enhance the performance of their applications. ISVs understand that these optimizations can be written once and used often, and that they will not be burdened with having to write additional performance code within their applications and then testing that code. In short, ISVs will have confidence that their applications will run on all the underlying hardware. Thus, as ISVs update, develop, and release software titles, they will use DirectX 6.0 (and future versions) to improve performance, decrease engineering costs, and shorten time to market. In addition to leveraging DirectX 6.0, performance from 3DNow! technology can also be gained by optimizing floating point routines within the application itself and within the graphics driver.

In conclusion, 3DNow! technology optimizations will provide an easy and compelling way for ISVs to gain significant 3D performance enhancements within the scope of work already planned and for title development that is planned for the future.

### *AMD Overview*

AMD is a global supplier of integrated circuits for the personal and networked computer and communications markets. AMD produces processors, flash memories, programmable logic devices, and products for communications and networking applications. The world's second-leading supplier of Windows compatible PC processors, AMD has shipped more than 100 million x86 microprocessors, including 55 million Windows compatible processors in the last five years.

Founded in 1969 and based in Sunnyvale, California, AMD has sales and marketing offices worldwide and manufacturing facilities in Sunnyvale; Austin, Texas; Bangkok, Thailand; Penang, Malaysia; Singapore; and Aizu-Wakamatsu, Japan. AMD had revenues of $2.4 billion in 1997. (NYSE: AMD).

### Cautionary Statement

This White Paper contains forward-looking statements that involve risks and uncertainties that could cause actual results to differ materially, including the market acceptance and successful production ramp of AMD processors, the impact of competitive products and pricing, the timely development of wafer fabrication process technologies, the effect of changing economic conditions, and such risks and uncertainties detailed from time to time in the company's SEC reports.