PVRVFrame

User Manual

Filename	:	PVRVFrame.User Manual.1.0.5.External.doc
Version	:	1.0.5 External Issue (Package: POWERVR SDK REL_3.0@2149525)
Issue Date	:	04 Sep 2012
Author	:	Imagination Technologies Ltd

Contents

List of Figures

1. Introduction

PVRVFrame is an emulation layer for desktop platforms that provides an OpenGL ES and EGL environment mimicking that available on PowerVR-enabled devices. It works by redirecting OpenGL ES API calls to the underlying OpenGL desktop implementation present on the development system. PVRVFrame is aimed at developers writing OpenGL ES 1.1 and OpenGL ES 2.0 applications for PowerVR-enabled embedded devices and is designed to mitigate the need to work directly with this hardware.

PVRVFrame is not per-pixel accurate, nor a performance simulator. The performance obtained when running OpenGL ES applications with PVRVFrame is not indicative of the performance obtained when running on real consumer hardware. PVRVFrame gives a qualitative preview of an OpenGL ES application only.

PVRVFrame is available for OpenGL ES 1.1, Common profile and OpenGL ES 2.0.

PVRVFrame must be used with the corresponding version of the Khronos OpenGL ES header files. These header files should be downloaded from the Khronos website; however versions are also bundled with PVRVFrame for convenience.

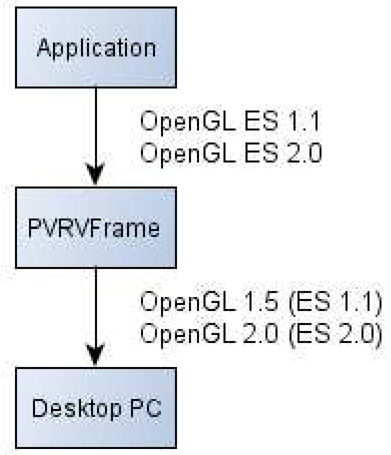

Figure 1-1 PVRVFrame Layer Overview

2. Package Contents

2.1. Windows

2.1.1. Open GL ES 1.1

`libEGL.dll:` PVRVFrame driver file for EGL1.4.

`libgles_cm.dll:` PVRVFrame driver file for OpenGL ES Common profile.

`libEGL.lib:` Import (“stub”) library for use when linking against libEGL.dll.

`libgles_cm.lib:` Import (“stub”) library for use when linking against libgles_cm.dll.

2.1.2. OpenGL ES 2.0

`libEGL.dll:` PVRVFrame driver file for EGL1.4.

`libGLESv2.dll:` PVRVFrame driver file for OpenGL ES 2.0.

`libEGL.lib:` Import (“stub”) library for use when linking against libEGL.dll.

`libGLESv2.lib:` Import (“stub”) library for use when linking against libGLESv2.dll.

2.2. Linux

2.2.1. OpenGL ES 1.1

`libEGL.so:` PVRVFrame driver file for EGL1.4.

`libgles_cm.so:` PVRVFrame driver file for OpenGL ES Common profile.

2.2.2. OpenGL ES 2.0

`libEGL.so:` PVRVFrame driver file for EGL1.4.

`libGLESv2.so:` PVRVFrame driver file for OpenGL ES 2.0.

2.3. Mac OS

2.3.1. OpenGL ES 1.1

`libEGL.dylib:` PVRVFrame driver file for EGL 1.4.

`libGLES_CM.dylib:` PVRVFrame driver file for OpenGL ES 1.1

2.3.2. OpenGL ES 2.0

`libEGL.dylib:` PVRVFrame driver file for EGL 1.4.

`libGLESv2.dylib:` PVRVFrame driver file for OpenGL ES 2.0

3. Requirements

3.1. Hardware

PVRVFrame requires hardware compatible with the underlying OpenGL implementation of the version of OpenGL ES being targeted.

The following is an indication of the requirements for emulation:

Emulating	Requires
OpenGL ES 1.1	OpenGL 1.5 +
OpenGL ES 2.0	OpenGL 2.0 +
OpenGL ES 3.0	OpenGL 3.3 +
OES_packed_depth_stencil	EXT_packed_depth_stencil
OES_framebuffer_object	EXT_framebuffer object
OES_texture_float	ARB_texture_float
OES_texture_half_float	ARB_texture_float
OES_vertex_half_float	ARB_half_float_pixel
OES_vertex_array_object	ARB_vertex_array_object
EXT_shader_texture_lod	ARB_shader_texture_lod
IMG_multisampled_render_to_texture	EXT_framebuffer_multisample

4. Installation

4.1. Windows

The DLL files that match the version of OpenGL ES that is to be targeted must be in a folder in the %PATH% environment variable, or in the directory the application is running from.

With the DLL files in place, a Visual Studio project can be set up as follows:

1. Create a new Visual C project, adding source and include files as necessary.

2. Download the Khronos include files:
    1. `‘gl.h’` and `‘egl.h’` for OpenGL ES 1.1
    2. `‘gl2.h’` and `‘egl.h’` for OpenGL ES 2.0

    OpenGL ES 1.1**:**

    It is good practice to store `‘gl.h’` and `‘egl.h’` in `‘GLES\’` and `‘EGL\’` subfolders respectively, ensuring these folders are in the include path.

    OpenGL ES 2.0:

    It is good practice to store `‘gl2.h’` and `‘egl.h’` in `‘GLES2\’ and ‘EGL\’` subfolders respectively, ensuring these folders are in the include path.

3. OpenGL ES 1.1:

    Copy: `OpenGL ES 1.1 header files`

    From: `<SDKPackage>\Builds\OGLES\Include\`

    To: Your project’s include path.

    OpenGL ES 2.0:

    Copy*:* `OpenGL ES 2.0 header files`

    From: `<SDKPackage>\Builds\OGLES2\Include\`

    To: Your projects include path.

4. Link the project to the supplied OGLES/OGLES2 libraries as described in Section 2 Package contents.

4.2. Linux & Mac OS

To set up a new project to run with PVRVFrame, follow the steps below:

1. Create a new makefile, adding source and include files as necessary.

2. Download the Khronos include files:
   1. `'gl.h'` and `'egl.h'` for OpenGL ES 1.1
   2. `'gl2.h'` and `'egl.h'` for OpenGL ES 2.0

   OpenGL ES 1.1**:**

   It is good practice to store `'gl.h'` and `'egl.h'` in `'GLES\'` and `'EGL\'` subfolders respectively, ensuring these folder are in the include path.

   OpenGL ES 2.0:

   It is good practice to store `'gl2.h'` and `'egl.h'` in a `'GLES2\' and 'EGL\'` subfolders respectively, ensuring these folders are in the include path.

3. OpenGL ES 1.x:

   Copy: `OpenGL ES 1.1 header files`

   From: `<SDKPackage>\Builds\OGLES\Include\`

   To: Your project's include path.

   OpenGL ES 2.0:

   Copy*:* `OpenGL ES 2.0 header files`

   From: `<SDKPackage>\Builds\OGLES2\Include\`

   To: Your projects include path.

4. Link the project to the supplied OGLES/OGLES2 libraries.

5. Run the following terminal command:

   Linux: `'export LD_LIBRARY_PATH=<lib folder >;$LD_LIBRARY_PATH'`

   Mac OS: `'export DYLD_LIBRARY_PATH=<lib folder>;$DYLD_LIBRARY_PATH'`

It is possible to check that the correct version of the libraries are being used and that the location of these libraries is set correctly; to do this run the following commands on an application built against the PVRVFrame libraries:

Linux: `'ldd <path to application>'`

Mac OS: `'otool –L <path to application>'`

5. SGX Supported Extensions

5.1. OpenGL ES 1.x

Extension	Comments
GL_EXT_multi_draw_arrays	
GL_IMG_read_format	
GL_IMG_texture_compression_pvrtc	
GL_IMG_texture_format_BGRA8888	
GL_OES_blend_equation_seperate	
GL_OES_blend_func_seperate	
GL_OES_blend_subtract	
GL_OES_byte_coordinates	
GL_OES_compressed_ETC1_RGB8_texture	
GL_OES_compressed_paletted_texture	
GL_OES_depth24	
GL_OES_draw_texture	
GL_OES_extended_matrix_palette	
GL_OES_fixed_pont	
GL_OES_framebuffer_object	GL_EXT_framebuffer_object support required
GL_OES_mapbuffer	
GL_OES_matrix_get	
GL_OES_matrix_palette	
GL_OES_point_size_array	
GL_OES_point_sprite	
GL_OES_query_matrix	
GL_OES_read_format	
GL_OES_rgb8_rgb8	
GL_OES_single_precision	
GL_OES_stencil_wrap	
GL_OES_stencil8	
GL_OES_texture_cube_map	
GL_OES_texture_env_crossbar	
GL_OES_texture_mirrored_repeat	
GL_EXT_multi_draw_arrays	

5.2. OpenGL ES 2.0

Extension	Comments
GL_EXT_discard_framebuffer	
GL_EXT_multi_draw_arrays	
GL_EXT_shader_texture_lod	
GL_IMG_multisampled_render_to_texture	GL_EXT_framebuffer_multisample support required GL_EXT_framebuffer_blit support required
GL_IMG_read_format	
GL_IMG_texture_compression_pvrtc	
GL_IMG_texture_format_BGRA8888	
GL_IMG_texture_npot	
GL_OES_compressed_ETC1_RGB8_texture	
GL_OES_depth_texture	
GL_OES_depth24	
GL_OES_element_index_uint	
GL_OES_frament_precision_high	
GL_OES_mapbuffer	
GL_OES_packed_depth_stencil	
GL_OES_required_internalformat	
GL_OES_rgb8_rgb8	
GL_OES_texture_float	
GL_OES_texture_half_float	
GL_OES_vertex_array_object	
GL_OES_vertex_half_float	

6. MBX Supported Extensions

6.1. OpenGL ES 1.x

Extension	Comments
GL_OES_byte_coordinates	
GL_OES_fixed_point	
GL_OES_single_precision	
GL_OES_matrix_get	
GL_OES_read_format	
GL_OES_compressed_paletted_texture	
GL_OES_point_sprite	
GL_OES_point_size_array	
GL_OES_matrix_palette	
GL_OES_draw_texture	
GL_OES_query_matrix	
GL_IMG_read_format	
GL_IMG_texture_compression_pvrtc	
GL_IMG_texture_format_BGRA8888	
GL_EXT_multi_draw_arrays	

7. Current Limitations

Precision Qualifiers

PVRVFrame does not support precision qualifiers in shaders. Shaders using precision modifiers will still function but the precision qualifiers are ignored; all precisions are assumed to be highp.

Thread Safety

PVRVFrame is not currently guaranteed to be thread safe.

Binary Shaders

Binary shaders are not supported.

Orthogonal Tex Env Combine

Orthogonal Tex Env Combine is not supported.

PBuffer

PBuffer is not supported on Mac OS. It is supported on Windows and Linux.

Colour Mask

Colour masking is not supported on the PowerVR MBX profile.

Texture Projection

Texture projection is not supported on the PowerVR MBX profile.

Context

The same context cannot be used to render to two different windows under Windows. This functions correctly under Linux and Mac OS.

glRenderBufferStorage

Calling glRenderBufferStorage with GL_STENCIL_INDEX8 does not work; DEPTH24_STENCIL8 (requires OES_packed_depth_stencil) should be used instead.

8. Related Materials

Software

- PVRTune
- PVRTrace

Documentation

- PVRTune User Manual
- PVRTrace User Manual
- PowerVR SDK User Guide

9. Contact Details

For further support contact:

devtech@imgtec.com

PowerVR Developer Technology
Imagination Technologies Ltd.
Home Park Estate
Kings Langley
Herts, WD4 8LZ
United Kingdom

Tel: +44 (0) 1923 260511
Fax: +44 (0) 1923 277463

Alternatively, you can use the PowerVR Insider forums:

www.imgtec.com/forum

For more information about PowerVR or Imagination Technologies Ltd. visit our web pages at:

www.imgtec.com