User's Manual

32

OMX Media Component

User's Manual: Common Part

— Preliminary —

Renesas Electronics
www.renesas.com

Rev.1.00 Aug. 29, 2014

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

Table of Contents

Figures

Tables

1. Overview

1.1. About This Document

This document is the User's Manual for OMX Media Component. It describes the specifications that are common to each OMX Media Component.

1.2. OMX Media Component Overview and Scope

The OMX Media Component specifications are based on the OpenMAX IL 1.1.2. For the standard specifications of the OpenMAX IL 1.1.2, see the Khronos Group, Inc. document [1]. This manual focuses on the extensions from the standard and the supplements for the standard.

The OMX Media Component consists of an OpenMAX IL Core and OpenMAX IL Components. Table 1-1 lists the libraries that are contained in the OMX Media Component Common Library.

Table 1-1 Libraries

Library Name	Description
Core Library	A library provides the OpenMAX IL Core functions.
MC Common Library	A library provides the OpenMAX IL Macro functions and the common implementation to each OMX Media Component.

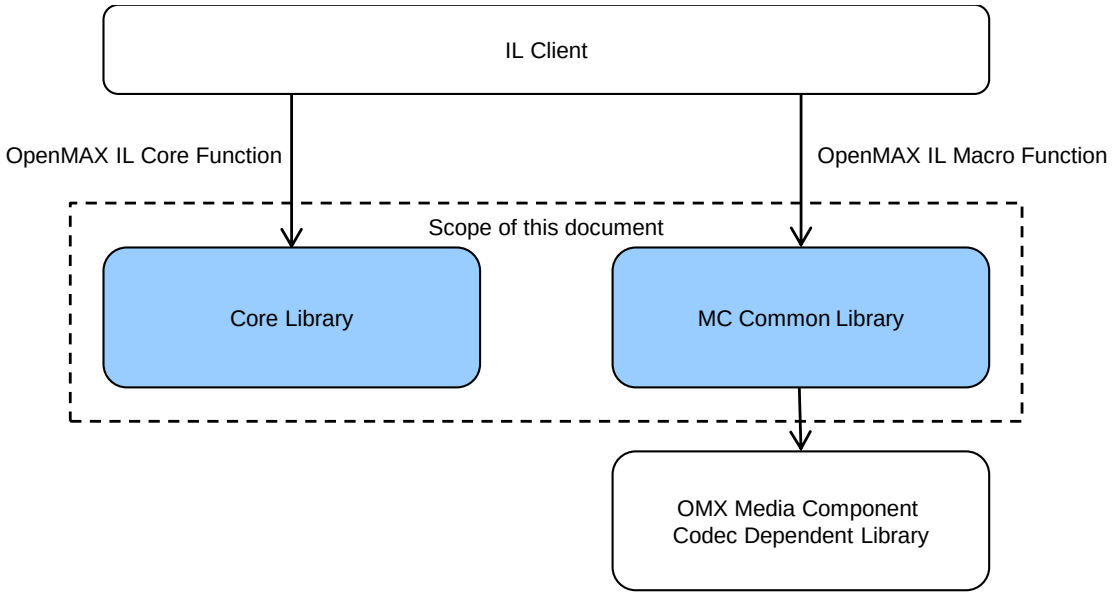

Figure 1-1 Software Stacks and Scope

This document describes the common functional specifications that the OMX Media Component Common Library provides to each Media Component. For the Media Component-dependent specifications, see each OMX Media Component User's Manual.

1.3. Attentions

The OMX does not necessarily assure interoperability for all media frameworks. To confirm the connectivity between a media framework and OMX Media Components, review the User's Manuals for each Media Component.

1.4. Required Header Files

Table 1-2 lists the header files that are required to use the OMX.

Table 1-2 Required Header Files

File name	Remarks
OMX_Types.h	Khronos Standard Headers
OMX_Index.h	
OMX_Core.h	
OMX_IVCommon.h	
OMX_Audio.h	
OMX_Video.h	
OMX_Image.h	
OMX_Other.h	
OMX_Component.h	
OMX_ContentPipe.h	
OMXR_Extension.h	Required to use the OMX extended definitions and structures

1.5. Related Documents

Table 1-3 lists the related documents.

Table 1-3 List of Related Documents

No.	Document Name	Remarks
[1]	OpenMAX Integration Layer Application Programming Interface Specification Version 1.1.2, September 1, 2008	http://www.khronos.org/registry/omxil/specs/OpenMAX_IL_1_1_2_Specification.pdf
[2]	OMX Integration Guide	OMX Integration Guide documents describe how to integrate the OMX to the system. The documents are distributed for each Operation System.

1.6. Terminology

Table 1-4 lists the terms that are used in this document.

Table 1-4 Terminology

Term	Abbreviation	Description
OMX	-	In this document, OMX means the Renesas implementation that is based on OpenMAX IL specifications.
OpenMAX IL	-	
OpenMAX IL Core	-	
Component	-	
Media Component	-	
IL Client	-	The layer of software that invokes the methods of the core or component.

2. API Reference

2.1. OpenMAX IL Core Functions

Table 2-1 lists the OpenMAX IL Core functions. In the table, 'X' indicates supported functions and '-' indicates NOT supported functions.

Table 2-1 OpenMAX IL Core Functions

Section	Function Name	Support
2.1.1	OMX_Init( )	X
2.1.2	OMX_Deinit( )	X
2.1.3	OMX_ComponentNameEnum( )	X
2.1.4	OMX_GetHandle( )	X
2.1.5	OMX_FreeHandle( )	X
2.1.6	OMX_SetupTunnel( )	-
2.1.7	OMX_GetContentPipe( )	-
2.1.8	OMX_GetComponentsOfRole( )	X
2.1.9	OMX_GetRolesOfComponent( )	X

2.1.1. OMX_Init( )

[Reference] Related document [1], 3.2.3.1

[Notes]
- In the case where OMX_Init( ) is called multiple times in the same application process, the OMX is initialized only at the first call. After the second call, OMX executes no operation and returns normal end.
- This function is not thread-safe. For the detail, see section 2.5.

[Remarks] None

2.1.2. OMX_Deinit( )

[Reference] Related document [1], 3.2.3.2

[Notes]
- In the case where OMX_Init( ) is called multiple times in the same application process, the IL client must call OMX_Deinit( ) as many times as OMX_Init( ) was called. The OMX is deinitialized only at the last call.
- OMX_Deinit( ) frees all the remained components forcibly that are not freed yet by OMX_FreeHandle( ).
- This function is not thread-safe. For the detail, see section 2.5.

[Remarks] None

2.1.3. OMX_ComponentNameEnum( )

[Reference] Related document [1], 3.2.3.3

[Notes]
- This function is not thread-safe. For the detail, see section 2.5.

[Remarks]
- For the name of Media Components, see individual Media Component User's Manual.

2.1.4. OMX_GetHandle( )

[Reference] Related document [1], 3.2.3.4

[Notes]
- This function is not thread-safe. For the detail, see section 2.5.

[Remarks] None.

2.1.5. OMX_FreeHandle( )

[Reference] Related document [1], 3.2.3.5

[Notes] − This function is not thread-safe. For the detail, see section 2.5.

[Remarks] None.

2.1.6. OMX_SetupTunnel( )

[Reference] Related document [1], 3.2.3.6

[Notes] − OMX_SetupTunnel( ) is not supported.

[Remarks] None.

2.1.7. OMX_GetContentPipe( )

[Reference] Related document [1], 3.2.3.7

[Notes] − OMX_GetContentPipe( ) is not supported.

[Remarks] None.

2.1.8. OMX_GetComponentsOfRole( )

[Reference] Related document [1], 8.2.4

[Notes] − This function is not thread-safe. For the detail, see section 2.5.

[Remarks] − For the role definition of Media Components, see individual Media Component User's Manual.

2.1.9. OMX_GetRolesOfComponent( )

[Reference] Related document [1], 8.2.3

[Notes] − This function is not thread-safe. For the detail, see section 2.5.

[Remarks] − For the role definition of Media Components, see individual Media Component User's Manual.

2.2. OpenMAX IL Macro Functions

Table 2-2 lists the OpenMAX IL Macro functions. In the table, 'X' indicates supported functions and '-' indicates NOT supported functions.

Table 2-2 OpenMAX IL Macro Functions

Section	Function Name	Support
2.2.1	OMX_GetComponentVersion( )	X
2.2.2	OMX_GetState( )	X
2.2.3	OMX_SendCommand( )	X
2.2.4	OMX_GetParameter( )	X
2.2.5	OMX_SetParameter( )	X
2.2.6	OMX_GetConfig( )	X
2.2.7	OMX_SetConfig( )	X
2.2.8	OMX_GetExtensionIndex( )	X
2.2.9	OMX_UseBuffer( )	X
2.2.10	OMX_AllocateBuffer( )	X
2.2.11	OMX_FreeBuffer( )	X
2.2.12	OMX_EmptyThisBuffer( )	X
2.2.13	OMX_FillThisBuffer( )	X
2.2.14	OMX_UseEGLImage( )	-

2.2.1. OMX_GetComponentVersion( )

[Reference] Related document [1], 3.2.2.1

[Notes] None.

[Remarks] None.

2.2.2. OMX_GetState( )

[Reference] Related document [1], 3.2.2.13

[Notes] None.

[Remarks] None.

2.2.3. OMX_SendCommand( )

[Reference] Related document [1], 3.2.2.2

[Notes]
- If the value of *nParam* is OMX_ALL for the OMX_CommandFlush, OMX_CommandPortDisable and OMX_CommandPortEnable command, the OMX Media Component issues an individual OMX_EventCmdComplete event for each port.
- During the state transition by an OMX_CommandStateSet command, this function results in an error.
- During a port operation by the OMX_CommandFlush, OMX_CommandPortEnable or OMX_CommandPortDisable, the following commands result in an error.
  - An OMX_CommandStateSet command.
  - An OMX_CommandFlush, an OMX_CommandPortEnable or an OMX_CommandPortDisable for the same port.

[Remarks] None.

2.2.4. OMX_GetParameter( )

[Reference] Related document [1], 3.2.2.8

[Notes]
- The *nSize* member of the *pComponentParameterStructure* must be set to the size of the structure. If an illegal size is specified, this function returns OMX_ErrorBadParameter.

[Remarks]
- For the supported indexes, see section 4.

2.2.5. OMX_SetParameter( )

[Reference] Related document [1], 3.2.2.9

[Notes]
- The *nSize* member of the *pComponentParameterStructure* must be set to the size of the structure. If an illegal size is specified, this function returns OMX_ErrorBadParameter.

[Remarks]
- For the supported indexes, see section 4.

2.2.6. OMX_GetConfig( )

[Reference] Related document [1], 3.2.2.10

[Notes]
- The *nSize* member of the *pComponentConfigStructure* must be set to the size of the structure. If an illegal size is specified, this function returns OMX_ErrorBadParameter.

[Remarks]
- For the supported indexes, see section 4.

2.2.7. OMX_SetConfig( )

[Reference] Related document [1], 3.2.2.11

[Notes]
- The *nSize* member of the *pComponentConfigStructure* must be set to the size of the structure. If an illegal size is specified, this function returns OMX_ErrorBadParameter.

[Remarks]
- For the supported indexes, see section 4.

2.2.8. OMX_GetExtensionIndex( )

[Reference] Related document [1], 3.2.2.12

[Notes] None.

[Remarks]
- For the strings for extended indexes, see each Media Component User's Manual.

2.2.9. OMX_UseBuffer( )

[Reference] Related document [1], 3.2.2.14

[Notes]
- Regarding the support for the OMX_UseBuffer( ) function and the detail memory specifications, see each Media Component User's Manual.
- The OMX_UseBuffer( ) cannot be called for the port on which the OMX_AllocateBuffer( ) is already called.
- The value of the *nSizeBytes* parameter must be the same value with the *nBufferSize* member of the OMX_PARAM_PORTDEFINITIONTYPE structure.

[Remarks] None.

2.2.10. OMX_AllocateBuffer( )

[Reference] Related document [1], 3.2.2.15

[Notes]
- The OMX_AllocateBuffer( ) cannot be called for the port on which the OMX_UseBuffer( ) is already called.
- The value of the *nSizeBytes* parameter must be the same value with the *nBufferSize* member of the OMX_PARAM_PORTDEFINITIONTYPE structure.

[Remarks] None.

2.2.11. OMX_FreeBuffer( )

[Reference] Related document [1], 3.2.2.16

[Notes]
- The IL client must not call this function for a buffer that the OMX Media Component has not returned to the IL client via the buffer callbacks.

[Remarks] None.

2.2.12. OMX_EmptyThisBuffer( )

[Reference] Related document [1], 3.2.2.17

[Notes]
- The IL client must not call this function for a buffer that the OMX Media Component has not returned to the IL client via the buffer callbacks.

[Remarks]
- For the details of the OMX_BUFFERHEADERTYPE structure, see section 5.1.1.

2.2.13. OMX_FillThisBuffer( )

[Reference] Related document [1], 3.2.2.18

[Notes]
- The IL client must not call this function for a buffer that the OMX Media Component has not returned to the IL client via the buffer callbacks.

[Remarks]
- For the details of the OMX_BUFFERHEADERTYPE structure, see section 5.1.1.

2.2.14. OMX_UseEGLImage( )

[Reference] Related document [1], 3.2.2.19

[Notes]
- OMX_UseEGLImage( ) is not supported.

[Remarks] None.

2.3. Callback Functions

Table 2-3 lists the OpenMAX IL function callbacks. These callbacks are set up via the OMX_GetHandle( ) function.

Table 2-3 OpenMAX IL Function Callbacks

Section	Fuction Name
2.3.1	(*EventHandler)( )
2.3.2	(*EmptyBufferDone)( )
2.3.3	(*FillBufferDone)( )

2.3.1. (*EventHandler)( )

[Reference] Related document [1], 3.1.2.9.1

[Notes] Table 2-4 shows the events passed via this callback. In the table, 'X' indicates supported events and '-' indicates NOT supported events.

[Remarks] None.

Table 2-4 Event Type

Event Type(*eEvent*)	Support	Reference
OMX_EventCmdComplete	X	Related document [1] 3.1.1.4.1
OMX_EventError	X	Related document [1] 3.1.1.4.2
OMX_EventMark	X	Related document [1] 3.1.1.4.3
OMX_EventPortSettingsChanged	X	Related document [1] 3.1.1.4.4 For Media Component dependent specifications, see each Media Component User's Manual.
OMX_EventBufferFlag	X	Related document [1] 3.1.1.4.5 For Media Component dependent specifications, see each Media Component User's Manual.
OMX_EventResourcesAcquired	-	
OMX_EventComponentResumed	-	
OMX_EventDynamicResourcesAvailable	-	
OMX_EventPortFormatDetected	-	

2.3.2. (*EmptyBufferDone)( )

[Reference] Related document [1] 3.1.2.9.2

[Notes] None.

[Remarks] − For the details of the OMX_BUFFERHEADERTYPE structure, see section 5.1.1.

2.3.3. (*FillBufferDone)( )

[Reference] Related document [1] 3.1.2.9.3

[Notes] None.

[Remarks] − For the details of the OMX_BUFFERHEADERTYPE structure, see section 5.1.1.

2.4. OMX Extended Functions

Table 2-5 lists the OMX extended functions.

Table 2-5 OMX Extended Functions

Section	Function Name	Description
2.4.1	OMXR_GetConfiguration( )	A function to get the path of the configuration file.
2.4.2	OMXR_SetConfiguration( )	A function to set the path of the configuration file.
2.4.3	OMXR_SetLogMode( )	A function to specify the log level for debugging purpose.

2.4.1. OMXR_GetConfiguration( )

[Definition]
```
OMX_ERRORTYPE OMXR_GetConfiguration(
    OMX_STRING          cConfigName,
    OMX_U32*            pu32Length
);
```

[Function] Gets the configuration file name.

[Arguments]

Variable Name	I/O	Description
cConfigName	O	A pointer to a string with the configuration file name.
pu32Length	I/O	The number of characters in the *cConfigName* string. If the cConfigName is NULL, the length of the configuration file name is set to pu32Length by the OMX. If the cConfigName is not NULL, the string buffer length is set to pu32Length by the IL client.

[Return Values] See section 6.

[Notes] None.

[Usage]
```
OMX_ERRORTYPE       err;
OMX_STRING    cConfigName;
OMX_U32         u32Length;

/* get the length of the configuration file name */
OMXR_GetConfiguration(NULL, &u32Length);
cConfigName = (OMX_STRING)malloc(u32Length);

/* get the configuration file name */
err = OMXR_GetConfiguration(cConfigName, &u32Length);
if (err != OMX_ErrorNone) {
        printf("Can't get configuration file name! error code = 0x%08x¥n", err);
        return -1;
}
```

2.4.2. OMXR_SetConfiguration( )

[Definition] OMX_ERRORTYPE OMXR_SetConfiguration(
OMX_STRING *cConfigName*
);

[Function] Sets a configuration file name.

[Arguments]

Variable Name	I/O	Description
cConfigName	I	A pointer to a null-terminated string with the configuration file name.

[Return Values] See section 6.

[Notes]
- The configuration file name set by this function is used in OMX_Init( ) function.
- When the configuration file name is not specified via this function, the default file name will be applied. For the default configuration file name, see related document [2].
- This function is not thread-safe. The IL client must not call this function from multiple threads in the same application process at the same time.

[Usage] OMX_ERRORTYPE err;

```
/* set the configuration file name */
err = OMXR_SetConfiguration("/usr/lib/omx/omx_config_base.txt");
if (err != OMX_ErrorNone) {
        printf("Can't set configuration file! error code = 0x%08x¥n", err);
        return -1;
}
```

2.4.3. OMXR_SetLogMode( )

[Definition] OMX_ERRORTYPE OMXR_SetLogMode(
OMX_U32 *u32LogMode*
);

[Function] Sets the log level for debugging purpose.

[Arguments]

Variable Name	I/O	Description
u32LogMode	I	A level for the debug logging. The log level can be specified for each module. The macro definitions for the log level are defined as following. OMXR_(module name)_LOG_LEVEL_(level) (module name) CORE : Log for OpenMAX IL Core UTIL : Log for OS dependent part CMN : Log for Media Component Common part VIDEO : Log for Video Domain AUDIO : Log for Audio Domain CNV : Log for the video converter part in Video Domain (level) ERROR, WARN, INFO, DEBUG The levels are defined as follows and ‘DEBUG’ is the most verbose logging level. ERROR < WARN < INFO < DEBUG The default level is ‘ERROR’ for all the modules.

[Return Values] See section 6.

[Notes]
- This function is not thread-safe. The IL client must not call this function from multiple threads in the same application process at the same time.

[Usage]

```
OMX_ERRORTYPE           err;
/*
    The following sample code sets the log level for CORE to ‘DEBUG’ and
    for CMN to ‘WARN’ respectively.
*/
err = OMXR_SetLogMode(OMXR_CORE_LOG_LEVEL_DEBUG |
OMXR_CMN_LOG_LEVEL_WARN);
if (err != OMX_ErrorNone) {
            printf("Can't set log mode! error code = 0x%08x¥n", err);
            return -1;
}
```

2.5. API Call Restrictions

2.5.1. Restrictions on OpenMAX IL Core Function Call

The OpenMAX IL Core Functions are not thread-safe. The IL client must call the functions exclusively in the following cases:

(1) The IL client calls the same OpenMAX IL Core function at the same time from multiple threads in the same process. For instance, thread A invokes OMX_Init( ) while thread B is calling OMX_Init( ).

(2) The IL client calls the different OpenMAX IL Core function at the same time from multiple threads in the same process. For instance, thread A invokes OMX_Deinit( ) while thread B is calling OMX_Init( ).

The OpenMAX IL Core functions must not be called from callback context.

2.5.2. Restrictions on OpenMAX IL Macro Function Call

The OpenMAX IL Macro Functions are basically thread-safe. But IL client must not call OMX_FreeHandle( ) and the other OpenMAX IL Macro functions at the same time from multiple threads in the same process.

The OpenMAX IL Macro functions can be called from callback context.

2.5.3. Restrictions on OMX Extended Function Call

See section 2.4.

3. State Transition

For the OpenMAX IL state transition model, see related document [1], 3.1.1.2. This section describes the supplement for the standard specifications.

3.1. Restrictions

- The OMX_StateWaitForResources state is not supported.

3.2. Valid OpenMAX IL Macro Function Calls

Table 3-1 shows the valid function calls for each state. In the table, 'X' indicates valid calls and '-' indicates invalid calls.

Table 3-1 Valid OpenMAX IL Macro Function Call for Each State

OpenMAX IL Macro Function / State	OMX_GetComponentVersion( )	OMX_SendCommand( )	OMX_GetParameter( )	OMX_SetParameter( )	OMX_GetConfig( )	OMX_SetConfig( )	OMX_GetExtensionIndex( )	OMX_GetState( )	OMX_UseBuffer( )	OMX_AllocateBuffer( )	OMX_FreeBuffer( )	OMX_EmptyThisBuffer( )	OMX_FillThisBuffer( )	OMX_UseEGLImage( )
OMX_StateLoaded	X	X[5]	X	X	X	X	X	X	X[1]	X[1]	X	-	-	-
OMX_StateIdle	X	X[5]	X	-	X	X	X	X	-	-	X	X[3]	X[3]	-
OMX_StateExecuting	X	X	X	-	X	X	X	X	-	-	X	X[6]	X[6]	-
OMX_StatePause	X	X	X	-	X	X	X	X	-	-	X	X[6]	X[6]	-
OMX_StateWaitForResources	X	X	X	X	X	X	X	X	X	X	X	-	-	-
OMX_StateInvalid	-	-	-	-	-	-	-	X	-	-	X	-	-	-
Disabled Port	X	X	X	X	X	X	X	X	X[2]	X[2]	X	X[4]	X[4]	-

X[1]: The call is valid during the state transition to OMX_StateIdle state.
X[2]: The call is valid during the port enable operation.
X[3]: The call is valid during the state transition to OMX_StateExecuting state.
X[4]: The call is valid during the port enable operation.
X[5]: The OMX_CommandMarkBuffer command is invalid.
X[6]: The call is invalid during the component state transition by OMX_CommandStateSet. Also the call is invalid for the port that is in a port disable operation or a port flush operation.

4. Indexes

This section describes the indexes that are available for all OMX Media Components. For Media Component dependent indexes, see each Media Component User's Manual.

4.1. Standard Indexes of OMX Media Component

Table 4-1 lists the OpenMAX IL standard indexes that are available for all OMX Media Components.

Table 4-1 Available Standard Indexes for OMX Media Component

Index	Corresponding Structure
OMX_IndexParamAudioInit	OMX_PORT_PARAM_TYPE
OMX_IndexParamVideoInit	OMX_PORT_PARAM_TYPE
OMX_IndexParamImageInit	OMX_PORT_PARAM_TYPE
OMX_IndexParamOtherInit	OMX_PORT_PARAM_TYPE
OMX_IndexParamStandardComponentRole	OMX_PARAM_COMPONENTROLETYPE
OMX_IndexParamCompBufferSupplier	OMX_PARAM_BUFFERSUPPLIERTYPE
OMX_IndexParamPortDefinition	OMX_PARAM_PORTDEFINITIONTYPE

5. Structures

This section describes the structures that are available for all OMX Media Components. For Media Component dependent structures, see each Media Component User's Manual.

5.1. Standard Structures

Table 5-1 lists the OpenMAX IL standard structures that are available for all OMX Media Components.

Table 5-1 OpenMAX IL Standard Structures

Section	Structure Name	Description
5.1.1	OMX_BUFFERHEADERTYPE	-
5.1.2	OMX_VERSIONTYPE	-
5.1.3	OMX_CALLBACKTYPE	-
5.1.4	OMX_PORT_PARAM_TYPE	-
5.1.5	OMX_PARAM_COMPONENTROLETYPE	-
5.1.6	OMX_PARAM_BUFFERSUPPLIERTYPE	-
5.1.7	OMX_PARAM_PORTDEFINITIONTYPE	-

5.1.1. OMX_BUFFERHEADERTYPE

[Reference] Related document [1] 3.1.2.7

[Notes] The following members must not be changed.

- *nSize*
- *nVersion*
- *nAllocLen*
- *pPlatformPrivate*
- *pInputPortPrivate*
- *pOutputPortPrivate*
- *nOutputPortIndex*
- *nInputPortIndex*

The following members are ignored on OMX_FillThisBuffer( ) calls.

- *nFilledLen*
- *nOffset*
- *hMarkTargetComponent*
- *pMarkData*
- *nTickCount*
- *nTimeStamp*
- *nFlags*

For the specifications of the *nOffset* and *nFlags* member on OMX_EmptyThisBuffer( ) calls, see each Media Component User's Manual.

The nFilledLen member on OMX_EmptyThisBuffer is set to 0 when EmptyBufferDone callback is returned.

The *pBuffer* member must not be changed unless otherwise specified in the Media Component User's Manual.

[Remarks] None.

5.1.2. OMX_VERSIONTYPE

[Reference] Related document [1] 3.1.2.4

[Notes] None.

[Remarks] None.

5.1.3. OMX_CALLBACKTYPE

[Reference] Related documents [1] 3.1.2.9

[Notes] None.

[Remarks] None.

5.1.4. OMX_PORT_PARAM_TYPE

[Reference] Related documents [1] 3.1.2.8

[Notes] − This structure is ignored on OMX_SetParameter( ) calls.

[Remarks] None.

5.1.5. OMX_PARAM_COMPONENTROLETYPE

[Reference] Related document [1] 8.2.2

[Notes] None.

[Remarks] None.

5.1.6. OMX_PARAM_BUFFERSUPPLIERTYPE

[Reference] Related document [1] 3.1.2.10

[Notes] None.

[Remarks] None.

5.1.7. OMX_PARAM_PORTDEFINITIONTYPE

[Reference] Related document [1] 3.1.2.12

[Notes] The following members are ignored.
- *bBufferContiguous* : fixed to OMX_FALSE
- *nBufferAlignment* : fixed to 0

For the following members, see each Media Component User's Manual.
- *nBufferCountActual*
- *nBufferCountMin*
- *nBufferSize*
- *format*

[Remarks] None.

6. Error Codes

6.1. OpenMAX IL Standard Error Codes

This section describes the supplement for related document [1] 3.1.1.3 and how to handle the error codes. The errors are classified by type of error handling.

6.1.1. Normal Return

Table 6-1 enumerates the normal return code. The Media Components continue normal operation.

Table 6-1 OpenMAX IL Standard Error Codes: Normal Return

Error Code	Description
OMX_ErrorNone	-
OMX_ErrorNoMore	-

6.1.2. Operation Errors

Table 6-2 enumerates the operation errors. After the return of these error codes, the Media Components continue the operation. But the API calls from the IL client may be incorrect, the API usage and the arguments should be checked.

Table 6-2 OpenMAX IL Standard Error Codes: Operation Errors

Error Code	Description
OMX_ErrorInvalidComponentName	-
OMX_ErrorComponentNotFound	-
OMX_ErrorSameState	-
OMX_ErrorIncorrectStateTransition	-
OMX_ErrorIncorrectStateOperation	-
OMX_ErrorNotImplemented	-
OMX_ErrorBadPortIndex	-
OMX_ErrorUnsupportedIndex	-
OMX_ErrorUnsupportedSetting	-
OMX_ErrorBadParameter	-

6.1.3. Critical Errors

Table 6-3 shows the critical errors. The OMX Media Component cannot continue with its operation. It requires a reallocation for the Media Component or a reset for system level. To reallocate the Media Component, call OMX_FreeHandle( ) on the Media Component that returned the critical error and call OMX_GetHandle( ) to create a new instance.

Table 6-3 OpenMAX IL Standard Error Codes: Critical Errors

Error Code	Description
OMX_ErrorTimeout	A timeout detected in the API function call. This error may occur by the CPU load or occupation by the other modules except the OMX Media Component.
OMX_ErrorHardware	The hardware failed to respond and OMX Media Component detected the failure.
OMX_ErrorUndefined	There was an error in calling OS system calls or an inconsistency detected internally.
OMX_ErrorInvalidState	The Media Component is in the OMX_StateInvalid state. In this state API function calls except OMX_GetState( ), OMX_FreeBuffer( ) and OMX_FreeHandle( ) results in this error.
OMX_ErrorInsufficientResources	There were insufficient memory resources or hardware resources to continue with the component operation. Make sure the resource allocation and the resource status at the system level.
OMX_ErrorInvalidComponent	An invalid component handle was specified in the API function call.
OMX_ErrorVersionMismatch	An invalid component version was specified in the API function call.
OMX_ErrorPortUnpopulated	The Media Component lost buffers of the port. This error may occur by calling OMX_FreeBuffer() in the status except a port disable operation and a state transition to OMX_StateLoaded.

6.1.4. Flow Control and Stream Errors

Table 6-4 enumerates the errors that indicate invalid flow control operation or corruption in input streams. The details depend on each Media Component. See each Media Component User's Manual.

Table 6-4 OpenMAX IL Standard Error Codes: Flow Control and Stream Errors

Error Code	Description
OMX_ErrorStreamCorrupt	-
OMX_ErrorUnderflow	-
OMX_ErrorOverflow	-

6.2. OMX Extended Error Codes

This section describes the OMX extended error codes and how to handle the errors.

6.2.1. Critical Errors

Table 6-5 shows the critical errors and how to handle the errors.

Table 6-5 OMX Extended Error Codes: Critical Errors

Error Code	Description
OMXR_ErrorFileNotFound	OMX_Init( ) returns OMXR_ErrorFileNotFound when the configuration file or the include files defined in configuration file were not found. Make sure that the configuration file is in the specified file path. For the installation of the configuration file, see related document [2].
OMXR_ErrorFileRead	OMX_Init( ) returns OMXR_ErrorFileRead when the OMX failed to access the configuration file. Make sure that the configuration file has a read permission and the file is readable.
OMXR_ErrorIllegalConfig	OMX_Init( ) returns OMXR_ErrorIllegalConfig when a syntax error was detected in the configuration file. Make sure the content of the configuration file is as delivered.

REVISION HISTORY	OMX Media Component User's Manual : Common Part

Rev.	Date	Description	
		Page	Summary
0.06	Dec. 20, 2013	―	Draft revision based on Japanese User's Manual Rev.0.06.
0.07	Jan. 31, 2014	―	Fixed typos.
0.08	Mar. 14, 2014	20	Removed FATAL log level.
0.09	May. 30, 2014	5	Fixed typos.
	Jul. 25, 2014	25	Add description of nFilledLen.
1.00	Aug. 27, 2014	―	Fixed typos. Added some descriptions.

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.3

OMX Media Component
User's Manual: Common Part

Publication Date: Rev. 1.00 Aug. 29, 2014

Published by: Renesas Electronics Corporation

OMX Media Component

User's Manual: Common Part

Renesas Electronics Corporation