Tegra Linux
PerfHUD ES Quickstart Guide

Version 10.7.2

Contents

Introduction

This quick start guide is designed to assist a new user in setting up and starting PerfHUD ES profiling of applications on a Tegra Linux device.

PerfHUD ES enables experimentation and in-depth analysis of OpenGL ES 2.x applications. Through data provided by the OpenGL ES driver and the Tegra SoC, developers are provided live system status, including deep GPU performance and bottleneck information, in order to help target optimizations where they are most needed.

The debugging capabilities of PerfHUD ES give full access to the state of the OpenGL ES pipeline, related textures and shaders, and all rendering states to help find the causes for improper setup, rendering anomalies, and performance issues.

Setup

Requirements

- A Windows-based host PC, ideally with minimum 2GHz CPU and 2GB RAM, plus an OpenGL 1.5 (GeForce 6-series) or better GPU. The "PerfHUD ES User Guide" installed during this setup process covers GPU requirements in more depth, as certain features are necessary for the host PC to visualize the graphics rendered on the Tegra device. Windows XP is the officially-supported Window OS for PerfHUD ES. Windows Vista and Windows 7 (32-bit and 64-bit) have had basic testing and seem to operate properly.
- An Ethernet connection to the Tegra device from the host PC, specifically with an IP address on the Tegra device that is reachable from the host PC. Specific details on setting up networking on your Tegra device is outside the scope of this document.
- A Linux OS installed on your target Tegra device with PerfHUD ES support. For public developers, the L4T 10.7.2 pack or later is required for PerfHUD support. Customers with LDK releases can add a PerfHUD ES support pack matching the branch. We have tested a Tegra 250 Devkit with both 10.7.1 and 10.7.2 LDK releases, and L4T 10.7.2 pack.

Installing the Windows PerfHUD ES Host

On the Windows host PC, you need the PerfHUD ES Host installer executable, which if you don't already have can be downloaded from the Tegra devsite:

http://tegradeveloper.nvidia.com/tegra/downloads

It should be a file named like "NVIDIA_PerfHUD_ES.exe" for the Windows installer. Once downloaded, run the installer, then:

1) Click "Next".
2) Decide to accept the EULA and then click "Next".
3) Decide on your installation location and then click "Next".
4) Click "Install" and wait for installation to complete.
5) Click "Finish".

We ***strongly*** recommend you read the "PerfHUD ES User Guide" document to learn how to use the PerfHUD ES host user interface to examine your GLES application. The installer adds a link to the doc in the the Start Menu under:

All Programs => NVIDIA Corporation => NVIDIA PerfHUD ES Tegra2

Installing PerfHUD ES Target Device Support

Note: If you are using the L4T 10.7.2 pack or later, you can skip to the Turn On/Off step.

At this time, LDK customers will need the "`tegra_linux_perfhud_10_7_1.tgz`" support pack for this next step – if you don't already have it, please contact your support representative. It is easiest to copy the archive to the Tegra device (using you favorite method, either copying to boot media from a host linux PC, or using remote file transfer like SCP), and then simply unpack it on-device using a remote shell, with:

```
tar -xzvf tegra_linux_perfhud_10_7_1.tgz
```

If you then execute '`ls`', you should see find now have '`perfhud.sh`', as well as two PerfHUD interposer libraries ('`.so`' files). The first time you run `perfhud.sh`, it will attempt to install the libraries properly (basically moving them to `/usr/lib`).

Turn On/Off PerfHUD Target Support

To enable PerfHUD support (and auto-install the support libs if not yet installed), execute:

```
./perfhud.sh on
```

If you later want to turn off PerfHUD support, simply execute:

```
./perfhud.sh off
```

Connecting to the Target Device

Prepare Target for Profiling

You should have installed the PerfHUD support pack on your Tegra device in the prior step.

To enable PerfHUD support on the target Tegra device, if you haven't already, execute:

```
./perfhud.sh on
```

Then, launch the application to be profiled on the Tegra device.

Start PerfHUD Host

Run the PerfHUD ES Host application on your Windows PC.

You will then be prompted with a dialog requesting the Target IP address.

Connect to Target over TCP/IP

If you are running just a single application, in a non-GLES-accelerated window manager:

1) Enter the IP address for your target into the dialog.
2) Click "Connect".

Also note that when there's only a single GLES application running, if you quit the application and then start it (or another GLES app) without exiting PerfHUD ES on the host PC, PerfHUD will automatically establish a connection to the new app instance without user intervention. This is useful for running a given app repeatedly as you find and fix issues.

If you are instead running *multiple* GLES applications simultaneously on the target, or a window manager that also leverages OpenGL ES in addition to applications, you'll want to:

1) Enter the IP address for your target into the dialog.
2) Click "Advanced"
3) Click "Refresh graphics process list"
4) Select your application from the process list
5) Click "Connect"

Common Problems

The windows ding.wav plays over and over again.

The sound effect is played each time a buffer object is uploaded, to disable open the view dropdown menu and deselect audio notifications.

The driver time never seems to indicate anything on the dashboard.

Please select a non-zero value for driver time sampling as the default is zero.

The speed bar is not visible and the frame profiler is not selectable.

Please ensure that your application is using the “EGL_NV_Perfmon” extension to acquire all application side timing information as detailed in the “PerfHUD ES User Guide” document.

Known Issues

Application hangs with target device log showing an error like "Output FIFO does not refill, context read is stuck".

There is no current workaround, please try restarting the target device and application. If the issue persists, please contact us. We are investigating the issue.

My application fails to run with GL_OUT_OF_MEMORY.

Please try restarting the target device and application. If the issue persists, please contact your support representative.

The timing graph varies wildly for a static scene.

There is no current workaround. We are investigating the issue.

The frame debugger takes a long time to complete or fails to complete.

Please try closing any other applications or increasing the memory available on your host PC.

The texture viewer does not appear.

Please try restarting the PerfHUD ES Host and reconnect to the target device.

The shader viewer shows "Microcode disassembler is not available" for vertex and fragment microcode modes.

Vertex and fragment microcode disassembly is not supported by PerfHUD ES on Tegra. Please select a source mode instead. The unsupported modes will be removed in a future release.

The driver time sampling interval is ignored after a lost connection.

Try setting the sampling interval to another value, then back again to your preferred value.

Framebuffer objects are displayed incorrectly when attached to a texture sampler.

There is no current workaround. We are investigating the issue.

The geometry viewer stutters on very complex geometry.

Please try closing other running applications on the host PC. Also, ensure that you have the latest drivers for your graphics card.

Vertical scroll bars appeared clipped along the vertical axis.

There is no current workaround. We are investigating the issue.

Portions of the Performance Dashboard overlay the Frame Debugger.

Please try restarting the PerfHUD ES Host to resolve this issue.

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com