ForceWare Graphics Drivers

Frame Synchronization User's Guide

Version 2.0

NVIDIA Corporation
July 27, 2005

Published by
NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050

Table of Contents

CHAPTER 1

ABOUT NVIDIA FRAME SYNCHRONIZATION

When presenting applications across multiple displays or projection systems, it is important that the displays operate in unison to create the appearance of a single display. Seamless presentation requires the following processes:

- Synchronizing the rendering of frames across all displays
- Synchronizing the swapping of front and back buffers

About This Document

This document explains how to use NVIDIA Frame Lock hardware and software to enable you to create one large virtual canvas using multiple displays. It contains the following main sections::

- "NVIDIA Frame Synchronization" on page 3

  explains how to set up the hardware and software on multiple systems for frame synchronization.

- "Using the OpenGL Extensions" on page 20

  explains how to use the OpenGL extensions supported by NVIDIA for synchronizing the display of frames among multiple displays as well as synchronizing buffer swaps among multiple application windows.

Other Documents

For details on using the NVIDIA Display Properties, see the latest version of the *ForceWare Graphics Drivers User's Guide*.

System Requirements

- Windows® 2000 or Windows® XP.
- Frame Synchronization Capable NVIDIA Graphics Product:
    - NVIDIA Quadro FX 3000G
    - NVIDIA Quadro G-Sync (must be used in conjunction with a Quadro FX 4400 or Quadro FX 4500 graphics card)
- NVIDIA Forceware Graphics Driver version 52.72 or later.

  Quadro FX 4500 support requires version 77.77 or later.

SLI Availability in Upcoming Driver Release

The next release of the NVIDIA Quadro graphics driver (scheduled for mid-September), will support SLI multi-GPU scalability combined with Genlock/Frame Lock functionality. This implementation will require one Quadro G-Sync card used in conjunction with two Quadro FX 4500 SLI cards.

CHAPTER 2

NVIDIA FRAME SYNCHRONIZATION

Visual computing applications that involve multiple displays, or even multiple windows within a display, can require special signal processing and application controls in order to function properly.

For example, in order to produce quality video recording of animated graphics, the display graphics must be synchronized with the video camera. As another example, applications presented on multiple displays must be synchronized in order to complete the illusion of a larger, virtual canvas.

This chapter explains how the NVIDIA Frame Lock graphics cards and Frame Synchronization software lets you synchronize windows and displays for various visual computing applications. It contains the following sections:

- "Frame Synchronization Principles" on page 4
- "Before You Begin" on page 5
- "How To Set Up Genlock" on page 6
- "How To Set Up Frame Lock" on page 13
- "Setting Up Genlock + Frame Lock" on page 19

Frame Synchronization Principles

NVIDIA Frame Synchronizing actually involves two main processes:

- **Genlock—**Synchronizing the displays to an external sync source
- **Frame Lock and Swap Sync—**Synchronizing applications across multiple systems.

Genlock

Genlock is the process of synchronizing the pixel scanning of one or more displays to an external synchronization source. NVIDIA genlock requires the external signal to be either TTL or composite, such as used for NTSC, PAL, or HDTV.

Frame Lock (with Swap Sync)

Proper synchronization of an application running on multiple displays involves the following two processes:

- **Frame Lock**

  Frame lock involves the use of hardware to synchronize the frames on each display in a connected system.

  When an application is displayed across multiple monitors, frame locked systems help maintain image continuity to create a virtual canvas. Frame lock is especially critical for stereo viewing, where the left and right fields must be in sync across all displays.

- **Swap Sync**

  Swap sync refers to the synchronization of buffer swaps of multiple application windows. By means of swap sync, applications running on multiple systems can synchronize the application buffer swaps between all the systems. Swap sync requires that the systems are frame locked.

Frame Lock + Genlock

Genlock can also be combined with frame lock. When several systems are connected together, a sync signal is fed from a master system to the other systems in the network and the displays are synchronized with each other.

Before You Begin

- Make sure that the graphics card and NVIDIA drivers are correctly installed. Refer to the instructions that came with your graphics card or drivers.
- The following is additional information concerning installation of the NVIDIA Quadro G-Sync card:
  - The NVIDIA Quadro G-Sync card must be used in conjunction with a Quadro FX 4400 or Quadro FX 4500 graphics card.
  - Install the NVIDIA Quadro G-Sync card into any available expansion slot within six inches of the Quadro FX 4400 or Quadro FX 4500 G-Sync connector
  - Use the cable that comes with your Quadro G-Sync card to connect the ***primary*** G-Sync connector to the Quadro FX graphics card G-Sync connector as shown in Figure 2.1.

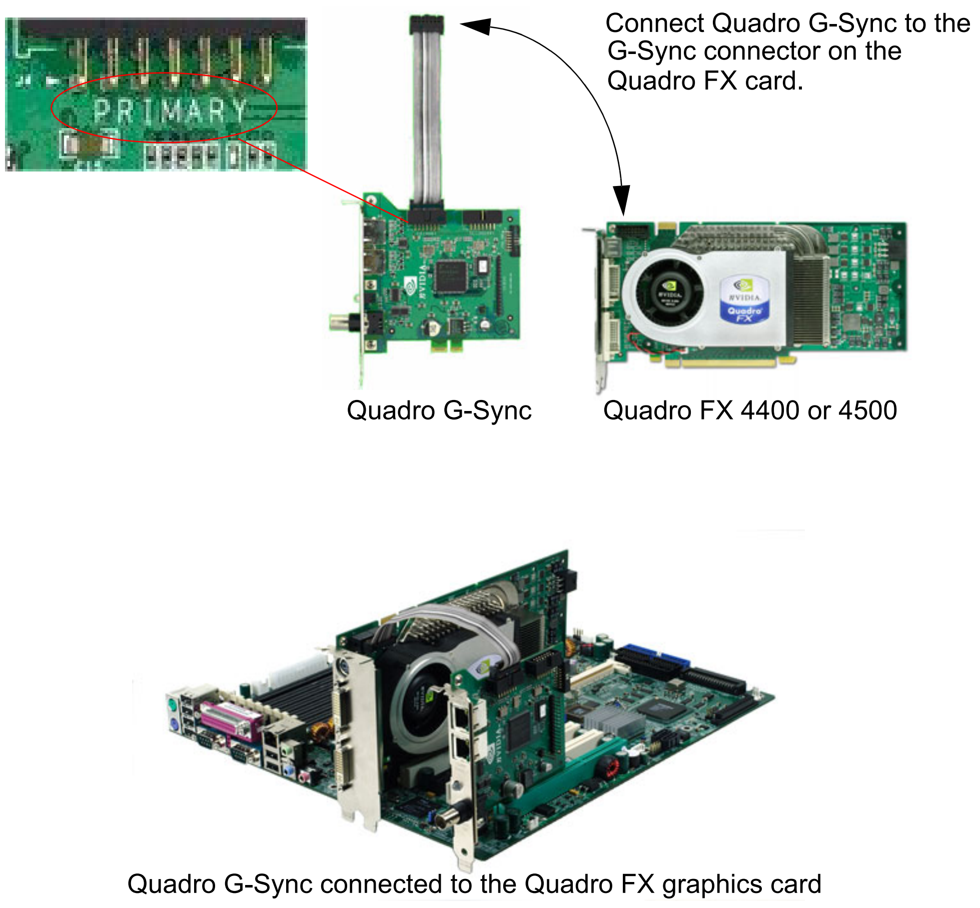

PRIMARY
Connect Quadro G-Sync to the G-Sync connector on the Quadro FX card.
Quadro G-Sync
Quadro FX 4400 or 4500
Quadro G-Sync connected to the Quadro FX graphics card

Figure 2.1 Connecting the NVIDIA Quadro G-Sync Card

How To Set Up Genlock

Genlock Quick Start Guide

The following are the basic steps to genlocking a system. Detailed instructions are provided in the section “Detailed Setup Instructions” on page 7.

1 Set Up the Hardware

Connect the external sync source to the BNC connector on the graphics card.

2 Set Up the Software

Use the Frame Synchronization property page to establish the system as the server, to choose the sync source, and configure the sync pulse.

Detailed Setup Instructions

Set Up the Hardware

1 Start the system, then connect the external source signal to the **House Sync** connector (BNC) of the graphics card.

2 Check the LED on the connector panel.

- A green LED means the hardware has detected the timing signal, and you can proceed to the next step.

  An occasional red flash is OK.

- A red LED means the hardware can't detect the timing signal.

  Check the signal source and correct the problem before proceeding.

Figure 2.2 illustrates a system that is connected to an external sync source.

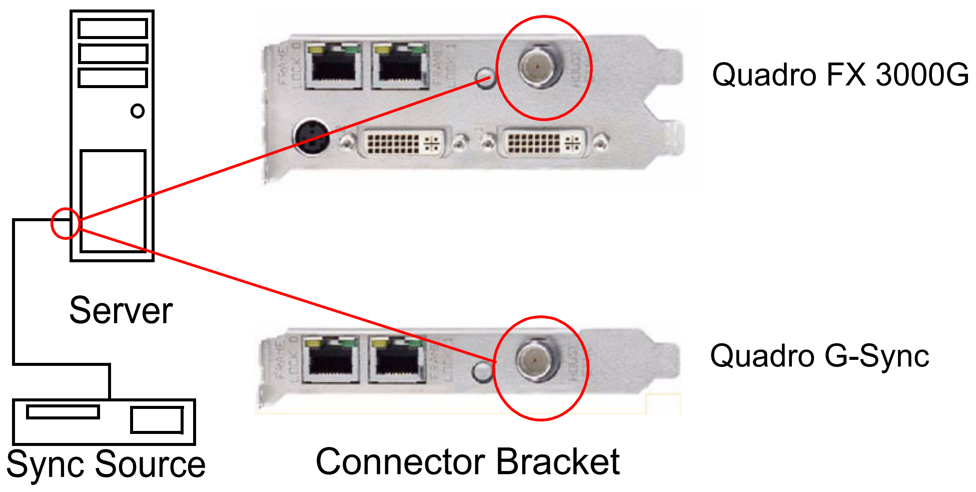
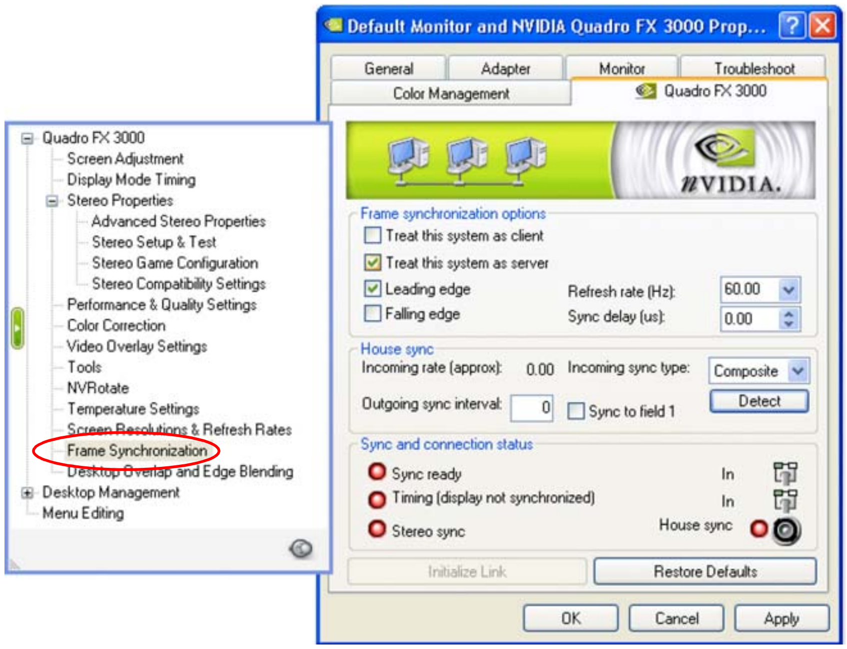


Figure 2.2 Example of Genlock Hardware Connections

Set Up the Software

1 Open the Windows Display Properties control panel and click Settings>Advanced to navigate to the NVIDIA graphics display properties page.

2 Click the **Frame Synchronization** tree item from the slide-out tray.



Default Monitor and NVIDIA Quadro FX 3000 Prop...

General | Adapter | Monitor | Troubleshoot
Color Management | Quadro FX 3000

- Quadro FX 3000
  - Screen Adjustment
  - Display Mode Timing
  - Stereo Properties
    - Advanced Stereo Properties
    - Stereo Setup & Test
    - Stereo Game Configuration
    - Stereo Compatibility Settings
  - Performance & Quality Settings
  - Color Correction
  - Video Overlay Settings
  - Tools
  - NVRotate
  - Temperature Settings
  - Screen Resolutions & Refresh Rates
  - Frame Synchronization
  - Desktop Overlap and Edge Blending
- Desktop Management
- Menu Editing

Frame synchronization options
- ☐ Treat this system as client
- ☑ Treat this system as server
- ☑ Leading edge
- ☐ Falling edge

Refresh rate (Hz): 60.00
Sync delay (us): 0.00

House sync
Incoming rate (approx): 0.00
Incoming sync type: Composite
Outgoing sync interval: 0
☐ Sync to field 1
Detect

Sync and connection status
- Sync ready — In
- Timing (display not synchronized) — In
- Stereo sync — House sync

Initialize Link | Restore Defaults

OK | Cancel | Apply

Figure 2.3 Frame Synchronization Page

3 Check the **Treat this system as Server** check box.

4 If the external signal is an interlaced signal—such as used to drive a 1080i60 display—then click the **Sync to field 1** check box.

For example, when the Sync to field 1 check box is checked, then only the first field of a 60Hz interlaced signal is used to synchronize the display. In this way the effective 120Hz field sync is converted to a 60Hz frame sync.

5 Click the Refresh Rate list arrow and choose a refresh rate that matches the external sync frequency.

If you do not know the external sync frequency, then pick a reasonable frequency from the list.

6 Click **Apply**.

The **Sync and connection status** group box indicates whether the display is genlocked to the external signal. The status LEDs should be green, as shown in Figure 2.4.

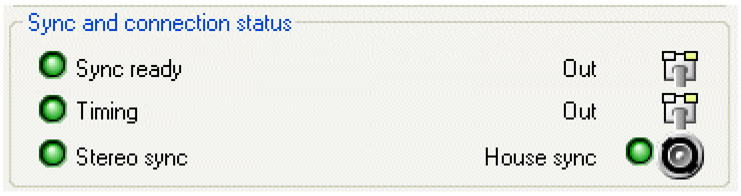


Figure 2.4 Sync and Connection Status for a Genlocked System

- If the Timing LED is red—indicating the display is not synchronized to the timing signal—then see Synchronizing to the External House Signal for further instructions.
- See "Understanding the Sync and Connection Status Indicators" on page 12 for more information about the status indicators.

Additional Setup Instructions

This section discusses the following topics that provide additional information and steps that may be needed to achieve successful genlock:

- "Restoring an Unstable Display" on page 9
- "Synchronizing to the External House Signal" on page 10
- "Understanding the Sync and Connection Status Indicators" on page 12

Restoring an Unstable Display

If your display becomes unstable and you want to restore it to its original configuration, do one or more of the following:

- Click **Restore Defaults** and then click **Apply**.
- Disconnect the external house sync.

Synchronizing to the External House Signal

This section describes other steps you may need to take after connecting a house sync signal for the first time, or after clicking **Restore Defaults** and then **Apply.**

Matching the External House Signal Frequency

After connecting the house sync, the external sync signal frequency is displayed in the section **Incoming Rate** in the control panel.

- If the rate that you previously selected from the list is different, then locate the matching frequency from the list and click **Apply**.
- If the rate is not listed, but the listed values are within 0.5 Hz of the actual signal, then enter the frequency directly into the list box and click **Apply**.

Adjusting for Interlaced External House Signals

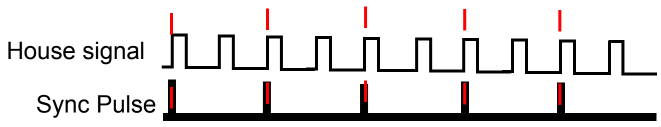
If the external signal is an interlaced signal, then check the **Sync to field 1** check box and click **Apply**.

Generating a Sync Frequency Slower than the External House Signal

If the external sync frequency is not support by your display because it is too high, you can use the external house signal to generate a sync signal of a reduced frequency by sampling the external signal.

- Enter a number in the **Outgoing sync interval** box, then click **Apply**.

  This number represents the number of external sync pulses to receive before generating the synchronization pulse. For example, if the house sync frequency is 120 Hz, then entering **1** in the Outgoing sync interval box results in a 60 Hz sync pulse.



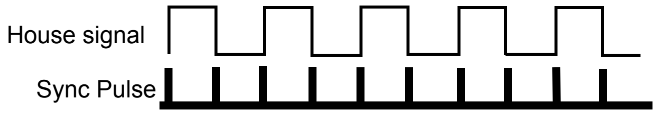
Outgoing Sync Interval value of 1

This action overrides any refresh rate that you may have selected from the refresh rate list.

Generating a Sync Frequency Faster than the External House Signal

You can double the sync frequency by triggering the sync pulse off both the leading and falling edge of the source sync signal. *This works only with TTL level signals*.

In the Sync Options section, click both the leading and falling edge check box.



Outgoing Sync triggered from leading and falling edge

This action overrides any refresh rate that you may have selected from the refresh rate list.

Detecting the Incoming Signal Type

NVIDIA Genlock supports the following external synchronization signal types:

- TTL (3.3 volt level):
- Composite Bi-level (NTSC or PAL sources use bi-level composite signals.)
- Composite Tri-level (HDTV sources commonly use tri-level composite signals.)

Normally, the software automatically detects the signal type that is being used. If necessary, you can force the software to detect the signal type as follows:

1 Click **Detect** to force detection of the signal type.

   The software tests each signal type and stops when it detects a match.

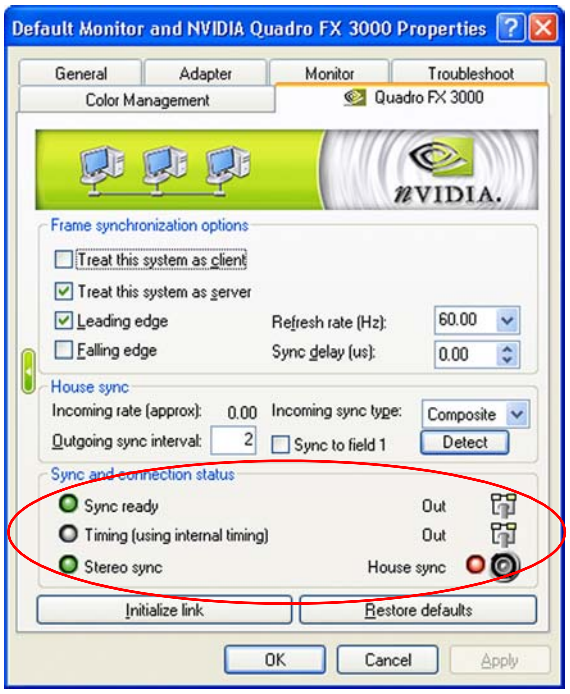
2 If the signal type cannot be detected, click the **Incoming sync type** list box arrow and then click the item corresponding to the signal source.

   Click Composite Auto-Detect if you want the frame lock board to classify the signal as bi-level or tri-level.

3 Click **Apply** to activate the changes.

Understanding the Sync and Connection Status Indicators

The **Sync and connection status** group box displays the status of the sync signals as well as the status of the BNC connection. See Figure 2.5.



Default Monitor and NVIDIA Quadro FX 3000 Properties

General | Adapter | Monitor | Troubleshoot | Color Management | Quadro FX 3000

Frame synchronization options

- [ ] Treat this system as client
- [✓] Treat this system as server
- [✓] Leading edge
- [ ] Falling edge

Refresh rate (Hz): 60.00
Sync delay (us): 0.00

House sync

Incoming rate (approx): 0.00
Incoming sync type: Composite
Outgoing sync interval: 2
- [ ] Sync to field 1

Detect

Sync and connection status

Sync ready — Out
Timing (using internal timing) — Out
Stereo sync — House sync

Initialize link | Restore defaults

OK | Cancel | Apply

Figure 2.5 Sync Status Indicators

- **House Sync** - Green indicates that a signal is present at the BNC connector.
- **Sync Ready** - Green indicates that a sync pulse is present.
- **Timing** - Indicates whether the timing is locked to the sync signal.
  - Green: The timing is locked to the house sync.
  - Grey: The timing is locked to the internal V-Sync.
  - Red: The display is not synchronized to the timing signal.
- **Stereo Sync** - Always green indicates that it is in sync with a timing signal.

  Note: The QuadroFX 3000G card typically takes a couple of seconds to achieve stereo sync. The Quadro G-Sync card, because it provides more accurate synchronization, typically requires about 30 seconds to achieve stereo sync.

How To Set Up Frame Lock

Frame Lock Quick Start Guide

The following are the basic steps to frame locking several systems. Detailed instructions are provided in the section "Detailed Setup Instructions" on page 14.

1 Set Up the Hardware

Connect all the systems together using standard CAT5 patch cabling.

WARNING! The voltage and signal on the frame lock ports are different from Ethernet signals. ***Do not connect a Frame lock port to an Ethernet card or network hub.*** Doing so can cause damage to the hardware.

2 Set Up the Server

Use the Frame Synchronization property page to establish the system as the server.

3 Set Up the Clients

Use the Frame Synchronization property page for each client to enable frame lock on that system, and to add an optional delay to the sync pulse before transmitting to other clients.

Detailed Setup Instructions

Set Up the Hardware

1 Daisy chain the graphics cards together using a standard CAT5 patch cable plugged into the external RJ45 connector.

- You can connect to any of the two RJ45 connectors located on the graphics card bracket.
- Each connector automatically configures itself as an input or output after all the connections are made and one system is configured as a synchronization server (see "Set Up the Server" on page 15). A flashing green LED indicates an input and a flashing yellow LED indicates an output.

  If there is no server assigned and connected, then both LEDs are a steady green.

WARNING! The voltage and signal on the frame lock ports are different from Ethernet signals. ***Do not connect a Frame lock port to an Ethernet card or network hub.*** Doing so can cause damage to the hardware.

2 Designate one of the cards to be the server device. The remaining cards are client devices.

The server determines the trigger pulse for the client devices. The trigger pulse is derived from the V-sync of the server video.

Figure 2.6 shows an example of four frame locked systems.

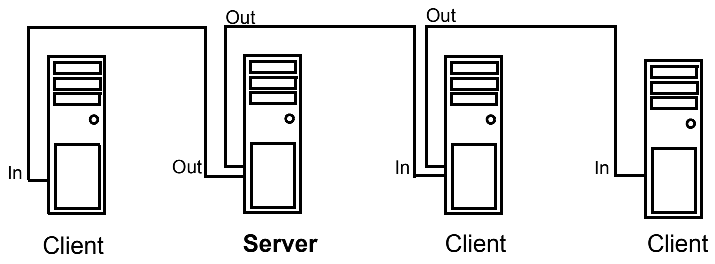
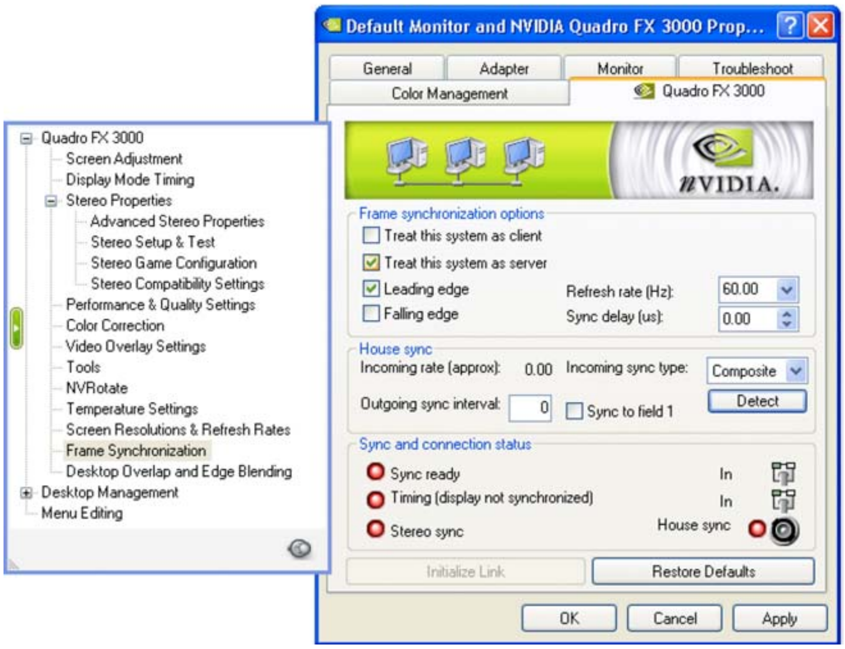


Figure 2.6 Example of Frame Lock Hardware Connections

Set Up the Server

1 Start the system that you have designated as the server, then open the Windows Display Properties control panel and click Settings>Advanced to navigate to the NVIDIA graphics display properties page.

2 Click the **Frame Synchronization** tree item from the slide-out tray.



Frame synchronization options
- ☐ Treat this system as client
- ☑ Treat this system as server
- ☑ Leading edge
- ☐ Falling edge
- Refresh rate (Hz): 60.00
- Sync delay (us): 0.00

House sync
- Incoming rate (approx): 0.00
- Incoming sync type: Composite
- Outgoing sync interval: 0
- ☐ Sync to field 1

Sync and connection status
- Sync ready — In
- Timing (display not synchronized) — In
- Stereo sync — House sync

Figure 2.7 Frame Synchronization Page

3 Check the **Treat this system as Server** check box.

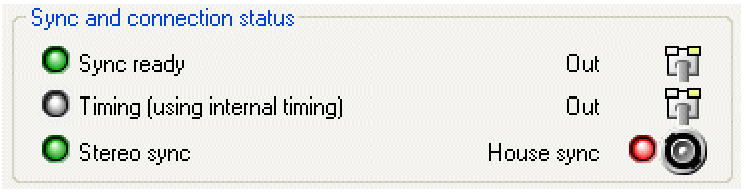
4 Configure the Sync Pulse Frequency

Click the **Refresh rate (Hz)** list box arrow and then click the frequency you want to use for the sync pulse.

The sync pulse frequency (refresh rate) must be supported by all the displays that are frame locked to the server.

5 Click **Apply** to activate the changes.

The **Sync and connection status** group box indicates the type of timing used for the frame lock server system. The Timing LED should be gray, as shown in Figure 2.8.



Sync and connection status

● Sync ready	Out	
● Timing (using internal timing)	Out	
● Stereo sync	House sync	●

Figure 2.8 Sync and Connection Status for a Frame Lock Server

Set Up the Clients

Using the OpenGL Extensions

You can set up the slave devices using the OpenGL extensions. See "Using the OpenGL Extensions" on page 20.

Using the Frame Synchronization page

1 For each slave system, start the system and then open the Windows Display Properties control panel and click Settings>Advanced to navigate to the NVIDIA graphics display properties page.

2 Click the Frame Synchronization tree item from the slide-out tray.

3 Click the **Treat this system as Client** check box.

4 If the next display downstream requires a slight delay, enter the sync offset (in microseconds) in the Sync Delay box.

The sync offset is the delay between the source trigger point and the actual sync pulse. See Figure 2.9. The value can be in the range of 0 to the time needed to display an entire frame.

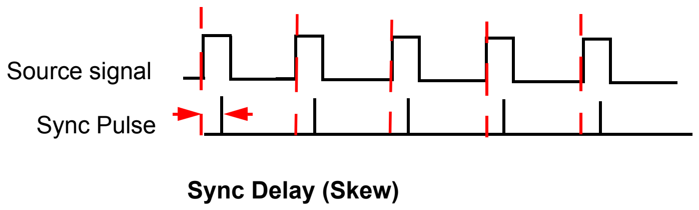


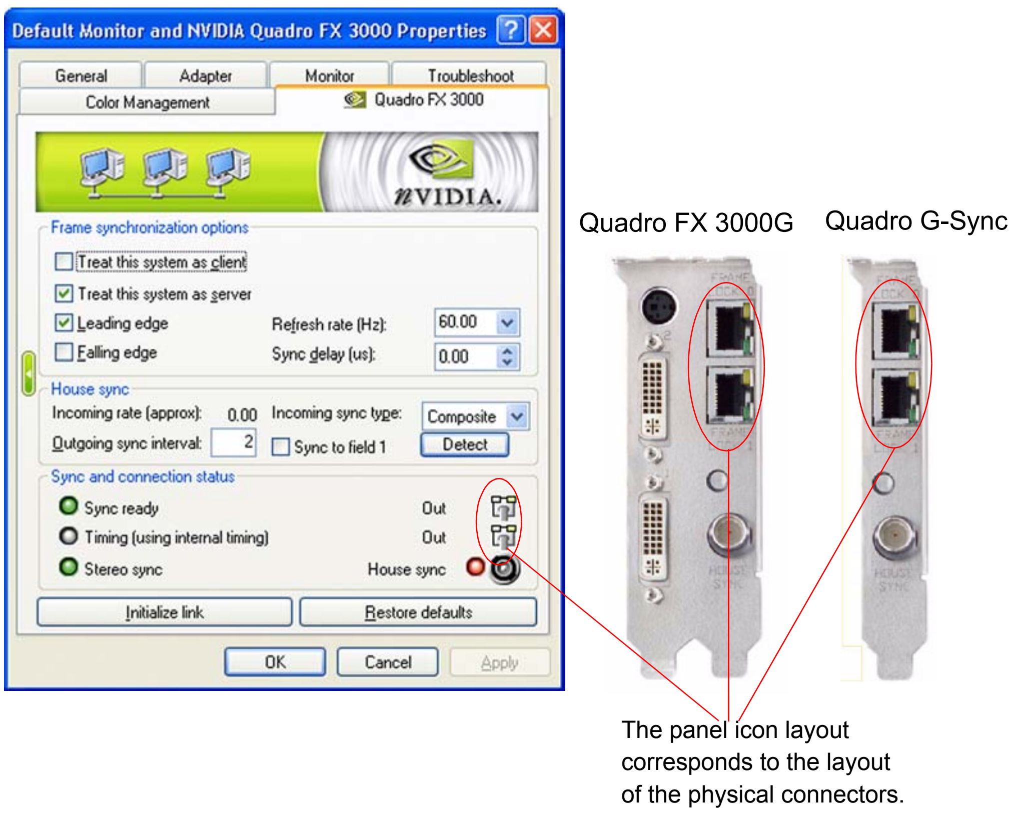
Figure 2.9 Offset Applied to Sync Pulse

5 Click **Apply** to activate the changes.

Testing the Connections

To test the connections, click **Initialize Link** on the Server control panel.

The **Sync and connection status** group box displays the status of the sync signals as well as the status of the RJ45 ports. The two ports are represented by connector icons. See Figure 2.10.



Default Monitor and NVIDIA Quadro FX 3000 Properties

General | Adapter | Monitor | Troubleshoot | Color Management | Quadro FX 3000

Frame synchronization options
- [ ] Treat this system as client
- [✓] Treat this system as server
- [✓] Leading edge
- [ ] Falling edge

Refresh rate (Hz): 60.00
Sync delay (us): 0.00

House sync
Incoming rate (approx): 0.00
Incoming sync type: Composite
Outgoing sync interval: 2
- [ ] Sync to field 1

Detect

Sync and connection status
- Sync ready — Out
- Timing (using internal timing) — Out
- Stereo sync — House sync

Initialize link | Restore defaults

OK | Cancel | Apply

.

In	The connector is receiving the sync signal from another card.
Out	The connector is sending the sync signal to another card.

Figure 2.10 Sync and Connection Status

Sync and Connection Status for the Server

- **Sync Status**
  - **Sync Ready** - Green indicates that a sync pulse is present.
  - **Timing** - Indicates whether the timing is locked to the sync signal.

    Grey: The timing is locked to the internal V-Sync.
  - **Stereo** - Always green indicates that it is in sync with a timing signal.
- **Connection Status**
  - **Out/Out** is the normal indicator for the server device.
  - **In/In** indicates that the system is not selected as a server or client, and is not Frame-lock enabled.

Sync and Connection Status for the Client

- **Sync Status**
  - **Sync Ready** - Green indicates that a sync pulse is present.
  - **Timing** - Indicates whether the timing is locked to the sync signal.

    Green: The timing is locked to the signal on the frame lock connector.

    Red: Either no signal on the frame lock connector is detected, or the lock to the signal has been lost.
  - **Stereo** - Steady green indicates that the client's stereo timing is locked to the server's stereo signal.

  Note: The QuadroFX 3000G card typically takes a couple of seconds to achieve stereo sync. The Quadro G-Sync card, because it provides more accurate synchronization, typically requires about 30 seconds to achieve stereo sync.
- **Connection Status**
  - **In/Out** or **Out/In** is the normal indicator for client devices.
  - **In/In** indicates that there is no connection at any of the connectors of the client device, or the system is not selected as a server or client, and is not Frame-lock enabled.

Swap Sync

The application controls synchronization of buffer swaps (swap sync) between applications. Proper connection and synchronization of the timing signal are required for proper synchronization of buffer swaps.

See "Using the OpenGL Extensions" on page 20 for instructions on how an application can use the extensions to accomplish swap sync.

Setting Up Genlock + Frame Lock

The procedures in the previous two sections describe the more common uses of NVIDIA Frame Sync. A third use involves a combination of genlock and frame lock.

In this setup, a set of client systems are connected to a system that is designated as the server, as described in "How To Set Up Frame Lock" on page 13. The difference is that the server system is genlocked to an external house sync, as described in "How To Set Up Genlock" on page 6.

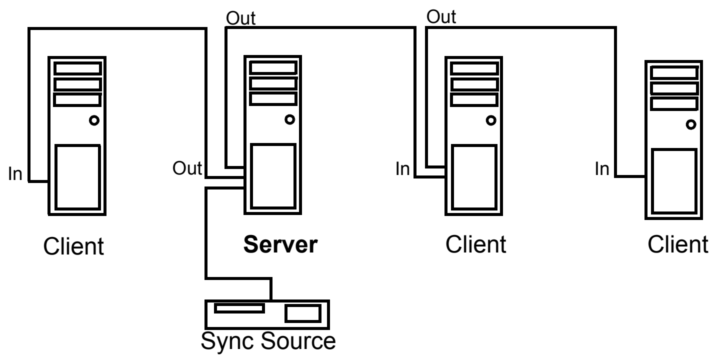


Figure 2.11 Example of Genlock + Frame Lock Configuration

CHAPTER
3

USING THE OPENGL EXTENSIONS

This chapter explains how to use the OpenGL extensions supported by the NVIDIA driver for accomplishing frame synchronization of applications.

- "Frame Synchronization Using the OpenGL Extensions" on page 21 explains the principles behind swap groups, and describes the OpenGL extensions to use to control frame lock and swap sync functions.
- "OpenGL Extensions" on page 26 details the API calls for the extensions.

Frame Synchronization Using the OpenGL Extensions

Swap sync refers to the synchronization of buffer swaps of multiple application windows. By means of swap sync, applications running on multiple systems can synchronize the application buffer swaps between all the systems.

Swap sync requires

- proper connection and synchronization of the timing signals
- a mechanism for binding several windows together so that buffer swaps can be synchronized across all windows

About Buffer Swaps

Key Points and Definitions

Buffer swaps are performed on windows.

- **Swap Groups**

  If buffer swaps must be synchronized across several windows on a single system, you can define a "group" that consists of the specified windows.

- **Swap Barriers**

  If buffer swaps must be synchronized across several systems, you can define a "barrier" that consists of the specified groups.

Buffer Swap Criteria

The criteria for buffer swaps involves when a window is ready to swap and when a group is ready to swap.

Window Buffer Swaps

Any `hDC` that is not a window—such as a non-visible rendering buffer—is always ready, otherwise the following criteria must be satisfied before a buffer swap for a window can be performed:

- The window itself must be ready, meaning:
  - A buffer swap command has been issued for it.
  - Its swap interval has elapsed.
- If the window belongs to a group, all the windows in the group must be ready.

- If the window belongs to a group and that group is bound to a barrier, all groups bound to that barrier must be ready.

Group and Barrier Buffer Swaps

- Buffer swaps for all windows in a swap group take place concurrently, and buffer swaps for all groups using a barrier take place concurrently.

  For barrier swaps, the vertical retraces of the screens of all the groups must also be synchronized, otherwise there is no guarantee of concurrency between groups.

- An implementation may support a limited number of swap groups and barriers, and may have restrictions on where the users of a barrier can reside.

  For example, an implementation may allow the users to reside on different display devices or even hosts. An implementation may return zero for any of **maxGroups** and **maxBarriers** returned by `QueryMaxSwapGroupsNV` if swap groups or barriers are not available in that implementation or on that host.

Frame Counter

The implementation provides a universal counter, or frame counter, among all systems that are locked together by swap groups/barriers. It is based on the internal synchronization signal which triggers the buffer swap.

- To obtain the current frame count, call `QueryFrameCountNV`().

- To reset the frame count back to zero, call, `ResetFrameCountNV`().

  In a system that has an NVIDIA frame-lock device installed and enabled, `ResetFrameCountNV()` succeeds only when the frame lock device is configured as a Master device.

Enabling and Configuring Genlock

See the section "OpenGL Extensions" on page 26 for detailed API descriptions of the extensions referred to in this section.

Enabling Genlock

Each display that is intended to be part of a frame-lock network must be enabled.

- To enable, call

  `wglEnableGenlockI3D()`

- To disable, call

  `wglDisableGenlockI3D()`

This call should be made before changing frame sync parameters in order to avoid synchronization problems.

- To query the state of a display, call

  `wglIsEnabledGenlockI3D()`

Configuring the Sync Trigger

The start of each frame[1] is synchronized to the sync trigger pulse. This pulse can be based on the vertical retrace signal of the frame lock device configured as the Master device, or it can be derived from an external video signal, like a house sync signal. Configuring the trigger pulse involves:

- Selecting the sync source signal
- Deriving the trigger pulse from the sync source based on the sync source edge, sampling frequency, and an optional delay

Select the Sync Source

- To select the source signal upon which to base the trigger pulse, call

  `wglGenlockSourceI3D()`

  and specify either the internal sync signal or an external sync signal.

- To query the sync source, call

  `wglGetGenlockSourceI3D()`

Configure the Trigger Pulse

The trigger pulse is derived from the sync source, and is the signal to which all frame-lock enabled boards are synchronized. Specify the trigger pulse as follows:

- **Set An Edge Mode**

  Set the sync pulse to coincide with the leading edge, the falling edge, or both the leading and falling edge of the source sync signal, by calling

  `wglGenlockSourceEdgeI3D()`

  and specifying either the rising edge, the falling edge, or both edges.

  To query the edge mode, call **`wglGetGenlockSourceEdgeI3D()`**.

- **Set a Sampling Value**

  Set the sync pulse as a function of samplings of the source signal, by calling

  `wglGenlockSampleRateI3D()`

1. The current hardware does not support triggering on field data, so you cannot specify triggering on left-only or right-only frames.

and specifying every ***n***th source pulse to use for the trigger pulse. The minimum value is 1. The maximum value is 6. As an example, if a value of 2 is used, then every other source pulse would generate a trigger pulse.

To query the sample value, call **`wglGetGenlockSampleRateI3D()`**.

- **Set an Optional Delay Value**

  If the next display downstream requires a slight delay, delay the trigger pulse (in pixels) by calling

  `wglGenlockSourceDelayI3D`()

  and specify a delay. The minimum delay value is 0. The maximum delay value is the number of pixel clocks needed to display an entire frame.

  To determine the maximum delay value:

  1. Call the function `wglQueryGenlockMaxSourceDelayI3D`() and obtain the values for maxLineDelay and maxPixelDelay.
  2. Calculate the maximum delay using the equation

     maxDelay = (maxLineDelay) x (maxPixelDelay)

Configuring for Swap Sync

Configuring Swap Groups

You can assign an OpenGL window to a swap group. There is a maximum number of allowable swap groups that can be created for any particular implementation.

- To determine the maximum swap group number, call

  `QueryMaxSwapGroupsNV()`

- To assign a window to a swap group, call

  `JoinSwapGroupNV()`

  and specify the swap group number.

  If `hDC` is already a member of a different group, it is implicitly removed from that group. If the swap group number is zero, the `hDC` is removed from the current group.

  To query the current swap group, call **`QuerySwapGroupNV`**().

Configuring Swap Barriers

You can bind a group to a barrier. There is a maximum number of barriers that can be created.

- To determine the maximum barrier number, call

  `QueryMaxSwapGroupsNV`()

- To bind a group to a barrier, call

  `BindSwapBarrierNV()`
  and specify the group and barrier numbers.

  If the barrier number is zero, then the group in unbound from the current barrier.

  To query the current barrier, call **`QuerySwapGroupNV`**().

OpenGL Extensions

This section describes the relevant OpenGL extensions supported by the NVIDIA display driver (to be shipped with the NVIDIA frame-lock hardware). The set of extensions supported for frame lock are exported as `WGL_I3D_genlock` and `WGL_NV_swap_group`.

These are exported in the WGL extension string which can be queried by `wglGetExtensionsStringARB`. This documentation covers the WGL versions of the extensions which are exported on Microsoft WindowsNT/XP systems. There is an equivalent Unix version of these extensions: `GLX_NV_swap_group`.

Exported WGL_I3D_genlock Functions

The exported functions for this extension are:

- `wglEnableGenlockI3D`
- `wglDisableGenlockI3D`
- `wglIsEnabledGenlockI3D`
- `wglGenlockSourceI3D`
- `wglGetGenlockSourceI3D`
- `wglGenlockSourceEdgeI3D`
- `wglGetGenlockSourceEdgeI3D`
- `wglGenlockSampleRateI3D`
- `wglGetGenlockSampleRateI3D`
- `wglGenlockSourceDelayI3D`
- `wglGetGenlockSourceDelayI3D`
- `wglQueryGenlockMaxSourceDelayI3D`

wglEnableGenlockI3D

This call enables genlock for all monitors connected to a device (device context **hDC**) of an OpenGL window. There is only a single genlock device for each graphics adapter regardless of the number of monitors supported by the adapter.

Function	**`BOOL wglEnableGenlockI3D(HDC hDC)`**
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.

Return Values	**`TRUE: Success`**
	`FALSE: Failure`

wglDisableGenlockI3D

This call disables genlock for a monitor.

Function	`BOOL wglDisableGenlockI3D(HDC hDC)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

wglIsEnabledGenlockI3D

This call queries the current genlock enable/disable state

Function	`BOOL wglIsEnabledGenlockI3D(HDC hDC, BOOL *pFlag)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*pFlag`**: - 0 = Genlock is disabled. 1 = Genlock is enabled.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

wglGenlockSourceI3D

This call sets the sync source upon which the frame lock sync trigger is based.

Function	`BOOL wglGenlockSourceI3D(HDC hDC, UINT uSource)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card. **`uSource`** - **`WGL_GENLOCK_SOURCE_MULTIVIEW_I3D`** Select the internal sync signal as the sync source. **`WGL_GENLOCK_SOURCE_EXTERNAL_SYNC_I3D`** Select the external house sync as the sync source.
Parameters Out	
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

wglGetGenlockSourceI3D

This call queries the current sync source.

Function	`BOOL wglGetGenlockSourceI3D(HDC hDC, UINT *uSource)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*uSource`** - **`WGL_GENLOCK_SOURCE_MULTIVIEW_I3D`** The internal sync signal is the sync source. **`WGL_GENLOCK_SOURCE_EXTERNAL_SYNC_I3D`** The external house sync is the sync source.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

wglGenlockSourceEdgeI3D

This call synchronizes the trigger pulse to the edge or edges of the source signal.

Function	`BOOL wglGenlockSourceEdgeI3D(HDC hDC, UINT uEdge)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card. **`uEdge`** - **`WGL_GENLOCK_SOURCE_EDGE_FALLING_I3D`** Selects the falling edge of the source. **`WGL_GENLOCK_SOURCE_EDGE_RISING_I3D`** Selects the rising edge of the source. **`WGL_GENLOCK_SOURCE_EDGE_BOTH_I3D`** Selects both edges of the source.
Parameters Out	
Return Values	**`TRUE:`** Success **`FALSE`**: Failure

wglGetGenlockSourceEdgeI3D

This call queries the current trigger pulse edge mode.

Function	`BOOL wglGetGenlockSourceEdgeI3D(HDC hDC, UINT *uEdge)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*uEdge`** - **`WGL_GENLOCK_SOURCE_EDGE_FALLING_I3D`** Selects the falling edge of the source. **`WGL_GENLOCK_SOURCE_EDGE_RISING_I3D`** Selects the rising edge of the source. **`WGL_GENLOCK_SOURCE_EDGE_BOTH_I3D`** Selects both edges of the source.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

`wglGenlockSampleRateI3D`

This call sets the trigger pulse as a function of the specified sync source sampling rate.

Function	`BOOL wglGenlockSampleRateI3D(HDC hDC, UINT uRate)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card. **`uRate`** - Specifies every ***n***th pulse of the sync source signal to use for the trigger pulse. For example, if **`uRate`** were set to a value of 2, every other sync source pulse would generate a trigger pulse. **`uRate`** must be in the range of 1–6.
Parameters Out	
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

`wglGetGenlockSampleRateI3D`

This call queries the current sync source sampling rate for the trigger pulse.

Function	`BOOL wglGetGenlockSampleRateI3D(HDC hDC, UINT *uRate)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*uRate`** - The ***n***th pulse of the sync source used for the trigger pulse. For example, if **`uRate`** were set to a value of 2, every other sync source pulse would generate a trigger pulse.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

`wglGenlockSourceDelayI3D`

This call specifies the delay to apply to the trigger pulse.

Function	`BOOL wglGenlockSourceDelayI3D(HDC hDC, UINT uDelay)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card. **`uDelay`** - The delay, in pixels, from the sync source trigger edge to the actual trigger pulse.
Parameters Out	
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

wglGetGenlockSourceDelayI3D

This call queries the current trigger pulse delay.

Function	`BOOL wglGetGenlockSourceDelayI3D(HDC hDC, UINT *uDelay)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*uDelay`** - The delay, in pixels, from the sync source trigger edge to the actual trigger pulse.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

wglQueryGenlockMaxSourceDelayI3D

This call queries the maximum line and pixel delay. Use these values to calculate the maximum trigger pulse delay (**`maxDelay`**), using the equation

`maxDelay` = (**`maxLineDelay`**) x (**`maxPixelDelay`**).

Function	`BOOL wglQueryGenlockMaxSourceDelayI3D(HDC hDC, UINT *uMaxLineDelay, UINT *uMaxPixelDelay)`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*uMaxLineDelay`** - The maximum number of raster scan lines per frame. This includes all visible and non-visible lines—such as vertical blanking lines— for all fields. **`*uMaxPixelDelay`** - The maximum number of pixels per raster scan line. This includes all visible and non-visible pixels such as horizontal blank.
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

Exported WGL_NV_swap_group Functions

The exported functions for this extensions are:

- `JoinSwapGroupNV`
- `BindSwapBarrierNV`
- `QuerySwapGroupNV`
- `QueryMaxSwapGroupsNV`
- `QueryFrameCountNV`
- `ResetFrameCountNV`

`JoinSwapGroupNV`

This call adds the OpenGL window with the device context `hDC` to the swap group specified by group. If `hDC` is already a member of a different group, it is implicitly removed from that group first.

Function	`BOOL JoinSwapGroupNV(HDC hDC, GLuint group);`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card. **`group`** - Swap group number to which the device context is to be assigned. The value must be between 0 and **`maxGroups`** (see "QueryMaxSwapGroupsNV" on page 33). If 0, the device context is unbound from its current group.
Parameters Out	
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

`BindSwapBarrierNV`

This call binds a swap group to a barrier.

Function	`BOOL BindSwapBarrierNV(GLuint group, GLuint barrier);`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card. **`group`** - Swap group number that is to be bound to a barrier. **`barrier`** - Barrier number to which the swap group is to be bound. Value must be between 0 and **`maxBarrier`** (see "QueryMaxSwapGroupsNV" on page 33).
Parameters Out	
Return Values	**`TRUE:`** Success **`FALSE:`** Failure

QuerySwapGroupNV

This call returns the swap group number and barrier number to which the device context is bound.

Function	`BOOL QuerySwapGroupNV(HDC hDC, GLuint *group, GLuint *barrier);`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*group`** - The group number to which the device context is bound, or assigned.
	`*barrier` - The barrier number to which the device context is bound.
Return Values	**`TRUE:`** Success
	`FALSE:` Failure. In this case, **`*group`** and **`*barrier`** are undefined.

QueryMaxSwapGroupsNV

This call returns the maximum number of swap groups and swap barriers that are supported by the implementation.

Function	`BOOL QueryMaxSwapGroupsNV(HDC hDC, GLuint *maxGroups, GLuint *maxBarriers);`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	**`*maxGroups`** - The maximum group number supported by the implementation.
	`*maxBarriers` - The maximum barrier number supported by the implementation.
Return Values	**`TRUE:`** Success
	`FALSE:` Failure. In this case, **`*maxGroups`** and **`*maxBarriers`** are undefined.

QueryFrameCountNV

`QueryFrameCountNV` returns in `count` the current frame counter for `swapGroup`.

This call returns the current frame count of the swap group.

Function	`BOOL QueryFrameCountNV(HDC hDC, GLuint *count);`
Parameters In	**`hDC`** - Device context for the frame lock card, or for a window residing on the frame lock card.
Parameters Out	*count - The current frame count of the swap group.
Return Values	**`TRUE:`** Frame count successfully retrieved.
	`FALSE:` Frame count retrieval failed.

ResetFrameCountNV

This call resets to zero the frame count of the swap group. The call succeeds only on the master device.

Function	`BOOL ResetFrameCountNV(HDC hDC);`
Parameters In	**`hDC`** - Device context for the frame lock card, or a window residing on the frame lock card.
Parameters Out	
Return Values	**`TRUE:`** Frame counter is successfully reset.
	`FALSE:` Frame counter not successfully reset.