Performance scaling and Depth/Alpha acquisition in DVI graphics clusters

Alan Heirich, Phillip Ezolt, Mark Shand, Erwin Oertli and Glenn Lupton
Hewlett-Packard Corporation
October 11, 2002

Abstract. This white-paper discusses DVI image acquisition in the context of Sepia, a commodity-based cluster visualization architecture developed by Hewlett-Packard. DVI acquisition performance is characterized by pipeline stall, DVI refresh rate, and head-of-line latency. These characteristics determine the scaling efficiency and interactive latency of a cluster in both sort-first and sort-last configurations. We present a model for DVI acquisition and show that optimal acquisition requires special support from graphics hardware vendors. Since this support is currently not provided, we discuss workaround methods for acquiring alpha and Z buffers from ATI and NVIDIA cards. These methods can be implemented on Linux and Windows using the standard vendor-supplied drivers.

1. Introduction

In recent years several cluster visualization architectures have been proposed or developed that support sort-first, sort-last or hybrid parallel rendering algorithms [HEI99,SV6,HUM02,STO01,BLA01]. The Sepia architecture (patents pending) [HEI99,MOL99,LOM01,ISA02] is distinguished among these by scaling independently in image size and data size while preserving unit cost per cluster node, and by supporting user-programmable compositing operators with ordering constraints, including Porter-Duff translucency operators and blending operators for volume rendering. The architecture is implemented atop a low-latency packet switching network, the *Virtual Interface Architecture (VIA)* [VIA99], as a modified network adaptor with FPGA coprocessor, extra memory and I/O attachments (see figure 1).

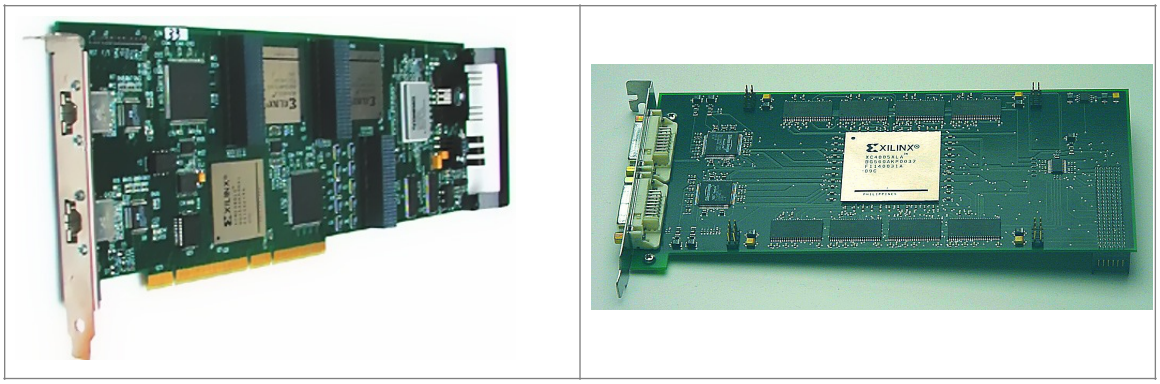

Figure 1. Sepia-2a prototype circa 2002. Main compositor board (left) features dual-Servernet network ports, 66-MHz PCI interface, 3 Xilinx FPGAs, SRAM and expansion connectors. DVI I/O daughterboard (right) features input and output DVI ports, 1 Xilinx

FPGA with DRAM, and attaches to the main board through an expansion connector. One set of boards is required by each rendering or display node in a cluster.

1. The Digital Video Interface (DVI) standard

The DVI 1.0 standard [DVI] is a serial high-speed digital connection providing a a maximum data rate of 495 Mbyte/s. The standard is primarily focused at providing a connection between a computer and its display device. Therefore neither flow control nor error detection is provided. In addition, Hot-plug detection, Display Data Channel, and Display Information Data provide Plug and Play capability.

2. OpenGL frame buffers

Commodity OpenGL accelerators typically organize their frame buffer contents into RGBA (red, green, blue, alpha) and ZZZS (three byte depth and stencil) tiled configurations. Acquiring the contents of the frame buffer by means of the DVI transmitter faces two problems: firstly, only RGB data is transmitted, that is, non-RGB data needs to be copied into RGB data prior to acquisition. Secondly, RGBA data may always be double-buffered whereas ZZZS data is typically single-buffered, and compressed as well as address-permuted in order to minimize frame buffer bandwidth. With it, ZZZS data cannot be acquired in parallel to rendering the next frame. In addition, direct acquisition of ZZZS data requires knowledge of their proprietary data format. This knowledge can only be acquired from graphics vendors.

2. Optimal DVI image acquisition

The performance of the image acquisition process has a major impact on the scaling and interactivity of applications that run on the visualization cluster, irrespective of the rendering paradigm or system software that is in use. Acquisition methods that stall the graphics pipeline yield poor scaling efficiency, and in these cases high rendering performance is sustained only by running at reduced frame rates or resolutions. Applications that acquire multiple buffers (for example, RGB and Z, or RGB,A,Z and stencil) typically multiply both acquisition latency and pipeline stall.

Optimal image acquisition incurs no pipeline stall. Pipeline stall can occur during the image acquisition process, for example as a delay while an Alpha, Depth, or Stencil channel is copied into an RGB buffer. It can also occur when applications are single buffered, for example because an application with single buffered Z cannot overlap acquisition and rendering of successive frames and must stall until acquisition is complete. Thus optimal acquisition requires full double buffering combined with direct acquisition of frame buffer content without copying or processing.

Pipeline stall can be death to application performance. For real-time applications at 60 frames per second, each millisecond of pipeline stall per frame reduces rendering performance by 6%. Acquisition methods that stall for several milliseconds per buffer can reduce rendering performance by 50% or more.

Optimal image acquisition incurs minimal head-of-line latency. This latency is the difference in time from when an application indicates a frame is complete to when pixel data is available to the network compositor. Pixel data is provided in interleaved form, for example RGBAZS, irrespective of the number of buffers that were acquired. Head-of-line latency increases with multiple buffers because all buffers must be acquired before pixels can be interleaved and issued to the compositor. Head-of-line latency is also affected by acquisition data rate through the DVI channel. The DVI refresh rate should always be set as high as possible, independent of the application frame rate, in order to minimize the time to acquire each buffer. Applications that acquire multiple buffers must use a DVI refresh rate that is a multiple of the application frame rate. For example, a 30 frame-per-second application that acquires RGB and Z requires at least a 60 Hz DVI refresh rate in order to acquire two buffers per frame.

Head-of-line latency is just one component of the overall end-to-end system latency, which is also affected by latency due to system de-synchronization, latency through the compositing network, and latency incurred by the display. Head-of-line latency can be reduced by implementing *cut-through* acquisition, in which pixels are released to the compositor as soon as the necessary data is available, concurrently with receiving the final buffer. Head-of-line latency can also be reduced for multiple buffer acquisitions by allowing buffers that are stored contiguously, such as RGBA, to be acquired in a single step.

The requirements for optimal DVI image acquisition can only be satisfied with special support from graphics vendors. Double buffered Z and stencil are not standard features of OpenGL. Low level control of the DVI transmitter addressing and framing is required to obtain raw frame buffer content with minimal head-of-line latency, and vendor-proprietary algorithms are required to decode raw ZS content.

3. Workarounds for DVI acquisition

Although graphics vendors don't currently provide the support needed for optimal DVI image acquisition workarounds can be used to acquire depth (on NVIDIA) and Alpha (on ATI and NVIDIA) using standard vendor-supplied drivers. The methods we employed display alternating full-screens of RGB and depth or alpha. This allows the application to use the full frame buffer resolution, while other techniques split the display into RGB and alpha/depth regions and require reduced or non-standard resolutions.

It is straightforward to implement an acquisition method where the application renders a frame and waits for an acquisition to complete. Unfortunately, this causes a lengthy pipeline stall. A higher-performance alternative overlaps rendering and acquisition by using signal handlers to control the acquisition independent of application rendering. Signal handlers are called when an RGB or depth/alpha acquisition is complete. The

signal handlers issue the appropriate OpenGL calls to place the proper RGB or depth/alpha data on the screen and start the next acquisition.

Using glReadPixels/glDrawPixels is the most straightforward choice for copying alpha/depth to the RGB buffer. However, these routines cause a slow round trip copy from the graphics card to system memory. A faster alternative is to perform all operations on the graphics card.

3.1 NVIDIA NV_copy_depth_to_color

NVIDIA drivers support an extension to copy 24-bit Z values to corresponding locations in the RGB buffer. This allows multiple buffer acquisition of RGB and Z with an intervening copy operation that destroys the value of the front RGB buffer. Since Z is single buffered this method incurs pipeline stall until after the copy operation completes, at which point the application can begin rendering the next frame. At 60 Hz DVI refresh this is approximately 17 milliseconds (to acquire RGB) plus 2.8 milliseconds (to copy Z to RGB at 1280x1024). A 30 frames-per-second renderer incurring 20 to 25 milliseconds of pipeline stall per frame experiences a 60% to 75% reduction in rendering performance. Minimum head of line latency using cut-through is equivalent to the RGB acquisition cost (17ms) plus the Z copy (2.8ms) or ~19.8ms.

It is possible to work around the single buffered Z by storing RGB to a buffer region and copying depth to the front buffer for acquisition. The application then continues rendering, but it has suffered a pipeline stall. The pipeline stall is at least the cost of RGB to buffer region copy (1.5 ms) plus the depth copy (2.8 ms), or ~4.3ms. The application must also wait for the previous acquisition to complete. The minimum acquisition time is equivalent to the signal handling cost (3 us) plus the depth acquisition cost (17ms) plus the RGB buffer region restore (1.4 ms) plus the RGB acquisition cost (17ms) or ~35.4ms. If the application's single-frame render time is greater than acquisition time (35.4 ms), then the application will not stall while the acquisition finishes. If the application's rendering time is less than the acquisition time, it will wait at most an additional 35.4 ms. The pipeline stall in the best case will be 4.3ms, and in the worst case be 39.7 ms. The minimum head of line latency with cut-through is equivalent to the signal handling cost (3 us) plus the RGB to buffer region copy (1.4ms) plus the depth copy (4.3ms) plus the depth acquisition cost (17ms) plus the RGB buffer region restore (1.4ms) or ~22.7ms.

3.2 Alpha acquisition by blending

Standard OpenGL blending operations can be used to copy alpha to RGB on an alpha-enabled framebuffer. By setting the glBlendFunc to (GL_DST_ALPHA, GL_ZERO) and then drawing a white square over the framebuffer, the alpha values are deposited in the RGB pixels. After the blend, the R, G, and B values are identical and equal to the alpha value of that pixel location. In contrast to depth, most modern OpenGL implementations have front and back buffers with independent alpha. This allows data to be acquired from the front buffer, while the application renders to the back buffer.

Signal handlers help the DVI acquisition using the following method. The application renders a frame to the back buffer. It then waits for the acquisition of previous frame's alpha values to complete. The application swaps the RGB from the back buffer to the front buffer and starts the DVI acquisition. At this point the application begins to render the next frame to the back buffer. When the RGB acquisition is finished, a signal handler is called which replaces the RGB in the front buffer with the alpha values and starts the alpha acquisition. The signal handler hides the DVI acquisition stall from the application, but the application still must wait for the previous acquisition to complete before it starts the next acquisition. The minimum acquisition time equals the signal handling cost (3 us) plus the RGB acquisition cost (17ms at 60Hz) plus the alpha blend cost (1.6ms at 1280x1024) plus the alpha acquisition cost (17ms) or ~35.6ms. If the application's single-frame rendering time is less than the acquisition time (35.6ms), the pipeline stall is the acquisition time (35.6ms) minus the rendering time. If the application's single-frame rendering time takes longer than the acquisition time (35.6ms), this method has no pipeline stall. The minimum head of line latency with cut-through is calculated by adding the signal handling cost (3 us) to the RGB acquisition cost (17ms) to the alpha blend cost (1.6ms) or it is ~18.6ms. The performance penalty of simultaneous use of the graphics card by the application and signal handler is unclear.

4. Summary

This white-paper has discussed the performance of DVI acquisition strategies in the context of a specific visualization cluster architecture, Sepia. The conclusions apply to other clusters that use DVI image acquisition. In the workshop presentation we intend to include an overview of the Sepia architecture and show taped demonstrations of several toy applications employing RGB, depth and Alpha acquisition on multiple displays.

Bibliography

[BLA01] W. Blanke, C. Bajaj, X. Zhang and D. Fussell. *A Cluster Based Emulator for Multidisplay, Multiresolution Parallel Image Compositing.* Computer Science & TICAM Technical Report, University of Texas at Austin (2001).

[DVI] Digital Video Interface working group. *Digital Video Interface (DVI) 1.0.*

[HEI99] Alan Heirich and Laurent Moll. *Scalable Distributed Visualization Using Off-theShelf Components.* In Proceedings of the IEEE 1999 Symposium on Parallel Visualization and Graphics (1999).

[HUM02] Greg Humphreys, Mike Houston, Yi-Ren Ng, Randall Frank, Sean Ahern, Peter Kirchner, and James T. Klosowski. *Chromium: A Stream Processing Framework for Interactive Graphics on Clusters.* In SIGGRAPH'02.

[ISA02] Michael Isard, Alan Heirich and Mark Shand. *Distributed Rendering of Interactive Soft Shadows.* In Procedings of the Fourth Eurographics workshop on Parallel Graphics and Visualization (PGV'02), to appear (2002).

[LOM01] Santiago Lombeyda, Laurent Moll, Mark Shand, David Breen and Alan Heirich. *Scalable interactive volume rendering using off-the-shelf components.* In Proceedings of the IEEE 2001 Symposium on Parallel and Large-Data Visualization and Graphics, pp. 115-121 (2001).

[MOL99] Laurent Moll, Alan Heirich, and Mark Shand. *Sepia: scalable 3D compositing using PCI Pamette.* In Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'99), April 1999.

[STO01] G. Stoll, M. Eldridge, D. Patterson, A. Webb, S. Berman, R. Levy, C. Caywood, M. Taveira, S. Hunt and P. Hanrahan. *Lightning-2: a high performance display subsystem for PC clusters.* In SIGGRAPH 2001, pp. 141-148.

[SV6] Hewlett-Packard Corporation. http://www.hp.com/workstations/products/immersive/sv6/faqs.html

[VIA99] VI architecture specification. http://www.viarch.com