3D*labs*®

GLINT® & PERMEDIA® 2

Software Release Note for Windows NT 4.0 Display & OpenGL Drivers

Release 2.11-1.1.24

Email: info@3dlabs.com
Web: http://www.3dlabs.com

3D*labs Inc.*
181 Metro Drive, Suite 520
San Jose, CA 95110
United States

Tel: (408) 436 3455
Fax: (408) 436 3458

3D*labs Ltd.*
Meadlake Place
Thorpe Lea Road, Egham
Surrey, TW20 8HE
United Kingdom

Tel: +44 (0) 1784 470555
Fax: +44 (0) 1784 470699

CONTENTS

1 Introduction

This note describes the Windows NT 4.0 Display Driver and OpenGL Installable Client Driver for GLINT 500TX & MX + DELTA boards, GLINT GMX, and PERMEDIA 2 boards. It also explains how to install these drivers.

Note: This document should be read in conjunction with the README.TXT file on the installation floppy, which contains details of any enhancements and/or bug fixes that have been made subsequent to these release notes being written.

1.1 Release Identity

Once the driver has been installed and the machine rebooted, the display driver release number can be determined by starting the Display Applet Control Panel in Control Panels directory.

Select the "3Dlabs" tab and click on the “Information” button. The Display Driver and OpenGL version numbers can be found in the Software section on this page.

1.2 Prerequisites

- Windows NT 4.0 (Build No 1381), Service Pack 3 is recommended. In fact SP3 is necessary for AGP cards to work and for DirectDraw to be hardware accelerated on PERMEDIA 2 based boards.
- Intel 486 processor or later or DEC Alpha.
- GLINT 500TX & MX + DELTA board, GLINT GMX board or PERMEDIA 2 board.

2 Installation

Before installing the software, power down the machine and install the GLINT or PERMEDIA board as per the hardware installation instructions. Boot the machine using the non-VGA boot option (new display drivers cannot be installed when the machine has been booted with the VGA boot option). Once booted and you have logged in as an Administrator, perform the following steps:

- Open the Display Control Panel in the Control Panels directory and click on the “Settings” tab.
- Press the “Display Type…” button. A new window titled "Display Type" will appear.
- Press the "Change..." button in this window. A window titled "Change Display" will appear.
- Press the "Have Disk ..." button in this window. A window titled "Install from Disk" will appear.
- Specify the path A:\. Insert the release floppy for your machine architecture into the drive and press OK. The "Change Display" window will appear with a list containing a single entry which says “3Dlabs Compatible Adapter”, select this item and press "OK".
- Then follow the instructions and quit the control panel applet. When asked if you want to restart the machine press "Yes".

Note: There are no options to select a given resolution at install time. When the machine reboots Windows NT 4.0 allows the video mode to be dynamically changed without the need for a reboot.

The machine will now shutdown. On restart again choose the non-VGA boot option. It will restart using the 3Dlabs board as the display device. This can be checked by opening the "Display" applet again and pressing the "Change Display Type..." button. The "Display Type" window should report that it is running on a GLINT or PERMEDIA display board.

To change the desired resolution, depth and frequency then open the Display Control Panel and select the required resolution, color depth and monitor frequency. This selected mode can be tested to ensure that it can be handled by the monitor. On some double buffered applications the higher refresh rate allows higher frame rates to be achieved. The display will change dynamically.

The above procedure installs the Windows NT display driver, GLINT control panel applet and the OpenGL installable client driver. Once the display resolution and pixel depth have been appropriately re-configured the machine is ready to run both Windows NT and OpenGL applications.

3 Resolutions and Color Depths

A full list of all modes is available via the Display Control Panel applet once the 3Dlabs driver has been installed and the system rebooted. Choose the "List all Modes" option to get this list.

4 3D Graphics & Double Buffering

The display driver contains an extension to allow 3D applications, and the OpenGL installable client driver (ICD), to drive the GLINT or PERMEDIA hardware. To provide a double buffering capability for these 3D applications the display driver provides the following features:

A screen-sized off-screen buffer is configured if the "DoubleBuffer.NumberOfBuffers" registry variable is ≥ 2 (see below). This buffer is used in 256, 32768 and True Color modes to provide BitBlt double buffering. The off-screen buffer is also used to provide full screen hardware double buffering if an application window covers the whole screen. Note that to provide double buffered accelerated capability there must be enough memory to contain two frame buffers. This may not be the case at high resolutions. Also note that on a PERMEDIA board the memory is unified and the frame buffer memory, depth buffer memory and texture memory all resides in the same place. On GLINT boards the video memory is separate from the depth buffer/ texture memory.

If a screen resolution is selected which cannot support double buffered OpenGL applications, then a warning message will be displayed in the 3Dlabs control panel.

4.1 Full Screen Double Buffering (All Boards)

If an application window covers the whole screen, the display driver will automatically switch to use a hardware double buffer mechanism, which can have a significant performance benefit. This mechanism will not be available to an application that has more than a small window border at the top of the screen. It will also be unavailable if, for example, a floating task bar (common on Windows NT 4.0) is at any edge other than the top of the screen, since the display driver will check and find that the application window does not cover the whole screen.

4.2 Full Flip Double Buffering (GLINT Boards Only)

Flip double buffering is available on GLINT boards. The flip double buffering is available for applications that run in a single window and is facilitated by all of the desktop rendering being written to both frame buffers. Note that when this is available, only the first double buffered window will use this approach, when there are multiple double buffered windows, blit double buffering is reverted to.

4.3 Blit Buffering (All Boards)

Blit double buffering is the simplest form and simply involves copying the contents of the back buffer into the displayed buffer.

Note that for the given configuration, the OpenGL ICD will always try and use the fastest double buffering method that it can. Also note that if the selected screen resolution/color depth is too high, then double buffered applications may start to run slowly, as they revert to using the Microsoft Generic OpenGL renderer.

5 3Dlabs Display Control Panel Tab

Some of the registry variables, detailed in the next section, can be conveniently changed via the 3Dlabs tab in the Display Control Panel. This tab allows both boot-time and run-time control over the configuration of OpenGL and other applications using the 3Dlabs display driver. Options that are not applicable to the currently installed graphics board will not be visible. The control panel is split into a number of pages as listed below. The pages are selected by clicking on one of the 3 configuration buttons at the top right of the window.

Note: It is necessary to have administrator privileges to change any settings in the control panel applet. If you do not have administrator privileges the options will be grayed out and cannot be changed.

5.1 Information Page – 2D Driver Build

For information only. This reports 4.0x.00.xxxx-xxxx corresponding to the File and Product Version information required for Window Hardware Quality Labs compliance under Windows95. The vendor is free to use the last four digits for their own use - in this case the daily build number that uniquely identifies the display driver binary.

5.2 Information Page – OpenGL ICD Build

For information only. This reports 4.0x.00.xxxx-xxxx corresponding to the File and Product Version information required for Window Hardware Quality Labs compliance under Windows95 (the same OpenGL ICD binary runs under both WindowsNT and 95). The vendor is free to use the last four digits for their own use - in this case the daily build number that uniquely identifies the Installable Client Driver (ICD) binary.

5.3 Information Page – OpenGL Release

For information only. This is the OpenGL version number that can also be queried by an OpenGL application at run-time. Applications will do this to determine what functionality they can expect to be provided by the implementation. Full OpenGL version 1.1 is supported by 3Dlabs' ICD. This release reports 1.1.23, where the last two digits identify the ICD internal release number (increments with every release).

5.4 Information Page – Chip Name

For information only. This reports the name of the 3Dlabs chipset on the board, such as 3Dlabs PERMEDIA 2 or 3Dlabs GMX 2000.

5.5 Information Page - Depth/Stencil Buffer

For information only. This reports the total amount of memory used by auxillary buffers for use in hardware hidden surface removal (z-buffering) and masking/clipping (stencil and GID planes).

5.6 Information Page - Texture Memory

For information only. This reports the total amount of graphics card memory available for storing texture maps - after allowing for full screen front, back and depth buffers. Smaller desktop colour depths and/or screen resolutions will free up more graphics card memory for texture use increasing the amount available for non-swappable textures (refer to the notes below on the texture memory manager).

5.7 Setup Page - Gamma Correction Adjustment

The gamma correction adjustment affects the entire screen display. The default gamma value is 1.0 and the allowable range of floating point values is 0.3 to 4.0.

5.8 Setup Page - Disable PCI Disconnect

Higher 2D graphics performance can be achieved by using PCI Disconnect. However this feature can sometimes adversely affect the performance of other devices, such as modems and sound cards.

Tick this option if you are experiencing problems with the performance of other devices.

5.9 OpenGL Page - SoftImage Version 3.51/3.7 Application support

Version 3.51/3.7of SoftImage requires this to be set to ensure the correct operation on the GLINT boards. Changing this option requires a re-boot of the system.

5.10 OpenGL Page - Number of DMA Sub-Buffers

Each DMA buffer is sub-divided into sub-buffers which are used in conjunction with an Interrupt DMA mechanism to reduce latency in the system. The number of

sub-buffers can be set here, setting it to 2 will disable the interrupt mechanism.

5.11 OpenGL – Wait For VBlank

Smooth animation of 3D applications can be achieved by rendering to an off-screen window/desktop sized colour buffer and copying or swapping the contents to the displayable front buffer at the completion of each frame. Enabling this option prevents tearing of the display by synchronizing the swap of the back and front buffers to the vertical blank retrace interval of the monitor display.

Leave this option unticked if the highest rendering frame rates of a double buffered application are desired (i.e. not locked to sub-multiples of the current display refresh rate).

6 OpenGL Texturing & Extensions

6.1 Efficient use of multiple textures

OpenGL applications that wish to render primitives with multiple texture maps will achieve much higher performance by avoiding the invoking of the different textures in immediate mode. There are two alternative options for efficient switching between multiple textures.

The first and much preferred option is to use the OpenGL texture object functionality. Texture objects are fully editable and may have their images and parameters altered at any time (unlike the use of textures in display lists). Details on texture object functionality is available in the OpenGL 1.1 specification and also in Appendix F in the 3Dlabs OpenGL Extensions document for further details. The performance gain using this approach will benefit performance for both the GLINT and PERMEDIA boards.

The second option is to define each texture (or array of mip map resolutions) within a display list - *with the limitation that only one texture is allowed per display list*. Switching between different textures is then achieved by referencing the appropriate display list. Since display lists are not editable in OpenGL, the OpenGL implementation is able to cache texture data defined within a display list. In effect the display list identifier acts as a texture handle. This caching cannot be performed when a texture is invoked in immediate mode since the application in this case is at liberty to have changed the texture data since any previous reference.

6.2 Texture Memory Cache Management

Texture data is stored in the local buffer memory on the graphics card. The memory available for textures is therefore constrained by the local buffer memory available. It is also constrained by the amount of local buffer memory already consumed for the depth buffer, stencil buffer, etc.. This amount varies according to the current display resolution in use, i.e. there is more memory available for textures when the display resolution (and therefore the size of the depth buffer, stencil buffer) is lowered.

On 3Dlabs OpenGL releases prior to version 1.1.14, if the condition is reached where there is insufficient local buffer memory to load a new texture then the OpenGL texture download will not succeed and will set the error code GL_OUT_OF_MEMORY. Textured primitives that expected to use this texture will not be rendered correctly. To improve on this behavior a scheme for swapping textures to/from system host memory is required. By setting aside a portion of texture memory on the graphics card for use as a texture cache and tracking when a texture switch takes place, textures can be reloaded to the cache as needed from a

copy kept in host memory when the texture was first downloaded. If the requested texture is already present in the cache, then no reload is performed.

Ideally for the greatest flexibility and most efficient use of available texture memory, all textures should be cacheable. However for a software texture cache manager there is a small performance overhead to be paid for this tracking plus any delay in reloading a swapped out texture (as a texture could be swapped out at any time by another OpenGL process). By allowing the user to specify the size of the texture cache through the use of the registry variable OpenGL.MaxTextureSize (as described above), an approximate balance between non-swappable and swappable textures can be made (and hence performance). Thus an application should load any real-time critical textures first as the texture manager will only place textures in cache and/or host memory if a space of sufficient size is not available in the non-swappable area of texture memory.

In order to guarantee all texture requests no matter how large, any texture whose size in texels is greater than the cache size will be silently filtered down to fit in the cache (while preserving aspect ratio). On the GLINT boards the size of the texture cache allows extra room for all lower mipmap level textures including border texels.

6.3 Texture Filter Modes

The default texture minification filtering for OpenGL involves mip-map filtering. This gives good textured rendering quality but at the cost of low performance. Much higher performance can be obtained by changing the default texture filtering such that the minification and magnification filtering modes are the SAME. Setting them to GL_LINEAR gives good quality bilinear filtering and improved performance. Setting both modes to GL_NEAREST will give nearest neighbor filtering and the fastest possible performance.

6.4 BGRA Extension

This extension provides an additional pixel color format for compatibility with the blue, green, red component ordering of Microsoft Windows DIB's (device independent bitmaps). Refer to Appendix E in the 3Dlabs OpenGL Extensions document for further details.

6.5 Palette Texture Extension

GLINT and PERMEDIA 2 both provide direct support for palette textures, where each texel represents an index into an on-chip RGBA (8-bits per component) lookup-table. An OpenGL palette texture extension has been defined by Microsoft which is supported by 3Dlabs OpenGL ICD from release 1.0.11. The supported texel depths depend on the chip. They are as follows:

GLINT GMX and GLINT MX : 1,2,4 and 8 bit texel depths.

GLINT 500TX : 1,2,4 bit texel depths.

PERMEDIA 2: 4 and 8 bit texel depths.

Besides improving texture performance and reducing the memory requirements for storing textures in the local buffer, by repeatedly updating the texture LUT, animation effects such as real-time color cycling are also possible. Refer to Appendices C and D for further details.

If many textures share the same look-up table (LUT), performance gains can be obtained with paletted texture objects by forcing the textures to share the same palette (particularly for 8-bit palette textures on the GLINT MX and PERMEDIA 2). The default behaviour when texture switching through calls to glBindTexture is to send down the LUT on every switch. This can be disabled by the EXT_shared_texture_palette extension described in Appendix I of the OpenGL Extensions document.

6.6 3Dlabs Driver extension

In addition to the extensions mentioned above, the 3Dlabs_DriverState extension has been added and is detailed at Appendix H in the OpenGL Extensions document. This extension is simply a mechanism for adding extra state to the Client Driver and adds extra control to the currently selected context.

6.7 Swap Hint Extension

The Microsoft defined extension GL_WIN_swap_hint is supported on GLINT and PERMEDIA boards. This extension allows the area of a window swapped by the SwapBuffers call to be restricted. This can give performance benefits when only a small area of the display is being updated at a time. For more information please contact Microsoft directly or search in the latest Win32 help files for glAddSwapHintRectWIN.

6.8 Kinetix Buffer Region Extension

This extension was specified and implemented as an optimisation for 3D Studio MAX 2. It can be freely used within other applications that could benefit from it. The extension allows areas of frame, depth and stencil buffer to be stored away and later repaired. 3D Studio MAX 2 uses the extension to optimise display when a single object is being edited within a complex scene. The extension is fully described in Appendix J.

7 3D Studio MAX

This release supports hardware acceleration of both 3D Studio MAX 1 and 2. 3D Studio MAX 1 is accelerated through the Autodesk propriety API, Heidi. 3D Studio MAX 2 is accelerated through OpenGL.

7.1 3D Studio MAX 1.x

Hardware acceleration is provided on MAX 1 through a Heidi driver. This driver, wglint.hdi, comes as standard with the display driver. Once the display drivers have been installed, the Heidi driver must be manually copied to your /3dsmax/drivers directory (or wherever you installed the application). Start MAX and go to the 'File/Preferences' dialog box. Click on the 'Viewports' tab and the select 'Choose Driver'. Ensure that the 'GLINT Hardware' option is selected. Restart MAX.

On PERMEDIA based boards best performance will be obtained when running with 32k colors. On GLINT based boards best performance will be provided when running in TrueColor display modes.

GLINT GMX based boards should be used with OpenGL acceleration under 3D Studio MAX 2. Although the application will run *you will not obtain full acceleration* when running under 3D Studio MAX 1.x with a Heidi driver.

When running MAX to ensure you are getting hardware acceleration go to the 'Help/About 3D Studio MAX' dialog. When running on GLINT hardware the driver section should read 'GLINT Hardware (MP)'.

7.2 3D Studio MAX 2

After installing these drivers start MAX and go to the 'File/Preferences' dialog box. Click on the 'Viewports' tab and the select 'Choose Driver'. Ensure that the 'OpenGL' option is selected. Shutdown MAX. Before restarting MAX, delete the MAX OpenGL configuration file that stores information about your OpenGL driver. Doing this will cause MAX to reset some of its options to get best performance from these drivers. The file is called oglgfx.ini and is located in your /3dsmax2/ directory (or wherever you installed MAX). If the file does not exist then simply restart MAX.

When running with OpenGL you can confirm that you are getting hardware acceleration by viewing the About from the Help menu. The driver box in the top right hand corner of the resulting dialog should read 'OpenGL (3Dlabs v1.1.23)'. If it reads 'OpenGL (Microsoft Corporation v.1.1.0)' then you are running through Microsoft's software OpenGL and should consult the OpenGL section of this document for more information on enabling OpenGL acceleration.

When running through OpenGL, 3D Studio MAX 2 defaults to not backface culling wireframe objects. This means that wireframe objects typically use twice as many lines as they do through a Heidi driver. This can result in poor performance in comparision to a Heidi driver. This behaviour can be changed from within the application. Go to the 'File/Preferences' dialog box. Click on the 'Viewports' tab and then select 'Configure Driver'. Enable the option 'Display Wireframe Objects Using Triangle Strips'. This causes wireframe objects to be backfaced culled and can drastically improve performance. The setting also causes shared edges within polygons to be drawn which may not appear as desired.

For the latest information for using 3D Studio MAX on GLINT & PERMEDIA boards please see the 3D Studio MAX FAQ at http://www.3Dlabs.com.

8 GLINT Event Logging

The 3Dlabs Display Driver registers a number of event log errors and warnings when problems are encountered. The events that can be logged include:

- no DMA support has been configured
- no interrupt driven DMA has been configured
- a non-cache coherent PCI bus has been detected which results in uncached DMA buffers.
- fewer than the required number of DMA buffers have been allocated.

After booting the driver it is advisable to check the system event log to determine the characteristics of your machine. For example, if an event log indicates that interrupt driven DMA has not been configured, this may be because the BIOS has not been configured for PCI interrupts.

To view the system event log, run the Event Viewer from the Administrative Tools program group. From the Log menu ensure that the System Log has been selected. Look for events with the Source type glint. Double click on these events to read the event message.

If no GLINT events are logged then everything is working perfectly. In this case interrupts are working, all DMA buffers have been allocated and the PCI bus is cache coherent.

9 Known Anomalies and Restrictions

9.1 PCI BIOS

Some PCI BIOS's may not assign correct physical addresses to PCI regions. Experience shows that this sometimes happens with the PCI region for the GLINT framebuffer. If this problem does arise, the Windows NT driver will boot but black areas will be seen on the screen. If this happens then a new physical address can be configured for the framebuffer by setting a registry variable. If this variable exists its value will override any address set up by the PCI BIOS.

If having booted the Windsows NT driver, black areas are seen on the screen, try setting this override variable as follows:

- run regedt32
- open the key `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\glint\Device0`
- From the Edit menu choose the "Add New Value" option.
- Set the Value Name to be "`PhysicalAddress.Region2`". Be careful to spell the name exactly as specified – it is case sensitive.
- Set the data type to be "REG_DWORD" and press OK.
- In the DWORD editor window set the physical address value (see below for suggestions) and press OK.
- Check that the entry has been created correctly and reboot the machine.

Selecting physical addresses in this way is an empirical task. An address must be chosen which does not conflict with any other in the system (the PCI address space is 4 GigaBytes in size so there is plenty to choose from). This task should be performed by the PCI BIOS but if it fails to do this the user must choose instead. A useful address to start with for the framebuffer is 0xA0000000. If this fails increment the address in units of 32MB. Another good starting address is 0x40000000. The final address chosen **must** be on a 32MB boundary.

Having created the Region2 registry variable and assigned it a value the machine should be rebooted. Continue modifying the address until the black areas of the screen do not appear. Normally, this will work after the first one or two addresses.

Note, this procedure is required very rarely. Generally, the user will never be required to perform these steps.

9.2 Display Driver

9.2.1 PCI Disconnect

As mentioned earlier, on occasions, the use of PCI Disconnect can cause other devices to suffer from performance problems, tick the 'Disable PCI Disconnect' option in the Control Panel Applet to disable this feature.

9.2.2 Software Cursor Corruption & GMX 2000

Colored cursors that cannot be handled by the RAMDAC on GLINT GMX boards have to be handled by software and when Page-Flip double buffering is enabled the system sometimes reads back the background data from the wrong buffer and the cursor looks incorrect.

9.3 OpenGL

- A mode change after running MultiGen Creator Version 2.0 for the first time can hang the system and/or corrupt title bar text. This is problem is still under investigation.
- The slow redraws of the texture palette in MultiGen Creator Version 2.0 on accelerated hardware is an application issue for which MultiGen will release a fix.
- When a constantly updating OpenGL application is running (such as the X29 or rollercoaster demos) screensavers run very slowly. This appears to be a problem with the task priority allocated to the screensaver process.
- When using the glaux library supplied by Microsoft, specifying that you require alpha planes in the visual is not satisfied by requesting a visual type of AUX_RGBA as opposed to AUX_RGB when calling auxInitDisplayMode(type). In these instances the hardware accelerated visual that will be returned in some modes may not have alpha planes. This is because the display driver exports visuals without alpha planes before those that do. This problem can be resolved in two ways: Firstly, if you have the source code, then when specifying the visual type you can OR in AUX_ALPHA, along with AUX_RGB (for Example auxInitDisplayMode(AUX_RGB | AUX_ALPHA)). Secondly, if source is not available, the following registry variable can be set to 1, which enables the visuals with alpha planes to be selected first.
  `3DExtensions.ExportAlpha`
  in:
  `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\glint\Device0`
  Setting this variable will result in a decrease in the performance of some applications as the driver must perform additional setup calculations for the graphics chip to cater for the Alpha value as well as R, G, and B.

- The conformance tests that involve mip-map texture filtering (miplin.c, mipsel.c and texbc.c) will fail on GLINT 500TX if a registry variable is not changed. `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\glint\Device0\`OpenGL.DisableMipMaps should be set to zero to enable the tests to pass. The default state is to have this set to one because this provides the best trade-off between image quality and performance for the majority of applications.
- In some cases, there is confusion over the meaning of the PFD_SUPPORT_GDI bit in the dwFlags field of the PIXELFORMAT descriptor. 3Dlabs have seen applications (for instance Open Inventor) which incorrectly assume that if this flag is set, rendering to bitmaps is supported by the visual. The Installable Client Driver does not support bitmap rendering so these applications fail. To enable these applications to work the exporting of PFD_SUPPORT_GDI can be disabled by setting the following registry variables in `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\glint\Device0 to` FALSE. The applications will then choose a Generic pixel format so using unaccelerated software rendering to draw to bitmaps.

  ```
  3DExtensions.SupportSingle
  3DExtensions.SupportDouble
  ```

  By default PFD_SUPPORT_GDI is set to TRUE for single buffer formats and FALSE for double buffered formats.

Note: Under Windows NT, Generic pixel formats that support double buffering and rendering via GDI are mutually exclusive. This is because GDI does not have the ability to render to the backbuffer. 3Dlabs have therefore chosen to set the default for double buffering, so as to be in line with the Microsoft implementation. However with care GDI rendering and double buffering may be mixed, so the latter registry variable will cause PFD_SUPPORT_GDI to be exported by double buffer formats, should an application benefit from this added functionality.

- When running multi-threaded applications it may be necessary to disable the use of the fast clear planes by setting the environment variable GLINT_DONT_USE_FCP to TRUE, or by checking the corresponding box in the GLINT control panel applet. This issue arises when more than one context is being used to render to the same window (e.g. OpenGL pipes screen-saver with multiple option selected). If this variable is set then this disables the use of 3Dlabs proprietary fast clear mechanism that allows the depth(Z) buffer to be cleared up to 16 times more quickly than normal. Typically this becomes significant for animation rates of 10Hz or higher in large windows.
- The standard maze screen saver does not get hardware accelerated. This is due to a bug in the Microsoft screen saver library. A customized accelerated 3Dlabs version that also supports linear filtering in the settings option has been provided with this release.