Mali™ GPU Shader Library

Version: 1.0

User Guide

ARM®

ARM DUI 0510A (ID101409)

Mali GPU Shader Library
User Guide

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
14 October 2009	A	Non-Confidential	First release for v1.0

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Unrestricted Access is an ARM internal classification.

Product Status

The information in this document is final, that is for a developed product.

Web Address

`http://www.arm.com`

Contents
Mali GPU Shader Library User Guide

List of Tables
Mali GPU Shader Library User Guide

List of Figures
Mali GPU Shader Library User Guide

Preface

This preface introduces the *Mali GPU Shader Library User Guide*. It contains the following sections:

About this book

This is the *Mali GPU Shader Library User Guide*. It provides guidelines for using the Shader Library to assist in the development of 2D and 3D graphics applications for Mali GPUs. This book is part of a suite belonging to the Mali Developer Tools.

Intended audience

This guide is written for system integrators and software developers who are writing OpenGL ES or OpenVG applications for the PC, using the Windows XP or Linux operating system, and might want to progress onto writing applications for the Mali GPU range in the future.

Using this book

This book is organized into the following chapters:

Chapter 1 ***Introduction***

Read this for an introduction to the Shader Library.

Chapter 2 ***Installation***

Read this for a description on how to install and start the tool.

Chapter 3 ***The Shader Library***

The Shader Library consists of information about how to use the Shader Library.

Glossary Read this for definitions of terms used in this book.

Typographical Conventions

The typographical conventions are:

italic	Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
`monospace`	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

mono`space`	Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`monospace italic`	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
`monospace bold`	Denotes language keywords when used outside example code.
< and >	Enclose replaceable terms for assembler syntax where they appear in code or code fragments. For example: `MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>`

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, `http://infocenter.arm.com`, for access to ARM documentation.

ARM publications

This guide contains information that is specific to the Mali GPU Software Tools. See the following documents for other relevant information:

- *Mali GPU Developer Tools Technical Overvi*ew (ARM DUI 501)
- *Mali GPU Performance Analysis Tool User Guide* (ARM DUI 0502)
- *Mali GPU Texture Compression Tool User Guide* (ARM DUI 0503)
- *Mali GPU Shader Development Studio User Guide* (ARM DUI 0504)
- *Mali GPU Demo Engine User Guide* (ARM DUI 0505)
- *OpenGL ES 1.1 Emulator User Guide* (ARM DUI 0506)
- *Mali GPU Binary Asset Exporter User Guide* (ARM DUI 0507)
- *OpenGL ES 2.0 Emulator User Guide* (ARM DUI 0511)
- *Mali GPU Offline Shader Compiler User Guide* (ARM DUI 0513).

Other publications

This section lists relevant documents published by third parties:

- *OpenGL ES 1.1 Specification* at `http://www.khronos.org`.
- *OpenGL ES 2.0 Specification* at `http://www.khronos.org`.
- *OpenGL ES Shading Language Specification* at `http://www.khronos.org`.
- *OpenVG 1.1 Specification* at `http://www.khronos.org`.

- *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2* (5th Edition, 2005), Addison-Wesley Professional. ISBN 0-321-33573-2.
- *OpenGL Shading Language* (2nd Edition, 2006), Addison-Wesley Professional. ISBN 0-321-33489-2.

Feedback

ARM welcomes feedback on this product and its documentation.

Feedback on this product

If you have any comments or suggestions about this product then contact `malidevelopers@arm.com` and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to `errata@arm.com`. Give:

- the title
- the number, ARM DUI 0510A
- the page numbers to which your comments apply
- a concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Chapter 1
Introduction

This chapter provides introductory information about the Mali GPU Shader Library.

It contains the following sections:

1.1 About the Mali GPU Shader Library

The Mali GPU Shader Library is a collection of example shader programs. These examples contain the ESSL vertex and fragment shader source files for shader programs and other information to help you start developing shader programs for the Mali GPU. You can use the shader programs as they are provided, modify them to suit your requirements or use them to learn and develop your own.

.

Chapter 2
Installation

This chapter provides information about installing the Shader Library. These procedure is described in:

- *Installing the Shader Library on Microsoft Windows* on page 2-2
- *Installing the Shader Library on Linux* on page 2-4.

2.1 Installing the Shader Library on Microsoft Windows

This section describes how to install the Shader Library on Microsoft Windows. It contains the following sections:

- *Installation requirements for the Shader Library on Microsoft Windows*
- *Installation procedure for the Shader Library on Microsoft Windows*.

Note

The Shader Library has been tested successfully on a 32-bit computer.

2.1.1 Installation requirements for the Shader Library on Microsoft Windows

To install the Shader Library on Microsoft Windows, you require:

- Microsoft Windows XP Professional Version 2002, service pack2
- 3 to 4MB is required for the shaders.

2.1.2 Installation procedure for the Shader Library on Microsoft Windows

To install the Shader Library on Microsoft Windows:

1. Locate the Mali Developer Download Center website at:

   `http://www.malideveloper.com`

2. Select the Shader Library package to download.

3. Run the file `Mali_GPU_Shader_Library_WinXP_v`*`m`*`.`*`n`*`.exe`

   where:

   m identifies the major version

   n identifies the minor version.

4. Select the required installation options and then click **Finish** to complete the installation.

By default, the Shader Library is installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Shader Library v`*`m`*`.`*`n`*

The Shader examples are installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Shader Library v`*`m`*`.`*`n`*`\ShaderExamples`

The *OrangeBook* examples are installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Shader Library v`*`m`*`.`*`n`*`\OrangeBook`

Note

OrangeBook examples are the high quality shaders described in the *OpenGL Shading Language 2nd edition*.

2.2 Installing the Shader Library on Linux

This section describes how to install the Shader Library on Linux. It contains the following sections:

- *Installation requirements for the Shader Library on Linux*
- *Installation procedure for the Shader Library on Linux*.

Note

The Shader Library has been tested successfully on a 32-bit computer.

2.2.1 Installation requirements for the Shader Library on Linux

To install the Shader Library on Linux, you require:

- Red Hat Enterprise Linux, Release 4
- `GNU tar` version 1.13 or higher
- 3 to 4MB is required for the shaders.

2.2.2 Installation procedure for the Shader Library on Linux

To install the Shader Library on Linux:

1. Locate the Mali Developer Download Center website at:

   `http://www.malideveloper.com`

2. Download the following package:

   `Mali_GPU_Shader_Library_RHEL4_v`*`m`*`.`*`n`*`.tar.gz`

   where:

   m identifies the major version

   n identifies the minor version.

3. To decompress the file:

   - open a command terminal and navigate to the directory where you have downloaded the package
   - type the following command:
     `tar -zxvf Mali_GPU_Shader_Library_RHEL4_v`*`m`*`.`*`n`*`.tar.gz`

By default, the Shader Library is installed in:

`ARM/Mali_Developer_Tools/Mali_GPU_Shader_Library_v`*`m`*`.`*`n`*

The Shader examples are installed in:

`ARM/Mali_Developer_Tools/Mali_GPU_Shader_Library_v`*`m.n`*`/ShaderExamples`

The *OrangeBook* examples are installed in:

`ARM/Mali_Developer_Tools/Mali_Shader_Library_v`*`m.n`*`/OrangeBook`

Note

OrangeBook examples are high quality shaders described in the *OpenGL Shading Language 2nd edition*.

Chapter 3
The Shader Library

This chapter describes how to use the Shader Library Tool. It contains the following sections:

3.1 About the Shader Library

The Shader Library consists of an extensive list of shader program examples and other related information. The purpose of the shader library is to assist with developing ESSL shader programs for the Mali GPUs, that is, it is not a library for the ARM linker.

Note

Shaders in the Shader Library render a variety of graphics effects with different levels of complexity.You can use the Mali Shader Development Studio to view the effects of these shader programs. Shader programs make use of the `shaders.shaderconfig` file to simplify the task of initializing and running a shader program on a particular object in Mali Shader Development Studio. See the *Mali GPU Shader Development Studio User Guide* for more information on how to create new eclipse projects from existing shader sources in shader library. You can also use these shader programs to develop application programs that use Mali Demo Engine and standard OpenGL ES API functions. See the *Mali GPU Demo Engine User Guide* for more information on using shaders.

3.1.1 OpenGL ES 2.0

Applications, based on OpenGL ES 2.0, must implement functionality that was available by default in OpenGL ES 1.1, such as lighting and texturing, in shader programs. This approach is particularly suitable for embedded device applications where resources are limited, because it provides greater flexibility in creating required visual effects using optimized shader programs, while minimizing the complexity of creating software drivers.

Note

When creating the OpenGL ES 2.0 API, to prevent duplicate functionality, Khronos removed a large part of fixed functionality from the graphics pipeline. This means that now graphics hardware can be programmed using shader programs supplied with applications. Consequently, when you draw objects using OpenGL ES 2.0, you must provide shader programs that specify how to draw the object.

3.2 Shader Effects

Table 3-1 shows the available shaders effects contained in the *ShaderExamples* and *OrangeBookExamples* folders. All of these shaders are directly supported by the Shader Development Studio using corresponding `shaders.shaderconfig` files.

Note

No existing applications in the Demo Engine support these shaders at present.

Table 3-1 Shader Library support

ShaderExamples folder	OrangeBookExamples folder
Demo 00 - Blank, Figure 3-1 on page 3-5	CH06 - Brick Effect, Figure 3-21 on page 3-15
Demo 01 - Red all Over, Figure 3-2 on page 3-5	CH10 - Cube Map, Figure 3-22 on page 3-15
Demo 02 - Normal Viewer, Figure 3-3 on page 3-6	CH10 - Earth, Figure 3-23 on page 3-16
Demo 03 - Raspberry Ripple, Figure 3-4 on page 3-6	CH11 - Glyph Bomb, Figure 3-24 on page 3-16
Demo 04 - Lighting, Figure 3-5 on page 3-7	CH11 - Lattice, Figure 3-25 on page 3-17
Demo 05 - Vertex Ripple, Figure 3-6 on page 3-7	CH11 - Stripe, Figure 3-26 on page 3-17
Demo 06 - Bump Mapping, Figure 3-7 on page 3-8	CH11 - Toy Ball, Figure 3-27 on page 3-18
Demo 07- Environment Mapping, Figure 3-8 on page 3-8	CH12 - Hemi Light, Figure 3-28 on page 3-18
Demo 08 - Textures, Figure 3-9 on page 3-9	CH12 - Spherical Harmonics Lighting, Figure 3-30 on page 3-19
Demo 09 - Edges, Figure 3-10 on page 3-9	CH12 - UberLight, Figure 3-30 on page 3-19
Demo 10 - Blended Textures, Figure 3-11 on page 3-10	CH14 - Chromatic Aberration, Figure 3-31 on page 3-20
Demo 11 - Bump and Environment Mapping, Figure 3-12 on page 3-10	CH14 - Diffraction, Figure 3-32 on page 3-20
Demo 12 - Gooch, Figure 3-13 on page 3-11	CH14 - DiffractionPerFragment, Figure 3-32 on page 3-20
Demo 13 - Mandelbrot, Figure 3-14 on page 3-11	CH14 - Reflection and Refraction, Figure 3-34 on page 3-21
Demo 14 - Brick Effect, Figure 3-16 on page 3-12	CH14 - WardBRDF, Figure 3-35 on page 3-22

Table 3-1 Shader Library support (continued)

ShaderExamples folder	OrangeBookExamples folder
Demo 15 - TextureCoordView, Figure 3-16 on page 3-12	
Demo 16 - VertexPosView, Figure 3-17 on page 3-13	
Demo 17 - Discard Fragment, Figure 3-18 on page 3-13	
Demo 18 - Fog Effect, Figure 3-19 on page 3-14	
Demo 19 - ToonShading, Figure 3-20 on page 3-14	

An example of each shader is now provided:

- *ShaderExample folder* on page 3-5
- *OrangeBookExample folder* on page 3-15.

3.2.1 ShaderExample folder

Figure 3-1 shows the blank shader:

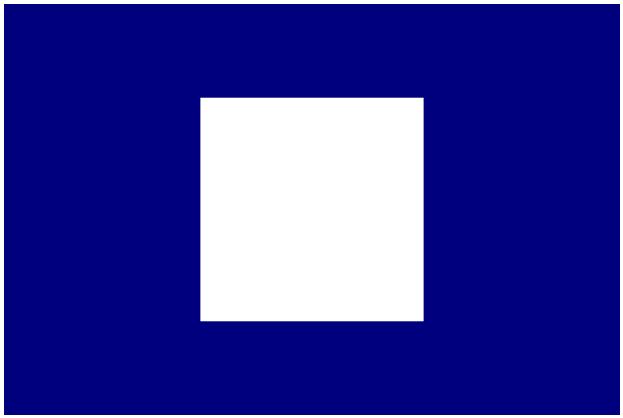

Figure 3-1 Demo 00 - Blank shader

Figure 3-2 shows the Red all Over shader:

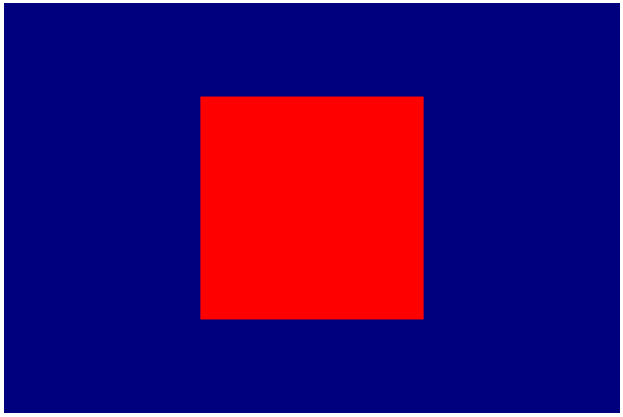

Figure 3-2 Demo 01 - Red all Over shader

Figure 3-3 shows the Normal Viewer shader:

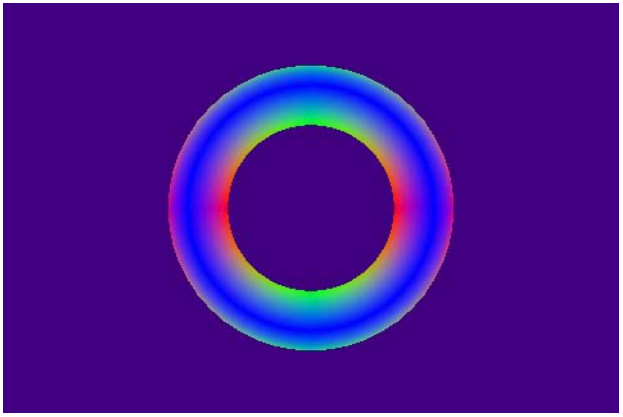

Figure 3-3 Demo 02 - Normal Viewer shader

Figure 3-4 shows the Raspberry Ripple shader:

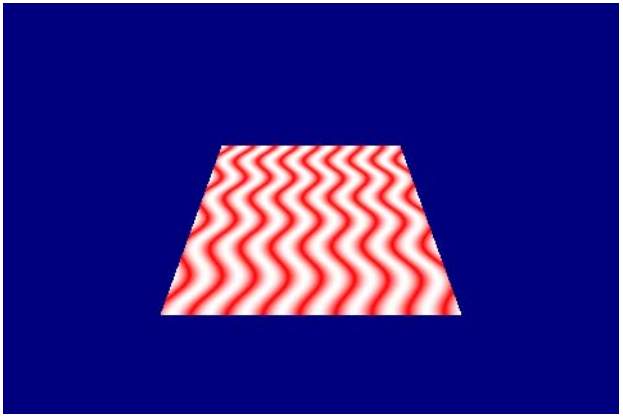

Figure 3-4 Demo 03 - Raspberry Ripple

Figure 3-5 shows the Lighting shader:

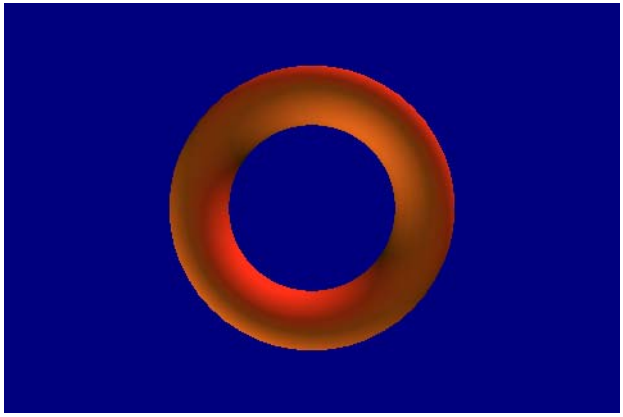

Figure 3-5 Demo 04 - Lighting shader

Figure 3-6 shows the Vertex Ripple shader:

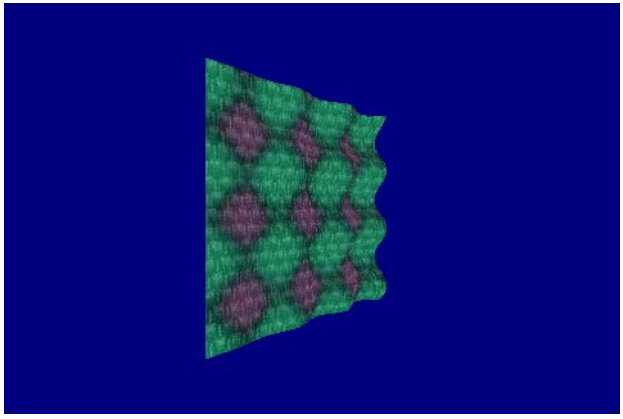

Figure 3-6 Demo 05 - Vertex Ripple shader

Figure 3-7 shows the Bump Mapping shader:

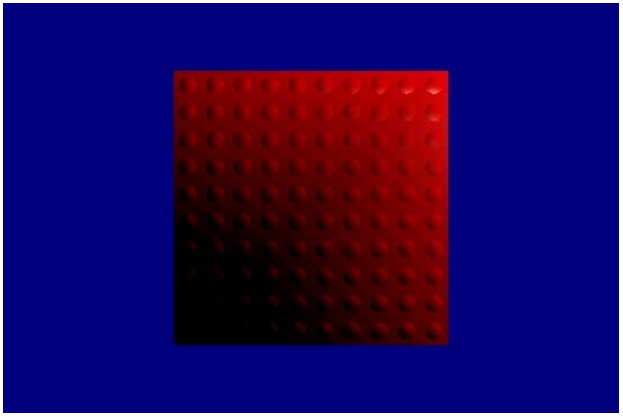

Figure 3-7 Demo 06 - Bump Mapping shader

Figure 3-8 shows the Environment Mapping shader:

Figure 3-8 Demo 07 - Environment Mapping shader

Figure 3-9 shows the Textures shader:

Figure 3-9 Demo 08 - Textures shader

Figure 3-10 shows the Edges shader:

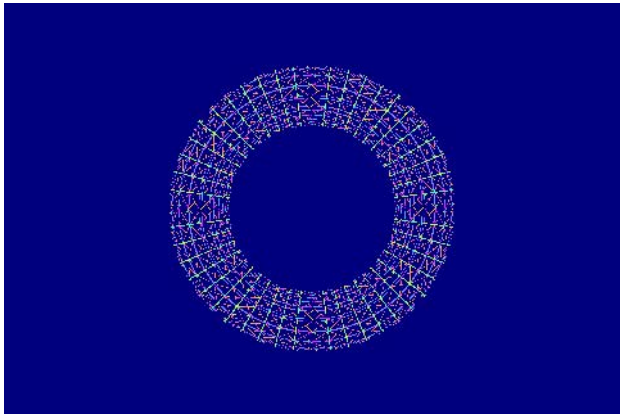

Figure 3-10 Demo 09 - Edges shader

Figure 3-11 shows the Blended Textures shader:

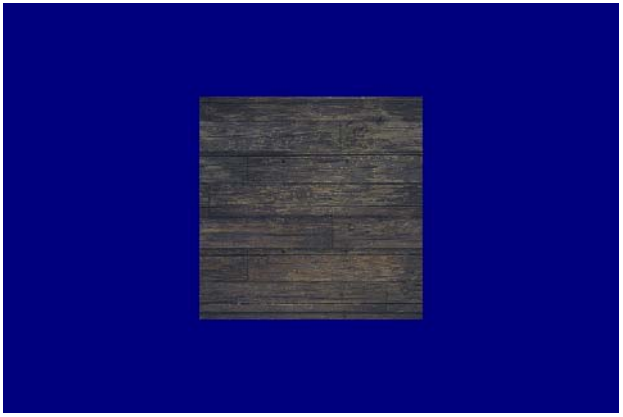

Figure 3-11 Demo 10 - Blended Textures shader

Figure 3-11 shows the Bump and Environment Mapping shader:

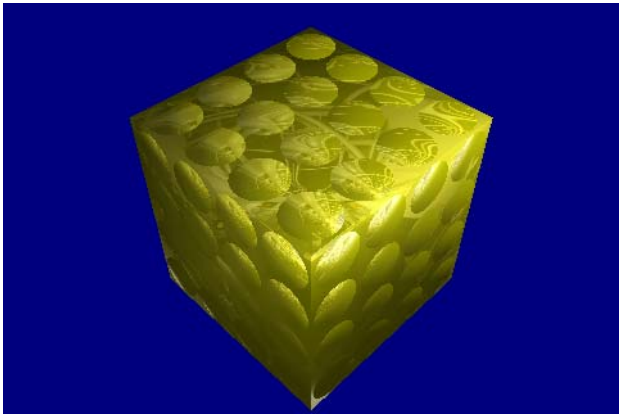

Figure 3-12 Demo 11 - Bump and Environment Mapping shader

Figure 3-12 on page 3-10 shows the Gooch shader:

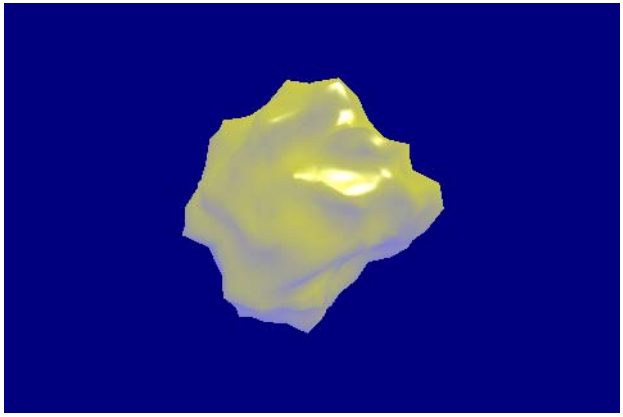

Figure 3-13 Demo 12 - Gooch shader

Figure 3-14 shows the Mandelbrot shader:

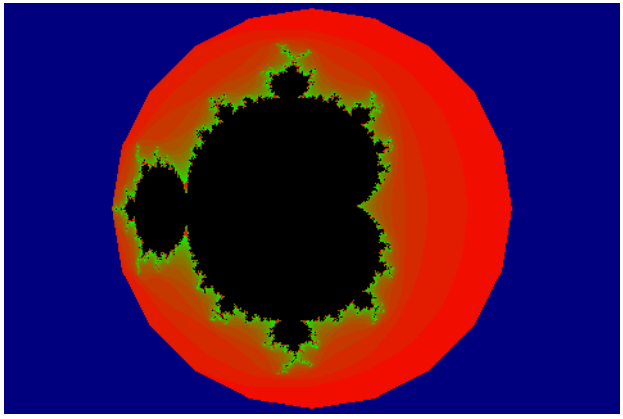

Figure 3-14 Demo 13 - Mandelbrot shader

Figure 3-15 shows the Brick Effect shader:

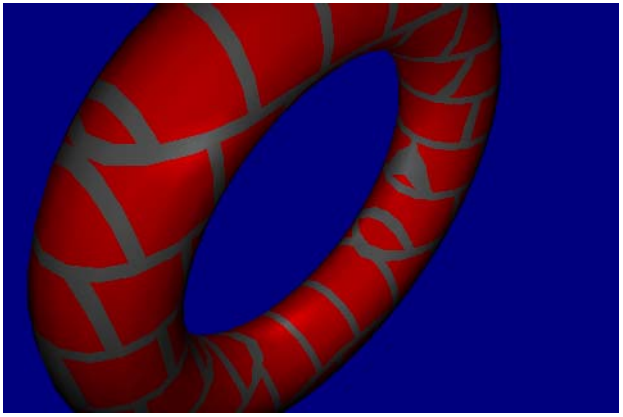

Figure 3-15 Demo 14 - Brick Effect shader

Figure 3-16 shows the TextureCoordView shader:

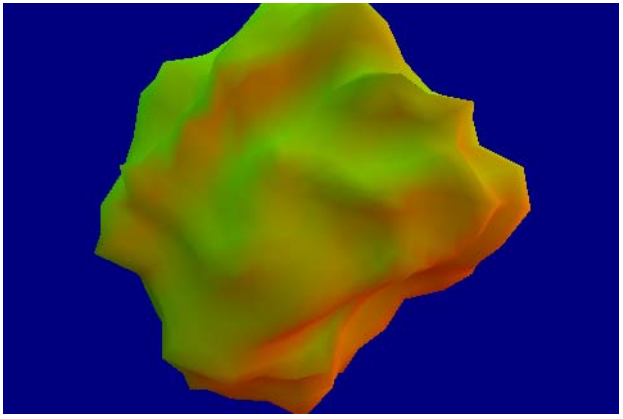

Figure 3-16 Demo 15 - TextureCoordView shader

Figure 3-17 shows the VertexPosView shader:

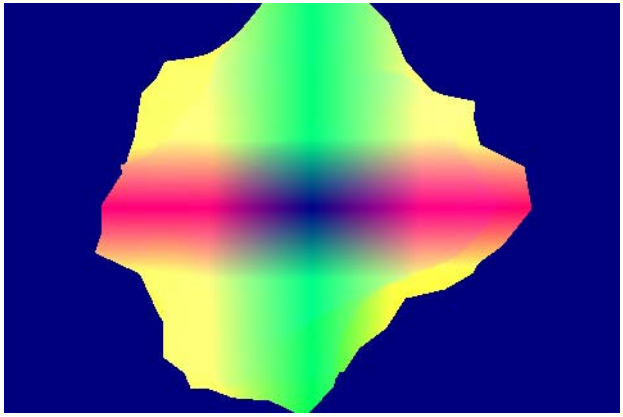

Figure 3-17 Demo 16 - VertexPosView shader

Figure 3-18 shows the Discard Fragment shader:

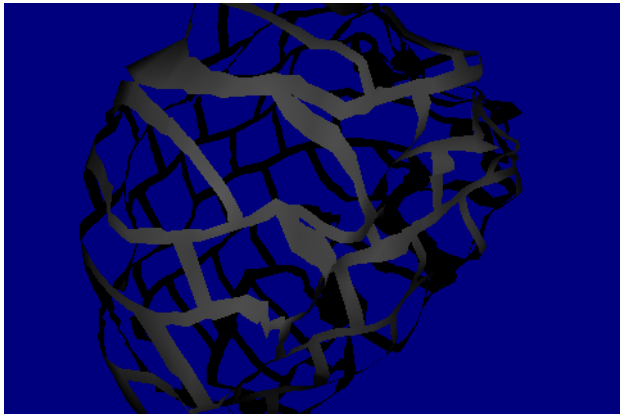

Figure 3-18 Demo 17 - Discard Fragment shader

Figure 3-19 shows the Fog Effect shader:

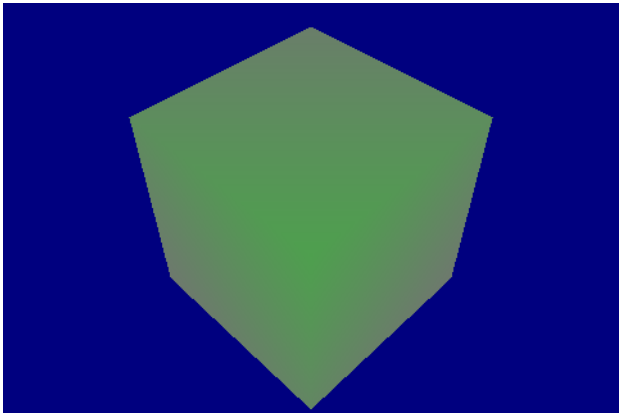

Figure 3-19 Demo 18 - Fog Effect shader

Figure 3-20 shows the ToonShading shader:

Figure 3-20 Demo 19 - ToonShading shader

3.2.2 OrangeBookExample folder

Figure 3-21 shows the Brick Effect shader:

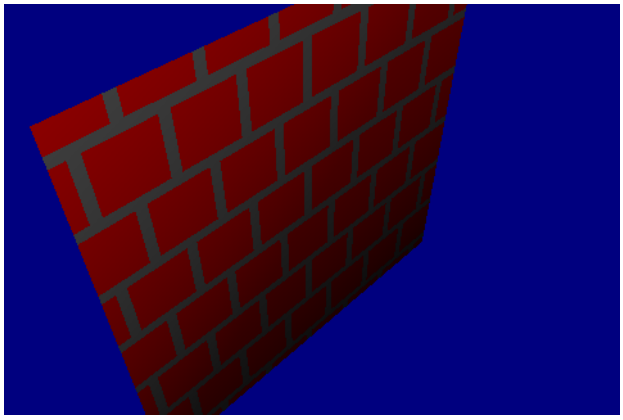

Figure 3-21 Brick Effect shader

Figure 3-22 shows the Cube Map shader:

Figure 3-22 Cube Map shader

Figure 3-23 shows the Earth shader:

Figure 3-23 Earth shader

Figure 3-24 shows the Glyph Bomb shader:

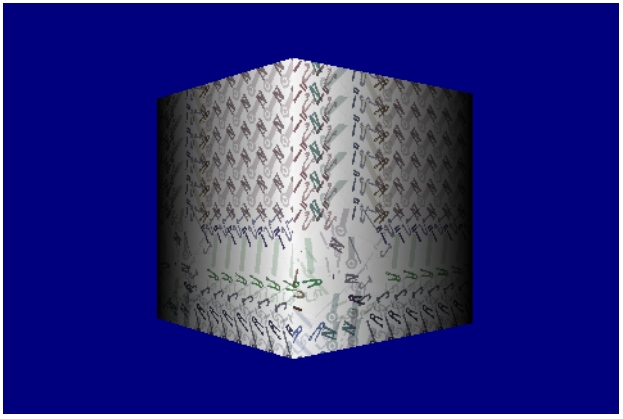

Figure 3-24 Glyph Bomb Ripple shader

Figure 3-25 shows the Lattice shader:

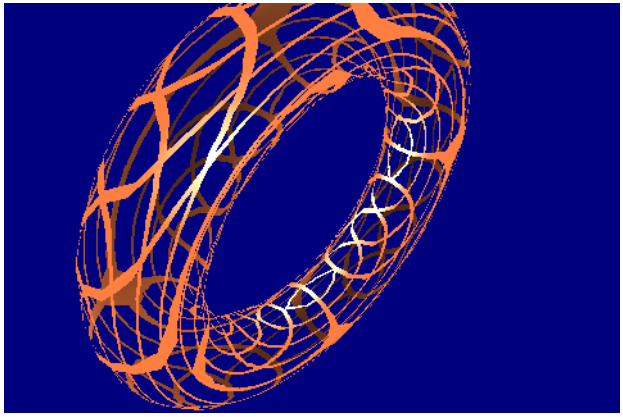

Figure 3-25 Lattice shader

Figure 3-26 shows the Stripe shader:

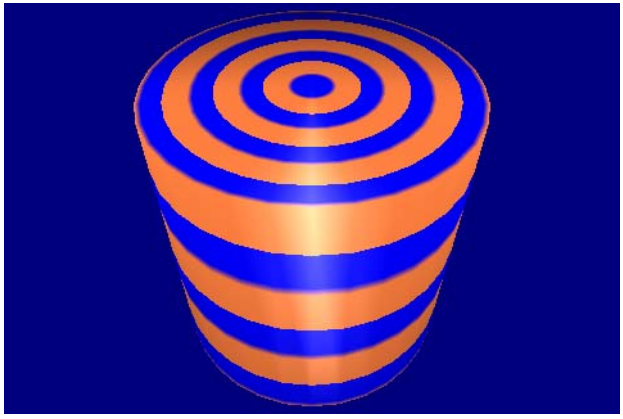

Figure 3-26 Stripe shader

Figure 3-27 shows the Toy Ball shader:

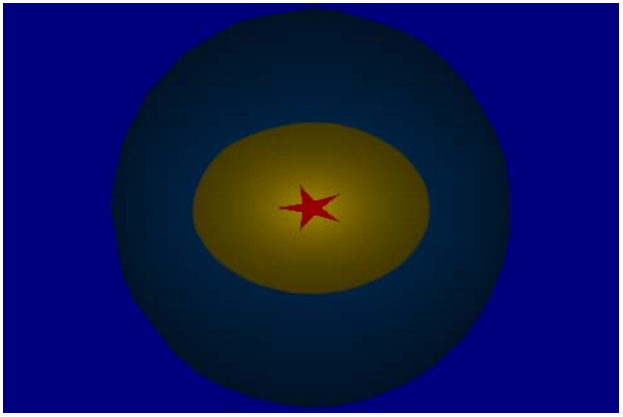

Figure 3-27 Toy Ball shader

Figure 3-28 shows the Hemi Light shader:

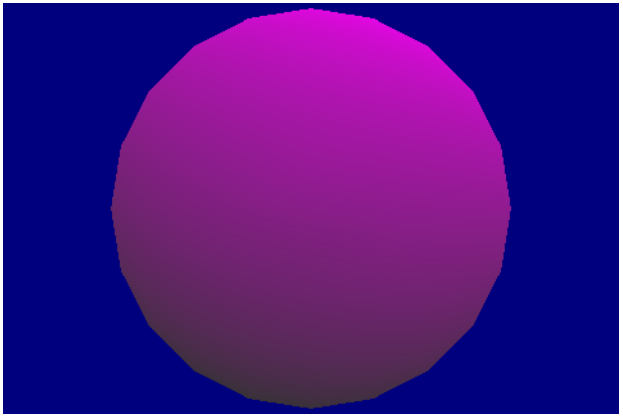

Figure 3-28 Hemi Light shader

Figure 3-29 shows the Spherical Harmonics shader:

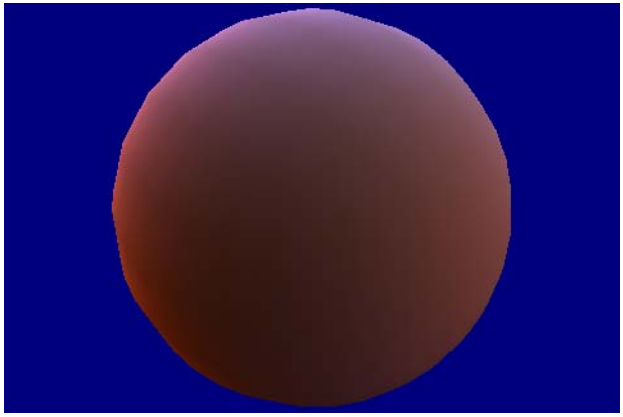

Figure 3-29 Spherical Harmonics shader

Figure 3-30 shows the Uber Light shader:

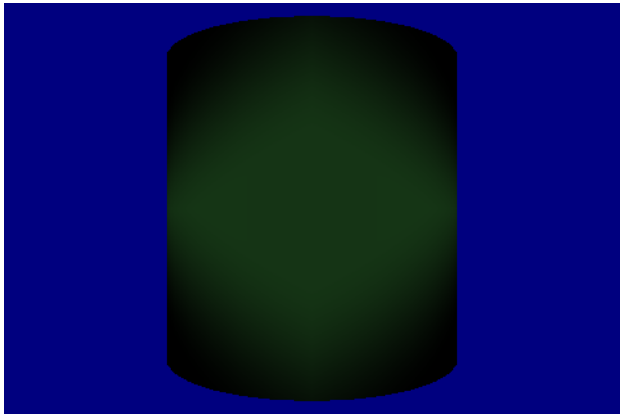

Figure 3-30 Uber Light shader

Figure 3-31 shows the Chromic Aberration shader:

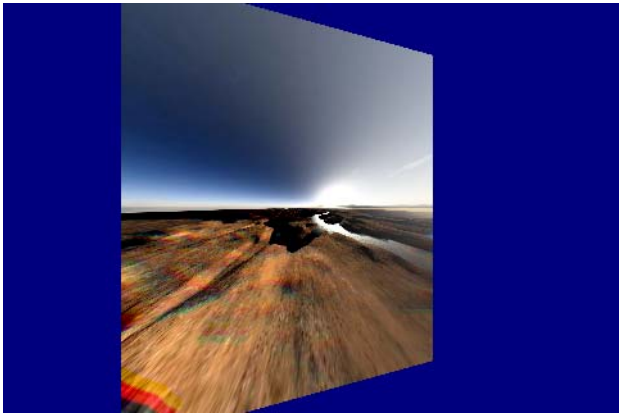

Figure 3-31 Chromic Aberration shader

Figure 3-32 shows the Diffraction shader:

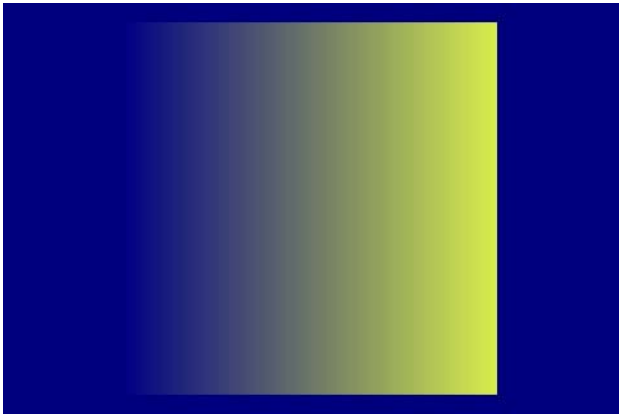

Figure 3-32 Diffraction shader

Figure 3-32 on page 3-20 shows the DiffractionPerFragment shader:

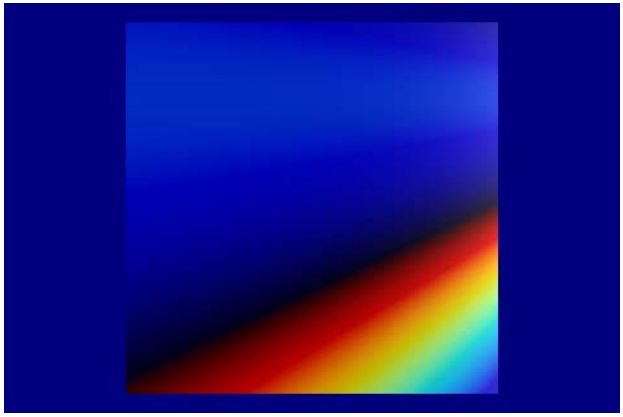

Figure 3-33 DiffractionPerFragment shader

Figure 3-34 shows the Reflection and Refraction shader:

Figure 3-34 Reflection and Refraction shader

Figure 3-35 shows the WardBRDF shader:

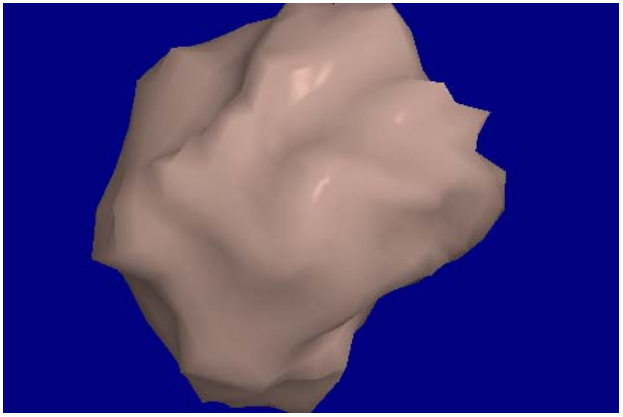

Figure 3-35 WardBRDF shader

Glossary

This glossary describes some of the terms used in ARM manuals. Where terms can have several meanings, the meaning presented here is intended.

Application Programming Interface (API)
A specification for a set of procedures, functions, data structures, and constants that are used to interface two or more software components together. For example, an API between an operating system and the application programs that use it might specify exactly how to read data from a file.

ESSL *See* OpenGL ES Shading Language.

Fragment shader A program running on the pixel processor that calculates the color and other characteristics of each fragment.

Graphics Processor Unit (GPU)
A hardware accelerator for graphics systems using OpenGL ES and OpenVG. The hardware accelerator comprises of an optional geometry processor and a pixel processor together with memory management. Mali programmable GPUs, such as the Mali-200 and Mali-400 MP GPUs, consist of a geometry processor and at least one pixel processor. Mali fixed-function GPUs, such as the Mali-55 GPU consist of a pixel processor only.

Instrumented drivers

Alternative graphics drivers that are used with the Mali GPU. The Instrumented drivers include additional functionality such as error logging and recording performance data files for use by the Performance Analysis Tool.

See also Performance Analysis Tool.

Mali A name given to graphics software and hardware products from ARM that aid 2D and 3D acceleration.

Mali Demo Engine

The Mali Demo Engine is a component of the Mali Developer Tools. The Mali Demo Engine library enables you to develop 3D graphics applications more easily than using OpenGL ES 2.0 alone.

Mali Demo Engine Library

A C++ class framework for developing OpenGL ES 2.0 applications for the Mali GPU. The Mali Demo Engine Library is a component of the Mali Developer Tools.

Mali Developer Tools

A set of development programs that enables software developers to create graphic applications.

OpenGL ES Shading Language (ESSL)

A programming language used to create custom shader programs that can be used within a programmable pipeline, on graphics hardware. You can also use pre-defined library shaders, written in ESSL.

Performance Analysis Tool

A fully-customizable GUI tool that displays and analyzes performance data files produced by the Instrumented drivers, together with framebuffer information.

See also Instrumented drivers, Performance data file.

Performance data file

Files that contain a description of the performance counters, together with the performance counter data in the form of a series of values and images. Performance data files are saved in `.ds2` format and can be loaded directly into the Performance Analysis Tool.

Performance variable

Data produced by the instrumented OpenGL ES 2.0 Emulator, that can be displayed and analyzed as statistical information in the Performance Analysis Tool.

Pixel A pixel is a discrete element that forms part of an image on a display. The word pixel is derived from the term Picture Element.

Shader A program, usually an application program, running on the GPU, that calculates some aspect of the graphical output. See fragment shader and vertex shader.

Shader Library A set of shader examples and other related information, designed to assist with developing shader programs for the Mali GPUs. The Shader Library is a component of the Mali Developer Tools.

Vertex shader A program running on the geometry processor, that calculates the position and other characteristics, such as color and texture coordinates, for each vertex.