

# ***OpenGL ES 1.1 lite***

## ***OS Adaptation Layer Specification***

### ***Notice***

*Ver. 1.3.3 / March 20,2007*  
*LF1000*



This document contains information on a product under development. MagicEyes reserves the right to change or discontinue this product without notice. © MagicEyes Digital Co.ltd 2007. All rights reserved.

---

### Important Notice

Information in this document is provided solely to enable system and software implementers to use MagicEyes products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

MagicEyes reserves the right to make changes without further notice to any products herein.

MagicEyes makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does MagicEyes assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in MagicEyes data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. MagicEyes does not convey any license under its patent rights nor the rights of others. MagicEyes products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the MagicEyes product could create a situation where personal injury or death may occur. Should Buyer purchase or use MagicEyes products for any such unintended or unauthorized application, Buyer shall indemnify and hold MagicEyes and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that MagicEyes was negligent regarding the design or manufacture of the part.

© MagicEyes Digital

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electric or mechanical, by photocopying, recording, or otherwise, without the prior written consent of MagicEyes Digital.

#### HOW TO REACH US:

MagicEyes Digital co., Ltd.

463-824 4F, Uniquet Bldg., 271-2 Seohyeon-Dong, Bundang-Gu  
Seongnam-City, Gyeonggi-Do, Korea  
TEL: 82-31-788-0300  
FAX: 82-31-707-5765

#### HOME PAGE:

<http://www.mesdigital.com>

#### DEVELOPER FORUM:

<http://forum.mesdigital.com>

## Table of Contents

<b>1. Overview.....</b>	<b>4</b>
1.1. History .....	4
<b>2. Operating System Adaptation Layer(OAL).....</b>	<b>5</b>
2.1. Overview .....	5
2.1.1. Interface function list.....	5
2.1.2. Function call flows .....	6
2.1.2.1. Initialization .....	6
2.1.2.2. Finalization .....	7
2.1.2.3. Rendering loop.....	7
2.2. Interface function reference.....	8
GLESOAL_Initialize.....	9
GLESOAL_Finalize .....	11
GLESOAL_SwapBufferCallback .....	12
GLESOAL_SetWindow.....	13
GLESOAL_GetWindowSize .....	14
GLESOAL_IsSoftwareSyncNeeded .....	15
GLESOAL_PushDisplayAddressPatch.....	오류! 책갈피가 정의되어 있지 않습니다.
GLESOAL_WaitForDisplayAddressPatched .....	17

## 1. Overview

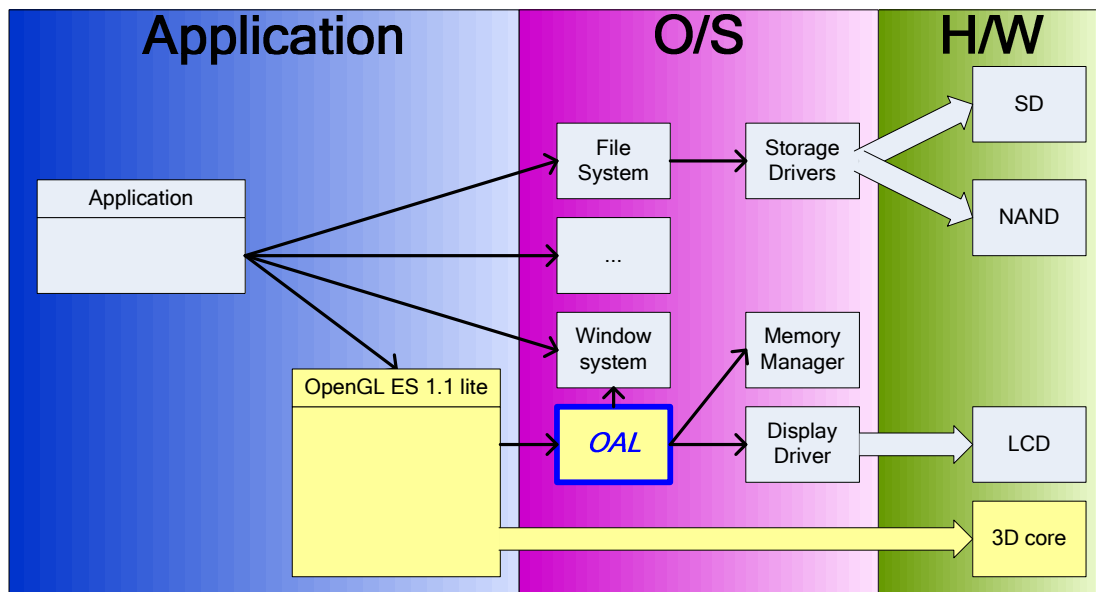
This document specifies the Operating System adaptation layer interface for OpenGL ES 1.1 lite. You must implement these functions for a OpenGL ES 1.1 lite application working on your operating system

### 1.1. History

Rev. Date	Modifier	Contents
2007/01/30	Yuni	first description
2007/02/03	Gamza	
2007/02/08	Gamza	Remove 1D/2D memory allocator Remove GLESOAL_GetVirtualAddressOf3DCore Modify GLESOAL_Inititalize function interface Remove return value of GLESOAL_SwapBufferCallback
2007/03/20	Gamza	Append GLESOAL_IsSoftwareSyncNeeded Append GLESOAL_PushDisplayAddressPatch Append GLESOAL_WaitForDisplayAddressPatched
2007/08/01	Gamza	Modify GLESOAL_PushDisplayAddressPatch interface for FSAA on MP2530F
2008/08/18	Yuni	Modify GLESOAL_PushDisplayAddressPatch to GLESOAL_SetDisplayAddress

## 2. Operating System Adaptation Layer(OAL)

The Operating System adaptation layer(OAL) is an interface between your own operating system and OpenGL ES 1.1 lite. So if you want to work OpenGL ES 1.1 lite on your system, you have to implement this layer.



### 2.1. Overview

OAL is a set of functions that interface between OpenGL ES 1.1 lite and OS. These functions can be grouped as follows:

- Call back functions  
OpenGL initialization/deinitialization/buffer swapping call back.
- Window information functions.  
OpenGL needs some information of a native window system

#### 2.1.1. Interface function list

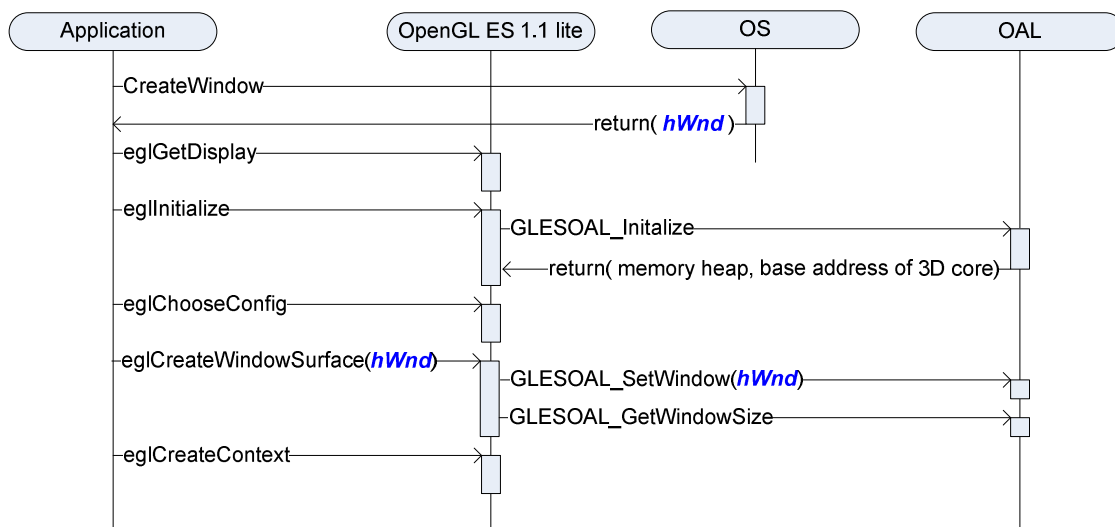
Group	Function	Brief description
Callback	GLESOAL_Initialize	Initialize the OS adaptation layer state.
	GLESOAL_Finalize	Finalize the OS adaptation layer state.

	GLESOAL_SwapBufferCallback	SwapBuffer call back function.
Window	GLESOAL_SetWindow	Set current native window used in OpenGL ES lite.
	GLESOAL_GetWindowSize	Get the window size from current native window
Buffer Swapping	GLESOAL_IsSoftwareSyncNeeded	Indicates whether software synchronization of buffer swapping is needed
	GLESOAL_SetDisplayAddress	Set the 3D layer address to the MLC of a display hardware.
	GLESOAL_WaitForDisplayAddressPatched	Wait for 3D layer address to be patched.

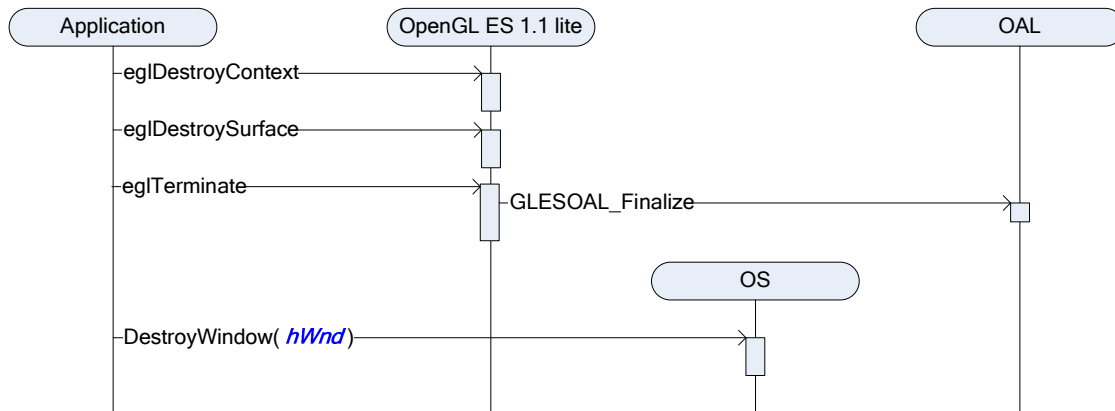
## 2.1.2. Function call flows

This section shows several sequence diagrams that describe the collaboration between application, OpenGL ES 1.1 lite and OS(OAL).

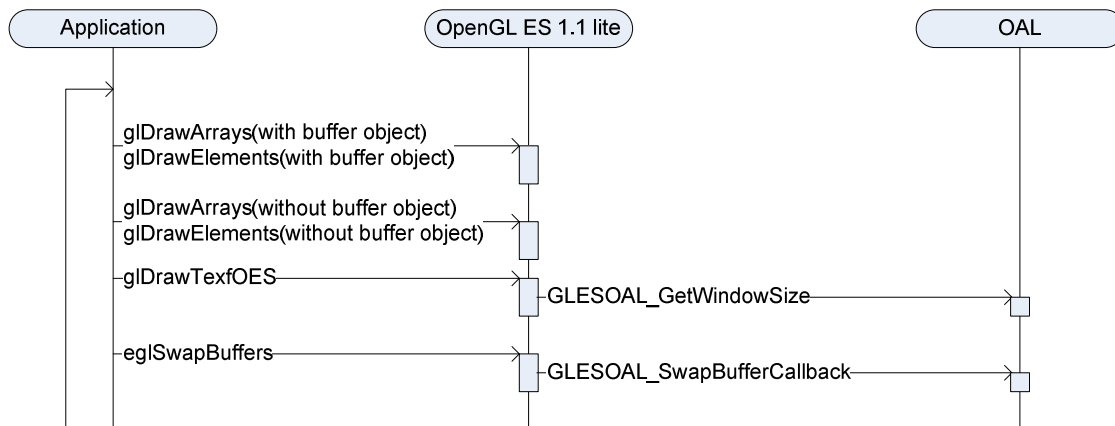
### 2.1.2.1. Initialization



### 2.1.2.2. Finalization



### 2.1.2.3. Rendering loop



## 2.2. Interface function reference

This section specifies reference of the OAL functions, prototype, parameters, responsibilities, related APIs and so on.

```
//-----
//
//
// interface between the OpenGL ES 1.1 lite and operating systems
//
//-----
#ifdef __cplusplus
extern "C" {
#endif

    typedef int GLESOALbool;

    typedef struct
    {
        unsigned int    VirtualAddressOf3DCore    ; // virtual address of the 3D core register

        unsigned int    Memory1D_VirtualAddress ; // must be 8byte aligned, non-cacheable
        unsigned int    Memory1D_PhysicalAddress; // must be 8byte aligned, non-cacheable
        unsigned int    Memory1D_SizeInMbyte     ; // size (Mbyte)

        unsigned int    Memory2D_VirtualAddress ; // must be 4Mbyte aligned, non-cacheable
        unsigned int    Memory2D_PhysicalAddress; // must be 4Mbyte aligned, non-cacheable
        unsigned int    Memory2D_SizeInMbyte     ; // size (Mbyte), must be multiple of 4
    } __OAL_MEMORY_INFORMATION__;

    GLESOALbool GLESOAL_Initialize( __OAL_MEMORY_INFORMATION__* pMemoryInformation );
    void        GLESOAL_Finalize ( void );
    void        GLESOAL_SwapBufferCallback( void );
    void        GLESOAL_SetWindow      ( void* pNativeWindow );
    void        GLESOAL_GetWindowSize( int* pWidth, int* pHeight );
    GLESOALbool GLESOAL_IsSoftwareSyncNeeded ( void );
    void        GLESOAL_SetDisplayAddress ( const unsigned int DisplayBufferPhysicalAddress );
    void        GLESOAL_WaitForDisplayAddressPatched ( void );

#ifdef __cplusplus
}
#endif
```



## GLESOAL\_Initialize

### Syntax

```
GLESOALbool GLESOAL_Initialize
(
    __OAL_MEMORY_INFORMATION__* pMemoryInformation
);
```

### Description

This function initializes whole the OS adaptation layer for working 3D graphics core on your system.

This is called by `eglInitialize()`. If this function returns `FALSE`, `eglInitialize()` returns `EGL_FALSE`.

The OpenGL ES 1.1 lite needs two kinds of non-cacheable memory heaps, and a non-cacheable virtual address of the 3D Core register.

The one kind of memory heap is called 'Memory1D', it is used for

- Hardware command queue. (128Kbytes)
- Buffer objects. ( It is defined by OpenGL ES 1.1 standard)
- H/W buffer objects. (It is defined by OpenGL ES 1.1 lite)

And the other one is called 'Memory2D', it is used for

- Three frame buffers and one depth buffer. ( Buffer resolution x 2 x 4 bytes )
- Textures.

At last, the virtual address of the 3D Core register is used for

- Accessing the 3D core register directly to control the 3D core.

This function has to provide these to OpenGL ES 1.1 lite. It should be returned by `pMemoryInformation` parameter.

```
typedef struct
{
    unsigned int    VirtualAddressOf3DCore    ; // virtual address of the 3D core register

    unsigned int    Memory1D_VirtualAddress ; // must be 8byte aligned, non-cacheable
    unsigned int    Memory1D_PhysicalAddress; // must be 8byte aligned, non-cacheable
    unsigned int    Memory1D_SizeInMbyte     ; // size (Mbyte)

    unsigned int    Memory2D_VirtualAddress ; // must be 4Mbyte aligned, non-cacheable
    unsigned int    Memory2D_PhysicalAddress; // must be 4Mbyte aligned, non-cacheable
    unsigned int    Memory2D_SizeInMbyte     ; // size (Mbyte), must be multiple of 4
} __OAL_MEMORY_INFORMATION__;
```

Member variable name	Description
VirtualAddressOf3DCore	The virtual address of base of the 3D core register. OpenGL ES 1.1 lite accesses the 3D core register directly by using this address. It must points a non-cacheable address space.

## OS Adaptation Layer Specification

Memory1D_VirtualAddress	Virtual base address of 1D memory heap
Memory1D_PhysicalAddress	Physical base address of 1D memory heap
Memory1D_SizeInMbyte	Size of 1D memory heap in Mbyte
Memory2D_VirtualAddress	Virtual base address of 2D memory heap
Memory2D_PhysicalAddress	Physical base address of 2D memory heap
Memory2D_SizeInMbyte	Size of 2D memory heap in Mbyte

Memory1D has the following restrictions

- Physical address and virtual address must points a non-cacheable address space.
- Physical address and virtual address must be 8byte aligned.

Memory2D has the following restrictions

- Physical address and virtual address must points a non-cacheable address space.
- Physical address and virtual address must be 4Mbyte aligned.
- The heap size must be a multiple of 4Mbytes.

Parameter

*pMemoryInfomation*

- [out] The memory information to be provided to OpenGL ES 1.1 lite.

Return Value

0

- OS adaptation layer initialization is failed.

1

- OS adaptation layer initialization is success.

Responsibilities

Responsibility	Driver	Core
Initialize 1D/2D memory heap	Memory manager	-
Allocate a virtual address to the corresponding 3D core	Memory manager	-

Functions which call it

`eglInitialize`

## GLESOAL\_Finalize

### Syntax

```
void GLESOAL_Finalize ( void );
```

### Description

This function finalizes OS adaptation layer when 3D graphics core working on your system, is finished. This is called at `eglTerminate()`.

### Parameter

None

### Return Value

None

### Responsibilities

Responsibility	Driver	Core
Hide the 3D layer from the screen.	Display driver	MLC
Finalize 1D/2D memory heap	Memory manager	-
Finalize the virtual address to the corresponding 3D core	Memory manager	-

### Functions which call it

`eglTerminate`

## GLESOAL\_SwapBufferCallback

### Syntax

```
void GLESOAL_SwapBufferCallback( void );
```

### Description

It is a callback function of the `eglSwapBuffers()` function. You have to display the 3D graphics layer and fit position and size of 3D graphics layer to the client area of the current native window.(the current native window is specified by `GLESOAL_SetWindow()` function)

### Parameter

None

### Return Value

None

### Responsibilities

Responsibility	Driver	Core
Display the 3D layer on the screen	Display driver	MLC
Fit a position and size of the displayed area to current native window.	Display driver	MLC
If the current native window size is equal or less than 0, turn off the 3D layer.	Display driver	MLC

### Functions which call it

`eglSwapBuffers`

## GLESOAL\_SetWindow

### Syntax

```
void GLESOAL_SetWindow( void* pNativeWindow );
```

### Description

This function assigns the native window when `eglCreateWindowSurface()` is called. The native window that is assigned through this function is used for getting window information such as a `GLESOAL_GetWindowSize()`.

### Parameter

*pNativeWindow*

- [in] the handle of native window. It is not a pointer but a `NativeWindowType`. If you calls `eglCreateWindowSurface()` function as following,

```
glesSurface= eglCreateWindowSurface(glesDisplay, configs[0], hWnd, configAttribs);
```

the OpenGL ES 1.1 lite calls this function as following

```
GLESOAL_SetWindow( (void*)hWnd );
```

### Return Value

None

### Responsibilities

Responsibility	Driver	Core
Set current native window	-	-

### Functions which call it

`eglCreateWindowSurface`

## GLESOAL\_GetWindowSize

### Syntax

```
void GLESOAL_GetWindowSize(  
    int* pWidth,  
    int* pHeight );
```

### Description

It returns the width and height of a native window that is set at GLESOAL\_SetWindow. Sometimes, OpenGL ES 1.1 lite calls this function to get the width and height of a native window.

### Parameter

*pWidth*

- [out] returns width value.

*pHeight*

- [out] returns height value.

### Return Value

None

### Responsibilities

Role	Driver	Core
Return the native window size where 3D layer is drawn	Window system	-

### Functions which call it

eglCreateWindowSurface

glDrawTexfOES

## GLESOAL\_IsSoftwareSyncNeeded

### Syntax

```
GLESOALbool GLESOAL_IsSoftwareSyncNeeded ( void );
```

### Description

This function indicates whether software synchronization of buffer swapping is needed.

If a system uses dual-display mode in MP2530F, this function must return 1.

### Parameter

*None*

### Return Value

*0*

- Software synchronization of buffer swapping is not needed.

*1*

- Software synchronization of buffer swapping is needed.

### Responsibility

Responsibility	Driver	Core
Indicates whether software synchronization of buffer swapping is needed	Display driver	-

### Functions which call it

`eglSwapBuffers`

## GLESOAL\_SetDisplayAddress

### Syntax

```
void GLESOAL_SetDisplayAddress (  
    ( const unsigned int DisplayBufferPhysicalAddress  
    );
```

### Description

This function sets the 3D layer address to the MLC of a display hardware.

### Parameter

*DisplayBufferPhysicalAddress*  
- [in] Display buffer address.

### Return Value

*None*

### Responsibility

Responsibility	Driver	Core
Set the 3D layer address to the MLC of a display hardware.	Display driver	-

### Functions which call it

`eglSwapBuffers`



## GLESOAL\_WaitForDisplayAddressPatched

### Syntax

```
void GLESOAL_WaitForDisplayAddressPatched ( void );
```

### Description

This function waits for 3D layer address that is pushed by GLESOAL\_SetDisplayAddress to be patched.

### Parameter

*None*

### Return Value

*None*

### Responsibility

Responsibility	Driver	Core
Wait for 3D layer address to be patched.	Display driver	-

### Functions which call it

`eglSwapBuffers`