

3D Solutions For Linux



- SGI Sample Implementation (SI)
- NVIDIA
- Xi Graphics
- Metrolink
- Workstation Vendors



SGI Sample Implementation (SI)



- Sample implementation used by SGI's OpenGL licensees
- Mature (in development since 1992)
- Open source in January 2000, includes
 - OpenGL 1.2
 - GLX 1.3
 - GLU 1.3
 - utilities & man pages



SGI Sample Implementation (SI)



- Open source SI, means SI-based IHV drivers can be open sourced
- Currently builds against XFree86 3.3.6
 - with indirect rendering
- GLX implementation part of XFree86 4.0
- GLU implementation used in Mesa
- Read-only cvs tree access + tarball
- oss.sgi.com/projects/ogl-sample



NVIDIA



- **Proprietary XFree86 4.0 drivers for OpenGL 1.2**
 - collaboration with SGI and VA Linux
- **Supports TNT*, GeForce, Quadro**
 - direct rendering, AGP 4X, NVIDIA extensions
- **Common code base with MS Windows drivers**
 - well tested



NVIDIA



- **Available from NVIDIA's web site**
 - Driver binaries for RedHat 6.1, 6.2 (build .94)
 - XFree86 4.0 X driver + GLX module
 - loadable kernel module
 - client-side GLX, OpenGL, GLU
 - kernel module comes with some source code
 - recompile against different kernel versions
- **XFree86 4.0 binaries from RedHat**
 - <ftp://rawhide.redhat.com/rawhide/i386/RedHat/RPMS/>



NVIDIA



- **Installation and troubleshooting FAQ available on NVIDIA's website**
- **Known bugs**
 - glXUseXFont + XFree86 4.0 xlib = core dump
 - Some SMP driver lock-ups
 - dlopen() interactions, single head only, ...
- **Configuration options**
 - lock to vertical sync, full screen AA



NVIDIA



- Open source driver for NV1, RIVA, TNT, GeForce
- XFree86 3.3.6
- Lower performance than proprietary driver
- www.nvidia.com/Products/Drivers.nsf



Xi Graphics 3D Accelerated-X



- Proprietary X server and OpenGL 1.2 drivers
- Support for many architectures
 - ATI, Matrox, 3Dlabs, Number9, 3dfx, S3
 - Laptops, multiple heads
- Entertainment & professional versions of the product
- www.xig.com



Metrolink



- Proprietary X server and OpenGL 1.2 drivers
- Broad support
 - ATI, NVIDIA, S3, 3Dlabs, Matrox, E&S
- Indirect rendering, little hardware acceleration
- www.metrolink.com



Workstation Vendors



• SGI 230/330/550 VPro series

- X86 w/ RedHat 6.1 + SGI driver/kernel overlay
- V3 and VR3 graphics
 - enhanced versions of NVIDIA GeForce & Quadro
- Enhanced version of NVIDIA/SGI driver
- 230, 330 shipping now
- www.sgi.com/workstations/230/



Workstation Vendors



• HP Visualize series

- currently X86-based, NVIDIA TNT2 graphics
- RedHat 6.2
- Beta version of X86-based, visualize-fx⁵, fx¹⁰
- Proprietary X server and open source kernel module to support direct rendering
- www.hp.com/visualize/products/linux/



Completeness & Conformance Issues for OpenGL & Mesa



- Conformance and Testing
- ABIs and Versioning
- Linux OpenGL Base Standard (oglbase)
- Extensions



Conformance and Testing



• OpenGL ARB conformance test

- 'must pass' test plus other tests
- must pass to call implementation OpenGL
 - not comprehensive, nor very strict (sanity test)
- distributed to OpenGL licensees by SGI
- SGI SI and Mesa pass most tests
 - doesn't necessarily mean SI-based IHV drivers pass



Conformance and Testing



• Vendor specific tests

- Microsoft WHQL tests - includes conformance
- ogst - SGI proprietary, but some licensees have it as well
- other vendor proprietary tests (3dfx, NVIDIA, ...)



Conformance and Testing



• Open source - glean

- test quality and performance
- C++ code
- can do side by side comparisons
- if you find a driver bug, send test case to glean team and it will be added to glean
- glean.sourceforge.org



ABIs and Versioning



- **OpenGL specification doesn't cover everything**

- covers names of symbols, tokens, behavior of function calls, versioning
- doesn't cover locations of header files, libraries, file names
 - /usr/lib/libGL.so -or- /usr/X11R6/lib/libGL.so, etc.

- **To release platform-independent apps, need everything standardized**



Linux/OpenGL Base Standard (oglbse)



- **OpenGL Application Binary Interface for Linux**
- **Like Linux Base Standard, but covers OpenGL runtime and sdk**
- **Representation from OpenGL providers and app vendors**
- **First version (1.0) last spring**



Linux/OpenGL Base Standard



- **For application deployment, specifies library packaging**

- data type mappings for IA32 (e.g., GLint == int)
- /usr/lib/libGL.so, libGL.so.1 - GLX + GL entry points
- /usr/lib/libGLU.so, libGLU.so.1 - GLU entry points
- the '.1' is for version 1.0 of the ABI, not OpenGL 1.0
- entry points for OpenGL 1.2, GLX 1.3, GLU 1.3
- thread-safe (pthreads)



Linux/OpenGL Base Standard



- **For SDK users, also includes locations and contents of header files**

- /usr/include/GL/glx.h - GLX 1.3
- /usr/include/GL/gl.h - OpenGL 1.2
- **GL_OGLBASE_VERSION**
- /usr/include/GL/glu.h - GLU 1.3
- /usr/include/GL/glex.h - OpenGL extensions
- /usr/include/GL/glxext.h - GLX extensions



Linux/OpenGL Base Standard



- **Recently released, not widely deployed**

- oss.sgi.com/projects/ogl-sample/ABI/

- **Incorporated into XFree86 4.01**

- **Unlikely to be compatible with existing OpenGL distributions**



OpenGL Versioning



- **Multiple versions of OpenGL specification (1.0, 1.1, 1.2)**

- **All are backward compatible**

- **Both compile-time and run-time version queries**

- `GL_VERSION_1_2`
- `glGetString(GL_VERSION)`
 - e.g., 1.2.0 vendor_specific_info



OpenGL Versioning



- **oglbase ABI includes symbols for OpenGL 1.2**

- Vendor may only ship 1.1 or 1.0 functionality
- so apps should perform a run-time check
 - adapt to lesser functionality - or -
 - abort

- **Other Unix vendors may adopt similar rules**



OpenGL Versioning



- **Example**

```
char* vers = glGetString(GL_VERSION);
int v12 = strncmp(vers, "1.2", 3) == 0;

#ifdef GL_VERSION_1_2
if(v12){
    glTexImage3D(...);
} else
#endif
{
    /* non 1.2 dependent code */
}
```



Extensions



- **Method for adding new functionality to OpenGL**

- ARB_multitexture, texenv_combine, ...

- **Important extensions become 'ARB' extensions**

- **Important extensions become part of base-line functionality**

- **Extensions are optional**



Extensions



• Interesting Extensions

- GL_ARB_multitexture
- GL_ARB_texture_env_add
- GL_ARB_multisample
- GL_ARB_texture_compression
- GLX_ARB_get_proc_address
- GL_EXT_blend_color
- GL_EXT_blend_subtract



Extensions



- **Only ARB extensions are included in the OpenGL specification**
- **ARB does provide rules for defining an extension**
 - manages registry of extension specs
 - oss.sgi.com/projects/ogl-sample/registry/
 - assigns token values, function names (avoids collisions)



Extensions



- **Portable applications (binary) may work with or without a particular extension**
 - reduce number of versions of application
- **Problem occurs when**
 - application is linked against extension entry point
 - target platform OpenGL doesn't include extension
 - unresolved symbols at run-time



Extensions



• Example

```
char* ext = glGetString(GL_EXTENSIONS);
multitex = strstr(ext, "ARB_multitexture" )!= NULL

#ifdef GL_ARB_multitexture
if (multitex){
    glActiveTextureARB(GL_TEXTURE1_ARB);
    ...
} else
#endif
{
    /* multi-pass texture code */
}
```



Extensions



- Solution, use soft references to symbols
- GLX_ARB_get_proc_address provides run-time queries
 - similar to dlsym()
- Application does run-time query for extension
- If present, does symbol lookup and uses it
- If not, use alternate code path



Extensions



• GLX_ARB_get_proc_address

```
char* ext = glGetString(GL_EXTENSIONS);
multitex = strstr(ext, "ARB_multitexture" )!= NULL

#ifdef GL_ARB_multitexture
if (multitex){
    PFNGLACTIVETEXTUREARBPROC glActiveTextureARBp =
        glXGetProcAddressARB("glActiveTextureARB");
    (*glActiveTextureARBp)(GL_TEXTURE1_ARB);
} else
#endif
{
    /* multi-pass texture code */
}
```



Extensions



- **GLX_get_proc_address** part of oglbase ABI
- **Similar functionality in MS Windows**
 - WGL_get_proc_address
 - WGL version context dependent, oglbase not
- **Likely to be supported by other Unix vendors**



Extensions for SDK Users



- **Problem with header files**
- **SGL maintains glxext.h and glxext.h as central definitions**
- **Idiotic MS** `PFN<unreadable_function_name>` typedefs included
 - for use with WGL/GLX_get_proc_addr
- **Include automatically in gl.h and glx.h**
 - unless `GL_GLEXT_LEGACY, GL_GLXEXT_LEGACY` defined



Extensions for SDK Users



- `GL_GLEXT_PROTOTYPES` to control whether function prototypes defined
- **Developers can download latest and greatest**
 - <http://oss.sgi.com/projects/ogl-sample/ABI/glxext.h>
- **Vendors may still have extra header files**
 - Proprietary extensions



Futures



- **Direct Rendering Infrastructure**
- **OpenGL Directions**
- **oglbase Directions**
- **Conformance**



Direct Rendering Infrastructure



- **Support more chips**
- **Port to other processors, OSes**
- **Track OpenGL evolution**
- **Performance tune**
- **Accommodate more vendor-specific implementations**
 - Avoid having vendors replace libGL.so with their own version



OpenGL Direction



- **Standardization of more ARB extensions**
- **Upgrades to the SI**
 - add ARB extensions
 - XFree86 4.0
 - rpm packages for GLU, man pages, etc
 - optimized geometry code
- **New code**
 - GLS, GLC, etc



oglbase



- **Resolve outstanding issues**
 - Definitions for other hardware (PowerPC, Alpha, ...)
- **Integrate into Linux Base standard**
- **Track ARB extensions and consider other extension symbols for inclusion in the ABI**



glean



- **Add more tests**
- **Build repository of test results for different platforms**



GLsetup



- **Version of Glsetup for Linux?**
- **www.glsetup.com**


