Introduction to Portable Programming with CGE PEX 5.1 Extensions 1

Purpose and Scope of this Guide

The purpose of this guide is to assist you in creating highly portable 3D graphic applications on platforms supporting the Common Open Software Environment. The programming interface that you will use for this purpose is known as the Common Graphics Environment PEX 5.1 Extensions (CGE PEX 5.1). Using this interface, you can create applications that are highly portable and interoperable on platforms that support CGE PEX 5.1. By following this guide, you should have success creating applications that meet this goal. You will find that the graphic portions of your application will compile effortlessly on all CGE PEX 5.1 platforms, eliminating the need for vendor specific *drivers* or code paths. You will also find that the graphical results on each CGE PEX 5.1 platform will be more complete and consistent, again reducing the need for vendor specific code paths.

This guide is written specifically for application developers who want to create portable and interoperable applications for the CGE PEX 5.1 platforms. This guide is not a PEX or a PEXlib tutorial. There are many other references available that serve as better and more general tutorial material. See Appendix A, “Other Useful References.” This guide is derived from the Common Graphics Environment PEX 5.1 Extensions specification which you

1

may obtain from your workstation vendor or from the X Consortium.[1] Or perhaps your vendor has integrated a version of this information with their PEX product.

While you may find some useful techniques in this guide that apply to creating portable vanilla PEX 5.1 applications, this guide does not specifically address this more general goal. Also, this guide does not include a tutorial on the new CGE PEX 5.1 functionality, specifically texture-mapping. It explains how to add texture mapping to your application so that it will work on all CGE PEX 5.1 platforms. Please refer to product documentation and other literature listed in Appendix A, "Other Useful References" to learn the fundamentals of features such as texture mapping.

This guide will help you interpret the basic CGE PEX 5.1 specifications, especially the more subtle, yet useful, information that is crucial to your application's success. You will also find a large number of hints and techniques that are the result of a cumulative total of years of experience in creating portable graphics applications in multi-vendor environments. PEX application programming is still in its infancy stage, so many of the solutions to common problems and difficulties have not yet disseminated through the industry. This guide contains the solutions to many of these problems and will help you put your PEX application well ahead of your competition's.

Motivation to Use CGE PEX 5.1 Extensions

We hope that you have chosen to create your next application with CGE PEX 5.1 for one or many of the following reasons:

Increased Portability

The CGE PEX 5.1 vendors are committed to supplying you with a CGE PEX 5.1 programming library that is consistent across all CGE PEX 5.1 platforms. There are no *missing* or *conflicting* function calls on any platform and all existing functions are implemented identically. In fact, some of the same code has been

1. You can obtain the specification from the X Consortium's FTP (File Transfer Protocol) server. The server's address is ftp.x.org. Login as "anonymous", giving your Internet address as the password. The specification can be found in: /contrib/PEXlib in the file CGE-PEX-51-Spec-V1.txt.Z.

shared by the vendors to implement the CGE PEX 5.1 library, so you might be using products on different vendors' platforms that were derived from the same source code base!

This promise of portability reduces the need for you to worry about programming graphics, an area that has historically been riddled with vendor extensions and incompatibilities.

Increased Interoperability

Your CGE PEX 5.1 application interoperates with another vendor's CGE PEX 5.1 server with no difficulty. This allows your application to execute on one machine, while interacting with a user running a CGE PEX 5.1 server on another workstation, even if the workstations come from different vendors. Interoperability gives your customers with large heterogeneous installations the ability to host or port applications onto fewer types of host machines, while still making them available for use from a wide variety of workstations.

Early Access to Some PEX 5.2 functionality

One of the most exciting benefits about CGE PEX 5.1 is that it provides you with the earliest consistent multi-vendor implementation and includes some of the functionality slated for PEX 5.2. The early availability of these functions will help you put more powerful applications in the hands of your customers without you having to resort to differing vendor-specific PEX extensions or switching to another graphics support package.

- Texture Mapping - This is the biggest and most exciting new feature in CGE PEX 5.1. Texture Mapping allows you to project pixmap-like images onto various PEX primitives, giving them appearances not possible with standard PEX.
- Anti-Aliasing - This allows you to create higher quality pictures by instructing the PEX server to draw specified primitives with a reduced "jaggies" effect. Some PEX vendors have provided this as a vendor extension in PEX 5.1 products. You can now count on it in CGE PEX 5.1.
- Transparency - Although this feature is a part of the PEX 5.1 specification, it was not required. It is now a part of CGE PEX 5.1, and offers you the ability to create objects that transmit light to varying degrees, that objects on the other side may appear clear or sheer, gauzy or diaphanous.

- Drafting Primitives - These are the common and useful circle, ellipse, and arc primitives that many vendors have provided as PEX 5.1 vendor extensions. They are now a common part of CGE PEX 5.1.

Note – It is impossible to guarantee that any of these functions will appear in PEX 5.2 in exactly the same way they appear in CGE PEX 5.1, because PEX 5.2 was not in a final state at the time CGE PEX 5.1 developed. However, the creators of CGE PEX 5.1 have done everything possible to follow PEX 5.2 directions and keep CGE PEX 5.1 as close to those directions as possible.

Higher Level of PEX 5.1 Functionality

One of the criticisms of PEX 5.1 is that too many features were left optional and that too few vendors implemented these optional functions. CGE PEX 5.1 raises the level of required implementation, giving you a graphics support system with a higher level of guaranteed functionality. This means that your application will have to make fewer queries for optional function. Fewer optional features means fewer possible configurations your application will have to handle with separate code paths.

Relationship of CGE PEX 5.1 to PEX 5.1

So, enough of the sales pitch. You're convinced, right? Now, what does all this mean to you if you are currently using PEX 5.1?

CGE PEX 5.1 Extensions As a Superset of PEX 5.1

Well, the name sure says it, but what do we really mean?

CGE PEX 5.1 is a set of extensions built on top of PEX 5.1 using the X Consortium standard rules and interoperability conventions for extending PEX 5.1. (See Chapter 8, "Interoperability Conventions"). These rules and guidelines are engineered to allow PEX 5.1 extensions to exist without affecting the operation of any core PEX 5.1 functions. This means that if you were to suddenly upgrade your graphics support software to CGE PEX 5.1, your PEX 5.1 applications would still work without modification.

If you want to take advantage of new CGE PEX 5.1 function (for example, texture mapping), you'll need to modify your application to make use of this function. Note that some of the PEX 5.1 optional function that is now required in CGE PEX 5.1 (for example, transparency) might suddenly start to work in your application.

CGE PEX 5.1 Programs in 5.1 Environments

If you add a CGE PEX 5.1 function, like texture mapping, to your application, your application may not work with a vanilla PEX 5.1 server. This is because the vanilla 5.1 server will not understand the CGE PEX 5.1 extensions and will probably return errors.

You'll need to decide if you want your application to continue to work with vanilla PEX 5.1 servers. If so, you'll have to add logic to your application to avoid the CGE PEX 5.1 functions when connected to a PEX 5.1 server.

Relationship of CGE PEX 5.1 to PEX 5.2

While moving your application to CGE PEX 5.1 gives you early access to PEX 5.2 function, it does not guarantee that your CGE PEX 5.1 program will port without effort to PEXlib 5.2. Since PEXlib 5.2 will be backwards compatible with PEXlib 5.1, the non-CGE portions of your application should port with no effort. However, the CGE-specific portions may need some tweaking to work with PEXlib 5.2.

At this stage, PEXlib 5.2 is still not completely defined, so the CGE vendors cannot predict all the changes that you will have to make. The intention is for changes to be limited to minor syntactical changes, but CGE vendors cannot guarantee this will be the case. You may choose to make the necessary changes when you upgrade your application from CGE PEX 5.1 to PEXlib 5.2. You can then make any CGE->5.2 changes as well as the changes required for your application to take advantage of any PEX 5.2 function not found in CGE PEX 5.1. This situation is illustrated in Figure 1-1 on page 6.

1

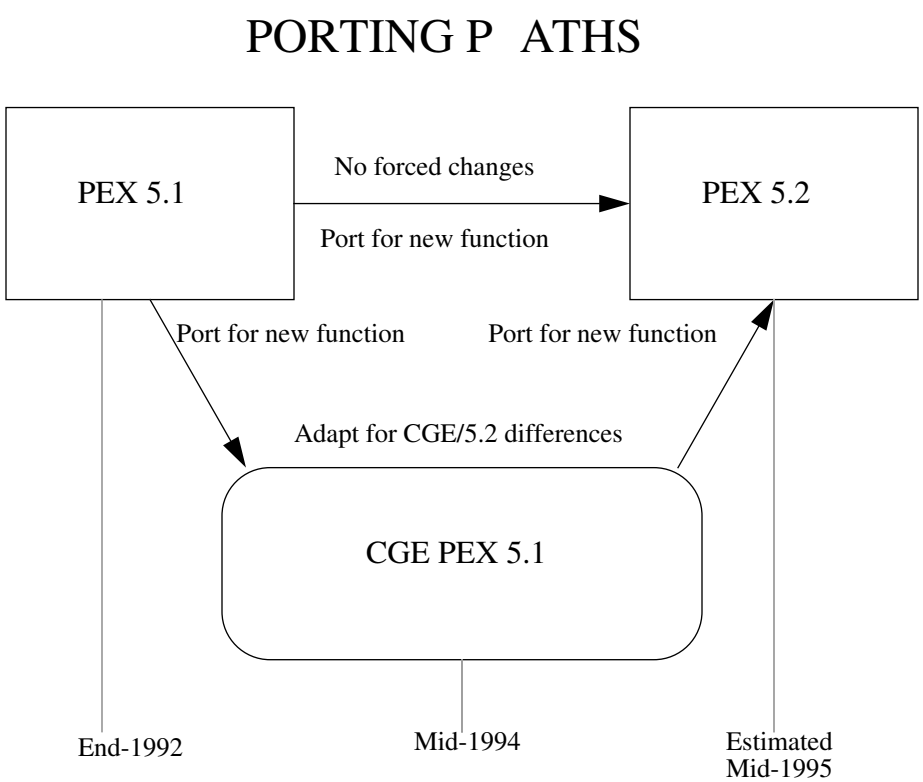

Figure 1-1 Porting Paths

CGE PEX 5.1 Extension Mechanics

This section explains how PEX 5.1 was extended to obtain CGE PEX 5.1. Skip this section if this sort of detail does not interest you.

CGE PEX 5.1 Follows PEX 5.1 Interoperability Conventions

See Chapter 8, "Interoperability Conventions" for more information. This information is also useful for general PEX 5.1 applications.

Vendor ID

To avoid conflicts between vendors adding extensions to PEX 5.1, the X Consortium assigns an 8-bit vendor ID to vendors wishing to implement extensions. The vendor ID is used in the protocol encoding of the extension to distinguish the extension protocol from another vendor's extensions.

The vendor ID 0x10 is IBM's vendor ID, but is being used by all vendors designing or implementing CGE PEX 5.1. CGE is using IBM's vendor ID because it has the least amount of conflict with existing IBM extensions. IBM ensures that other IBM extensions do not conflict with the CGE PEX 5.1 extensions.

According to the PEX 5.1 Interoperability Conventions, 16-bit quantities, such as Output Command types and Enumerated Type indices, are encoded with values starting at 0x9000, while 32-bit quantities like Escape opcodes start at 0x80100000.

New Output Commands Encoded As Vendor-Specific Output Commands

CGE PEX 5.1 specifies new Output Commands (OCs) as vendor-specific OCs, setting the high bit on and using the vendor ID for the next 7 bits. This is in contrast to the current 5.2 direction of using GDPs and GSEs for extension OCs. Besides inquiry, you should see little functional difference between the two methods. You can query the existence of the new OCs by getting the CGE PEX 5.1 enumerated type information for CGE PEX 5.1 output commands.

1

New Requests Encoded As PEXEscape and PEXEscapeWithReply

The new CGE PEX 5.1 requests are converted by PEXlib to PEXEscape or PEXEscapeWithReply requests, as this is the only way to add requests to PEX 5.1.

Extended Renderer Attributes and Extended Pipeline Context Attributes Accessed with New Requests

CGE PEX 5.1 adds several attributes to the renderer and the pipeline context. Due to protocol restrictions, you cannot access these new attributes with the PEX 5.1 requests. Therefore, CGE PEX 5.1 provides requests to change and get these extended attributes.

You cannot create either of these resources with the extended attributes; you must create them with PEX 5.1 requests and then later change them with the new CGE PEX 5.1 requests (escapes).