Professional 3D Graphics

THE MEMORY SYSTEM MAKES THE DIFFERENCE

Jan Bjernfalk
Director, Product Marketing
REALimage™ Technology

v. 1.1, February 11, 1999

EVANS & SUTHERLAND
Desktop Graphics
P.O. Box 58700
Salt Lake City, Utah 84158-0700
http://www.es.com

PROFESSIONAL 3D GRAPHICS:
THE MEMORY SYSTEM MAKES THE DIFFERENCE

INTRODUCTION

While high-performance 3D graphics is becoming pervasive on the PC platform, it has become clear that the needs of the professional workstation user are not necessarily the same as those of the general consumer. To address this fact, there have emerged graphics solutions aimed specifically at the professional user, which are differentiated from their consumer-market counterparts in the areas of feature set, performance, and architecture. The Evans & Sutherland REALimage™ 3D graphics controller, as implemented in AccelGALAXY, is an excellent example of such technology aimed specifically at the professional user.

In this paper, we will analyze the some of the most important characteristics that distinguish a professional 3D graphics card from those in the consumer space. In the end, we will see that the key to many of these advantages is the use of a more sophisticated and robust memory system. Consumer 3D graphics cards typically employ a single, one-port memory system (sometimes referred to as a "Graphics Unified Memory", or "GUM", which is then shared across all of the various 3D functions (texture mapping, frame buffer, video buffer). Professional 3D cards, like AccelGALAXY, have memory systems with independent ports and memory devices dedicated to each of the key 3D functions.

"NO COMPROMISE" VIDEO RESOLUTION

One of the important differences between the consumer and the professional user is the type of tradeoff they are willing to make on video resolution versus 3D performance. While consumers may have very high demands for 3D performance, the driving applications behind this are almost always games.

Games have the characteristic that they require high 3D performance, but are designed to run at relatively low screen resolutions, like 640x480 or 800x600. Because of this, games fit well into the compromise inherent in a Graphics Unified Memory. At lower resolutions, there is more memory bandwidth available to 3D-specific graphics functions. This is good news, because at higher screen resolutions (e.g. 1600x1200) it is easy to lose as much as one-half of the total memory bandwidth or more to the video refresh process (see Table 3).

There are also times when the consumer will want relatively high resolution, but this is likely to be when doing primarily 2D tasks, such as interacting with the Windows™ desktop, editing a speadsheet, reading e-mail, etc. In this situation, it is perfectly acceptable for video refresh to occupy the majority of the graphics memory's bandwidth, since no 3D processing is required.

The professional user, on the other hand, will not generally accept this sort of compromise. Whether the application is mechanical design, digital content creation, geoscience, or any other 3D-intensive task, the professional user will not trade resolution for graphics performance. This type of user spends many hours of the day looking at 3D images on the screen, and wants the highest resolution possible without giving up any 3D performance. Furthermore, professional 3D applications (like Pro/ENGINEER, 3D Studio Max, Maya, etc) are *designed* to be run at high resolution.

When a professional user runs one of these applications on a consumer-style 3D graphics board with a unified memory, graphics memory becomes the bottleneck and the overall performance is far less than what the specifications of the controller might indicate. At this point, advanced on-chip architectural features such as susperscalar processing and multiple texture pipelines don't help much, because the single, external memory bank has become overloaded, leaving much of the on-chip logic idle.

REALimage addresses this challenge by employing a dual-port frame buffer memory architecture. The video refresh task has a dedicated memory port, independent of the one used for 3D graphics update processing. This has the benefit that as video resolution is increased, there is hardly any degradation of 3D graphics rendering performance. Thus, the tradeoff and the compromise are eliminated.

"NO COMPROMISE" RENDERING FEATURES

A similar problem to the video tradeoff described above is the tradeoff relative to enabling various 3D graphics features, such as Z buffering, stencil, transparency, anti-aliasing, etc. Enabling each of these features increases the amount of data that must be read from and written to the frame buffer memory (see Table 1).

Again, with a consumer-style 3D graphics card, there is only one memory system to handle all tasks. Therefore, as the professional 3D application enables these features, there will be a corresponding decrease in overall pixel fill-rate performance, because the memory system once again is the bottleneck. For the casual user this may not be a problem. But serious professional users are not comfortable thinking of Z buffering and anti-aliasing as "options". These are features that professionals need turned on most of the time.

The REALimage solution to this problem lies in the use of a unique frame buffer memory technology (Mitsubishi 3DRAM[tm]). 3DRAM has on-chip all of the registers, cache, and ALUs required to handle the complete read-modify-write memory loop involved in 3D pixel-arithmetic operations, including Z-buffer comparisons, stencil, transparency, anti-aliasing, and fog. The result is that all of these operations run at full speed; the user is not penalized for turning them on.

ANTI-ALIASED LINES

Many professional engineering applications, especially those for mechanical design and drafting, represent data as vector lines drawn on the screen. When designing complex parts, such as automobiles and airplanes, the amount of vector data presented on-screen can become huge -- often surpassing 1 million lines.

Because the applications need to represent these lines as being quite thin, the notion of smoothing out the jagged edges from pixelization, or anti-aliasing, becomes very important. It becomes even more important when there are a million lines displayed on the screen, as it could otherwise be impossible to visually resolve the location of the individual lines. For the mechanical designer who spends most the day staring at a computer screen running Pro/ENGINEER, it becomes an absolute necessity to have high-quality anti-aliased lines.

To this end, today's professional 3D graphics cards typically support an anti-aliased vector primitive, whereas most consumer-level cards do not carry this added circuitry. On the quantitative side, this vector primitive makes most mechanical and other vector-oriented applications run much faster. Some benchmarks, such as the vector-oriented CDRS-03 portion of Viewperf (which draws anti-aliased lines), run considerably faster on hardware that supports the anti-aliased line primitive. On the qualitative side, anti-aliased vector images are much more pleasing to the eye, and make it easier to see and understand the intended visual content.

REALimage technology is widely recognized as producing the cleanest, sharpest anti-aliased lines in the industry. Indeed, for REALimage to render a 10-pixel long, 1-pixel wide, anti-aliased line a read-modify-write of approximately 24 pixels is required. Without specific hardware acceleration, achieving this quality of line is prohibitive in terms of performance.

PER-PIXEL HARDWARE ATTRIBUTES

Per-Window A/B Buffer Selection

Most 3D graphics accelerators available today support the concept of double-buffering, whereby smooth motion is achieved by using two decoupled sets of color buffers (one for update, one for video refresh). Double-buffering is

easy to implement so long as it is applied to a full-screen visual; all that is required to accomplish a buffer swap is to exchange the pointers to the start of the video and update buffers. This works fine for games, since the 3D "window" is almost always full-screen.

Unfortunately, the way most people use many professional 3D applications is to have multiple 3D windows open at the same time, in a fully windowed environment. These various 3D windows may be multiple views of the same object, or even views of different objects, but more importantly these 3D windows will tend to update (and hence buffer swap) asynchronously.

There are at least two ways for a 3D controller to solve the problem of multiple asynchronously updating 3D windows. A robust solution is to have the video refresh hardware decide, on a pixel-by-pixel (or window-by-window) basis, which of two buffers to draw video data from. This implies that the video process must draw simultaneously from both buffers, along with additional data in each pixel (such as a Window-ID) to help decide which buffer to display from. This is the approach used by REALimage-based controllers, and it takes advantage of the fact that there is a dedicated memory port for video data. Thus, there is sufficient bandwidth available to draw both buffers for all pixels, and make the actual selection on a per-pixel basis before sending video data to the RAMDACs.

In systems where memory bandwidth is more severely limited, such as with a single Graphics Unified Memory architecture (or perhaps any system where frame buffer memory is single-ported), it is not practical to employ the above solution. To conserve bandwidth, video must necessarily be restricted to fetch only one color buffer per pixel and no additional attributes. In this situation, the typical solution to the asynchronous 3D window problem is as follows: (1) render the new window frame into a buffer in off-screen memory, then (2) copy the contents of that buffer into the window region in on-screen memory (the video buffer). While this approach conserves video bandwidth, it has the disadvantage that it adds the time required to do the final copy (BLT) to the task of rendering each frame.

Most 3D performance benchmarks in use today only focus on the full-screen window case. While these benchmarks may be quite useful, they ignore the effects of multiple asynchronous windows, which is a substantial factor in "real world" applications.

Window Clipping

Related to the above scenario of multiple 3D windows on the desktop at once, it is easy to imagine a situation where, because of windows overlapping each other, one or more of the 3D windows has a visible on-screen shape that is non-rectangular.

In a system using the copy-offscreen-buffer approach to 3D rendering, the process is further complicated because now software must clip the incoming

buffer to the non-rectangular on-screen shape. This is not terribly complex, but does have the impact of further slowing the 3D update process.

REALimage technology deals with this problem by having special bits for each pixel, which are designated as the Window-ID. Thus, the 3D rendering pipeline can proceed as if it were rendering into a normal rectangular window. During the read-modify-write operations for each new pixel (which happens completely inside the 3DRAM), an additional comparison is made to ensure that any pixel to which is attempted a write, belongs to the same Window-ID that the 3D pipeline is rendering into. With this technique, and the extra hardware support provided by the 3DRAM, the process of clipping a 3D window to any arbitrary shape is handled without any software overhead or performance degradation.

Overlay Planes

Pop-up menus, dialog boxes, and other temporary display fields are common in any interactive application. Some professional 3D accelerators, including those based on REALimage, provide additional dedicated "overlay" bit planes. These overlay planes can be rendered into, like any other color planes, but have the advantage that doing so does not disturb any data in the main color buffers. They also have the attribute that any pixel can be "transparent" in its overlay plane so one can "see through" to the data in the main color buffers below.

These attributes make overlay planes attractive for use with displaying temporary interactive fields, as well as for many other uses. Many professional 3D applications, such as Softimage, Maya, and DesignWave, are designed to take advantage of graphics cards that have this overlay capability.

Implicit in the overlay plane functionality is the requirement that more data must be fetched by the video refresh process; the overlay part of the pixel must first be tested to see if it is transparent. If so, then the color in the main buffer must be fetched and passed along to video. Because of this added burden on video bandwidth, overlay functionality is typically not included on consumer-level 3D graphics cards.

MEMORY ARCHITECTURE

As we have described, consumer-level 3D graphics devices are typified by their use of a single, one-port bank of DRAM devices. This single memory, at a minimum, is shared across the purposes of : (1) Update frame buffer, (2) video display buffer, and (3) texture buffer.

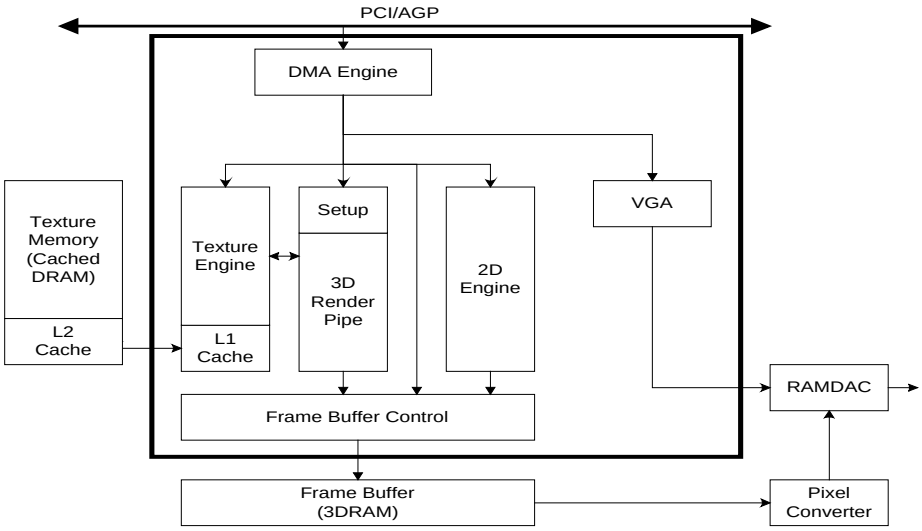
Because memory bandwidth so quickly becomes the limiting factor in these architectures, the most common differentiator (and also predictor of performance) of consumer graphics cards is the bit-width of the shared memory bank. If the rendering controller is well-designed (with today's

competitive market, most successful ones are), it is extremely likely that the memory system, not the controller, will limit the performance of typical professional workloads. Micro-architectural features such as multiple pixel pipelines, dual texture caches, and superscalar processing, while perhaps academically interesting, only help to ensure that the memory remains the real bottleneck.

Today, state-of-the-art memory on a consumer 3D graphics card is considered to be a 128-bit wide SDRAM array, typically with a clock rate of 100 MHz[1]. This would yield a theoretical peak bandwidth of 1600 Mbytes/second. However, given the diverse demands on the memory, and the frequent arbitration and bus reversals, it is difficult to imagine that even the best controllers could achieve a sustained utilization of better than 75% in most cases. This observation would lead us to downrate the effective available bandwith to 1200 Mbytes/second.

The memory architecture on a professional-grade 3D card can be quite different in order to satisfy the greater workload presented. For example, a REALimage-based 3D card has a dedicated memory for textures, and a dual –port frame buffer memory which gives independent access for both update rendering and video refresh.

REALimage Architecture:



[1] 125 MHz SDRAMs are available in late 1998, and 200 MHz parts possibly before the end of 1999.

Because a REALimage graphics card has a dedicated memory port for each of the three primary 3D memory functions, a user can take full advantage of any of the texture, render, and video capabilities without compromising the others.

Also, the memory used on REALimage 3D cards is *application-specific.* These memories have architectural advantages that make them more efficient than ordinary commodity memory. For example, the texture memory is implemented with CDRAM, a variant of SDRAM that includes an on-chip cache memory. This cache is used by REALimage as the Level-2 cache in the texture memory hierarchy. Having a 2nd level cache off-chip from the REALimage controller allows more efficient use and scheduling of the DRAM array.

We have already described some of the attributes of the 3DRAM. From an architectural standpoint, one of the biggest benefits of 3DRAM is that, while rendering 3D graphics, pixels need only flow in one direction from the controller to the memory. Because of the unidirectional flow, and because the texture and video functions have independent memory ports which do not interfere with this flow of pixels, the REALimage technology is capable of sustaining 3D fill rates that are over 90% of the theoretical maximum derived from clock rate only. And most importantly, because all pixel data processing is done on-board the 3DRAM™ memory, there is no slowdown as features like 32-bit color, Z-buffer, Stencil, Transparency, Fog, Anti-aliasing, and Window-clipping are turned on.

MEMORY BANDWIDTH ANALYSIS

Table 1 - Frame Buffer Update Bandwidth Requirements by Mode

Pixel Mode	Bytes Read [1]	Bytes Written	Mbyte/s @ 100 Mpix/s [2]
RGBA:16	0	2	200
RGBA:32	0	4	400
RGBA:16 +Z/s:32 [3]	4	6	1000
RGBA:32 + Z/s:32	4	8	1200
RGBA:32 + Z/s:32 + Blend [4]	8	8	1600
RGBA:32 + Z/s:32 + Blend + WID [5]	9	8	1700

Notes:
(1) This column represents the number of bytes that must be read from frame buffer memory for each pixel rendered.

(2) Total frame-buffer bandwidth, in megabytes/sec, required to achieve a fill rate of 100 M pixels/sec.

(3) 16-bit R,G,B, Alpha, with 32-bit Depth (Z) plus Stencil

(4) "Blend" means that new color is mixed with existing color in frame buffer. Required for anti-aliasing, transparency, and fog.

(5) "WID" = Window I.D. Required when doing window-clipping.

Table 2 - Texture Memory Bandwidth Requirements

Texture Mode	Bytes Read [1]	Mbyte/s @ 100 Mpix/s
16-bit Bi-linear	4	400
16-bit Tri-linear	6	600
32-bit Bi-linear	8	800
32-bit Tri-linear	12	1200

Notes: (1) REALimage™ implementation requires, on average, 2 (bilinear case) or 3 (trilinear case) texels read from texture memory for each pixel rendered. Theoretically, for extremely large polygons, an ideal implementation could approach 1 or 1.5, respectively. However, this is simply not realistic for polygons less than 100 pixels in size.

Table 3 - Video Memory Bandwidth Requirements

Video Mode	Bytes Read	Mbyte/s @ 1280x1024 / 85 Hz	Mbyte/s @ 1600x1200 / 85 Hz
24-bit RGB	3	332	490
RGB:24 + Overlay:8	4	442	653
RGB:24 (both buffers) + Overlay:8 + WID:8 [1]	8	884	1306

Notes: (1) Required to support independent, asynchronous per-window double-buffer control.

Table 4 - Total Bandwidth Available, 128-bit Graphics-Unified-Memory

SDRAM Clock	100 MHz	125 MHz
Peak Bandwidth	1600 MB/s	2000 MB/s
Available Bandwidth (1)	1200 MB/s	1500 MB/s

Notes: (1) Assumes that rendering controller can achieve an average memory scheduling efficiency of 75%, which is fairly aggressive.

INTERPRETTING THE DATA

One of the many things illustrated by the data in the above tables is that as high-quality rendering features are turned on, such as 32-bit color, Z-buffering, anti-aliasing, and high-resolution video, the demand on memory bandwidth climbs dramatically.

To determine how a particular application would perform on a 128-bit 3D graphics card relative to a REALimage-class card, first choose a quality/functionality level from each of Tables 1, 2, and 3. Then add the bandwidths required from each table. If the sum is less than the available bandwidth as shown in Table 4, then one can expect that the 128-bit card would perform proportionately slower in that scenario.

Example 1: 16-bit RGB, No Z, No Blend, 16-bit bilinear, 1280x1024 @ 85 Hz

Frame Buffer BW Required:	200 (@ 100 Mpix/s)
Texture Memory BW:	400
Video BW Required:	332
Total:	932

This is a very simple scenario. For a 128-bit card with 100 MHz SDRAM, there would be 1200 MB/s available, which is more than enough to handle this scenario with either a consumer-level card or a REALimage card.

Example 2: 32-bit RGB, Z, Blend, 16-bit bilinear, Overlay, 1600x1200 @ 85 Hz

Frame Buffer BW Required:	1600 (@ 100 Mpix/s)
Texture Memory BW:	400
Video BW Required:	653
Total:	2653

This is a fairly typical scenario for a mid-range professional application. For a 128-bit card with 100 MHz SDRAM, there would be 1200 MB/s available, which is only 45% of the bandwidth required to match the REALimage-based card.

Example 3: 32-bit RGB, Z, Blend, 32-bit Trilinear, WindowClip, Overlay, Mult. Indep. 3D Window Swapping, 1600x1200 @ 85 Hz

Frame Buffer BW Required:	1700 (@ 100 Mpix/s)
Texture Memory BW:	1200
Video BW Required:	1306
Total:	4206

This is a high-end scenario, typical of a professional power-user. For a 128-bit card with 100 MHz SDRAM, there would be 1200 MB/s available, which is only 29% of the bandwidth required to match the performance of the REALimage-based card.

NOTE: All of the above examples consider only the case of a single monitor video output. If the application requires driving dual monitors, which is possible with the E&S AccelGALAXY, the video bandwidth requirements are *double* those shown above.

SUMMARY

We have seen that when faced with workloads typical of the professional 3D graphics user, even a well-designed consumer-market 3D graphics card can perform at ***less than half*** of the rates provided by a professional-grade 3D card like AccelGALAXY. This is true, despite what can be very impressive, but also misleading, hardware specs for the controller itself. Peak internal rates of a 3D controller become irrelevant when the workload overwhelms the memory system.