QUALCOMM
CDMA Technologies

Adreno™ Unified Emulator User Guide

80-VN411-1 A

August 24, 2011

Submit technical questions at:
https://support.cdmatech.com

QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121-1714
U.S.A.

Terms and Conditions of Usage

This document may refer to or describe parts whose manufacture, use, sale, offer for sale, or importation into the United States are limited or prohibited by one or more U.S. Federal Court injunctions against QUALCOMM Incorporated. This document is not to be construed as an offer to sell such parts for use or importation into the U.S., nor should it be construed as assistance in making, using, selling, offering to sell, or the importation of any product in the U.S. containing such parts. This document is intended solely to provide information for those products that are outside the scope of the injunction. Recipient's download and/or use of the information in this document constitutes agreement with these terms.

Contents

Figures

Tables

Revision history

Revision	Date	Description
A	Jan 2009	Initial release
B	Feb 2011	Revision – removed deprecated ES Control features
C	August 2011	Revision – added S3D to ES Control

1 Introduction

1.1 Purpose

Qualcomm's OpenGL® ES 2.0 Emulator is designed to emulate functionality that will be exposed by future mobile Qualcomm hardware. The purpose of this emulator is to aid developers in their design of OpenGL ES 2.0 games and applications.

1.2 Scope

This document is for developers that design OpenGL ES 2.0 games and applications.

1.3 Conventions

Function declarations, function names, type declarations, and code samples appear in a different font, e.g., `#include`.

1.4 References

Reference documents, which may include QUALCOMM®, standards, and resource documents, are listed in Table 1-1. Reference documents that are no longer applicable are deleted from this table; therefore, reference numbers may not be sequential.

Table 1-1 Reference documents and standards

Ref.	Document	
	Qualcomm	
Q1	*Application Note: Software Glossary for Customers*	CL93-V3077-1
	Standards	
S1	*OpenGL ES*	http://www.khronos.org/opengles/
S2	*Khronos OpenGL ES API Registry*	http://www.khronos.org/registry/gles/
	Resources	
R1	*ATI Products and Technologies*	http://ati.amd.com/products/

1.5 Technical assistance

For assistance or clarification on information in this guide, submit a case to Qualcomm CDMA Technologies at https://support.cdmatech.com/.

If you do not have access to the CDMATech Support Service website, register for access or send email to support.cdmatech@qualcomm.com.

1.6 Acronyms

For definitions of terms and abbreviations, refer to [Q1].

2 Overview

Qualcomm's OpenGL ES 2.0 Emulator is designed to emulate graphics functionality on mobile Qualcomm hardware. The purpose of this emulator is to aid developers in their design of OpenGL ES 2.0 games and applications.

The emulator has five major features:

- Support of core OpenGL ES 2.0 functionality
- Support for many important OpenGL ES 2.0 extensions
- Support of core OpenGL ES 1.x functionality
- Support for many OpenGL ES 1.x extensions
- Support for EGL 1.3

Refer to [S1] for OpenGL ES and EGL specifications.

3 Prerequisites

Check the following before running the emulator:

- Recommended hardware is ATI Radeon X1300 or higher. ATI Radeon 9500 is the minimum required hardware. Equivalent hardware from other vendors may work, but it is not tested.
- If you are running ATI graphics hardware, install the latest Catalyst drivers from [R1]. This is important, as there have been recent fixes that affect the emulator.
- To run the emulator, Visual Studio 2008 must be installed.
- This version of the emulator requires Windows XP, Windows Vista or Windows 7.

4 Architecture

The emulator works by running on top of the desktop OpenGL 2 driver. Figure 4-1 illustrates the different components of the emulator, as well as interactions between these components.

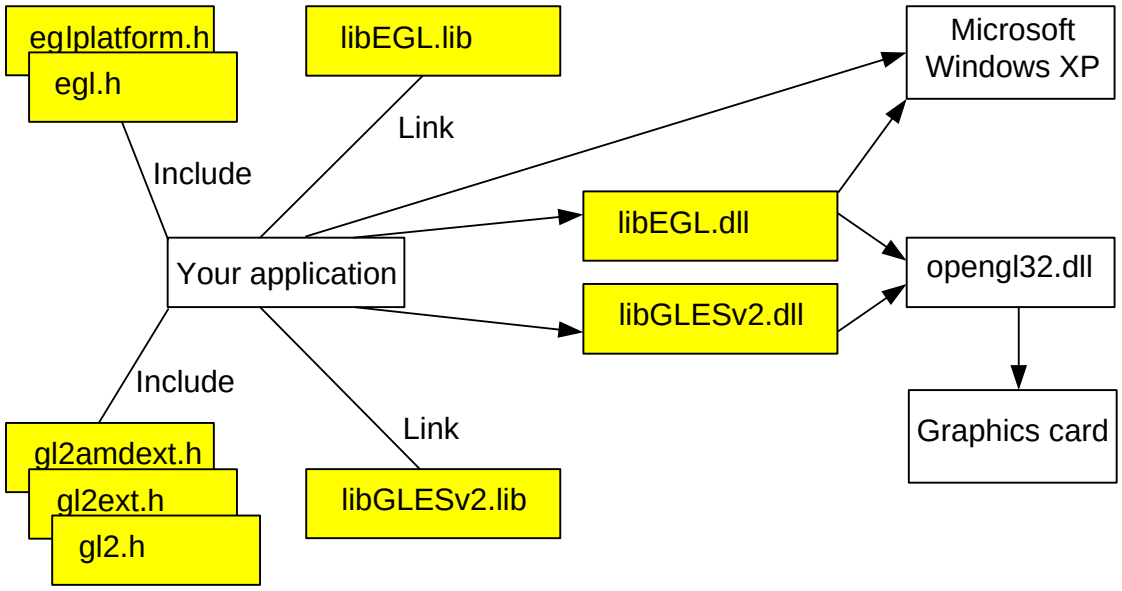

Figure 4-1 Components of OpenGL ES 2.0

In the figure:

- All the yellow-highlighted sections are part of the OpenGL ES 2.0 Emulator
- egl.h and eglplatform.h are standard EGL header files from Khronos that allow your game to use EGL 1.3
- gl2.h and gl2ext.h are standard OpenGL ES 2.0 header files from Khronos that allow your game to use core OpenGL ES 2.0 and some standard extensions.
- gl2amdext.h gives access to additional OpenGL ES 2.0 extensions
- Link your game against libEGL.lib and libGLESv2.lib (real hardware will have similarly named libraries to link your game against)
- libEGL.dll is the emulator for EGL 1.3 (real hardware equivalent will be the EGL 1.3 driver)
- libGLESv2.dll is the emulator for OpenGL ES 2.0 (real hardware equivalent will be the OpenGL ES 2.0 driver)

5 Extensions

Following is a current list of supported extensions supported by the emulator. This list includes nonratified extensions whose support is dependent on ratification:

- GL_AMD_alpha_test
- GL_AMD_compressed_3DC_texture
- GL_AMD_compressed_ATC_texture
- GL_AMD_logic_op
- GL_AMD_shader_binary_Z400
- GL_EXT_texture_filter_anisotropic
- GL_EXT_texture_type_2_10_10_10_REV
- GL_OES_compressed_ETC1_RGB8_texture
- GL_OES_compressed_paletted_texture
- GL_OES_depth_texture
- GL_OES_depth24
- GL_OES_element_index_uint
- GL_OES_fragment_precision_high
- GL_OES_packed_depth_stencil
- GL_OES_rgb8_rgba8
- GL_OES_texture_3D
- GL_OES_texture_float
- GL_OES_texture_float_linear
- GL_OES_texture_half_float
- GL_OES_texture_half_float_linear
- GL_OES_texture_npot
- GL_OES_vertex_half_float
- GL_OES_vertex_type_10_10_10_2
- GL_OES_standard_derivatives

For extension specifications of all ratified extensions, refer to [S2].

6 Caveats

Following is a list of emulator caveats:

- There are many cases where the emulator's software workload is different than what will exist on real ES hardware. Because of this, do not use the emulator to judge software performance.
- All precision qualifiers in shaders are ignored and the default shader precision for your desktop graphics card is used instead.
- Full OpenGL ES 2.0 error checking is not supported, but it is very close.
- Shader invariance rules are ignored.
- Varying packing rules are not followed.
- Shader constant expressions are not evaluated.
- Since the emulator is built on top of desktop OpenGL, any bugs in your desktop OpenGL driver will affect the emulator. As of this document's writing, the FBO implementations in desktop OpenGL drivers are incomplete.
- Real devices may support only 16-bit or 32-bit display buffers. The emulator only works under 32-bit display.
- The max values for Qualcomm's future GPU are used for constant states like MAX_TEXTURE_SIZE, MAX_VARYING_VECTORS, ...
- MAX_VERTEX_ATTRIBUTES is a special case where 15 is exported instead of 16 like future hardware supports.
- Vertex texture fetches are not supported.
- ATI desktop hardware does not support non-power of two textures with a wrap mode other than clamp. These will punt to software and be unbearably slow on the emulator even though the real mobile hardware supports this.

7 ES_Control – OpenGL ES 2.0 Emulator Control Panel

ES_Control (see Figure 7-1) is a control panel used to set configuration options for Qualcomm's OpenGL ES Emulator. Qualcomm's OpenGL ES Emulator is designed to control various aspects OpenGL ES 2.0 application running on a Qualcomm handheld GPU.

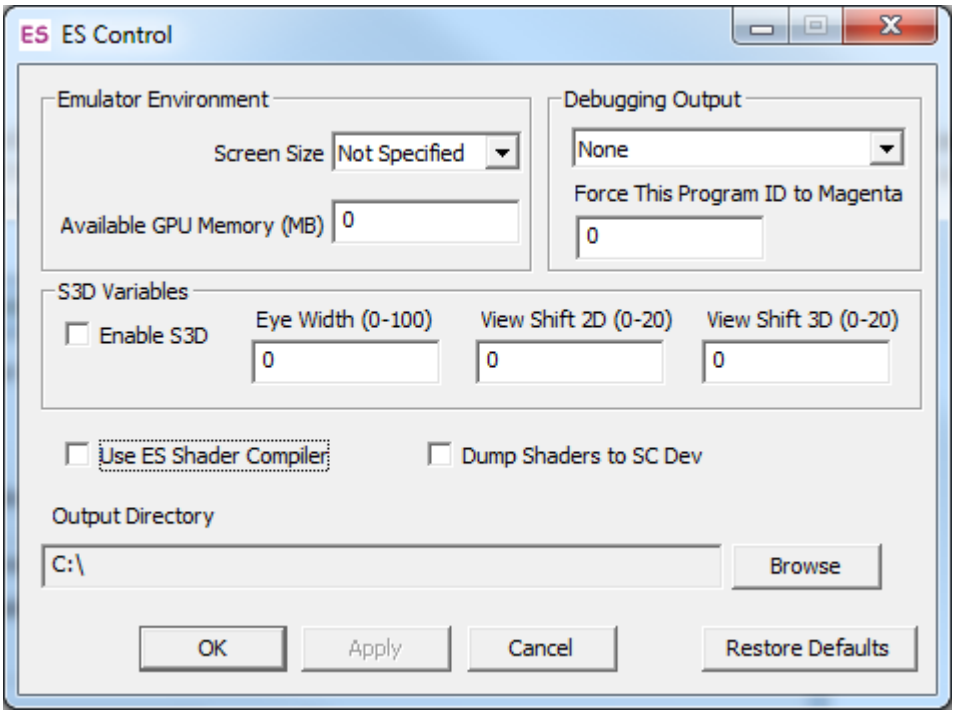

Figure 7-1 ES_Control graphical interface

7.1 Emulator environment

7.1.1 Screen size

This is used to select the desired screen width and height.

The width and height are used for max viewport dimensions and max render buffer sizes. When Not Specified is selected, the max sizes are 2048.

7.1.2 Available GPU memory (MB)

This is used to set the amount of memory available to the GPU for textures, VBOs, FBOs, and color/depth/stencil buffers (0 means there is no limit).

With desktop GPUs the drivers can swap textures and VBOs out to system memory when video memory is exhausted. However, on mobile platforms there is a limited amount of memory that can be used. Whenever the amount of memory used exceeds this number a GL_OUT_OF_MEMORY error is generated.

7.1.3 Debugging output

This is used to check for emulator errors and OpenGL ES errors at the end of every entry point and pipe these errors to the selected output method:

- None – This method performs no error checking. This mode provides best performance.
- Debugger Break – When a debugger is attached an int3 is triggered on any error, which makes it possible to break on the entry point causing the error. Text describing any errors is also printed to the debugger output. If a debugger is not running the application will seem to crash on the first error detected.
- Debugger Output – When a debugger is attached, text describing any errors is printed to the debugger output.
- Text File – A text file called ES2_Debug.txt is created and any errors are written into this file.
- App Window – The most recent error detected is displayed at the top of the application's window.

7.1.4 Force this program ID to magenta

This is used to change the fragment shader that is attached to the selected program ID to output magenta.

This is useful for debugging shaders. The program ID can be most easily attained by dumping detailed performance logs to a file and viewing the ES2_PerfDumpShaders.txt file.

7.1.5 S3D Variables

These variables are used to render an ES 2.0 emulated application in stereo. This current implementation will split the application screen in half and render the output on each side, one for the left eye and one for the right. The variables that are defined in this section of the ES Control tool are used to globally assign the spatialization of the images. Note that it is up to the application to increase the width of the application if more horizontal resolution is required.

7.1.5.1 Enable S3D

Enables the S3D mode on. This is global; any ES 2.0 emulated application will be affected by this.

7.1.5.2 Eye Width

This is the amount of eye separation between the two images. The greater the value, the further apart the 'eyes' are (increases the overall stereoscopic nature of the resulting image, but can lead to unnatural effects at high values).

7.1.5.3 View Shift 2D

The amount to shift 2D polygons by (i.e. polys that are rendered with the z-buffer turned off).

7.1.5.4 View Shift 3D

The amount to shift 3D polygons by (i.e. polys that are rendered with the z-buffer turned on).

7.1.6 Use ES shader compiler

This is used to enable true ES 2.0 shader parsing and validates that shaders will run on Qualcomm's ES 2.0 hardware.

When this option is set, shaders are run through the same parser and shader compiler that will be used with real hardware. This should be turned off only if a shader that should be compiling and linking fails. If such a case is encountered, report a bug to Qualcomm's handheld developer relations.

7.1.7 Dump shaders to SC Dev

This is used to create a text file called ES2_DumpShadersScDev.txt. All the shader pairs are added to this file in a format readable by scltool.

7.1.8 Output directory

This is used to set the output directory for all out files generated by the emulator.

These files include ES2_Debug.txt, ES2_PerfDumpSumary.txt, ES2_PerfDumpFrames.txt, and ES2_PerfDumpShaders.txt.