SPARCstation 10ZX and SPARCstation ZX Graphics Technology

A White Paper

Sun

Contents

Introduction 1

The design and development of graphics accelerators is a complex and resource-intensive task. To build the most powerful graphics processors possible, designers track and use the most advanced component and manufacturing technologies available to them. And as a result, some graphics workstations deliver near real-time simulation and modeling — necessarily at a significant price.

High performance and low price are the dual, significant factors in the design and production of a graphics device. Therefore, designers should focus their energy on understanding how to maximize performance while reducing component and manufacturing costs.

The SPARCstation™ 10 ZX and SPARCstation ZX graphics workstations are the first generation of an innovative graphics architecture that reflects the characteristics of today's advanced processor and systems technologies. The SPARCstation 10 ZX and SPARCstation ZX graphics workstations feature a novel architecture for advanced graphics rendering and display at extremely compelling price and performance points.

Note – Throughout this document, the SPARCstation 10 ZX and SPARCstation ZX workstations will be referred to simply as the SPARCstation ZX workstation.

Reducing Bottlenecks in Graphics Performance

Few users can afford the cost of the hardware necessary to provide industry-leading performance. Instead, users are compelled to determine the most effective performance and functionality for their particular application.

With conventional graphics processor design, the history is really one of bottlenecks. Any point in the graphics pipeline could be a bottleneck. And advances in graphics technology simply move the bottleneck from one place to another.

3-D geometry graphics performance is directly dependent on the available floating point processing capability since the calculation of transformations and lighting is compute-intensive. However, current very large scale integration (VLSI) technology has effectively driven the price of floating point performance below $2 per megaflop (MFLOP), and the graphics bottleneck has now moved to the drawing bandwidth of video RAM.

The key to building the SPARCstation ZX system lay in understanding what portion of the graphics pipeline is scalable (and therefore less prone to bottlenecks), and what is not. If the VRAM provides a natural barrier beyond which drawing processors do not need to scale, then why not construct a graphics system with low-cost drawing processors, but replicated based on optimal VRAM size?

The same reasoning applies to the manipulation and processing of the geometry data. Traditional graphics processors are designed to handle a large range of data types and geometry primitives. The resulting wide variation finds its way to the floating point processor, where diversity leads to complexity, cost, and performance limits. The simpler the floating point processor, the faster it can perform its task. And with the vector nature of 3-D graphics operations, multiple floating point units make sense. The SPARCstation ZX design embodies this philosophy with four, independent custom floating point units, each delivering a equivalent of 150 MFLOPS for a total of 600 million floating point operations dedicated to graphics computation.

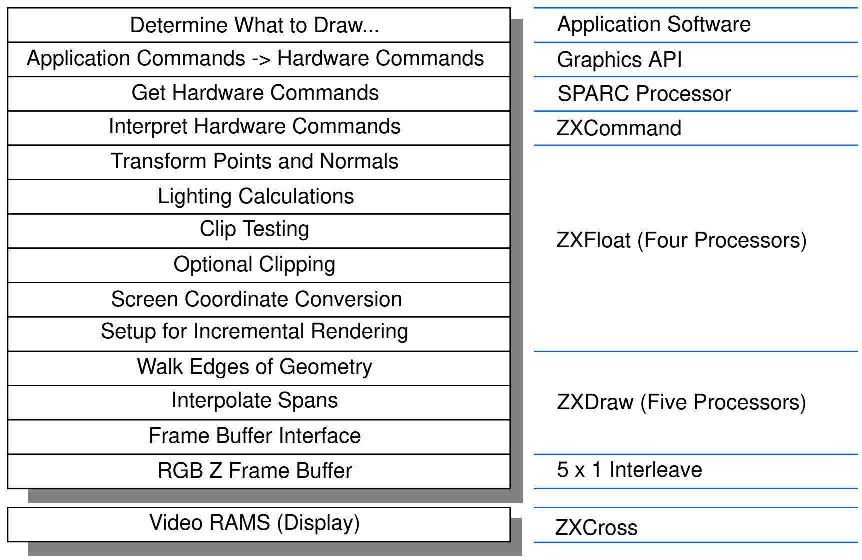

Figure 1.1 Graphics Rendering Stages in the SPARCstation ZX System

Developers of the SPARCstation ZX workstation began by working backward from the limitation of memory bandwidth imposed by VRAM. Design choices were based upon the idea that breaking down complex geometry at the beginning state of graphics rendering could more effectively define and control the processing imposed on individual floating point units. Providing scalable processing to rendering operations such as edge walking and span interpolation resulted in greater performance. Finally, the timing constraints associated with the cross-connection of multiple drawing processors accessing VRAM provided a gating factor for defining the optimum number of drawing processors.

Another, very important benefit emerged as a result of this architecture — the resulting graphics rendering would optimize as well for *individual primitives* as it would for multiple ones. Given how typical graphics programs are written, this has the effect of substantial increases in performance for most applications.

The SPARCstation ZX workstation was developed with a philosophy of low cost and internal scalability, based on the assumption that the scalable factors in graphics processing are the number of VRAM planes and the amount of floating point resources. Careful analysis of graphics applications and accelerators and how they process and render data has led to the design and development of the SPARCstation ZX system.

This white paper explores the architecture, capabilities, and resulting benefits of the SPARCstation ZX graphics accelerator.

Features of the SPARCstation ZX Accelerator 2

The SPARCstation ZX mid-range 3-D graphics accelerator targets users with requirements in mechanical computer aided design (MCAD), animation and simulation, molecular modeling, scientific visualization, mapping and other 3-D market segments.

The SPARCstation ZX accelerator is optimized to accelerate the geometry-oriented XGL™ foundation graphics library. The XGL library includes 2-D and 3-D graphics primitive functions that support a wide variety of graphics-based applications. The XGL library is the foundation interface upon which most industry standard and third-party graphics application program interfaces (APIs) are built, including OpenGL©, PHIGS, HOOPS©, PEXlib, and others.

Key SPARCstation ZX Accelerator Features

This section briefly describes some key SPARCstation ZX features. These features are described in additional detail in Chapter 3, *SPARCstation ZX Accelerator Architecture*.

Bit Planes

The SPARCstation ZX accelerator includes a total of 96 bit planes. The planes are organized into groups providing support for 24-bit double-buffered rendering, with 24-bit Z-buffering. Ten planes are provided for window ID support, eight are used for overlays, and six are used as fast clear planes to accelerate double-buffering.

24-Bit Image Planes

The SPARCstation ZX accelerator includes a 24-bit color frame buffer that enables the simultaneous display of a continuous range of colors from a palette of 16.7 Million. The image planes store data in one of two color models: 24-bit RGB true color, or 8-bit indexed color. The color model is selected on a per pixel basis by the window ID planes.

24-Bit Double-Buffering

Double-buffering is used by the SPARCstation ZX accelerator to display dynamic, flicker-free animation. An image is rendered into off-screen (back) memory and then switched to display (front) memory for viewing. The front and back buffers are continually swapped as new images are rendered, providing dynamic, smooth updates. This isolates the viewer from the rendering process, and avoids the display of partially rendered images.

The accelerator uses two sets of 24-bit planes for the highest possible image fidelity when double-buffering. Applications can use double-buffered image planes as if they were 24-bits deep. Each 24-plane image group is broken down into 8 planes (bits) each for red, green, and blue primary colors. Each double-buffered window is allocated a unique window ID to control the display of the buffers for that window.

24-Bit Z-Buffer

The SPARCstation ZX accelerator includes support for a full 24 bits of Z-buffer. The Z-buffer provides hardware assistance for hidden surface removal, which facilitates faster dynamic rendering of 3-D objects. A 24-bit deep Z-buffer provides approximately 16.7 million discrete steps in Z, or 256 times the precision of a 16-bit Z-buffer.

Hardware Window IDs

The SPARCstation ZX accelerator includes ten independent window ID planes for multiple, independently double-buffered windows. Window ID planes are used in both the window system and geometry pipeline, for definition of the color mode (for example, 8-bit, 24-bit, and so on), double-buffering, overlay plane control, selection of colormaps, and fast clear plane usage. Up to 80 windows may be independently buffered without mutual interaction. In previous designs, double-buffering between windows required synchronization. With the SPARCstation ZX accelerator, double-buffering can occur whenever the processor is ready; it does not require direct synchronization. Each double buffered window has a unique identifier stored in a hardware table that can be toggled to switch the front and back buffers.

Overlay Planes and Hardware Cursor

The SPARCstation ZX accelerator includes eight overlay planes for simultaneous 8-bit application rendering. Unlike many graphics processors, the cursor is managed by independent logic in a custom application-specific integrated circuit (ASIC) — that is, a hardware cursor, rather than in the overlay planes. This helps enable fast tracking of the mouse and incurs no overhead of the system to update the cursor position on the screen. The overlay planes further enable pop-up menus and 8-bit applications to appear without damaging underlying images (and their windows) and eliminates overhead of redisplay of the image planes.

Fast Screen Clear Planes

To achieve the effect of smooth motion, many image frames per second need to be generated quickly. Smooth motion is important when 3-D objects need to be viewed dynamically, such as in 3-D modeling, simulation, and animation. This effect requires that the image memory, including the Z-buffer be cleared before each redisplay, representing a fixed per-frame overhead. Hence the screen clear speed is critical to high interactivity. Using special features of the video RAM, the SPARCstation ZX accelerator uses six independent fast clear planes to clear the entire image and Z-buffer in less than 200 microseconds. The six fast clear planes can be set up to provide plane clearing of any combination of planes (e.g. 3 double-buffered windows, or 6 single-buffered ones).

Dual Rendering Paths

The SPARCstation ZX accelerator uses two separate rendering paths for optimized management of 2-D integer and 2-D/3-D floating point geometry data. A direct path to the frame buffer enables the host CPU to bypass floating point transformations, thereby providing low latency access to the frame buffer. Additional logic in the control and rendering ASICs also provides two independent data paths — one for complex geometry (such as lit triangles), and another for low-level raster operations.

The dual rendering paths ensure that non-geometry oriented rendering is unencumbered by the geometry logic of the graphics processor, while 2-D/3-D floating point data is managed and processed in an efficient manner.

The independent rendering paths enable multiple processes to access the SPARCstation ZX accelerator simultaneously. In the case of a multiprocessor configuration, even greater throughput can be realized with each processor updating graphics on alternate bus cycles.

Transformation Pipeline

The SPARCstation ZX architecture incorporates a highly efficient graphics pipeline through the implementation of custom ASICs consisting of a command ASIC, multiple floating point ASICs, and multiple rendering ASICs. The command ASIC optimizes the dispatch of data for transformation through a unidirectional bus that ensures effective load balancing is performed on the floating point ASICs.

Each floating point ASIC delivers the equivalent of 150 MFLOPS of floating point performance, with an aggregate of 600 MFLOPS.

The command ASIC converts a variety of numeric data formats into a small number of isolated drawing primitives. The consolidation of data formats simplifies the floating point transform processor logic, leading to computational optimization and resulting in higher performance.

Additional Features

Single Primitive Optimization

The SPARCstation ZX architecture provides for efficient management and rendering of both chained and individual primitives. Applications frequently do not render data as chained primitives, but rather render each primitive one at a time. The result usually leads to lower than the optimized performance claimed by graphics product specifications.

In the SPARCstation ZX accelerator, the command ASIC recognizes both chained and unchained primitives. The ASIC deconstructs complex geometry and passes it on to the floating point ASICs one primitive at a time. The custom floating point ASICs are optimized for very specific graphics operations, defined by their individually dedicated static RAM (SRAM), and are connected in parallel for maximum performance.

Multiple Rendering Modes

The frame buffer's 24-bit image planes can be configured (on a per-window basis) as 8-bit or 24-bit images. All 8-bit and 24-bit applications run simultaneously on the screen. SunPHIGS™ graphics library, XGL graphics library, XIL™ imaging library, Direct Xlib™ library, and Xlib applications can be written in any one of these modes.

This broad selection of display capabilities provides compatibility with existing applications and utilities. Common applications and tools can be run in 8-bit mode and still take advantage of the window system acceleration. Applications needing nondestructive windows or pop-up menus can use the 8-bit mode. Windows running these applications and tools can be intermingled with accelerated 24-bit windows without any restrictions.

The SPARCstation ZX accelerator also provides support for 1-bit stencil mode rendered in either the overlay or image planes. This feature is ideal for applications that draw non-destructive images including text and window pop-up menus.

Fast Scrolling Text and Fast Raster Operations

Fast window system performance is another key to the SPARCstation ZX accelerator. For example, the dual rendering paths enable 24-bit rasters to be moved or BITBLT'd from one location on the screen to another extremely quickly. In addition, there is no performance degradation when windows are occluded.

Dynamic Shading

Shading is used for rendering realistic 3-D models. The SPARCstation ZX accelerator provides a two types of shading: Flat or faceted and smooth or Gouraud. Flat or faceted shading assigns a polygon one color. With flat shading, side-by-side polygons may have different shades of color thereby creating the facet effect.

Smooth or Gouraud shading blends color across each polygon. This is a more compute-intensive and complex task, but creates much more realistic results. The SPARCstation ZX accelerator speeds the rendering and dynamics of both faceted and Gouraud shading by performing the lighting and shading calculations in the floating point and rendering logic. By optimizing for shading, applications can render both Gouraud and flat-shaded objects with no difference in performance.

APIs based on the XGL graphics library have full access to all shading methods supported on the SPARCstation ZX accelerator.

32 Hardware-Supported Light Sources

The SPARCstation ZX accelerator implements 32 independent light sources directly in hardware. Light source types include ambient, positional, directional, and spot. APIs and applications layered above the XGL graphics library have access to all light sources. Additional light sources (beyond the 32 in firmware) can be used, but revert to software for computation.

Anti-Aliased Vectors

The SPARCstation ZX accelerator supports line (vector) anti-aliasing through the XGL graphics library. While additional computation is required for anti-aliasing, the calculations have minimum performance impact, rendering anti-aliased vectors at rates approaching that of aliased vectors. Because most complex geometric models have a large percentage of short (less that 10 pixel) vectors, performance for anti-aliasing very nearly approximates the performance for aliased vectors.

NURBS Curve and Surface Support

Non-uniform rational B-splines (NURBS) curves and surfaces are concise yet powerful and general representations of a wide variety of simple and complex geometries. Trimmed NURBS are increasingly more prominent in both informal and formal industry standards for geometric modeling and computer graphics. NURBS include powerful mathematical properties that provide ease of manipulation and processing.

From a high-level mathematical description, a graphics system tessellates curved geometry into lines (in the case of curves) or triangles (in the case of surfaces) whose sizes depend on the spatial resolution of the curve or surface to the eye, and on the complexity of the geometry.

XGL NURBS curves are general enough for users to generate uniform curves such as Bezier curves, or uniform B-Splines. The order of the curves can be up to twenty-five (most CAD applications use three or four). XGL attributes offer complete control of the precision associated with the curve rendering, such as dynamic tessellation to optimize the trade-off between number of points computed and rendering time. The XGL Library also offers a constant subdivision attribute to control the distance between two consecutive knots.

Dynamic Tessellation with NURBS

The SPARCstation ZX accelerator and the XGL library provide very powerful NURBS facilities, including the ability to dynamically tessellate NURBS surfaces based upon application (and user) control. Dynamic tessellation in XGL is mapped directly to the SPARCstation ZX firmware, where internal algorithms provide optimum performance for NURBS calculation and tessellation computation.

Transparency

The XGL library supports two transparency methods on the SPARCstation ZX accelerator for 3-D surfaces: screen-door transparency and alpha-blending (two-pass) transparency. Transparency can be invoked through the XGL 3D Context attributes assigned to the primitive to be rendered.

Screen door transparency enables a device-dependent screen door to render the object using a mesh through which some of the primitives pixels are rendered. If the blended method is used, blending equations determine how the surface pixel values are blended with the background color or existing pixel values. Transparency may be achieved in a number of ways, dependent upon available display hardware. For example, transparency may be directly supported on some devices or may use stipple patterns on others.

The SPARCstation ZX accelerator microcode optimizes both screen-door and alpha-blending transparency methods to deliver the same high performance for either.

Depth Cueing

Depth cueing assists observers with perception of depth by fading the primitive's colors by an amount that corresponds with the distance of the primitive from the observer. Typically the amount of fading increases with the distance.

Applications that use depth cueing often combine this effect with motion through animation. Two common applications that use depth cueing are molecular modeling and flight simulation. In the case of molecular modeling, the background color is usually black. Depth perception can be enhanced by fading the more distant parts of the molecule to black, making them appear dimmer. Oscillation of the molecule about a vertical axis enhances the depth perception of the object.

The XGL library supports two methods for depth-cueing: *linear* and *scaled*. In linear depth-cueing, the colors of the primitives are the same as the non-depth cued colors at the front of the Virtual Device Coordinate (VDC) viewport. At the back of the viewport, the colors completely change to the specified depth cue color (RGB) or the color at the bottom of the color ramp (for indexed color). Between these two boundaries, all colors modulate to a linear combination of the original color and the depth cue color.

Scaled depth-cueing is a superset of linear depth cueing. Scaled depth-cueing is linear between two predefined reference planes, that can be placed at arbitrary depths in virtual device coordinates. The scale factors for the front and back planes must fall within values between 0.0 and 1.0. The scale factors outside the linear region are clamped at the values of the reference planes closest to them.

The SPARCstation ZX accelerator performs both linear and scales depth cueing through microcode. In both cases, depth cueing is performed on a per-pixel basis with no performance penalty versus non-depth cueing rendering. Depth cueing is fully compliant with ANSI/ISO standards.

Hardware Picking

Picking is the ability to select individual geometry (vectors, polygons, markers, text) from a large model and receive more information describing that selected geometry. Hardware picking makes the response time almost instantaneous. Both the SunPHIGS graphics library and XGL graphics library support hardware picking with the locator or pick device.

The SPARCstation ZX accelerator supports hardware picking through the ZXDraw ASIC. The ZXDraw processor includes logic for bounding box definition. ZXDraw stores the current bounding box and pick ID (associated with the geometry) in its internal registers an, on pick selection, sends the pick ID back to the application (through the graphics API).

The SPARCstation ZX graphics pipeline includes sufficient latency to ensure that rendered geometry primitives and pick IDs will always be in the correct order. Hence pick selections are guaranteed to always be correct.

Multiple Resolution Support

The ZXCross ASIC performs output pixel multiplexing, pseudo-color look-up, and gamma correction. A variety of monitor resolutions are supported, ranging from interlaced NTSC at 640x480 pixels to 1280x1024 non-interlaced.

The support for multiple resolution frees the user to select the most appropriate solution meeting their requirements, based on application requirements, image fidelity, and price. Table 2.1 lists the resolutions possible on the SPARCstation ZX system.

Resolution	Frequency	Interlaced/Non-Interlaced
1280 x 1024	76 Hz	Non-interlaced
1280 x 1024	67 Hz	Non-interlaced
1152 x 900	76 Hz	Non-interlaced
1152 x 900	66 Hz	Non-interlaced
1024 x 768	76 Hz	Non-interlaced
1024 x 768	60 Hz	Non-interlaced
640 x 480	60 Hz	NTSC, Interlaced
770 x 575	50 Hz	PAL, Interlaced
960 x 680	108 Hz Stereo	Non-Interlaced, 54 Hz Field Rate per Eye
960 x 680	112 Hz Stereo	Non-Interlaced, 56 Hz Field Rate per Eye

Table 0.1 SPARCstation ZX Resolution and Display Parameters

SPARCstation ZX Accelerator Architecture 3

The SPARCstation ZX accelerator is an innovative desktop graphics subsystem designed to heavily leverage recent advances in systems integration including all custom circuitry, and optimized logic for maximum rendering performance within the constraints of VRAM access.

The SPARCstation ZX accelerator uses 11 custom designed ASIC chips. The use of custom ASICs substantially reduced the requirement for discrete chips and glue logic, thus providing addition size and power reduction through higher integration.

The architecture also relies on component-level and system optimization, ranging from multiple custom floating point logic chips (ZXFloat) specifically designed for transformations, to pipeline control logic (ZXCommand) that reduces the complexity of primitive processing. Additional performance is realized by multiple specialized rendering ASICs (ZXDraw) that perform five-way interleaving of display memory.

To facilitate maximum performance for non-graphics applications, the SPARCstation ZX accelerator employs dual rendering paths so that the window system can access the frame buffer directly, independent of the 3-D graphics pipeline.

3

System Overview

The SPARCstation ZX accelerator can fetch vertex data directly from the host system memory. This data enables the accelerator to render complex graphics at high speed into its frame buffer with little or no assistance from the host.

The system can be either a bus master or slave, depending on its current operation. For example, a graphics application may prepare data and pass it to the graphics library (API) being used. The graphics API in turn passes the data to the XGL library, which efficiently formats the data and instructs the processor to fetch it from memory. The fetch operation initiated by the graphics processor efficiently off-loads this processing (fetch and put) from the CPU, resulting in better overall system performance. This capability is known as direct memory access (DMA), and is an integral part of the SPARCstation ZX architecture.

The SPARCstation ZX accelerator is optimized to deliver maximum rendering performance for both SBus and MBus-based SPARC systems. The components of the system, further illustrated in Figures 3.1 and 3.2, are:

- *Host CPU*. The CPU of the SBus workstation in which the SPARCstation ZX graphics accelerator is installed. The SPARCstation ZX accelerator relies on the SBus for host-to-accelerator communication and hence must be installed in SBus-based workstations only, including desktop, deskside, and data center systems. In workstations that support multiprocessing, there may be a second CPU, which can be employed for faster application execution such as 3-D display-list oriented programs.
- *System Memory.* The system memory is an MBus slave. The MBus is the processor to memory bus, which is also used as a processor-to-processor bus on multiprocessing systems. As with other systems, the system memory is used by the CPU for storage of instructions and data. It is also used for temporary storage of vertex data that is to be processed by the SPARCstation ZX graphics accelerator.
- *I/O Devices.* These can be either SBus slaves or masters, such as disk controllers and communications interfaces.
- *SPARCstation ZX Graphics Accelerator.* The accelerator is a special purpose computing pipeline optimized for 3-D graphics. It is both an SBus slave and a direct virtual memory access (DVMA) master.

- *SBus.* The SBus is a high performance, byte-addressed, 32- or 64-bit data-path, synchronous bus. It provides a 32- or 64-bit virtual address that allows for DVMA by all bus masters. This greatly simplifies software memory management and provides a 28-bit physical address for slaves. The SPARCstation ZX can generate interrupts on the SBus to inform the host of various graphics processor conditions.

SPARCstation 10ZX - MBus-based Systems

The SPARCstation 10ZX includes the MBus, a processor bus optimized for multiprocessor-capable systems. The following components are part of the SPARCstation 10ZX and other MBus-based systems.

- *MBus to SBus Interface (MSI).* The MSI provides a mechanism for communication between the host CPU, its memory, and SBus-based devices such as the SPARCstation ZX workstation
- *MBus.* The host CPU uses the MBus as a high speed bus for communication between the CPU, system memory, and other CPUs (in a multiprocessor configuration). The MBus is a 32-bit, byte-addressed bus, typically running at 40 MHz. The host CPU communicates with the SPARCstation ZX accelerator across the MBus through the MSI and via processor memory.

SPARCstation ZX - Non-MBus Systems

The SPARCstation ZX workstation, in this case the non-Mbus based platform, uses only the SBus for graphics processor to memory/CPU communication. The SBus is a 32-bit, byte-addressable peripheral bus running at 25 MHz.

For accelerated 3-D graphics, the host CPU builds lists of vertex data in system memory. It then tells the SPARCstation ZX graphics accelerator (slave) that the data is ready. The accelerator then becomes a DVMA master, fetches the vertex data directly from system memory, and processes it. While this is taking place, the host CPU is free to perform other tasks.

The host CPU (as a master) can also perform unaccelerated graphics operations (text or 2-D graphics) by writing directly into the frame buffer (slave).

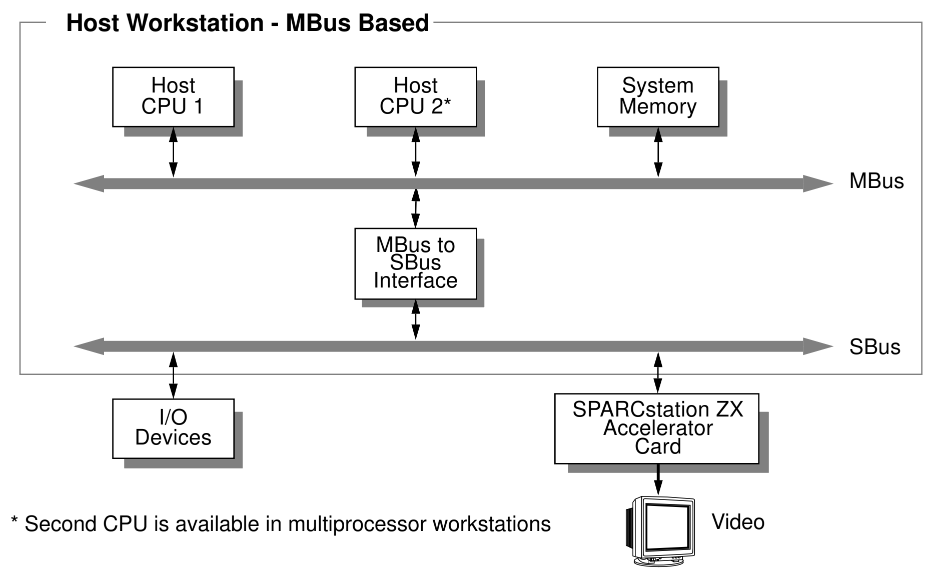

Figure 3.1 High-level Overview: SPARCstation Host and SPARCstation 10ZX Graphics

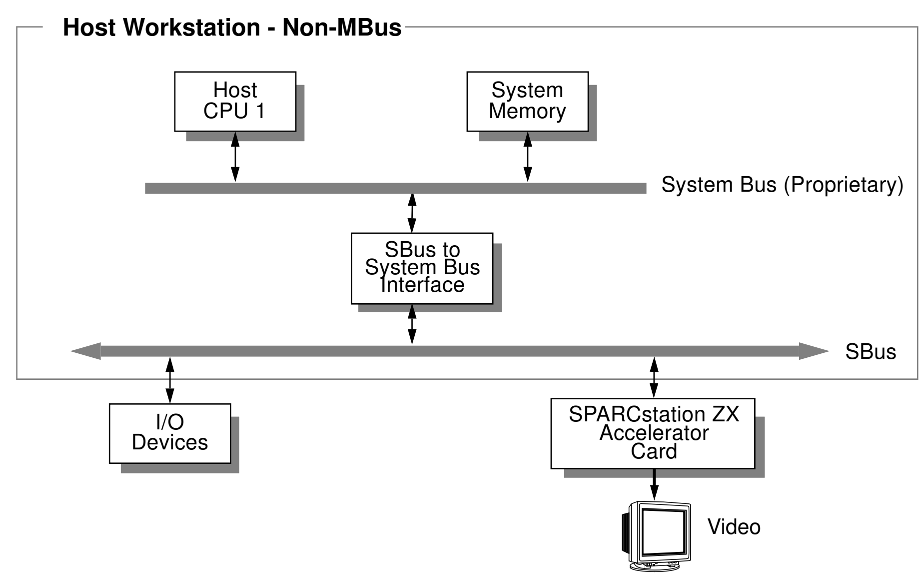

Figure 3.2 High-level Overview: SPARCstation Host and Non-MBus SPARCstation ZX Graphics

SPARCstation ZX Graphics Processor Architecture

Figure 3.3 provides a block diagram of the SPARCstation ZX graphics accelerator. Each of the individual blocks are described below.

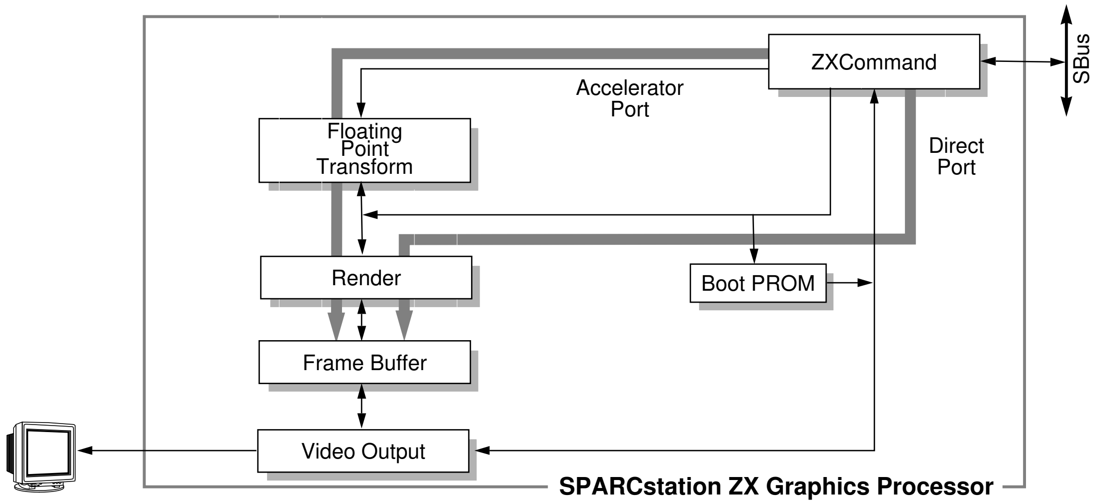

Figure 3.3 High-level Overview: SPARCstation ZX Graphics Processor

Command

The ZXCommand ASIC converts chained vertex data in a variety of numeric formats into a number of isolated drawing primitives. The incoming vertex data contains position data, vertex and facet normal and color information, edge highlight flags, and chaining information. Normal and color information are represented as 32-bit floating point numbers. This data is converted into isolated triangle and line commands (all with floating point arguments) before being sent to the floating point chips.

Boot PROM

The Boot PROM is a 64 KB programmable read-only memory that initializes the SPARCstation ZX accelerator when the workstation is powered on. The SPARCstation ZX Boot PROM is completely compliant with the SPARC Open Boot PROM Specification. The programmability of the boot PROM enables simplified upgrading on the PROM contents without the necessity of replacing the device.

Floating Point Transform

The Floating Point Transform section consists of four custom ZXFloat ASIC each with their own static RAM (SRAM) components. Each ZXFloat is a specialized floating point processor which is optimized for low-level graphics operations that scales linearly in performance. The SRAM memory components contain microcode used by the ZXFloat chips to perform their specialized operations.

The ZXCommand ASIC can access the state of each ZXFloat chip to determine which are busy and which are available. The ZXCommand chip sends accelerator port primitives to an available ZXFloat, which processes the primitive and signals the ZXCommand chip when it has completed processing. The ZXCommand chip then enables the results to be sent from the ZXFloat chip to the Render section.

The ZXFloat chips convert individual dot, vector, and triangle data from 3-D model coordinates (MC) to 3-D world coordinates (WC) to 3-D device coordinates (DC) represented by frame buffer pixel locations and Z-buffer values. ZXFloat also performs lighting calculations that result in three floating-point values (red, green and blue) for each vertex. These values are a function of the color and surface properties of the triangle, the position of the lights, and the angle that the light hits the surface. ZXFloat also performs window clipping on the rendered image.

Render

The Render section consists of five ZXDraw ASICs. Each ZXDraw controls one-fifth of the frame buffer memory, which is organized into a five-by-one interleave factor. After receiving their input, each ZXDraw operates independently, providing up to five times rendering speed improvement for many operations, and rendering up to the maximum VRAM memory bandwidth.

The ZXDraw ASIC has two internal data paths for rendering: an accelerated path and a direct path. The accelerated geometry path contains drawing hardware for 3-D images, while the direct path allows access to the frame buffer for window system and applications that do not need to pass through the transformation pipeline.

For 3-D accelerated operations, ZXDraw converts the dot, vector, or triangle parameters into pixel operations for the frame buffer. For the vertices, ZXDraw receives various values, such as the slopes of the sides of a triangle, and the corresponding increments for the red, green, and blue color values. ZXDraw fills in all the intermediate pixels outlined by the vertices.

ZXDraw is also responsible for raster operations, vector anti-aliasing, alpha transparency, and Z-buffer algorithms.

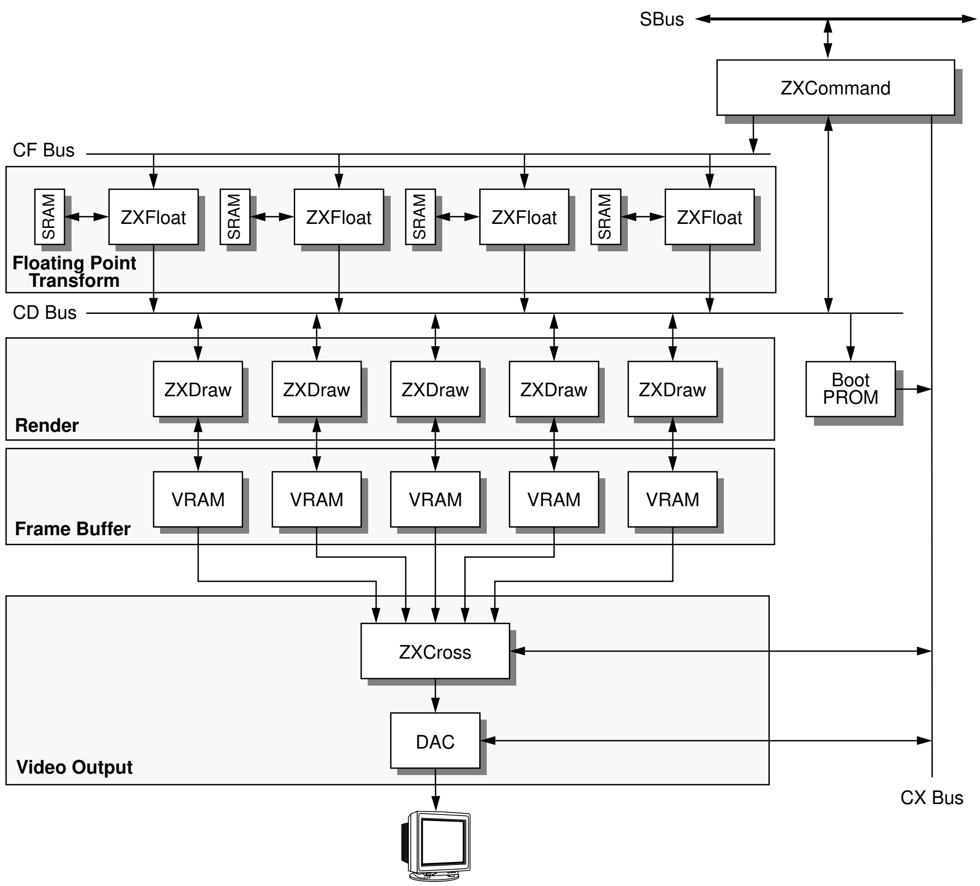

Figure 3.4 SPARCstation ZX Graphics Processor Block Diagram

Frame Buffer

The SPARCstation ZX frame buffer consists of 1280 by 1024 pixels by 96 planes. The memory planes are organized as follows:

- 48 image planes, which can be organized as:
  - Two RGB true color buffers of 24 planes each
  - Six indexed color buffers of eight planes each
- Eight overlay planes
- 24 Z-buffer planes
- Six P window ID planes
- Four Q window ID planes
- Six fast clear planes that can be organized into any combination (for example, three sets of double-buffered windows, six single-buffered window, and so on)

The SPARCstation ZX frame buffer supports multiple resolutions, as illustrated in Figure 3.5. It can also operate in a nonstandard quad-buffered stereo mode of 960 by 680 pixels. A stereo output is provided to switch left and right shutters on stereo goggles or a monitor face plate.

The SPARCstation ZX accelerator has no conventional cursor plane in the frame buffer, because this capability is provided by the ZXCross ASIC. Figure 3.7 on page 27 illustrates the organization of the SPARCstation ZX memory planes.

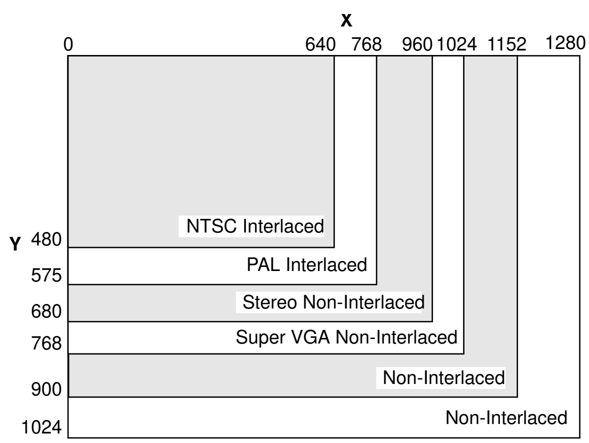

Figure 3.5 SPARCstation ZX Frame Buffer Resolutions

Image Planes

The 48 image planes in the frame buffer hold a color value for each pixel to be displayed. The image planes can store image data using one of two color models:

- 24-bit RGB true color
- 8-bit indexed color

The color model is selected on a per-pixel basis by the window ID planes.

In the 24-bit true color model, two separate 24-bit planes are dedicated to displaying the 16.7 million color gamut. Each 24-bit plane is divided evenly into eight bit planes each for the three primary colors: red, green and blue. The two frame buffers are referred to as buffer A and buffer B. As the system displays the image in buffer A, the next stage in the image's transition is being drawn in buffer B. When the contents of buffer B are complete, the display is switched from buffer A to buffer B. Figure 3.6 illustrates the relationship between each 24-bit buffer and the display image.

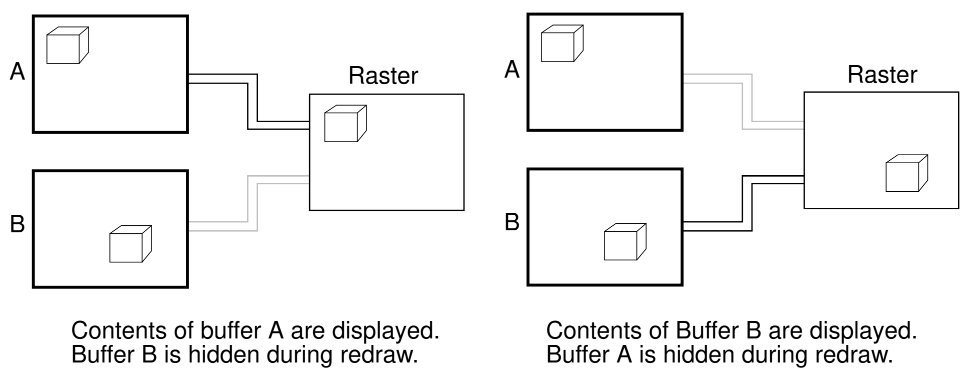

Figure 3.6 SPARCstation ZX Accelerator Double-Buffering

Z-Buffer (Depth) Planes

The 24-bit Z-buffer plane stores the depth value for the last pixel written at a given location into the image write buffer. Z-buffering enables the system to make the portions of an object visible that are nearer to the viewer and hide the portions that should be concealed by other portions of the object. This process is referred to as *hidden surface removal*.

Hidden surface removal is performed on the SPARCstation ZX accelerator with the ZXDraw ASIC. The process tests to determine which faces are in front and removes those surfaces that should be hidden. First, the Z-buffer value at each pixel is set to infinity so that any value written to a pixel is nearer to the viewer than the initial pixel value. As the application draws the object, ZXDraw compares the depth of the face with the depth of the last value written to the Z-buffer for that pixel. If the new face is not farther away, the color is written to the image memory, and ZXDraw stores the new depth value in the Z buffer, overwriting any previous value for that pixel. If the face is farther away, the face's pixel is discarded and nothing is changed.

Overlay Planes

The eight overlay planes on the SPARCstation ZX accelerator serve as an extra eight-bit indexed color frame buffer. The overlay data can be transparent or solid. The overlay plane can also be made visible or invisible (via an enable bit). However, changing the contents or visibility of the overlay buffer does not alter the image buffer contents. The overlay is used to run the user's desktop, and 8-bit applications not requiring 3-D graphics acceleration.

Overlay planes provide non-destructive annotation of underlying 8- or 24-bit images. They enable an image (a mail tool, for example), to be temporarily superimposed over another image. In this manner, the data in the image beneath is not changed or affected; the image need not be redrawn when the overlay image is removed.

Fast Clear Planes

The SPARCstation ZX accelerator includes six fast clear planes to rapidly clear the screen between frames so that animation of objects appears smooth. The six fast clear planes implement the fast clear function for three selected double-buffered image windows. Each fast clear plane pair can be assigned to clear one double-buffered window at optimized hardware speed. Before the start of a new frame, the appropriate fast clear plane is cleared to all zeros, using a special high-speed clear mode, indicating that the values stored in the image and Z-buffer planes are invalid for a specified window.

As pixels are rendered into the buffer, a '1' is written into the fast clear plane at the pixel location, indicating that the image and depth are now valid. During display refresh, all valid pixels (Fast Clear = 1) are displayed using the color value stored in the image or overlay buffers. Invalid pixels (Fast Clear = 0) are displayed using the color value stored in the fast clear background color assigned to that fast clear set.

The SPARCstation ZX accelerator produces extremely fast clearing of image and Z-buffer planes, with clear rates under 200 microseconds. This clear rate ensures the smooth update of successive images.

Window ID Planes

The SPARCstation ZX includes a set of ten window ID planes, which are segmented into six P planes and four Q planes. The window ID planes are separated in order to support uncorrelated overlays.

During image, Z-buffer and overlay plane writes, the current window ID value is compared with the stored window ID for each pixel. If the values are not equal, no write occurs.

During display cycles (50 to 112 times per second, depending on display configuration), the stored Q window ID value is used to determine overlay transparency, while the stored P code is used to specify the current image display buffer, and output color model for each pixel on the screen. If the display is not transparent, the Q window ID works identically to the P window ID.

The ten-bit window ID acts as an index into the window look-up table in the ZXCross ASIC to define the window's display properties:

- Color mode selection: 24-bit true color or 8-bit indexed color
- Double-buffered image plane window control
- Selection of color maps
- Specification of the three fast clear window sets

Video Output

The Video Output section consists of the ZXCross ASIC and a random-access memory digital-to-analog converter (RAMDAC). The ZXCross ASIC contains the window ID look-up tables (LUTs) and the color look-up tables, along with the programmable video timing generation and hardware cursor generation logic.

The ZXCross ASIC performs output pixel multiplexing (buffer and channel selection), pseudo-color look-up, and gamma correction (as required). A variety of monitor resolutions are supported, ranging from interlaced NTSC at 640x480 pixels to 1280x1024 non-interlaced.

Look-up Tables

The window ID look-up tables define the window's display properties, such as the color mode, double-buffering, and so on. The color look-up tables are, in essence, color maps, which contain a selection of colors for a particular application. Color look-up tables are used primarily in eight-bit indexed mode.

In indexed color mode, an eight-bit value from the image plane addresses a location in the color look-up table. The look-up table has as many entries as there are pixel values, that is, there are 256 (0 to 255) color map entries for the eight bits ($2^8 = 256$). Each of the 256 possible bit combinations references an entry in the colormap, rather than directly dictating the intensity of the CRT electron beam.

The output RAMDAC contains a third type of look-up table, known as a *gamma correction* table. Gamma correction is an adjustment to the normal color mapping to make up for non-linearity of the luminescent phosphor in color CRTs. The gamma correction look-up table is available to all types of color models.

Video Timing

The SPARCstation ZX accelerator incorporates a programmable video timing generator in order to support multiple display resolutions. The timing generator has programmable registers that contain information such as pixel start and stop points, horizontal blanking pulse, sync pulse, and other variables. A programmable frequency synthesizer generates different pixel clock frequencies necessary to support the multiple display resolutions.

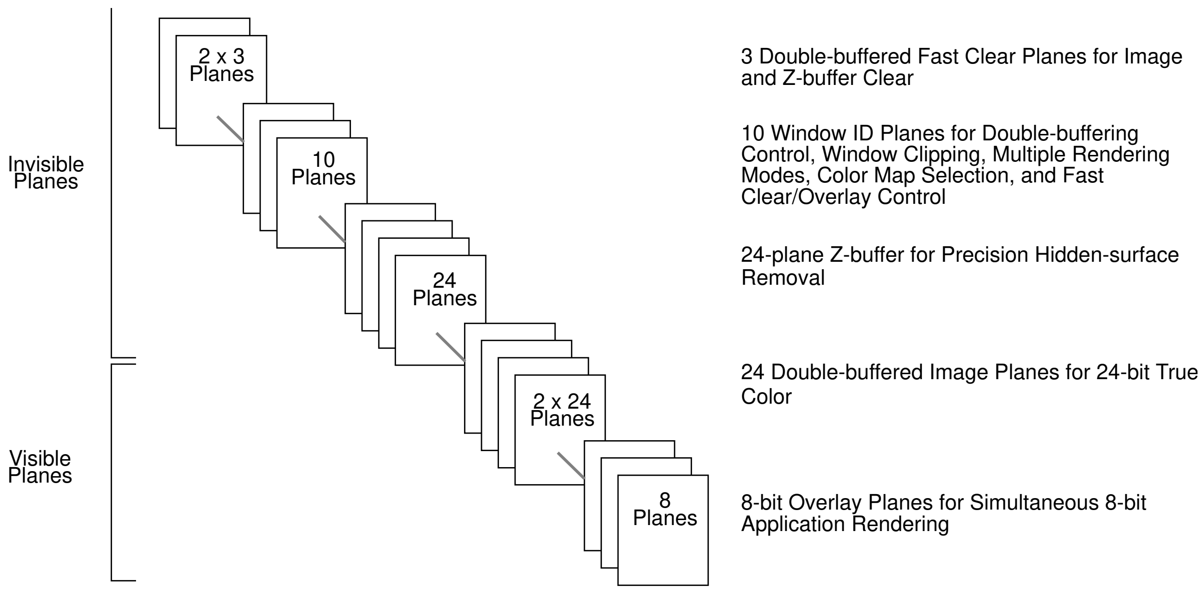

Figure 3.7 SPARCstation ZX Accelerator Graphics Plane Groups

Hardware Cursor

Rather than implementing a conventional cursor plane in the frame buffer, the SPARCstation ZX accelerator provides hardware cursor generation logic directly in the ZXCross ASIC. The cursor information is limited to 32x32 pixels. Cursors larger than 32x32 must be rendered in software.

The cursor color uses one bit of information per pixel, selecting between two cursor colors. Cursor enabling capability is made possible by an additional bit per pixel, used to disable or enable the cursor color.

Host Interface

The SPARCstation ZX processor communicates with the host workstation as an SBus slave, a DVMA master, and by generating interrupts on the SBus. The host uses the slave interface to directly access the graphics processor memory and devices. The SPARCstation ZX processor uses the DVMA interface to get graphics geometry data from the host system memory to the accelerator port, and to move raster data from the host system memory to the frame buffer.

DVMA is utilized with contiguous geometry only, and is not used to move attribute data from memory to the accelerator. Attribute data and non-contiguous geometry data use programmed I/O for their transfer to the SPARCstation ZX processor.

SPARCStation ZX Accelerator Data Paths

The SPARCstation ZX accelerator uses two, discrete paths for rendering data between the host system and the frame buffer. A direct path gives an application bidirectional access to the frame buffer directly through the ZXCommand and ZXDraw render logic. The accelerated path gives an application unidirectional access to the frame buffer through ZXCommand, the ZXFloat floating point transform logic, and the ZXDraw render logic.

The accelerated path is optimized for 3-D graphics applications. It accepts vertex data (such as triangle strip, polyline, dot) commonly used in such applications. The direct path to the Frame Buffer is provided for other applications (such as text update and scroll, and image load and store) that are not accelerated.

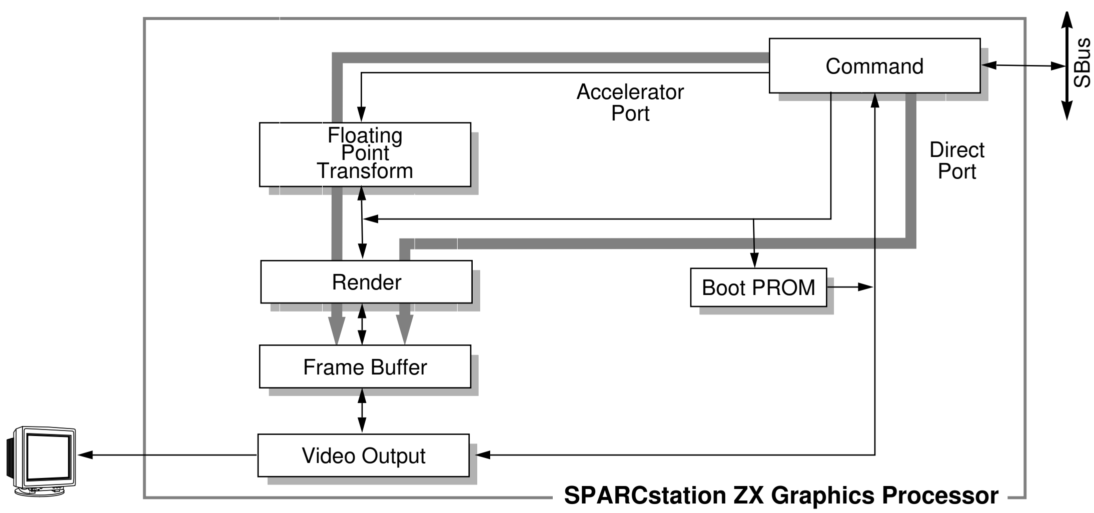

Figure 3.8 SPARCstation ZX Accelerator Port versus Direct Port

Direct Path

The direct path to the frame buffer enables the host to bypass the floating point transformations for those applications that are better served by host-based software. This includes complex primitives that are not supported through the accelerator port. Tasks such as the OpenWindows™ window server, 2-D integer geometry graphics, and image processing applications run on the host CPU and use the direct path to the SPARCstation ZX frame buffer.

The direct path also enables the host to access other SPARCstation ZX graphics processor device components, for example, to update the status and control registers on some of the graphics processor logic devices. This path is also used to access the Boot PROM and the look-up tables in the Video Output section.

The SPARCstation ZX accelerator is an SBus slave when the host uses the direct port to read or write the frame buffer, and for a number of accelerator port functions. This is the path primarily used by the window system. Support is provided for rendering text, area moves, and fill.

Accelerated Path

The accelerated path to the frame buffer goes through the Floating Point Transform section. This path is used to render three basic primitives: dots, vectors, and triangles. The accelerator also can render antialiased dots, antialiased vectors, and triangle strips. Various data formats may be used, for chained or isolated data, color information, and packed or floating point normals.

The SPARCstation ZX accelerator usually uses programmed I/O to fetch data (via the CPU) from system memory. However when the data consists of chained vertices, DVMA is used. After the host CPU tells the accelerator that vertex data is ready, ZXCommand fetches the pipeline input from system memory using DVMA. ZXCommand converts the chained vertex data into a small number of drawing primitives with floating point coordinates. It then passes these primitives to the Floating Point Transform section, which transforms the data into screen-space, fixed-point rendering parameters and passes its output to the Render section. The Render section converts the rendering parameters into pixel operations for the frame buffer. The pixel operations include various blend and transparency operations along with window ID and Z-buffer checks. The frame buffer stores the color, depth, and transparency for each pixel on the display. The contents of the frame buffer are continuously scanned out to the video display monitor.

SPARCstation ZX Processor Packaging

The SPARCstation ZX accelerator uses a double-wide SBus form factor consisting of two boards, stacked vertically. In the space allocated for two SBus cards (approximately 5x9 inches), the graphics processor offers 96 bit planes of DRAM/VRAM, 11 custom (4 unique) ASICs, associated SRAM, and a RAMDAC.

This density was achieved in two ways. First, optimum use was made of the available space by integrating all required functions into custom ASICs. The use of custom ASICs helped limit the number of chips by reducing the 'glue' chips required by discrete generic components. Virtually all the components on the SPARCstation ZX accelerator are VLSI or RAM, providing both high functional density and very high reliability for the product.

Even with the density achieved through high levels of integration, the SPARCstation ZX graphics processor requires more square inches of board space than a desktop would allow. So the circuitry was "folded" into a 2-board assembly, nearly doubling the available area. Figure 3.9 illustrates the SPARCstation ZX graphics processor.

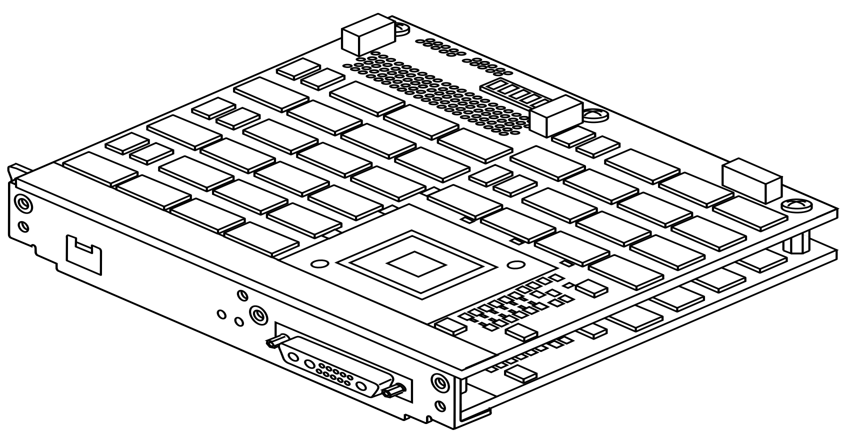

Figure 3.9 SPARCstation ZX Graphics Processor Packaging

Stereo Capability

Stereopsis is a means for providing depth perception to a user viewing a computer image on the screen. It computes separate images for the right and left eye, and then displays each image to the respective eye. The SPARCstation ZX graphics processor provides stereopsis capability by syncing separate images, individually computed for the left and right eyes, to an external sync mechanism attached to a viewing system worn by the user.

To use stereo, an application provides two viewing matrices (one for each eye) instead of the usual single matrix, and computes and sends the data for each eye. The application then activates stereo mode. The OpenWindows environment and the SPARCstation ZX hardware then do the rest of the work.

Where application windows activate stereo mode, the display will automatically alternate between the left and right views (as defined by the matrices), in sync with the shutter mechanism. For all non-stereo windows, the display will automatically provide the same view to both eyes.

The SPARCstation ZX accelerator supports two stereo modes:

- 960x680 at 108 Hz Non-interlaced (54 Hz per eye)
- 960x680 at 112 Hz Non-interlaced (56 Hz per eye)

Adding Stereo to the SPARCstation ZX System

The stereo connector for connection between stereo goggles and the SPARCstation ZX graphics accelerator is a female, three-conductor, mini-phone connector, as illustrated in Figure 3.10.

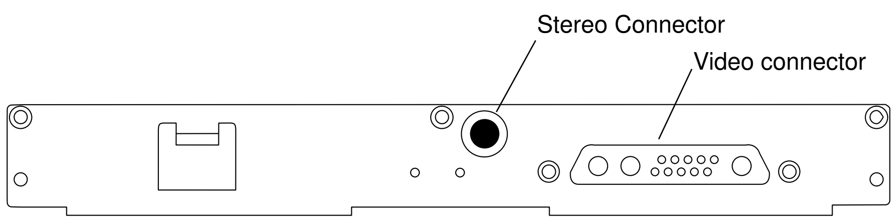

Figure 3.10 SPARCstation ZX Graphics Processor Stereo Connector

The stereo signal is a TTL-level, 50 percent duty cycle signal that switches between left and right stereo shutters. The polarity of the signal may be reversed through software control.

Stereo cables and goggles for use with the SPARCstation ZX system are available from the following sources:

StereoGraphics Corporation
2171–H East Francisco Blvd.
San Rafael, CA 94901
(415) 459-4500
FAX: 415-459-3020

Tektronix, Inc.
Display Product Division
P.O. Box 500
Beaverton, Oregon 97077

SPARCstation ZX Accelerator Software 4

SPARCstation ZX Software Interfaces

The SPARCstation ZX accelerator is supported by all Solaris® graphics and window systems APIs, including the XGL graphics library, the XIL imaging library, the Direct Xlib library, PostScript, and the OpenWindows (X Windows) environment. Industry standard X extension libraries such as Xlib and PEXlib are also available and are accelerated when appropriate. Emerging standards such as the X Imaging Extension (XIE), and Display PostScript® (DPS), together with their related APIs (such as XIElib and DPSlib) will be supported in the future.

As a geometry device, the SPARCstation ZX system principally accelerates the XGL geometry library, and APIs that are layered above it. However because the SPARCstation ZX accelerator incorporates a direct port for pixel-based rendering, imaging interfaces and window systems are also accelerated. The following sections briefly describe the foundation graphics interfaces and the functions accelerated on the SPARCstation ZX processor. Figure 4.1 illustrates the Solaris graphics and windowing APIs and their relationship to the SPARCstation ZX system.

Geometry Interfaces

As a 3-D solids graphics accelerator, the SPARCstation ZX system requires a geometry-oriented graphics API to take full advantage of its features. Developers may use a number of industry standard and proprietary interfaces available under the Solaris operating system to render graphics.

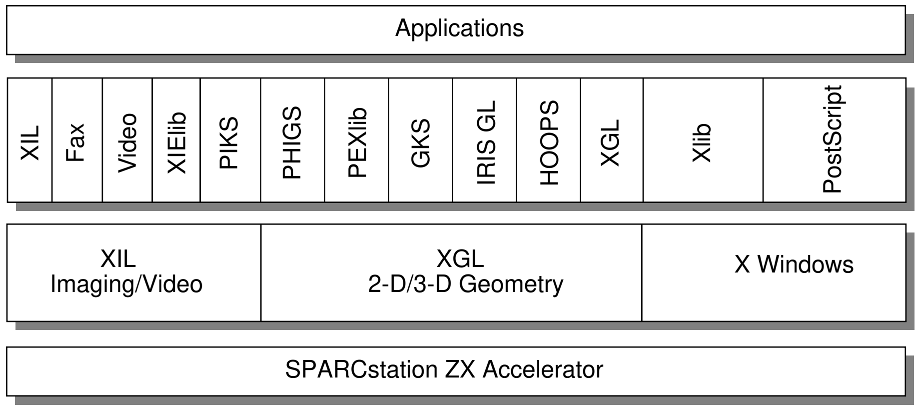

Figure 4.1 Solaris Foundation Graphics Libraries and Layered Interfaces

XGL Library

The XGL foundation geometry library provides the functionality and performance required by applications requiring geometry manipulation and display. It optimizes 2-D and 3-D rendering, including high-quality lighting and shading, NURBS, meshes, texture mapping, antialiasing, and transparency. XGL incorporates flexible geometry manipulation through separate and distinct 2-D and 3-D pipelines. The XGL library includes sophisticated state-of-the-art features such as dynamic tessellation of NURBS surfaces, as well as flexible pipelines for lighting and transformation.

The SPARCstation ZX accelerator was developed to map as closely as possible to the feature set and architecture of the XGL library. Individual operations defined in XGL and represented by the XGL 'loadable interfaces' are replaced by driver-level code that maps directly to microcode on the SPARCstation ZX accelerator. Among the key features in the XGL library accelerated by the SPARcstation ZX processor are:

- Transformations
  Both 2-D (3x2) and 3-D (4x4) transformations
- Geometry Attributes
  Common attributes including color, line type, fill pattern and textures, etc.
- Lighting and Shading
  XGL lighting models, including flat and Gouraud as well as up to 32 light sources (positional, directional, spot, and ambient)

- Non-Uniform Rational B-Splines (NURBS)
  NURBS capability, including optimized dynamic tessellation support
- Transparency
  Both screen-door and alpha blended transparency
- Anti-Aliasing
  Anti-aliased vector support
- Depth Cueing
  Both linear and scaled
- Texture Mapping
  2-D texturing of 3-D surfaces at multiple points in the rendering pipeline

The XGL library has an object-oriented interface that hides the low-level details of implementation from the application developer. As such, it is architected so that it may evolve as an interface, permitting new algorithms to be implemented within XGL without incompatibly changing the application program interface bindings.

Third-Party Geometry Interfaces

Third-party geometry libraries available on Solaris range from many implementations of PHIGS and GKS, to HOOPS©, IrisGL©, OpenGL© and a large number of other interfaces. In most cases, these products are layered above the XGL library, and hence access the same functionality and experience the same performance on the SPARCstation ZX accelerator.

PEX and PEXlib

PEX and PEXlib are a 3-D standards-based wire protocol and X Extension API (for PEX) respectively. PEXlib and PEX both uses the XGL library and therefore take advantage of the mapping between the XGL library and the SPARCstation ZX microcode.

Imaging Interfaces

While the SPARCstation ZX processor is principally a 3-D geometry accelerator, some of its features are important to imaging applications. This consideration is perhaps most important to users with multiple application requirements including 3-D rendering.

For example, a user (or developer) of animation software might use the SPARCstation ZX accelerator for 3-D geometry manipulation in modeling and simulating an animation sequence, but rely on the SPARC CPU performance, and the SPARCstation ZX accelerator's 24-bit display and video output for generation of the compute-intensive final rendering.

A user of both imaging and geometry applications can also benefit from the SPARCstation ZX architecture. Imaging applications using the XIL library uses the direct path when rendering on the SPARCstation ZX.

XIL Library

The XIL library is a foundation imaging library providing the common functions required by many imaging applications. It defines how imaging operations, such as display, image manipulation, compression and decompression, are carried out.

Many low-level XIL functions are accelerated by the SPARCstation ZX accelerator. These operations are principally associated with raster manipulation and can include:

- Color Map Manipulation
- Raster Operations
- Image Scaling and Transformation
- Image Clipping

The XIL library is an object-oriented interface that hides the details of its implementation from the programmer. This enables the internals of the XIL library to be extended without changing the interface in an incompatible way. C language bindings provide compatibility with other tools available under the Solaris operating environment including a large number of software development products.

Third-Party Imaging Interfaces

Third-party imaging interfaces usually extend the capabilities and user-interface facilities of the XIL library to include broader functionality such as image processing and manipulation, visualization, and machine and industrial vision.

Third-party interfaces layered above the XIL library may exploit its capabilities on the SPARCstation ZX accelerator. Some APIs, such as those used for mapping, image processing or visualization may use the XIL library for 2-D image processing while relying on the XGL library for 3-D manipulation.

XIE and XIElib

XIE and XIElib are a 2-D standards-based wire protocol and related X extension API currently under study by various standards organizations. XIE and XIElib address the requirement for a standard imaging library within the distributed X Windows environment. While XIE and XIElib are not yet available, they are expected to use the XIL library on all Solaris platforms when they become available. As a result, XIElib applications will experience the same performance and where appropriate, the same functionality afforded to the XIL library by the SPARCstation ZX accelerator.

OpenWindows Interfaces

The OpenWindows environment is an X Windows compliant server. Applications that use the X protocol, running both locally or remotely, will receive some acceleration from the SPARCstation ZX system. OpenWindows interfaces, including those listed below, use the direct path when rendering on the SPARCstation ZX accelerator.

The SPARcstation ZX accelerator supports efficient implementations of the OpenWindows window system and its APIs by providing support in the following key areas:

- A direct path is provided from the host interface through ZXCommand to the frame buffer. Independent frame buffer state (separate from the accelerator port) is maintained.
- The frame buffer provides window ID planes and clipping, supporting arbitrarily shaped windows.
- The frame buffer supports raster functions, including local Boolean operations and local read/modify/write access to the frame buffer.
- Stencil access mode provides faster text operations, writing horizontal spans of up to 32 pixels in a single host write request.
- The frame buffer supports double-buffering and output color map selection on a per window basis, using window ID planes.

- The frame buffer supports fast fills of rectangular frame buffer regions with constants, such as setting window IDs.
- Eight overlay planes in the frame buffer can be treated as a separate desktop from the image and depth planes.

Xlib

Xlib is a 2-D integer-oriented X Windows API. Any features of the OpenWindows window systems accelerated by the SPARCstation ZX accelerator may also be accelerated with Xlib.

Direct Xlib Library

The Direct Xlib library is an implementation of the X Windows-based Xlib library, which has been designed to use Sun's Direct Graphics Access (DGA) technology. DGA uses a shared memory mechanism to synchronize rendering between the graphics interface and the window system server. As a result, graphics rendering may take place independent of the window server, thus achieving higher performance.

Direct Xlib is call-for-call compatible with the Xlib library, but through the use of DGA is able to be directly implemented upon the host display accelerator. While standard Xlib also achieves modest acceleration on the SPARCstation ZX system through the window server, application developers who desire maximum performance for their locally running Xlib applications should consider using the Direct Xlib library.

PostScript

The PostScript (and Display PostScript) API was developed by Adobe Systems, initially as a printer control language. Its device independence and the general trend toward WYSIWYG display rendering has led to the availability of PostScript on Sun Workstations®.

On the SPARCstation ZX system, PostScript rendering takes place through the direct path. Therefore, PostScript-based APIs enjoy the same characteristic acceleration as the OpenWindows window system. Features such as colormap manipulation, stencil and raster operations, non-rectangular area fill and Boolean operations are all accelerated on the SPARCstation ZX processor.

Summary 5

3-D graphics technology is often as complex to design and implement as the computing platforms they complement. Similar design considerations contribute to their complexity, from heat and power constraints to performance and coupling to related subsystems such as I/O, memory, and so forth.

Engineering design considerations are relevant to both computing and graphics subsystems. And increasingly, high-level integration, custom design, and recognizing system bottlenecks have become significant possible areas of performance and price improvement.

The SPARCstation ZX accelerator leverages an understanding of the architectural constraints of today's graphics subsystems to provide new levels of price/performance and flexibility. The SPARCstation 10ZX system introduces the performance of multiprocessing to computationally-intensive graphics tasks, while the SPARCstation ZX system is the highest performance and lowest-cost desktop solids workstation available today.

The design of the SPARCstation ZX accelerator benefits from an analysis of bottlenecks in graphics processing, and a leveraging of component technologies into the highest possible level of integration. The resulting graphics systems design achieves new levels of size reduction, functionality, and, above all, performance. For the first time, solids modeling design can occur on every desktop and users, unencumbered by working solely with wireframe models, can achieve new levels of design construction and visualization.

Coincident with the availability of SPARCstation ZX technology is the maturation of Sun's XGL graphics library. The XGL library, now in its third generation and the basis of many third-party graphics APIs, has added powerful new features, ranging from transparency and anti-aliasing to texture mapping and dynamic NURBS tessellation. Many of the XGL library features map efficiently to the microcode of the SPARCstation ZX accelerator, and all deliver substantially higher performance than any previous Sun platform.

The SPARCstation ZX system includes additional, powerful features ranging from 8-bit overlay support, and multiple resolutions, to stereo, 24-bit Z-buffering, extensive window ID planes, and up to 32 individual lights in microcode.

Developers and end users alike will find the SPARCstation ZX system an attractive complement to their application solutions. Complete support for and optimized acceleration of other Solaris interfaces including the OpenWindows environment, the XIL library and third-party APIs that layer above them further increases the broad applicability of the SPARCstation ZX workstation for today's demanding graphics user.

References

Solaris XGL Reference Manual, SunSoft Inc.
Solaris XGL Programmer's Manual, SunSoft Inc.

Solaris XIL Imaging Library Reference Manual, SunSoft Inc.
Solaris XIL Imaging Library Programmers Guide, SunSoft Inc.

SunPHIGS Reference Manual, Sun Microsystems Computer Company
SunPHIGS Extensions Reference Manual, Sun Microsystems Computer Company
SunPHIGS Porting Guide, Sun Microsystems Computer Company

Direct Xlib 2.0 Users Manual, Sun Microsystems Computer Company

OpenWindows User's Guide, SunSoft Inc.
OpenWindows Programmers Guide, SunSoft Inc.

X Window System User's Guide, O'Reilly & Associates. Inc.
Xlib Reference Manual, O'Reilly & Associates. Inc.
*Xlib Programming Manua*l, O'Reilly & Associates. Inc.

X Protocol Reference Manual, O'Reilly & Associates. Inc.
PEX Protocol Specification Version 5.0P, MIT

PEXlib Specification and C Language Binding, Hewlett-Packard Company, Version 5.1

PEXlib Programming Manual, O'Reilly & Associates. Inc., 1992
X Image Extension Protocol Reference Manual, MIT X Consortium, Public Review Draft Version 4.1.2

The Display PostScript System, Adobe Systems Inc., 1990
An Introduction to Computer Graphics Concepts, Addison-Wesley
The Renderman Companion, Upstill, Addison-Wesley

Sun Microsystems
Computer Corporation

A Sun Microsystems, Inc. Business

For U.S. Sales Office locations, call: 800 821-4643
In California: 800 821-4642

Australia: (02) 844 5000
Belgium: +32 2 730 38 11
Brazil: 011-55-11-887-9011
Canada: 416 477-6745
Caribbean: +1 415 688-9464
Finland: +358-0-502 27 00

France: (1) 30 67 50 00
Germany: (0) 89-46 00 8-0
Hong Kong: 852 802 4188
Italy: 039 60551
Japan: (03) 3221-7021
Korea: 822-563-8700

Latin America: 415 688-9464
Mexico: 001-525-580-5229
The Netherlands: 033 501234
New Zealand: (04) 499 2344
Nordic Countries: +46 (0) 8 623 90 00
PRC: 861-831-5568

Singapore: 224 3388
Spain: (91) 5551648
Switzerland: (01) 825 71 11
Taiwan: 2-514-0567
UK: 0276 20444
Elsewhere in the world, call
Corporate Headquarters:
415 960-1300
Intercontinental Sales:
415 688-9000

Printed in USA