PVRHub

User Manual

Filename	:	PVRHub.User Manual
Version	:	PowerVR SDK REL_3.5@3523383a External Issue
Issue Date	:	17 Apr 2015
Author	:	Imagination Technologies Limited

Contents

List of Figures

List of Tables

1. Introduction

1.1. Document Overview

The purpose of this document is to serve as a complete user manual for the PVRHub Android Application and the PVRHub Linux scripts. It includes installation instructions, a guide to the functionality of the two applications, a complete listing of all interface options and preferences for each of the applications, as well as instructions on their usage.

1.2. Software Overview

PVRHub is an application for Android and Linux that allows for the control and setup of the PVRTrace Recording Libraries. This helps the user avoid manual setup and control. PVRHub also includes the PVRPerfServer functionality to transmit performance data to an instance of PVRTune. There are two main components to PVRHub:

- **Recording Libraries**: These are shim libraries that are installed on a platform and capture all calls to that platform's native graphics libraries. These calls are captured and written into a `.pvrtrace` file for reading back by the PVRTrace GUI.
- **PVRPerfServer**: This is an application which runs on the target device and reads counters and registers from the hardware and either writes it into a `.pvrtune` file, or transmits that data to PVRTune over a network.

Note: On devices that have not been rooted, users must package the PVRTrace recorder libraries into their APK.

Note: For more information about unrooted instructions, see the "PVRTrace Quick Start Guide for Unrooted Android Devices" distributed with the PowerVR Graphics SDK.

2. Installation

2.1. Android

When installing PVRHub it is important to remember that the current version of PVRHub is only compatible with Android 4.1 (Jellybean) and later versions. To install PVRHub, on the local machine run:

```
adb install PVRHub.apk
```

Note: For full functionality, PVRHub needs to be running on a rooted device.

2.2. Linux

With the SDK package successfully installed, PVRHub can be installed in one of two ways, per terminal, or per user.

2.2.1. Per Terminal

To install PVRHub per terminal, complete the following instructions:

1. Open a terminal and enter the following:

```
cd <PVRHubFolder>/<Platform>/
```

Where `<PVRHubFolder>` is the folder in which PVRHub is installed and `<Platform>` is the Linux platform PVRHub is to be ran on (i.e., Linux_ARMv7, Linux_x86_32, etc.).

2. In the same terminal enter the following:

```
source Scripts/envsetup.sh
```

3. Repeat this process for each terminal in which PVRHub is to be used.

2.2.2. Per User

To install PVRHub per user, enter the following into the BASHRC file of the desired user:

```
if [ -f /Path/to/PVRHub/<Platform>/Scripts/envsetup.sh ]; then
    cd /Path/to/PVRHub/<Platform>/
    source Scripts/envsetup.sh
    cd
fi
```

Future interactive shells opened by the user are now able to use PVRHub.

3. Usage

3.1. Android

When installed on an Android device, PVRHub can trace and profile applications, collecting information about their performance for later analysis.

3.1.1. Installing PVRTrace Recording Libraries

If PVRHub can gain root access, click `Install PVRTrace`. If that option is unavailable, but the device is rooted, run PVRTraceInstaller through a command line, as follows:

1. Get a root ADB connection (`adb root`) or shell (`adb shell`. Run as `su`).
2. Run PVRTraceInstaller.

```
/data/data/com.powervr.PVRHub/bin/PVRTraceInstaller -i
```

Note: To uninstall the PVRTrace Recording Libraries on a rooted device when the Uninstall PVRTrace option is unavailable, run the following:

```
/data/data/com.powervr.PVRHub/bin/PVRTraceInstaller -u
```

Note: For more information regarding the extra features of PVRTraceInstaller, see Appendix A.

3.1.2. Tracing Applications

To trace an application using PVRHub:

1. If using network recording, go to the `PVRTrace Record` screen (see Section 4.1.3) and enable `Network Wait` if needed.
2. From the same screen select the desired start and end frames and set the output file.
3. On the `Home` screen (see Section 4.1.1) choose `Start PVRTrace Record` and launch the desired applications that need to be traced.
4. Run PVRTrace GUI and either open the output file or follow the necessary steps for network recording.

Note: The PVRTrace Recording Libraries need to be installed for this functionality to be available.

3.1.3. Profiling Applications

To profile an application using PVRHub, select `Start Profiling` from the GUI.

Note: If the PVRTrace Recording Libraries are installed, API level profiling is also available.

3.2. Linux

PVRHub consists of a number of scripts that, once installed as described in Section 2.2, can be used to profile or trace an application. These scripts are detailed in Table 1.

Table 1. Scripts for profiling or tracing an application

Script	Description
pvr_trace	Used to trace a single application.
pvr_profile	Used to profile a single application.
start_tracing.sh	Used to trace multiple applications.
stop_tracing.sh	Used to stop tracing after start_tracing.sh is used.

3.2.1. Trace a Single Application

To trace a single application complete the following instructions:

1. Open a terminal.
2. Enter the following:

```
pvr_trace <application>
```

Where <application> is the path to the application to be traced.

3. Open the output trace file in PVRTrace GUI.

3.2.2. Trace Multiple Applications

Any application launched between calling start_trace.sh and stop_trace.sh is traced. To trace multiple applications:

1. Open a terminal.
2. Enter the following:

```
source start_tracing.sh
```

3. Launch an application.
4. Exit the application.
5. Open the output file in PVRTrace.
6. Repeat steps 2-4 for desired applications.
7. Enter the following:

```
source stop_tracing.sh
```

3.2.3. Profile a Single Application

To profile a single application complete the following instructions:

1. Open a terminal.
2. Enter the following:

```
pvr_profile <application>
```

Where <application> is the path to the application to be profiled.

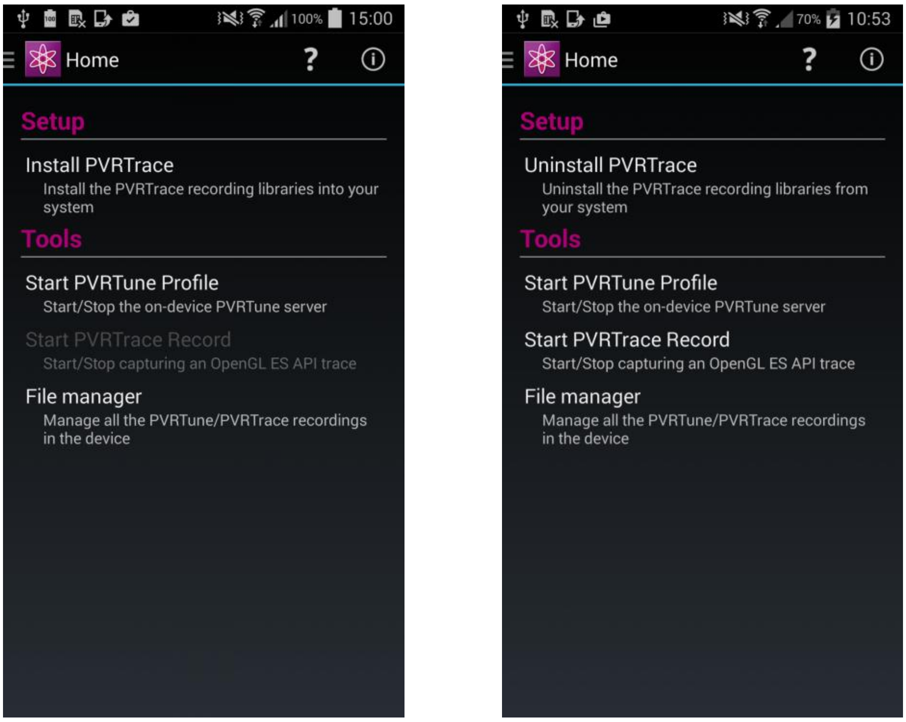
3. Open PVRTune and connect to the device running the application to be profiled.

4. PVRHub Overview

This section provides an overview of PVRHub for Android.

4.1.1. Home Screen

Figure 1 depicts the PVRHub `Home` screen. The various options appearing in the interface are described in Table 2.



100% 15:00

Home

Setup

Install PVRTrace
Install the PVRTrace recording libraries into your system

Tools

Start PVRTune Profile
Start/Stop the on-device PVRTune server

Start PVRTrace Record
Start/Stop capturing an OpenGL ES API trace

File manager
Manage all the PVRTune/PVRTrace recordings in the device

70% 10:53

Home

Setup

Uninstall PVRTrace
Uninstall the PVRTrace recording libraries from your system

Tools

Start PVRTune Profile
Start/Stop the on-device PVRTune server

Start PVRTrace Record
Start/Stop capturing an OpenGL ES API trace

File manager
Manage all the PVRTune/PVRTrace recordings in the device

Figure 1. PVRHub main screen

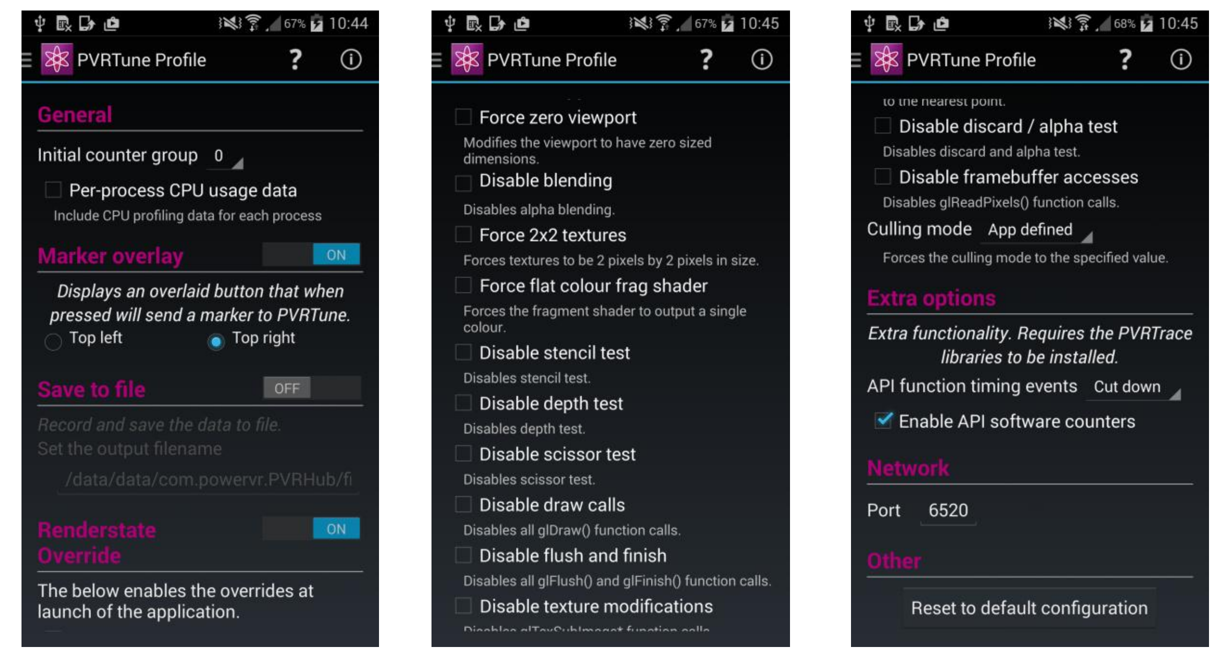
Table 2. Description of options on the PVRHub main screen

Option	Description
`Setup: Install PVRTrace`	Installs the PVRTrace Recording Libraries.
`Setup: Uninstall PVRTrace`	Removes the PVRTrace Recording Libraries.
`Tools: Start PVRTune Profile`	Starts PVRPerfServer. In addition, if any of `Enable Software Counters`, `Enable Renderstate Override`, or `API Function Timing Events` is enabled, the PVRTrace Recording Libraries, if installed, are set to profiling mode. *Note: When profiling, the PVRHub screen is highlighted with a green border.*
`Tools: Stop PVRTune Profile`	Stops PVRPerfServer. This option is only available when profiling whole systems or multiple applications.

Option	Description
`Tools: Start PVRTrace Record`	Starts tracing the selected application or enables tracing for all new applications. *Note: When tracing, the PVRHub screen is highlighted with a red border.*
`Tools: Stop PVRTrace Record`	Disables tracing for all new applications.
`Tools: File manager`	Manage the recordings stored on the device.

4.1.2. PVRTune Profile Screen

Figure 2 depicts the `PVRTune Profile` screen. The various options appearing in the interface are described in Table 3.



67% 10:44

PVRTune Profile

General

Initial counter group 0

□ Per-process CPU usage data

Include CPU profiling data for each process

Marker overlay ON

Displays an overlaid button that when pressed will send a marker to PVRTune.

○ Top left ● Top right

Save to file OFF

Record and save the data to file.

Set the output filename

/data/data/com.powervr.PVRHub/fi

Renderstate Override ON

The below enables the overrides at launch of the application.

67% 10:45

PVRTune Profile

□ Force zero viewport

Modifies the viewport to have zero sized dimensions.

□ Disable blending

Disables alpha blending.

□ Force 2x2 textures

Forces textures to be 2 pixels by 2 pixels in size.

□ Force flat colour frag shader

Forces the fragment shader to output a single colour.

□ Disable stencil test

Disables stencil test.

□ Disable depth test

Disables depth test.

□ Disable scissor test

Disables scissor test.

□ Disable draw calls

Disables all glDraw() function calls.

□ Disable flush and finish

Disables all glFlush() and glFinish() function calls.

□ Disable texture modifications

68% 10:45

PVRTune Profile

to the nearest point.

□ Disable discard / alpha test

Disables discard and alpha test.

□ Disable framebuffer accesses

Disables glReadPixels() function calls.

Culling mode App defined

Forces the culling mode to the specified value.

Extra options

Extra functionality. Requires the PVRTrace libraries to be installed.

API function timing events Cut down

☑ Enable API software counters

Network

Port 6520

Other

Reset to default configuration

Figure 2. PVRTune Profile screen

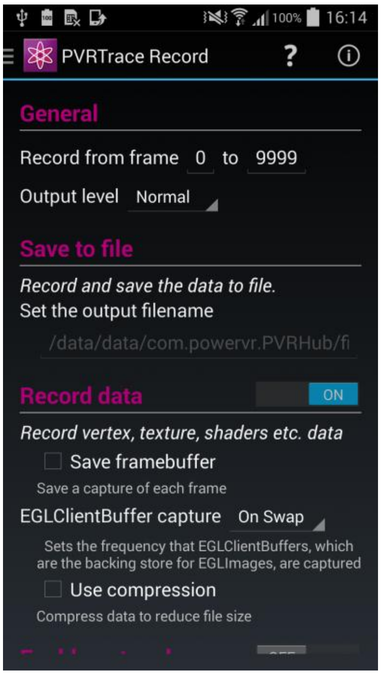
Table 3. Description of options on the PVRTune Profile screen

Option	Description
`General: Initial counter group`	The initial active counter group when PVRPerfServer starts.
`General: Per-process CPU usage data`	Enables or disables the inclusion of CPU profiling data for each process.
`Marker overlay`	Displays an overlaid button that, when pressed, will send a marker to PVRTune.
`Save to file`	This option, when toggled on, saves the profiling data to a file.
`Save to file: Set the output filename`	This states the location and file name of the output PVRTune file.
`Renderstate Override`	If set to `true`, this will enable PVRTune capabilities to toggle different selections of RenderStates. Refer to the "PVRTune User Manual" for further details.

Option	Description
`Renderstate Override: Force zero viewport`	Modifies the viewport to have zero sized dimensions.
`Renderstate Override: Disable blending`	Disables alpha blending.
`Renderstate Override: Force 2x2 textures`	Forces textures to be 2 pixels by 2 pixels in size.
`Renderstate Override: Force flat colour frag shader`	Forces the fragment shader to output a single colour.
`Renderstate Override: Disable stencil test`	Disables the stencil test.
`Renderstate Override: Disable depth test`	Disables the depth test.
`Renderstate Override: Disable scissor test`	Disables the scissor test.
`Renderstate Override: Disable draw calls`	Disables all `glDraw()` function calls.
`Renderstate Override: Disable flush and finish`	Disables all `glFlush()` and `glFinish()` function calls.
`Renderstate Override: Disable texture modifications`	Disables `glTexSubImage2D()` and `glTexSubImage3D()`. This disables the updating of texture sub-regions.
`Renderstate Override: Disable texture filtering`	Disables texture filters. All texture filters are set to the nearest point.
`Renderstate Override: Disable discard/alpha test`	Disables the alpha test.
`Renderstate Override: Disable framebuffer accesses`	Disables `glReadPixels()` function calls.
`Renderstate Override: Culling mode`	Forces the culling mode used in the application.
`Extra options: API function timing events`	Enables or disables the sending of timing data describing the API calls made to the driver. *Note: Setting this to* `Verbose` *may affect performance.*
`Extra options: Enable API software counters`	Enables or disables the sending of API software counters and additional counter set, gained from the API through the use of the PVRTrace Recording Libraries.
`Network: Port`	The port that PVRPerfServer binds to for connecting to PVRTune.
`Other: Reset to default configuration`	Selecting this option resets profile settings to default.

4.1.3. PVRTrace Record Screen

Figure 3 depicts the `PVRTrace Record` screen. The various options appearing in the interface are described in Table 4.



PVRTrace Record

General

Record from frame 0 to 9999

Output level Normal

Save to file

Record and save the data to file.

Set the output filename

/data/data/com.powervr.PVRHub/fi

Record data ON

Record vertex, texture, shaders etc. data

□ Save framebuffer

Save a capture of each frame

EGLClientBuffer capture On Swap

Sets the frequency that EGLClientBuffers, which are the backing store for EGLImages, are captured

□ Use compression

Compress data to reduce file size

Figure 3. PVRTrace Record screen

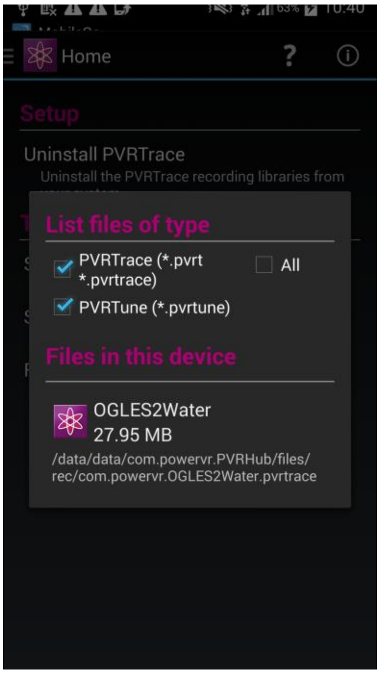
Table 4. Description of options on the PVRTrace Record screen

Option	Description
`General: Record from frame`	Allows selection of the first and last frames of a traced application to record.
`General: Output level`	Amount of information output during recording.
`Save to file`	This option, when toggled on, saves the trace data to a file on the device.
`Save to file: Set the output filename`	This states the location and file name of the output trace file.
`Record data`	With this option toggled on, the PVRTrace Recording Libraries record all buffer contents, all render state changes, every shader, every texture, etc. This option allows the Image Analysis Window of the PVRTrace GUI to be used.
`Record data: Save framebuffer`	With this option ticked, the PVRTrace Recording Libraries save the contents of the framebuffer at the end of every frame.
`Record data: EGLClientBuffer capture`	Defines the frequency that EGLClientBuffers are sampled when recording.
`Record data: Use compression`	Enables compression of the trace data to reduce output file size.
`Enable network`	This option, when toggled on, states whether the PVRTrace Analysis GUI records the trace directly via the network.
`Enable network: Network wait`	With this option ticked, the PVRTrace Recording Libraries cause the application to wait in the first OpenGL call until the PVRTrace Analysis GUI signals to continue.

Option	Description
`Enable network: Set buffer size`	Sets the size of the buffer that is filled before sending data to the PVRTrace Analysis GUI when using Network recording. In general the default option is sufficient, but in some cases this option may need to be changed depending on network settings.
`Other: Append PVRTrace version`	Adds the PVRTrace version to the results of: • `eglQueryString(display, EGL_VERSION)` • `glGetString(GL_RENDERER)`
`Other: Reset to default configuration`	Selecting this option resets API trace settings to default.

4.1.4. File Manager Screen

Figure 4 depicts the `File manager` screen. The various options appearing in the interface are described in Table 5.



List files of type

☑ PVRTrace (*.pvrt *.pvrtrace) □ All

☑ PVRTune (*.pvrtune)

Files in this device

OGLES2Water
27.95 MB
/data/data/com.powervr.PVRHub/files/rec/com.powervr.OGLES2Water.pvrtrace

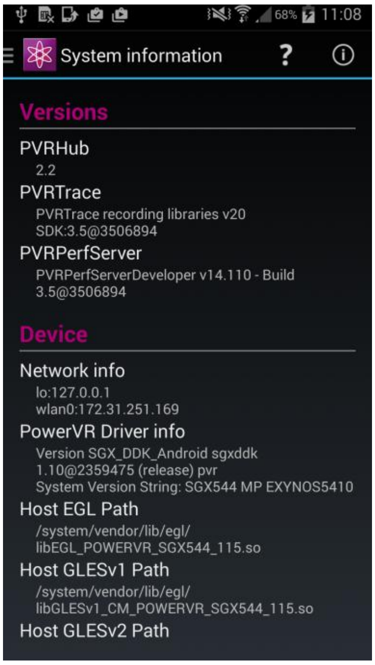
Figure 4. File manager screen

Table 5. Description of options on the File manager screen

Option	Description
`List files of type`	Specifies the file types to list in `Files in this device`.
`Files in this device`	Lists the files in this device that have the specified file types selected in `List files of type`.

4.1.5. System Information Screen

Figure 5 depicts the `System information` screen, which displays versioning and host details.



Versions

PVRHub
2.2

PVRTrace
PVRTrace recording libraries v20
SDK:3.5@3506894

PVRPerfServer
PVRPerfServerDeveloper v14.110 - Build
3.5@3506894

Device

Network info
lo:127.0.0.1
wlan0:172.31.251.169

PowerVR Driver info
Version SGX_DDK_Android sgxddk
1.10@2359475 (release) pvr
System Version String: SGX544 MP EXYNOS5410

Host EGL Path
/system/vendor/lib/egl/
libEGL_POWERVR_SGX544_115.so

Host GLESv1 Path
/system/vendor/lib/egl/
libGLESv1_CM_POWERVR_SGX544_115.so

Host GLESv2 Path

Figure 5. System information screen

5. Contact Details

For further support, visit our forum:
http://forum.imgtec.com

Or file a ticket in our support system:
https://pvrsupport.imgtec.com

To learn more about our PowerVR Graphics SDK and Insider programme, please visit:
http://www.powervrinsider.com

For general enquiries, please visit our website:
http://imgtec.com/corporate/contactus.asp

Appendix A. Extra Features of PVRTraceInstaller

The extra features of the PVRTraceInstaller are:

- **System report generator**: When a report is generated, it is written in `/data/local/tmp/PVRTraceInstaller_report.txt` which contains all the relevant information of the Android system. By executing PVRTraceInstaller with `-g` it will generate a report and terminate.

```
/data/data/com.powervr/PVRHub/bin/PVRTraceInstaller -g
```

- **System recovery**: PVRTraceInstaller can also be used to restore the system in case the trace libraries were wrongly installed. By executing PVRTraceInstaller with `-r` it will restore the system and terminate.

```
/data/data/com.powervr/PVRHub/bin/PVRTraceInstaller -r
```