PVRScope

User Manual

Filename	:	PVRScope.User Manual.mht
Version	:	1.2f (PowerVR SDK 2.03.23.1153)
Issue Date	:	12 Aug 2008
Author	:	PowerVR

Contents

List of Figures

Error! No table of figures entries found.

1. Introduction

PVRScope is a utility library that can be used to access the hardware performance counters in PowerVR SGX. For example, SGXPerfServer uses PVRScope.

The hardware performance counters may be used to optimise software to better utilise the hardware.

1. Usage

PVRScope is supplied as a C/C++ header and a library file.

1.1. Compile

To call the PVRScope functions, add the location of "PVRScope.h" to your include paths and add "`#include <PVRScope.h>`" to the relevant C/C++ file.

1.1.1. Usage

Use of PVRScope is very simple.

1. During initialisation, call `PVRScopeInit()`, this allocates and returns the PVRScope data area.
2. Call `PVRScopeGetPerfCounters()` once to retrieve the number of counters, an array of `SPVRScopeCounter` structs describing the counters, and to allocate memory in which the current counter values will be received. The `SPVRScopeCounter` struct contains the counter's name; a suggested maximum value (0 indicates no suggestion) – e.g. this might be used to scale a graph's Y axis; and the hardware performance counter group it belongs to.
3. As and when desired call `PVRScopeReadPerfCountersThenSetGroup()`; this will fill the counter-value array with the current counter values then change the active performance counter group. Typically the group ID should be 0xffffffff in order to leave the active group unchanged, unless it is desired to change it in which case pass the new group ID.
4. To retrieve the timing data for a render use `PVRScopeTimingRetrieve()`.The timing data struct returned contains: a frame number; a value to indicate the activity described by the timing data, for example 3D core or TA core activity; the start and end times (in hw clocks, hw microseconds and host microseconds); and the clock speed of the SGX. To use `PVRScopeTimingRetrieve()` the generation of the timing data needs to be enabled beforehand by calling `PVRScopeTimingEnable()`. This function is also used to disable it.
5. Finally, call `PVRScopeDeInit()` to shutdown PVRScope and free the allocated memory.

1.2. Link

On Linux, link your application to "libpvrscope.a" or "libpvrscope.so" as applicable, and on other platforms link your application to "PVRScope.lib".