White Paper

# The SGI® Onyx® 3000 Series with InfinitePerformance™ Graphics—Architecture and Programming

Bob Kuehne, SGI

## Abstract

The SGI Onyx 3000 series of visualization systems with InfinitePerformance graphics represents the latest product offering in the decade-long history of scalable graphics products from SGI. The SGI Onyx 3000 series with InfinitePerformance graphics is the first SGI product to perform graphics scaling using powerful low-cost pipes in conjunction with a new hardware compositor module. This system allows unprecedented levels of host and graphics scalability, resulting in the highest geometry performance commercially available today. This white paper will detail the overall system architecture and focus on graphics software techniques available for developers to fully utilize this system.

## 1.0 Introduction

The Onyx 3000 series with InfinitePerformance graphics is a highly scalable next-generation graphics powerhouse system from SGI. The Onyx family workstations combine the shared-memory programming that users have come to expect with the unrivaled scalability of the SGI® Origin® 3000 series architecture, adding unprecedented graphics scalability. The result is the most scalable graphics system on the planet. The Onyx 3000 series with InfinitePerformance graphics allows applications to easily scale to work with ever-increasing data-set size, model complexity, and visual realism.

The Onyx 3000 series has numerous advantages over other systems on the market. First, the Onyx 3000 series system architecture provides the highest bandwidth interconnect available, which speeds moving data throughout the system—from disk, to memory, to CPU, and to graphics. The Onyx 3000 series with InfinitePerformance graphics moves data faster than any other graphics system available. Second, the Onyx 3000 series with InfinitePerformance graphics has the lowest memory latency of any system in the industry, allowing the fastest available access to memory. Third, only the Onyx 3000 series combines this architecture with the simplicity of a shared-memory programming model. Fourth, the Onyx 3000 series is highly modular, allowing the system itself to scale to include more pipes and more channels as a customer's demands increase. Finally, the Onyx 3000 series with InfinitePerformance graphics allows applications to use standard SGI® graphics APIs to seamlessly take advantage of this system scalability.

## 2.0 The Architecture of the Onyx 3000 Series with InfinitePerformance Graphics

The workstations of the Onyx 3000 series with InfinitePerformance graphics share the same high-bandwidth, low-latency SGI® NUMA architecture found in all Onyx and Origin family products and systems. Details about the Origin 3000 series can be found on the SGI Web site. The workstations in the Onyx 3000 series with InfinitePerformance graphics contain two new components:

- V-brick: a compact graphics module containing up to two independent InfinitePerformance graphics pipes
- Scalable graphics compositor: a zero-latency, stereo-capable, genlock-capable, frame-synchronous digital hardware compositor

### 2.1 The V-Brick

The V-brick is a new graphics module consisting of up to two independent InfinitePerformance graphics pipes based on modified versions of the proven VPro™ V12 graphics processor. The VPro V12 is the workhorse of the SGI graphics line, providing high polygon rates and high fill rates.

Up to 16 independent InfinitePerformance graphics pipes can be installed in a single Onyx 3000 series system. These independent graphics pipes can be deployed in a variety of ways that increase the flexibility and power of the system. For example, four independent pipes could be used to drive four single displays or to drive a single 4-way composited display, bringing up to four times the graphics processing power to bear on your data. Systems with more than four independent InfinitePerformance graphics pipes can mix and match usage modes that adapt to system load throughout the day.

### 2.1 The Scalable Graphics Compositor

The scalable graphics compositor (or compositor) allows zero-latency compositing of two or four independent InfinitePerformance pipes simultaneously. Four independent pipes combined through the compositor together are referred to as a 4-way InfinitePerformance graphics pipe. Stereo, genlock, and frame-synchronization support are all provided in hardware. Two video outputs are available on the compositor: one DVI-D and one HD-15 analog.

A variety of visual formats are supported, including:

- 1280x1024_60/75, output: DVI/13W3
- 1280x1024_96s, output: 13W3
- 1600x1200_60, output: DVI/13W3
- 1920x1200_60, output: 13W3

## 3.0 System Configuration

One key advantage of the SGI Onyx 3000 series architecture is its modularity, which allows the systems to be configured in many different ways. This white paper will not address the numerous ways in which a system can be configured, because that topic is covered in other guides, but will look at ways in which pipes can be configured through software tools available on the system.

### 3.1 Configuring a Static Compositor Tiling

The Onyx 3000 series with InfinitePerformance graphics ships with a software tool to allow compositing arrangements to be set statically by a system administrator. This tool, sgcombine, is similar in intent to ircombine, used with the Onyx 3000 series with InfiniteReality3® graphics. Whereas ircombine lets a user describe which areas of the screen will get output on different channels, sgcombine describes how different channels will combine into one logical frame buffer. The tool lets the user visually array the input-independent InfinitePerformance pipes into some logical output arrangement and then download that arrangement throughout the system, configuring independent InfinitePerformance pipes and the compositor simultaneously.

The central sgcombine window (Fig. 1) displays the composited output area. Users can select the video format shared by the compositor and the independent InfinitePerformance pipes and select an external sync format. Selecting the video format for the compositor will automatically select a format for the independent InfintePerformance pipelines and configure external sync. Users may also select an initial tiling setup with the Initial Tiling Mode menu. The User Guide and System Setup Guide both provide additional detail on the operation of this tool.

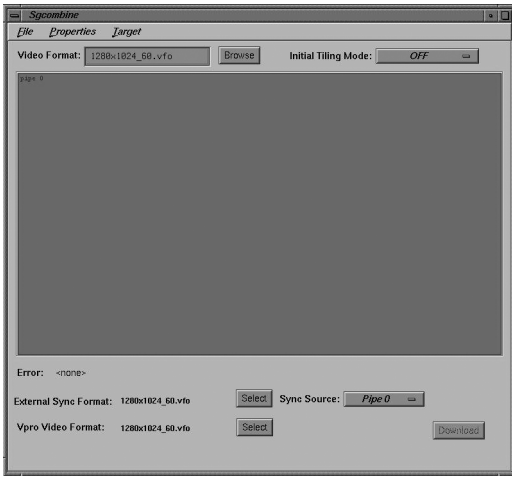


Fig. 1. sgcombine static compositor assignment

## 4.0 Scalable Software for Scalable Systems

The Onyx 3000 series with InfinitePerformance graphics is a scalable hardware system that allows software to use up to four independent InfinitePerformance pipelines simultaneously to address a larger problem than any single pipeline can address alone. To achieve good software scalability, an application must do some additional work to ensure the most efficient rendering. The Onyx 3000 series with InfinitePerformance graphics performs a type of compositing known as spatial, or 2D, compositing. This operation requires specific data-distribution techniques. This section of the paper will discuss the various aspects of performance, from bottlenecks to tuning to APIs.

### 4.1 2D Compositing

2D or spatial compositing, or the division of a logical resultant rendered image into subregions or subimages, is a powerful technique to allow two types of performance scaling simultaneously. First, geometric performance can scale well, provided the geometric data is balanced equally among the four pipes in the compositor. Second, fill performance can scale well, given that each pipe is required to fill the pixels only for the subregion that it contributes to the final image.

An easy initial approach to get an application "compositor-ready" is to select equally sized regions for each of the four pipelines. This technique is virtually identical to that required to implement a Powerwall (three pipes, side by side). For an applica-

tion to use effectively either of these modes, Powerwall or composited, the application must first possess several characteristics and abilities:

- View-frustrum cull-enabled
- Multiprocess or multithreaded
- Texture and display list management enabled

View-frustum culling minimizes the data sent to each pipe by sending only the data needed by that pipe. In an ideal application for the SGI Onyx 3000 series with InfinitePerformance graphics, each independent InfinitePerformance pipeline would perform exactly 25% of the workload, enabling a 4x performance improvement. Owing to view-frustum culling overlap (the effects produced from a single object spanning several pipelines), this efficiency is difficult to achieve. Or, said differently, a 4x performance boost is possible, but your results may differ. View-frustrum culling efficiency is under active research and quantification.

Multiprocessing is the second essential application prerequisite to using the Onyx 3000 series with InfinitePerformance graphics effectively. Without multiprocessing, each of the four independent InfinitePerformance pipes in a single Onyx 3000 series InfinitePerformance pipe would render in serial fashion, making the overall rendering the same performance (realistically, a bit slower) as a single independent InfinitePerformance pipeline: no benefit whatsoever. However, if each pipeline can render in parallel, and the application can keep each pipe busy without stalling the other pipelines, a high degree of scalability can be achieved. Parallelism is essential, and can be difficult, but the Onyx 3000 series with InfinitePerformance graphics is an SGI NUMA system that enables applications to simply and efficiently perform basic, thread-level, shared-memory parallelism. This allows an application to avoid the arduous task (and lower bandwidth and performance) of writing to a cluster and focus instead on solving the problem at hand.

### 4.2 Per-Pipe Display and Texture ID Management

An application must also provide for some of the inherent difficulties of writing to multiple graphics adapters. Specifically, because each independent InfinitePerformance pipeline is really a separate pipeline, an application cannot assume that the same texture and display list IDs will be available on each independent InfinitePerformance pipeline. This means that each texture and display list must be generated and downloaded to each independent InfinitePerformance pipeline. The good news is that most multipipe applications have already encountered this issue and built structure to manage the complexity. However, for those developers who have not, it is easy to do this with a set of structures as outlined below.

An example of some C++ pseudocode to manage a list of unique IDs per pipe follows. First, use the structure below to manage associations of texture or display list names to particular IDs:

```
using namespace std;
typedef map< string tex_or_dl_name,
        GLuint id > name_id_map;
```

And use this structure to manage pipe name (or other ID if you prefer, use perhaps the full display ID such as ":0.0," ":0.1," etc.) to above ID mapping:

```
typedef map< string sub_pipe_name,
        name_id_map > pipe_map;
```

The combination of these two structures makes it simple to manage the multipipe texture issues. For example, the following pseudocode allows the developer to allocate texture IDs for multiple pipes:

```
int texid;

// this is to make ":0.0" current
pseudo_glxMakeCurrent( this_pipe );
glGenTextures( 1, \&texid );
pipe_map[":0.0"]["blue_marble.rgb"]
        = texid;

// this is to make ":1.0" current
pseudo_glxMakeCurrent( other_pipe );
glGenTextures( 1, \&texid );
pipe_map[":1.0"]["blue_marble.rgb"]
        = texid;
```

This code can then be used later to retrieve the ID and can be used in another context, such as to download textures to each pipe:

```
// this is to make ":0.0" current
pseudo_glxMakeCurrent( this_pipe );
glTexImage2D(
  pipe_map[ ":0.0" ][ "blue_marble.rgb" ],
  0, GL_RGB, 256, 256,
  GL_RGB, GL_BYTE, pixels );

// this is to make ":1.0" current
pseudo_glxMakeCurrent( other_pipe );
glTexImage2D(
  pipe_map[ ":1.0" ][ "blue_marble.rgb" ],
  0, GL_RGB, 256, 256,
  GL_RGB, GL_BYTE, pixels );
```

The point of the above code is to illustrate that a simple layered indirection scheme makes trivial the seemingly complex task of managing potentially unique texture and display list ids across multiple 4-way composited pipes and independent InfinitePerformance pipelines.

### 4.3 Compositing Issues

Once you've got the desired data rendering on each independent pipeline, the next task becomes how to composite them together effectively. The SGI Onyx 3000 series with InfinitePerformance graphics offers a diverse set of compositing operations, from simple linear stripes, horizontal or vertical, to combined horizontal and vertical splits, or any combination of the above. Fig. 2 shows a few possible compositing modes. Fig. 3 examines in detail one possible compositing arrangement, showing a nonuniform set of regions, aggregated in a 4-way composition through the compositor.

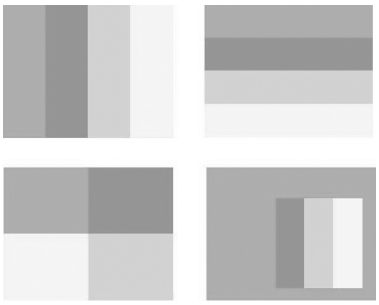

Fig. 2. Four types of possible composited output from the InfinitePerformance compositor

As mentioned in section 3.1, a static compositing arrangement may be chosen. However, there are inherent limits to the scalability of this technique. For example, to achieve good performance, data must be well distributed among the pipes—evenly balanced data scales well. However, in a static compositor arrangement, it's possible for a user of an application to position all of the data within a single independent pipeline, say in the far upper-left corner of the 4-way composited window. This scenario would leave the remaining three pipelines idle. Although fixed compositing arrangements are easy to set up (fire-and-forget), they are much harder to get to perform effectively with dynamic data.

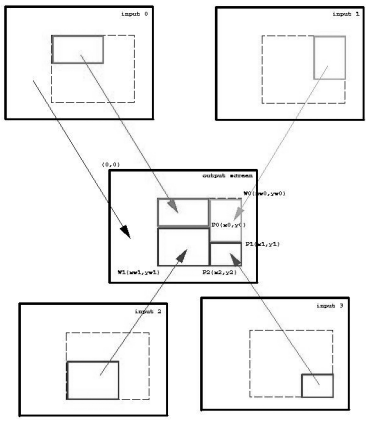

Fig. 3. A detailed example of the regions from independent InfinitePerformance pipes combined in a 4-way composition

A much better way to do compositing is to check how your data is distributed on a per-frame basis and then adjust the compositing regions so that each is doing a roughly equal portion of the work. It is not always possible to adjust on a per-frame basis, and, in fact, owing to the time involved in downloading new configurations through the system, it's not always desirable to do this per frame. However, it is a good idea to adjust the compositing as your data changes, say every N number of frames. It adds a bit more workload to an application to select a good composition, but it is quite easy to get better performance with this technique than to simply use a static composition arrangement. SGI provides a number of Software Development Toolkits that support the ability to dynamically adjust composited arrangements, as detailed in section 4.4.

### 4.4 Compositing APIs

The Onyx 3000 series with InfinitePerformance graphics is fully supported across the SGI graphics software API suite. Several levels of support exist, from low-level control over each independent InfinitePerformance pipeline and the compositor directly, to higher-level APIs managing window control and context management, to still higher-level APIs managing the entire scenegraph and all of the previous tasks as well. All the APIs referenced above are shipping with SGI systems today.

The highest level API that SGI produces that supports all of our graphics systems is OpenGL Performer™. OpenGL Performer is a scenegraph that will manage

multiprocessing, parallel graphics rendering, view-frustrum culling, and state-sorting, and will provide a host of other performance tools to ensure that your data renders as fast as possible on whatever system you're using. See the OpenGL Performer Web site for details about this API. OpenGL Performer fully supports the SGI Onyx 3000 series with InfinitePerformance graphics.

The medium-level API to use with the Onyx 3000 series with InfinitePerformance graphics is OpenGL Multipipe™ SDK. Already in use at numerous software vendors around the world, OpenGL Multipipe SDK takes care of the drudgery of multipipe management and the difficulties of parallel programming while still allowing an application to use its existing scenegraph. OpenGL Multipipe SDK fully supports the Onyx 3000 series with InfinitePerformance graphics.

The lowest-level API to use with the Onyx 3000 series with InfinitePerformance graphics is known as the GLX™ Hyperpipe API. This API has existed for nearly four years and is specifically designed to support dynamic reconfiguration of pipes in logical groupings. This is precisely what is needed to fully control the Onyx 3000 series with InfinitePerformance graphics.

### 4.5 GLX Hyperpipe

GLX Hyperpipe is an API that allows a developer to fully control the Onyx 3000 series with InfinitePerformance graphics scalable graphics compositor and the regions being composited from the V-bricks. GLX Hyperpipe (or more simply, hyperpipe) is a simple API, yet it allows some very powerful operations. The API consists of a number of calls that allow graphics hardware and software to interact. Quoting the "man hyperpipe" man page:

"This extension provides a means for configuring and managing a group of rendering pipes that work together to produce a single display. Typically, a hyperpipe application will be multithreaded, with one thread per pipe; each thread needs to create its own rendering context. The hyperpipe extension allows these rendering threads to communicate with the hardware."

The API allows a user to:

- Determine the physical configuration of a hyperpipe network
- Configure a hyperpipe
- Manage GLXSwapBuffers correctly
- Redirect resize parameters

The hyperpipe API is the definitive SDK if you have an application that can't use one of the higher-level toolkits.

## 5.0 Scalable Software

The hardest part of any application is scalability. Although the Onyx 3000 series with InfinitePerformance graphics performs one extremely difficult part of the scalable graphics process—the compositing—a large part of this difficult task is still shouldered by the developer. Classes and books have been written on the subject, but at the end of the day, developing scalable software is a difficult task. However, SGI provides proven software APIs to make that process much easier. The SDKs produced by SGI that are discussed in section 4.4 are a good starting point, and the SGI Developer Forum provides another excellent opportunity to learn about the details of scalable software. SGI has been enabling developers to write the most scalable applications for more than 20 years, through software and hardware, and the InfinitePerformance graphics system continues that great tradition.

## 6.0 Conclusion

The Onyx 3000 series with Infinite Performance graphics is the latest high-performance graphics supercomputer from SGI. Combining state-of-the-art graphics subsystems, zero-latency compositing, and industry-standard graphics APIs, the Onyx 3000 series with InfinitePerformance graphics provides a completely scalable graphics hardware environment. With careful attention to culling and multiprocessing details, the Onyx 3000 series with InfinitePerformance graphics can scale to address the largest problems. The Onyx 3000 series with InfinitePerformance graphics is the best choice for developers and customers who need the absolute standard in graphics scalability and performance.

References


SGI Origin 3000 Series. *www.sgi.com/origin/3000/*

SGI VPro Graphics. *www.sgi.com/workstations/octane2/graphics.html*.

SGI Onyx 3000 Series Systems with InfinitePerformance Graphics. 2002.

SGI InfinitePerformance: Scalable Graphics User's Guide. 2002.

SGI Onyx 3000 Series Systems with InfiniteReality Graphics. *www.sgi.com/onyx3000/*

SGI® Reality Center™. *www.sgi.com/realitycenter/*

SGI OpenGL Performer. *www.sgi.com/software/*

SGI OpenGL Multipipe SDK. *www.sgi.com/software/*

Cok, K., A. Commike, B. Kuehne, T. True. Developing Efficient Graphics Software. SIGGRAPH 1999 Course Notes (1999).

SGI Developer Forum. *www.sgi.com/developers/*, 2002.



**Corporate Office**
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
(650) 960-1980
**www.sgi.com**

North America 1(800) 800-7441
Latin America (52) 5267-1387
Europe (44) 118.925.75.00
Japan (81) 3.5488.1811
Asia Pacific (65) 771.0290





3213 (1/02)

J13291