**Document Type: Detail Design**

**Document Name:**

**ADSP OMX RDR**

Renesas Design Vietnam Co. Ltd

R-Car Software Solution 2 Group

Middleware 1 Team

Renesas Electronics Corporation

Table of Contents

[1. Overview 3](#_Toc6318628)

[2. Function list 4](#_Toc6318629)

[3. Detail information 5](#_Toc6318630)

[3.1 Data type and Macro definition 5](#_Toc6318631)

[3.2 Function definition 6](#_Toc6318632)

[3.2.1 OMX Capture 6](#_Toc6318633)

[3.2.1.1 CAPTURE\_SetConfig\_Send 6](#_Toc6318634)

[3.2.1.2 CAPTURE\_SetConfigIndex 7](#_Toc6318635)

[3.2.1.3 CAPTURE\_Setup 8](#_Toc6318636)

[3.2.1.4 CAPTURE\_RuntimeInit 9](#_Toc6318637)

[3.2.1.5 CAPTURE\_GetParam 10](#_Toc6318638)

[3.2.1.6 CAPTURE\_TimeStamp 11](#_Toc6318639)

[3.2.1.7 CAPTURE\_GetParameter 12](#_Toc6318640)

[3.2.1.8 CAPTURE\_SetParameter 13](#_Toc6318641)

[3.2.1.9 CAPTURE\_ComponentInit 14](#_Toc6318642)

[3.2.1.10 CAPTURE\_ComponentDeInit 15](#_Toc6318643)

[3.2.1.11 CAPTURE\_ComponentCreate 16](#_Toc6318644)

[3.2.2 OMX RENDERER 18](#_Toc6318645)

[3.2.2.1 RENDERER\_SetConfig\_Send 18](#_Toc6318646)

[3.2.2.2 RENDERER\_SetConfigIndex 19](#_Toc6318647)

[3.2.2.3 RENDERER\_Setup 20](#_Toc6318648)

[3.2.2.4 RENDERER\_RuntimeInit 21](#_Toc6318649)

[3.2.2.5 RENDERER\_GetParam 22](#_Toc6318650)

[3.2.2.6 RENDERER\_TimeStamp 23](#_Toc6318651)

[3.2.2.7 RENDERER\_GetParameter 24](#_Toc6318652)

[3.2.2.8 RENDERER\_SetParameter 25](#_Toc6318653)

[3.2.2.9 RENDERER\_ComponentInit 26](#_Toc6318654)

[3.2.2.10 RENDERER\_ComponentDeInit 27](#_Toc6318655)

[3.2.2.11 RENDERER\_ComponentCreate 28](#_Toc6318656)

[3.2.3 OMX Codec Base 30](#_Toc6318657)

[3.2.3.1 xaomx\_create\_buffer 30](#_Toc6318658)

[3.2.3.2 xaomx\_response\_process 32](#_Toc6318659)

[4. Revision history 33](#_Toc6318660)

List of Figures

[Figure 1‑1 The software architecture 3](#_Toc6318661)

[Figure 3‑1 CAPTURE\_SetConfig\_Send flowchart 6](#_Toc6318662)

[Figure 3‑2 CAPTURE\_SetConfigIndex flowchart 7](#_Toc6318663)

[Figure 3‑3 CAPTURE\_Setup flowchart 8](#_Toc6318664)

[Figure 3‑4 CAPTURE\_RuntimeInit flowchart 9](#_Toc6318665)

[Figure 3‑5 CAPTURE\_GetParam flowchart 10](#_Toc6318666)

[Figure 3‑6 CAPTURE\_TimeStamp flowchart 11](#_Toc6318667)

[Figure 3‑7 CAPTURE\_GetParameter flowchart 12](#_Toc6318668)

[Figure 3‑8 CAPTURE\_SetParameter flowchart 13](#_Toc6318669)

[Figure 3‑9 CAPTURE\_ComponentInit flowchart 14](#_Toc6318670)

[Figure 3‑10 CAPTURE\_ComponentDeInit flowchart 15](#_Toc6318671)

[Figure 3‑11 CAPTURE\_ComponentCreate flowchart 17](#_Toc6318672)

[Figure 3‑12 RENDERER\_SetConfig\_Send flowchart 18](#_Toc6318673)

[Figure 3‑13 RENDERER\_SetConfigIndex flowchart 19](#_Toc6318674)

[Figure 3‑14 RENDERER\_Setup flowchart 20](#_Toc6318675)

[Figure 3‑15 RENDERER\_GetParam flowchart 22](#_Toc6318676)

[Figure 3‑16 RENDERER\_TimeStamp flowchart 23](#_Toc6318677)

[Figure 3‑17 RENDERER\_GetParameter flowchart 24](#_Toc6318678)

[Figure 3‑18 RENDERER\_SetParameter flowchart 25](#_Toc6318679)

[Figure 3‑19 RENDERER\_ComponentInit flowchart 26](#_Toc6318680)

[Figure 3‑20 RENDERER\_ComponentDeInit flowchart 27](#_Toc6318681)

[Figure 3‑21 RENDERER\_ComponentCreate flowchart 29](#_Toc6318682)

[Figure 3‑22 xaomx\_create\_buffer flowchart 31](#_Toc6318683)

[Figure 3‑23 xaomx\_response\_process flowchart 32](#_Toc6318684)

List of Table

[Table 2‑1 Function list of OMX Capture 4](#_Toc6318685)

[Table 2‑2 Function list of OMX Renderer 4](#_Toc6318686)

[Table 2‑3 Function list of OMX Codec base 4](#_Toc6318687)

[Table 3‑1 XAOMXCapture type structure information 5](#_Toc6318688)

[Table 3‑2 XAOMXRenderer type structure information 5](#_Toc6318689)

[Table 3‑3 Macro definitions of OMX Capture 5](#_Toc6318690)

[Table 3‑4 Macro definitions of OMX Renderer 5](#_Toc6318691)

[Table 2‑1 Function list of OMX Capture 4](#_Toc6318692)

[Table 2‑2 Function list of OMX Renderer 4](#_Toc6318693)

[Table 2‑3 Function list of OMX Codec base 4](#_Toc6318694)

[Table 3‑1 XAOMXCapture type structure information 5](#_Toc6318695)

[Table 3‑2 XAOMXRenderer type structure information 5](#_Toc6318696)

[Table 3‑3 Macro definitions of OMX Capture 5](#_Toc6318697)

[Table 3‑4 Macro definitions of OMX Renderer 5](#_Toc6318698)

# Overview

In this chapter, overview of OMX Capture/Renderer interface is explained.

User Application

OMX TDM

OMX Renderer/Capture

OMX Equalizer

User Space

ADSP Interface for Linux (base)

Kernel Space

ARM

ADSP Driver for Linux

Audio HW

ADSP

NC/EC

ADSP Framework

Renderer/Capture Plugin

TDM Plugin

Equalizer Plugin

Codec

This document’s target is in side of red square.

DAC/ADC

SRC/CTU/MIX/ DVC/ SSI/

ADMA

Figure 1‑1 ****The software architecture****

# Function list

The following is list of functions:

Table 2‑1 Function list of OMX Capture

|  |  |  |
| --- | --- | --- |
| **No.** | **Function Name** | **Outline** |
| 1 | CAPTURE\_SetConfig\_Send | Prepare message contain volume value want to set parameters for XF\_SET\_PARAM command to send to plug-in. |
| 2 | CAPTURE\_SetConfigIndex | Check the index of parameter structure is correct. |
| 3 | CAPTURE\_Setup | Capture prepare codec setup parameters. |
| 4 | CAPTURE\_RuntimeInit | Gets the command index of 1st and 2nd DMA devices |
| 5 | CAPTURE\_GetParam | Get DMA channel 1, 2 value from response structure. |
| 6 | CAPTURE\_TimeStamp | Calculate timestamp for current output buffer and next buffer. |
| 7 | CAPTURE\_GetParameter | Get Capture parameter. |
| 8 | CAPTURE\_SetParameter | Set Capture parameter. |
| 9 | CAPTURE\_ComponentDeInit | Destroy Capture component and private data. |
| 10 | CAPTURE\_ComponentInit | Initialize base codec interface, parameters, PCM and Capture format. |
| 11 | CAPTURE\_ComponentCreate | Create base component and initialize the Capture component. |

Table 2‑2 Function list of OMX Renderer

|  |  |  |
| --- | --- | --- |
| **No.** | **Function Name** | **Outline** |
| 1 | RENDERER\_SetConfig\_Send | Prepare message contain volume value want to set parameters for XF\_SET\_PARAM command to send to Renderer plug-in. |
| 2 | RENDERER\_SetConfigIndex | Check the index of parameter structure is correct. |
| 3 | RENDERER\_Setup | Renderer prepare codec setup parameters. |
| 4 | RENDERER\_RuntimeInit | This is a dummy function, not use in Renderer runtime initialization. |
| 5 | RENDERER\_GetParam | This is a dummy function. |
| 6 | RENDERER\_TimeStamp | Calculate timestamp for current input buffer and next buffer. |
| 7 | RENDERER\_GetParameter | Get Renderer parameter. |
| 8 | RENDERER\_SetParameter | Set Renderer parameter. |
| 9 | RENDERER\_ComponentDeInit | Destroy Renderer component and private data. |
| 10 | RENDERER\_ComponentInit | Initialize base codec interface, parameters, PCM and Renderer format. |
| 11 | RENDERER\_ComponentCreate | Create base component and initialize the Renderer component. |

Table 2‑3 Function list of OMX Codec base

|  |  |  |
| --- | --- | --- |
| **No.** | **Function Name** | **Outline** |
| 1 | xaomx\_create\_buffer | Allocate buffer header for OMX, set buffer pointer and request ADSP to map this buffer to optimize latency. |
| 2 | xaomx\_response\_process | Handle response received massage from proxy. To reduce audio Latency on Android this function add an opcode XF\_MMAP\_THIS\_BUFFER. |

# Detail information

This section describes detail information of data types, macro definitions.

## Data type and Macro definition

Table 3‑1 XAOMXCapture type structure information

|  |  |
| --- | --- |
| Member name | Outline |
| XAOMXCodecBase base | Generic codec structure. |
| OMX\_AUDIO\_PARAM\_PCMMODETYPE sPCM | PCM-specific parameters (output port). |
| XAOMX\_AUDIO\_PARAM\_CAPTURE sCAPTURE | Capture parameters. |

Note:

Structure XAOMXCodecBase is a base codec structure of OMX in omx-codec-base.h

Structure OMX\_AUDIO\_PARAM\_PCMMODETYPE is a PCM format description in OMX\_Audio.h

Structure XAOMX\_AUDIO\_PARAM\_CAPTURE is a parametric Capture settings structure.

Table 3‑2 XAOMXRenderer type structure information

|  |  |
| --- | --- |
| Member name | Outline |
| XAOMXCodecBase base | Generic codec structure. |
| OMX\_AUDIO\_PARAM\_PCMMODETYPE sPCM | PCM-specific parameters (input port). |
| XAOMX\_AUDIO\_PARAM\_RENDERER sRENDERER | Renderer parameters. |

Note:

Structure XAOMX\_AUDIO\_PARAM\_RENDERER is a parametric Renderer settings structure.

Table 3‑3 Macro definitions of OMX Capture

|  |  |  |
| --- | --- | --- |
| Macro | Value | Outline |
| NUM\_OUTPUT\_BUFFERS | 4 | Total amount of output buffers. |
| OUTPUT\_BUFFER\_LENGTH | 4096 | Default output buffer length. |
| BUFFER\_ALIGNMENT | 32 | Required data alignment of Capture. |

Table 3‑4 Macro definitions of OMX Renderer

|  |  |  |
| --- | --- | --- |
| Macro | Value | Outline |
| NUM\_INPUT\_BUFFERS | 4 | Total amount of input buffers. |
| INPUT\_BUFFER\_LENGTH | 4096 | Default input buffer length. |
| BUFFER\_ALIGNMENT | 32 | Required data alignment of Renderer. |

## Function definition

### OMX Capture

#### CAPTURE\_SetConfig\_Send

DD\_API\_RDR\_01\_001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 CAPTURE\_SetConfig\_Send(XAOMXCodecBase \*pBase,  xf\_set\_param\_msg\_t \*msg,  OMX\_PTR pParam) | | | |
| **Function** | Prepare message contain volume value want to set parameters for XF\_SET\_PARAM command to send to plug-in. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Capture component. |
| xf\_set\_param\_msg\_t\* | msg | O | Message send to Capture plugin. |
| OMX\_PTR | pParam | I | Parameter want to send to plug-in. |
| **Return value** | length | | Number of parameters want to set for Capture plugin. | |
| 0 | | Do not set volume to plug-in. | |
| **Description** | * CAPTURE\_SetConfig\_Send processing:   - Prepare volume rate value into command message. | | | |

[Covers: FD\_API\_RDR\_006]

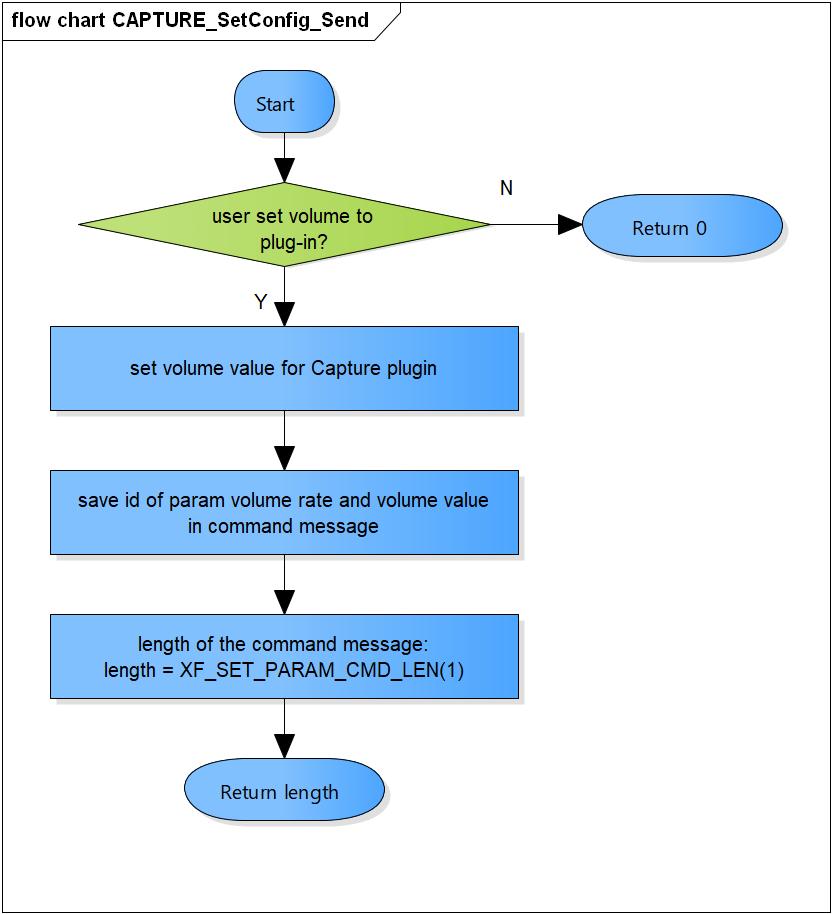


Figure 3‑1 CAPTURE\_SetConfig\_Send flowchart

#### CAPTURE\_SetConfigIndex

DD\_API\_RDR\_01\_002

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE CAPTURE\_SetConfigIndex(OMX\_INDEXTYPE nIndex) | | | |
| **Function** | Check the index of parameter structure is correct. | | | |
| **Arguments** | Type | Name | I/O | Description |
| OMX\_INDEXTYPE | nIndex | I | Index of parameter structure.  Valid range: XAOMX\_IndexParamAudioCapture |
| **Return value** | XA\_NO\_ERROR | | Normal return. | |
| OMX\_ErrorUnsupportedIndex | | Wrong Index of structure. | |
| **Description** | * CAPTURE\_SetConfig\_Send processing:   - Check the index of parameter structure is correct. | | | |

[Covers: FD\_API\_RDR\_006]

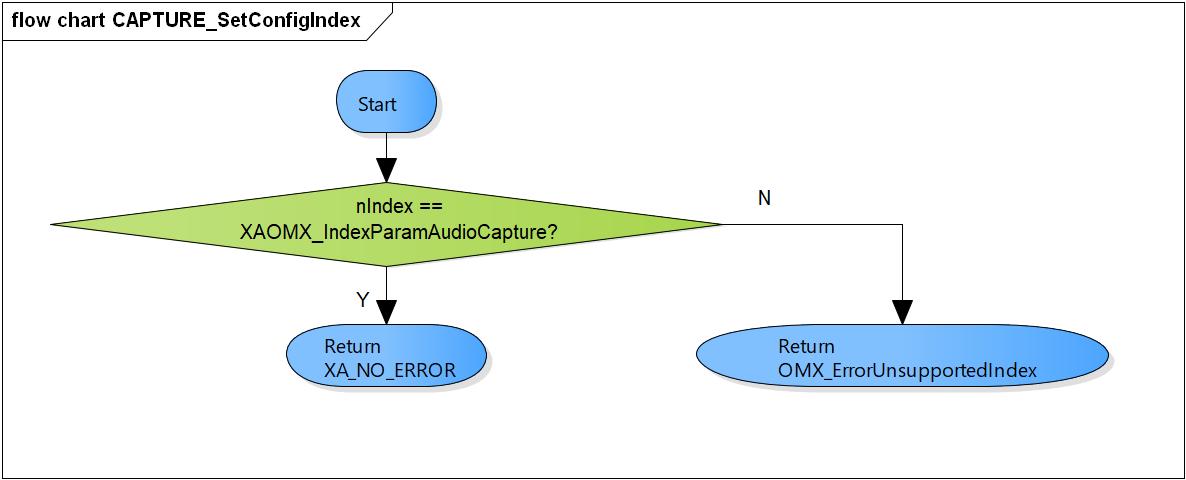


Figure 3‑2 CAPTURE\_SetConfigIndex flowchart

#### CAPTURE\_Setup

DD\_API\_RDR\_01\_003

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 CAPTURE\_Setup(XAOMXCodecBase \*pBase, xf\_set\_param\_msg\_t \*msg) | | | |
| **Function** | Capture prepare codec setup parameters. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Capture component. |
| xf\_set\_param\_msg\_t\* | msg | O | Message send to Capture plugin. |
| **Return value** | length | | Number of parameters want to set for Capture plugin. | |
| **Description** | * CAPTURE\_Setup processing:   - Prepare PCM width, channels, sample rate, input1, 2; DMA channel 1, 2; out sample rate, volume rate and ring-buffer number values into command message. | | | |

[Covers: FD\_API\_RDR\_003]

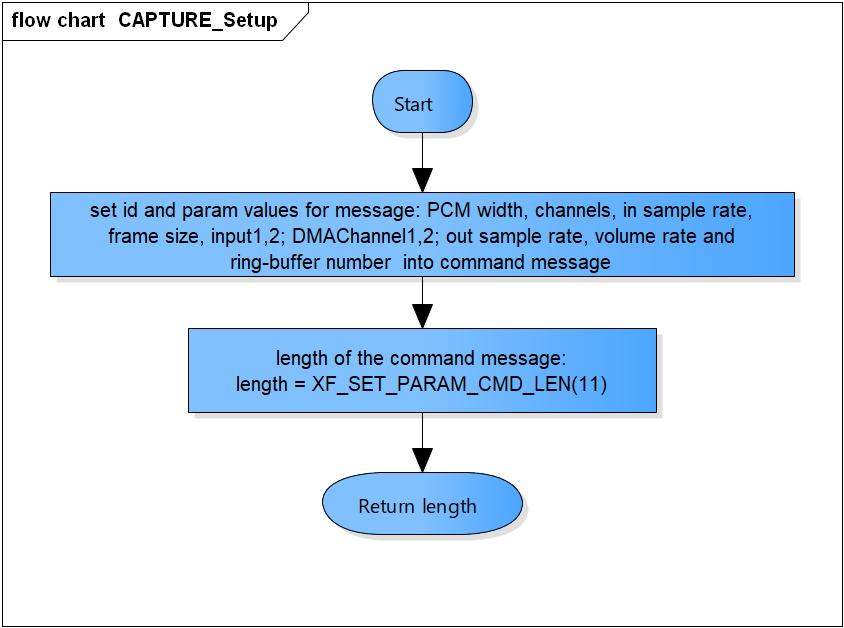


Figure 3‑3 CAPTURE\_Setup flowchart

#### CAPTURE\_RuntimeInit

DD\_API\_RDR\_01\_004

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 CAPTURE\_RuntimeInit(XAOMXCodecBase \*pBase, xf\_start\_msg\_t \*msg) | | | |
| **Function** | This function gets the command index of 1st and 2nd DMA devices | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Capture component. |
| xf\_start\_msg\_t\* | msg | O | Message send to Capture plugin. |
| **Return value** | length | | Number of parameters want to set for Capture plugin. | |
| **Description** | * CAPTURE\_RuntimeInit processing:   - Save DMA channel 1, 2 into command structure. | | | |

[Covers: FD\_API\_RDR\_003]

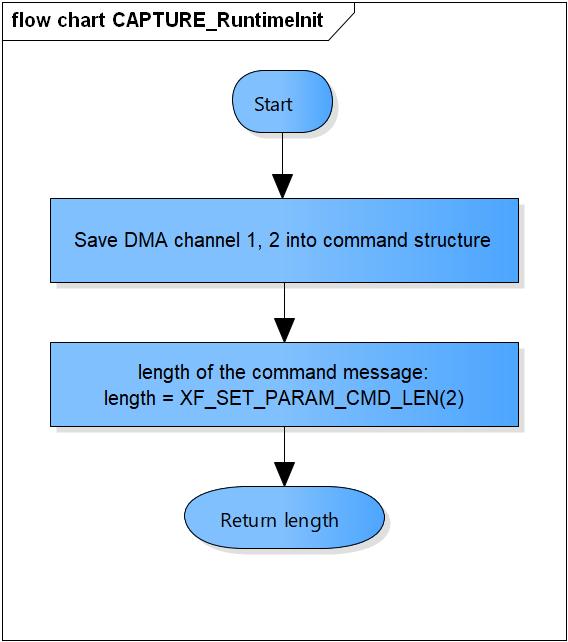


Figure 3‑4 CAPTURE\_RuntimeInit flowchart

#### CAPTURE\_GetParam

DD\_API\_RDR\_01\_005

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 CAPTURE\_GetParam(XAOMXCodecBase \*pBase,  xf\_get\_param\_msg\_t \*msg,  u32 length) | | | |
| **Function** | Get DMA channel 1, 2 value from response structure. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I/O | Pointer to base codec structure of Capture component. |
| xf\_get\_param\_msg\_t \* | msg | I | Message receive from Capture plugin. |
| u32 | lenghth | I | Length of message. |
| **Return value** | 0 | | Normal return. | |
| -EBADF | | The message length is invalid. | |
| **Description** | * CAPTURE\_GetParam processing:   - Get DMA channel 1, 2 values from response structure. | | | |

[Covers: FD\_API\_RDR\_003]

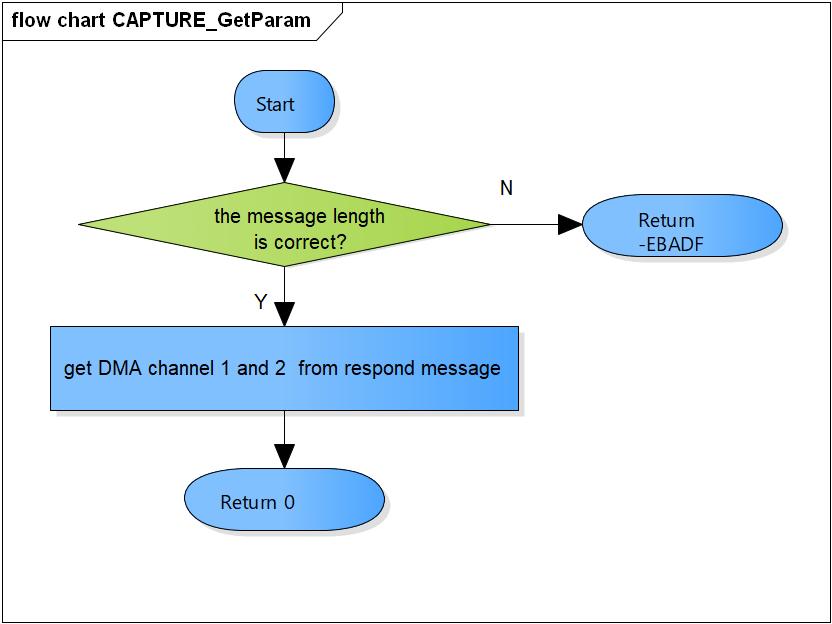


Figure 3‑5 CAPTURE\_GetParam flowchart

#### CAPTURE\_TimeStamp

DD\_API\_RDR\_01\_006

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static void CAPTURE\_TimeStamp(XAOMXCodecBase \*pBase,  OMX\_BUFFERHEADERTYPE \*pBufHdr) | | | |
| **Function** | Calculate timestamp for current output buffer and next buffer. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I/O | Pointer to base codec structure of Capture component. |
| OMX\_BUFFERHEADERTYPE\* | msg | O | Output buffer. |
| **Return value** | None. | | | |
| **Description** | * CAPTURE\_TimeStamp processing:   - Add current timestamp to the output buffer.  - Calculate timestamp for the next buffer. | | | |

[Covers: FD\_API\_RDR\_003]

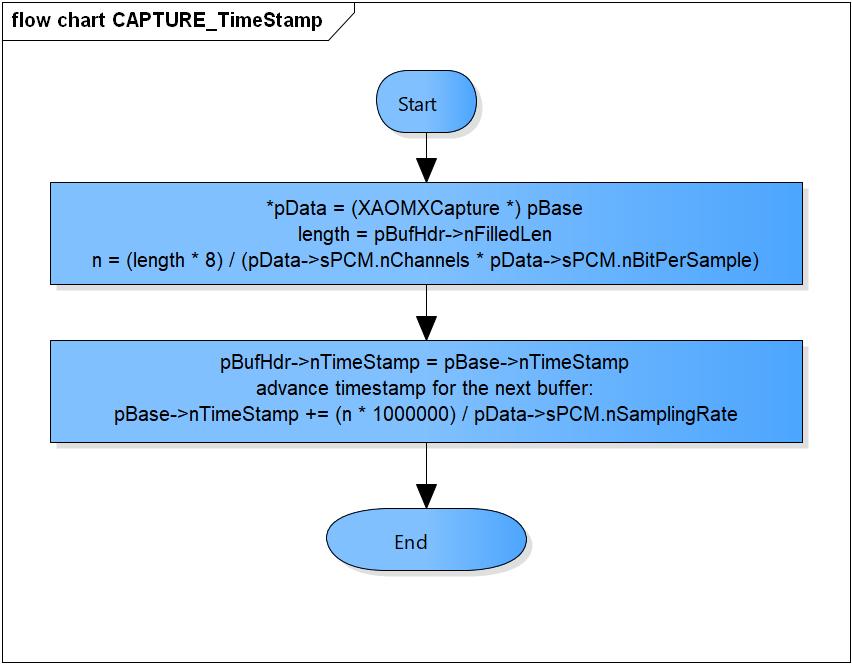


Figure 3‑6 CAPTURE\_TimeStamp flowchart

#### CAPTURE\_GetParameter

DD\_API\_RDR\_01\_007

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE CAPTURE\_GetParameter(XAOMXCodecBase \*pBase,  OMX\_INDEXTYPE nIndex,  OMX\_PTR pParam) | | | |
| **Function** | Get parameter and save to pParam. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I/O | Pointer to base codec structure of Capture component. |
| OMX\_INDEXTYPE | nIndex | I | Index of parameter structure.  Supported index:  OMX\_IndexParamAudioPcm  XAOMX\_IndexParamAudioCapture  OMX\_IndexParamCompBufferSupplier |
| OMX\_PTR | pParam | O | Pointer to parameter structure want to save. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorUnsupportedIndex | | Wrong Index of structure. | |
| **Description** | * CAPTURE\_GetParameter processing:   - Get parameters of OMX\_AUDIO\_PARAM\_PCMMODETYPE, XAOMX\_AUDIO\_PARAM\_CAPTURE, OMX\_PARAM\_BUFFERSUPPLIERTYPE structure base on nIndex. | | | |

[Covers: FD\_API\_RDR\_004]

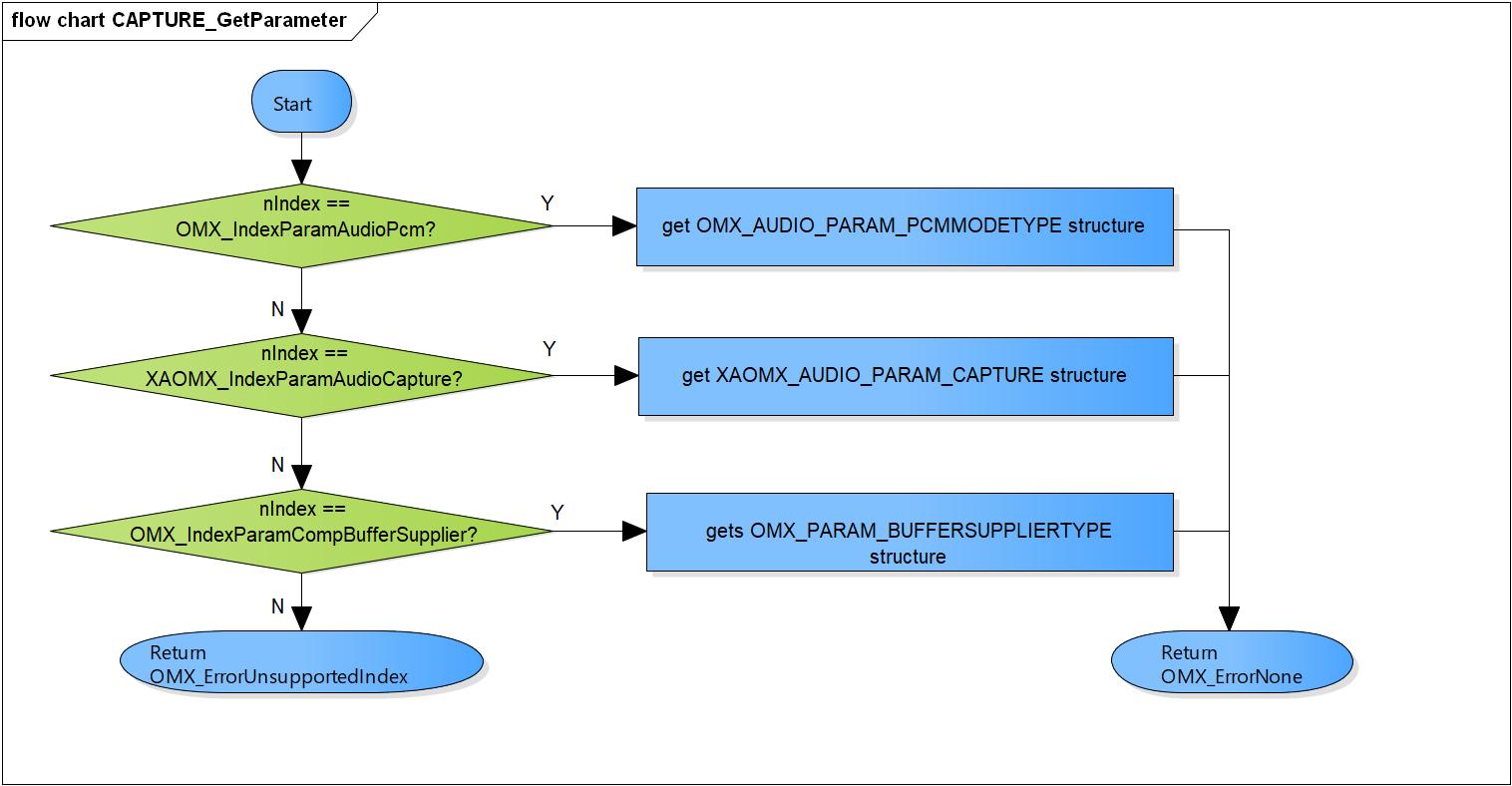


Figure 3‑7 CAPTURE\_GetParameter flowchart

#### CAPTURE\_SetParameter

DD\_API\_RDR\_01\_008

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE CAPTURE\_SetParameter(XAOMXCodecBase \*pBase,  OMX\_INDEXTYPE nIndex,  OMX\_PTR pParam) | | | |
| **Function** | Set parameter from pParam and save to parameter structure. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I/O | Pointer to base codec structure of Capture component. |
| OMX\_INDEXTYPE | nIndex | I | Index of parameter structure.  Supported index:  OMX\_IndexParamAudioPcm  XAOMX\_IndexParamAudioCapture  OMX\_IndexParamCompBufferSupplier |
| OMX\_PTR | pParam | I | Pointer to parameter structure want to set. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorUnsupportedIndex | | Wrong Index of structure. | |
| OMX\_ErrorBadPortIndex | | Port index is invalid. | |
| **Description** | * CAPTURE\_SetParameter processing:   - Set parameters of OMX\_AUDIO\_PARAM\_PCMMODETYPE, XAOMX\_AUDIO\_PARAM\_CAPTURE, OMX\_PARAM\_BUFFERSUPPLIERTYPE structure base on nIndex. | | | |

[Covers: FD\_API\_RDR\_005]

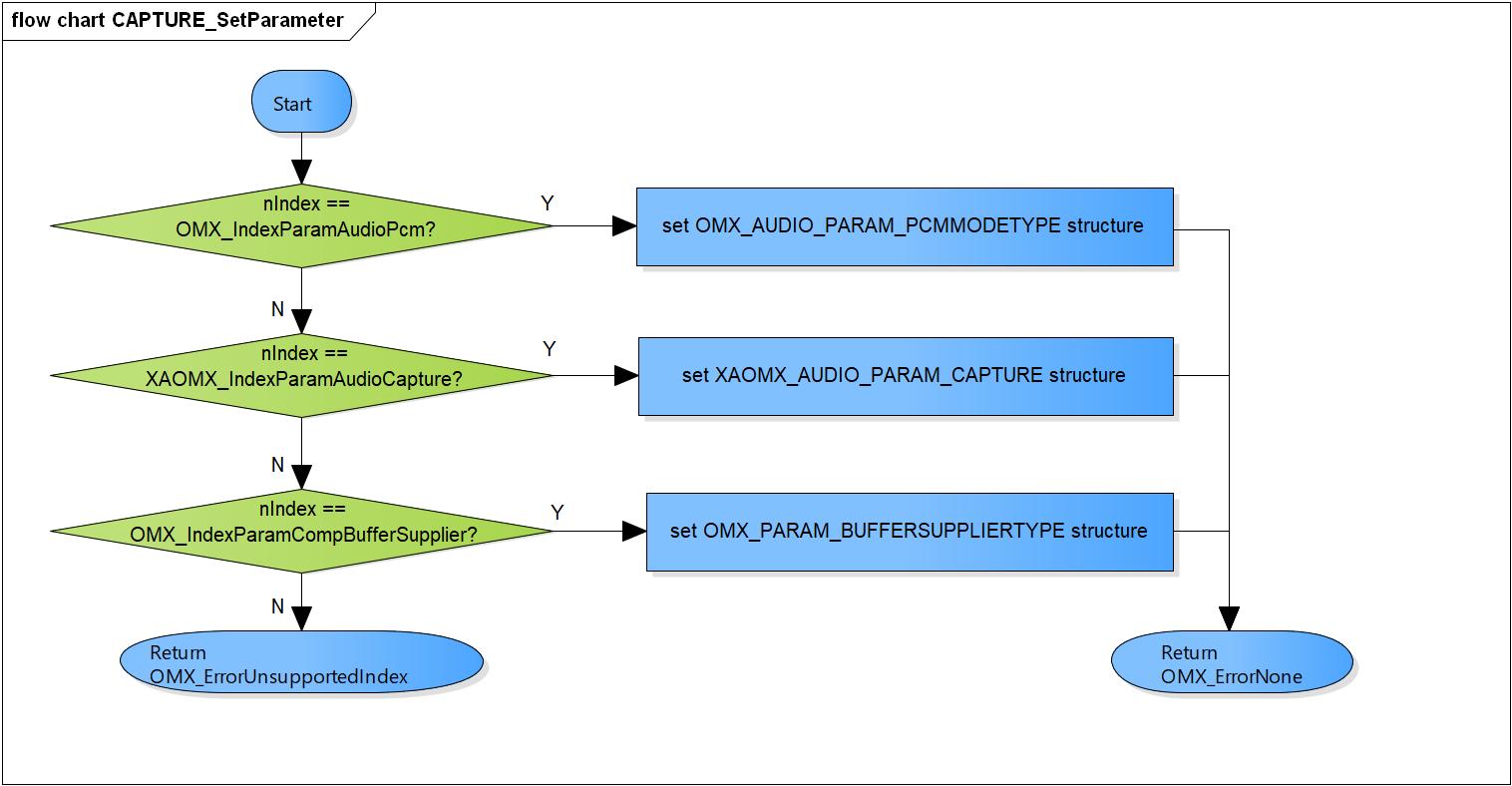


Figure 3‑8 CAPTURE\_SetParameter flowchart

#### CAPTURE\_ComponentInit

DD\_API\_RDR\_01\_009

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE CAPTURE\_ComponentInit(xf\_proxy\_t \*proxy,  OMX\_HANDLETYPE hComponent,  OMX\_PTR pAppData,  OMX\_CALLBACKTYPE \*pCallbacks) | | | |
| **Function** | Initialize base codec interface, parameters, PCM and Capture format. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_proxy\_t \* | proxy | O | OpenMAX interface proxy. |
| OMX\_HANDLETYPE | hComponent | O | Pointer to Capture component handle. |
| OMX\_PTR | pAppData | I | Pointer to pAppData of Capture component. |
| OMX\_CALLBACKTYPE | pCallbacks | I | Pointer to Callbacks of Capture component. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorInsufficientResources | | Initialize base codec interface is fail. | |
| **Description** | * CAPTURE\_ComponentInit processing:   - Initialize base codec interface, parameters, PCM and Capture format. | | | |

[Covers: FD\_API\_RDR\_001]

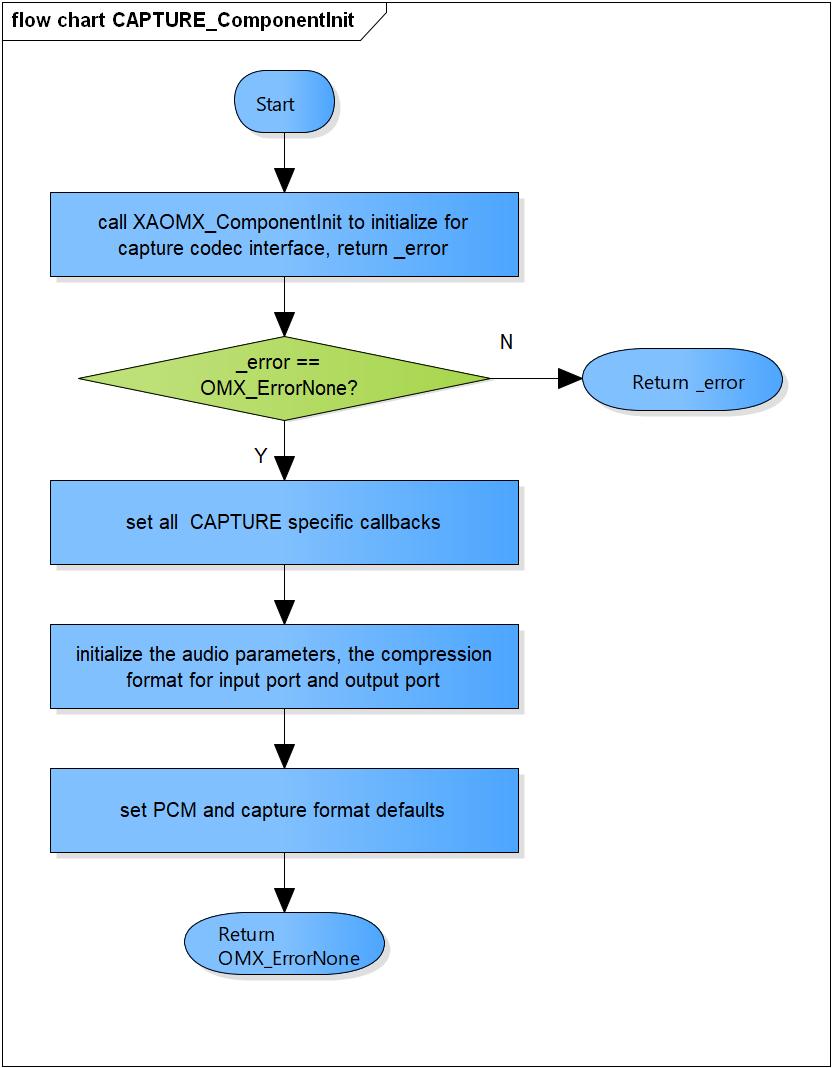


Figure 3‑9 CAPTURE\_ComponentInit flowchart

#### CAPTURE\_ComponentDeInit

DD\_API\_RDR\_01\_010

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE CAPTURE\_ComponentDeInit(OMX\_HANDLETYPE hComponent) | | | |
| **Function** | Destroy Capture component and private data. | | | |
| **Arguments** | Type | Name | I/O | Description |
| OMX\_HANDLETYPE | hComponent | O | Pointer to Capture component handle. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorBadParameter | | Private data is NULL. | |
| **Description** | * CAPTURE\_ComponentDeInit processing:   - Destroy Capture component and private data. | | | |

[Covers: FD\_API\_RDR\_002]

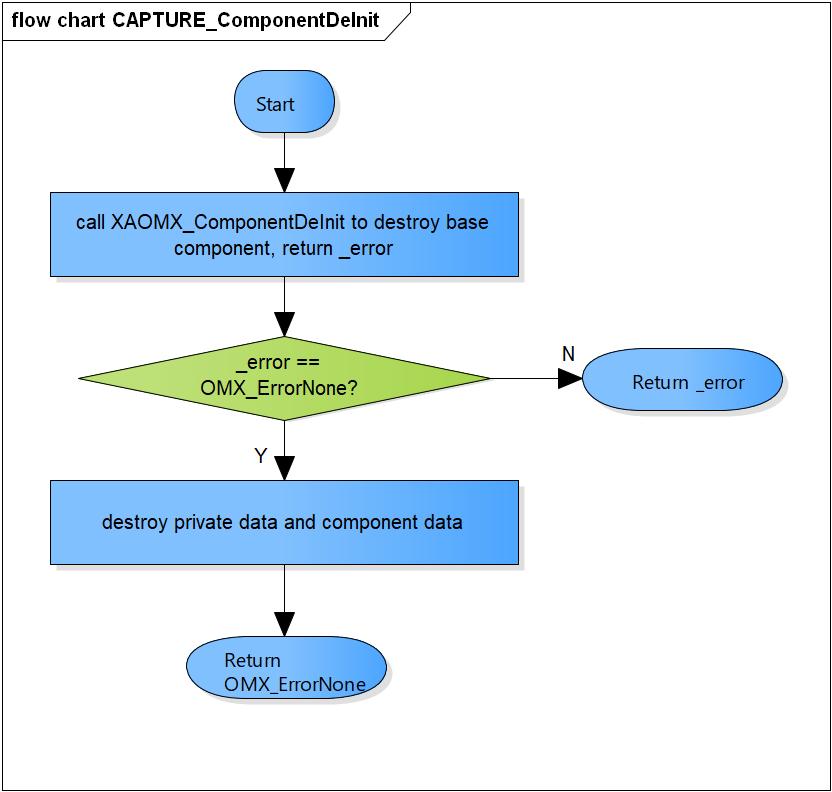


Figure 3‑10 CAPTURE\_ComponentDeInit flowchart

#### CAPTURE\_ComponentCreate

DD\_API\_RDR\_01\_011

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | OMX\_ERRORTYPE CAPTURE\_ComponentCreate(xf\_proxy\_t \*proxy, OMX\_HANDLETYPE \*hComponent, OMX\_PTR pAppData, OMX\_CALLBACKTYPE \*pCallbacks) | | | |
| **Function** | Create base component and initialize the Capture component. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_proxy\_t\* | proxy | O | OpenMAX interface proxy. |
| OMX\_HANDLETYPE\* | hComponent | O | Pointer to Capture component handle. |
| OMX\_PTR | pAppData | I | Pointer to pAppData of Capture component. |
| OMX\_CALLBACKTYPE \* | pCallbacks | I | Pointer to Callbacks of Capture component. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorInsufficientResources | | Cannot create the base component.  Cannot allocate private memory.  Initialize the Capture component is fail. | |
| **Description** | * CAPTURE\_ComponentCreate processing:   - Create the base component.  - Allocate the private memory.  - Initialize the Capture component. | | | |

[Covers: FD\_API\_RDR\_001]

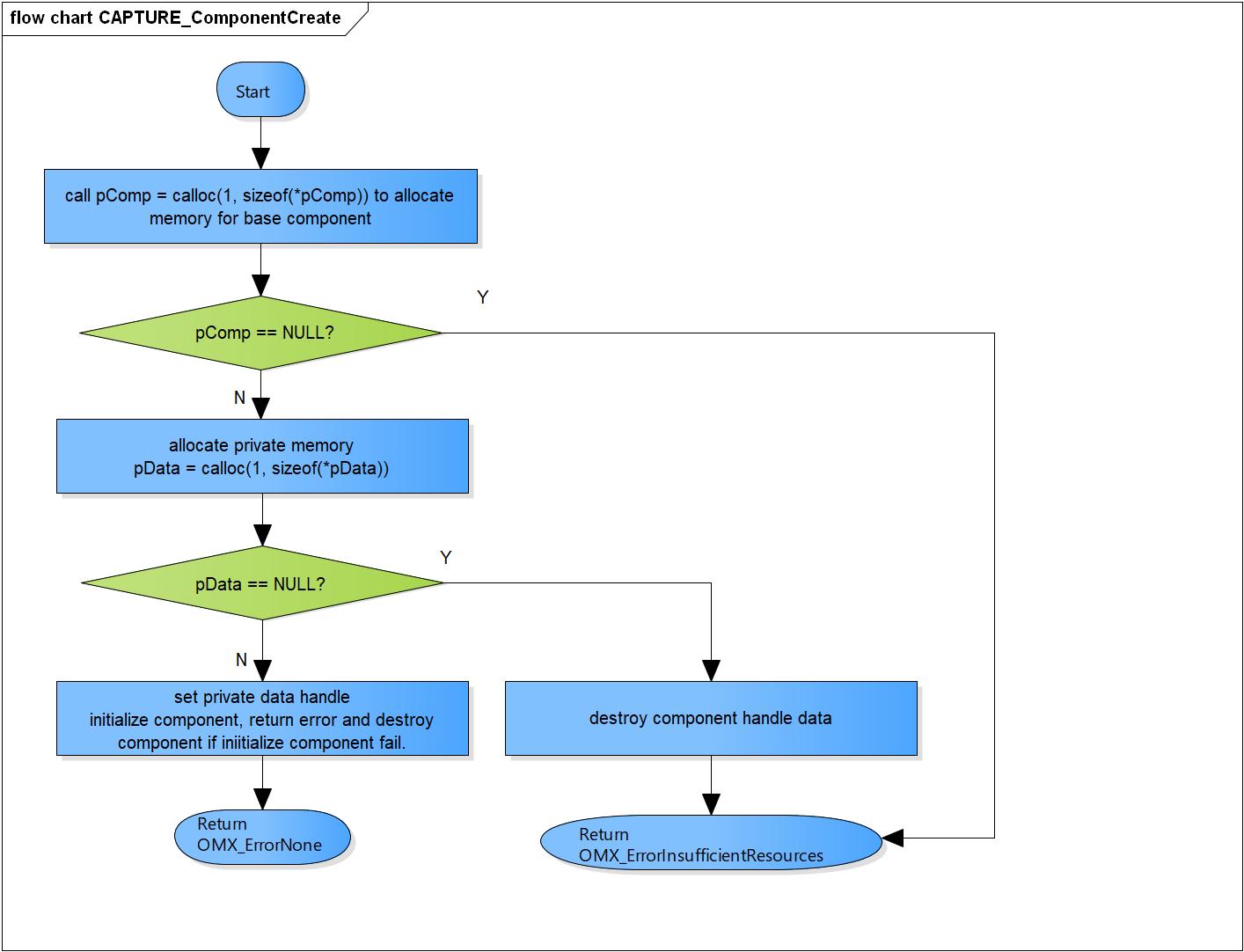


Figure 3‑11 CAPTURE\_ComponentCreate flowchart

### OMX RENDERER

#### RENDERER\_SetConfig\_Send

DD\_API\_RDR\_01\_012

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 RENDERER\_SetConfig\_Send(XAOMXCodecBase \*pBase,  xf\_set\_param\_msg\_t \*msg,  OMX\_PTR pParam) | | | |
| **Function** | Prepare message contain volume value want to set parameters for XF\_SET\_PARAM command to send to Renderer plug-in. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Renderer component. |
| xf\_set\_param\_msg\_t\* | msg | O | Message send to Renderer plugin. |
| OMX\_PTR | pParam | I | Parameter want to send to plug-in. |
| **Return value** | length | | Number of parameters want to set for Renderer plugin. | |
| 0 | | Do not set volume to plug-in. | |
| **Description** | * RENDERER\_SetConfig\_Send processing:   - Prepare volume rate value into command message. | | | |

[Covers: FD\_API\_RDR\_006]

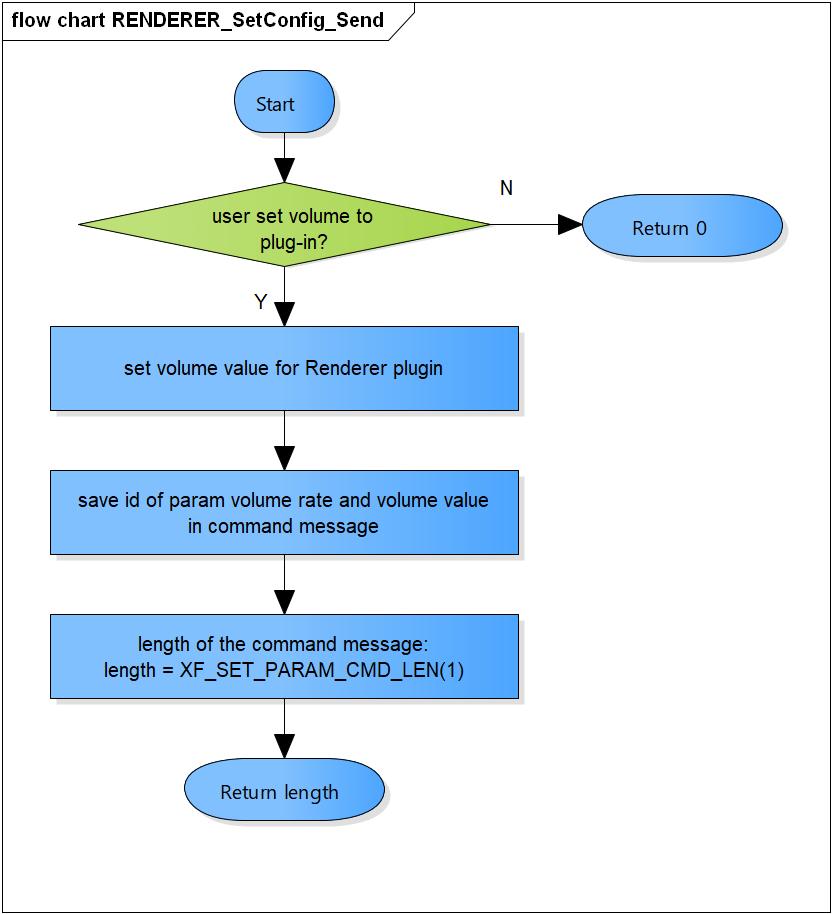


Figure 3‑12 RENDERER\_SetConfig\_Send flowchart

#### RENDERER\_SetConfigIndex

DD\_API\_RDR\_01\_013

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE RENDERER\_SetConfigIndex(OMX\_INDEXTYPE nIndex) | | | |
| **Function** | Check the index of parameter structure is correct. | | | |
| **Arguments** | Type | Name | I/O | Description |
| OMX\_INDEXTYPE | nIndex | I | Index of parameter structure.  Supported value:  XAOMX\_IndexParamAudioRenderer |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorUnsupportedIndex | | Wrong Index of structure. | |
| **Description** | * RENDERER\_SetConfig\_Send processing:   - Check the index of parameter structure is correct. | | | |

[Covers: FD\_API\_RDR\_006]

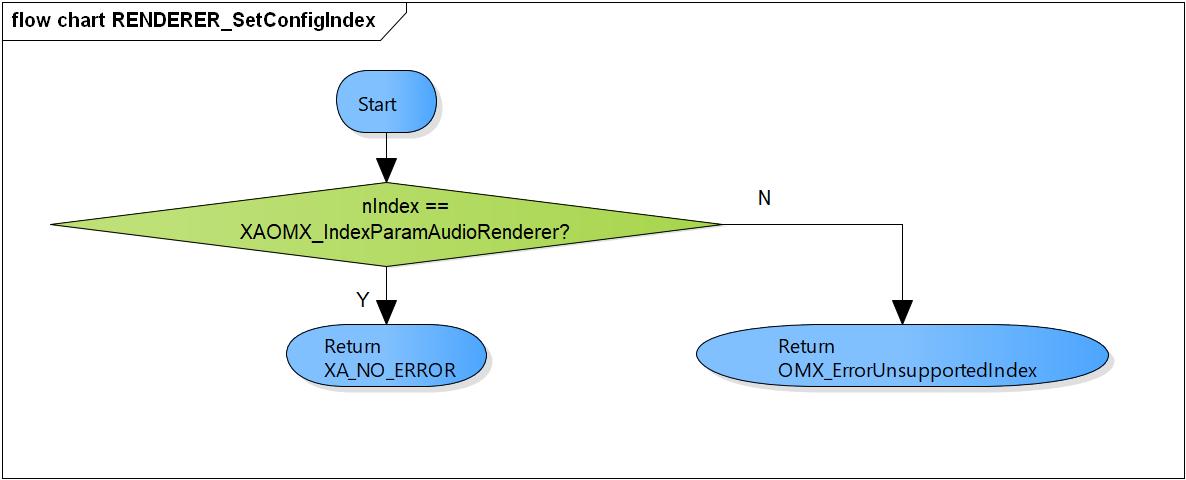


Figure 3‑13 RENDERER\_SetConfigIndex flowchart

#### RENDERER\_Setup

DD\_API\_RDR\_01\_014

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 RENDERER\_Setup(XAOMXCodecBase \*pBase, xf\_set\_param\_msg\_t \*msg) | | | |
| **Function** | Renderer prepare codec setup parameters. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Renderer component. |
| xf\_set\_param\_msg\_t\* | msg | O | Message send to Renderer plugin. |
| **Return value** | length | | Number of parameters want to set for Renderer plugin. | |
| **Description** | * Renderer\_Setup processing:   - Prepare PCM width, channels, sample rate, output1, 2; DMA channel 1, 2; out sample rate, volume rate, out channel and mix control values and ring-buffer number into command message. | | | |

[Covers: FD\_API\_RDR\_003]

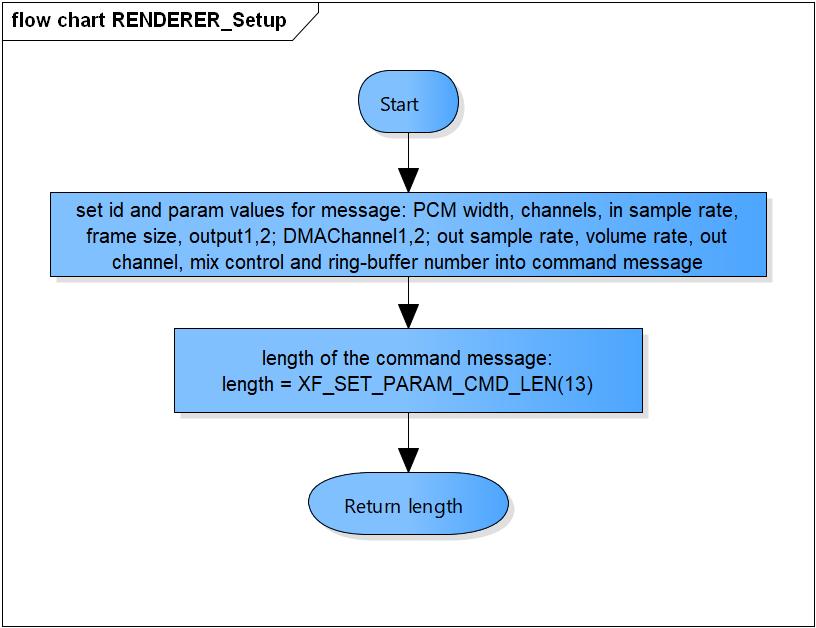


Figure 3‑14 RENDERER\_Setup flowchart

#### RENDERER\_RuntimeInit

DD\_API\_RDR\_01\_015

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 RENDERER\_RuntimeInit(XAOMXCodecBase \*pBase, xf\_start\_msg\_t \*msg) | | | |
| **Function** | This is a dummy function, not use in Renderer runtime initialization. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Renderer component. |
| xf\_start\_msg\_t\* | msg | O | Message send to Renderer plugin. |
| **Return value** | 0 | | No parameter need to set to plugin. | |
| **Description** | * RENDERER\_RuntimeInit processing:   - Return 0. | | | |

[Covers: FD\_API\_RDR\_003]

#### RENDERER\_GetParam

DD\_API\_RDR\_01\_016

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 RENDERER\_GetParam(XAOMXCodecBase \*pBase,  xf\_get\_param\_msg\_t \*msg,  u32 length) | | | |
| **Function** | This is a dummy function. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I/O | Pointer to base codec structure of RENDERER component. |
| xf\_get\_param\_msg\_t \* | msg | I | Message receive from RENDERER plugin. |
| u32 | lenghth | I | Length of message. |
| **Return value** | 0 | | Normal return. | |
| -EBADF | | The message length is invalid. | |
| **Description** | * RENDERER\_GetParam processing:   - Check the message length is correct and return 0. | | | |

[Covers: FD\_API\_RDR\_003]

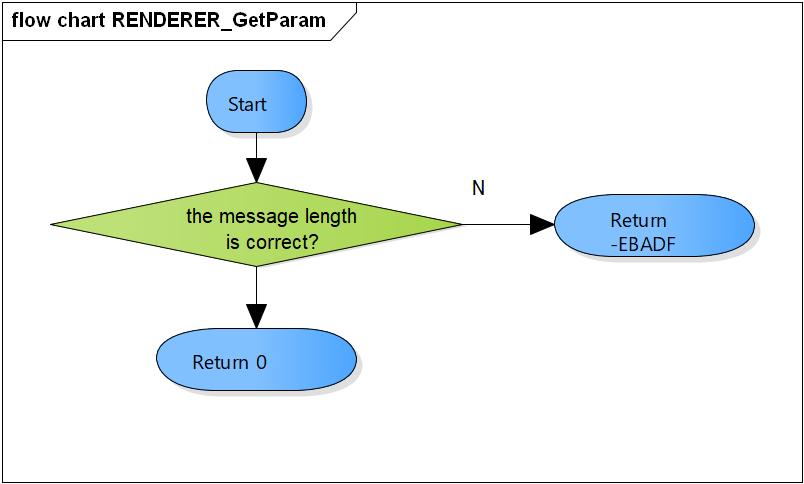


Figure 3‑15 RENDERER\_GetParam flowchart

#### RENDERER\_TimeStamp

DD\_API\_RDR\_01\_017

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static void RENDERER\_TimeStamp(XAOMXCodecBase \*pBase,  OMX\_BUFFERHEADERTYPE \*pBufHdr) | | | |
| **Function** | Calculate timestamp for current input buffer and next buffer. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of RENDERER component. |
| OMX\_BUFFERHEADERTYPE\* | msg | O | Input buffer. |
| **Return value** | None. | | | |
| **Description** | * RENDERER\_TimeStamp processing:   - Add current timestamp to the Input buffer.  - Calculate timestamp for the next buffer. | | | |

[Covers: FD\_API\_RDR\_003]

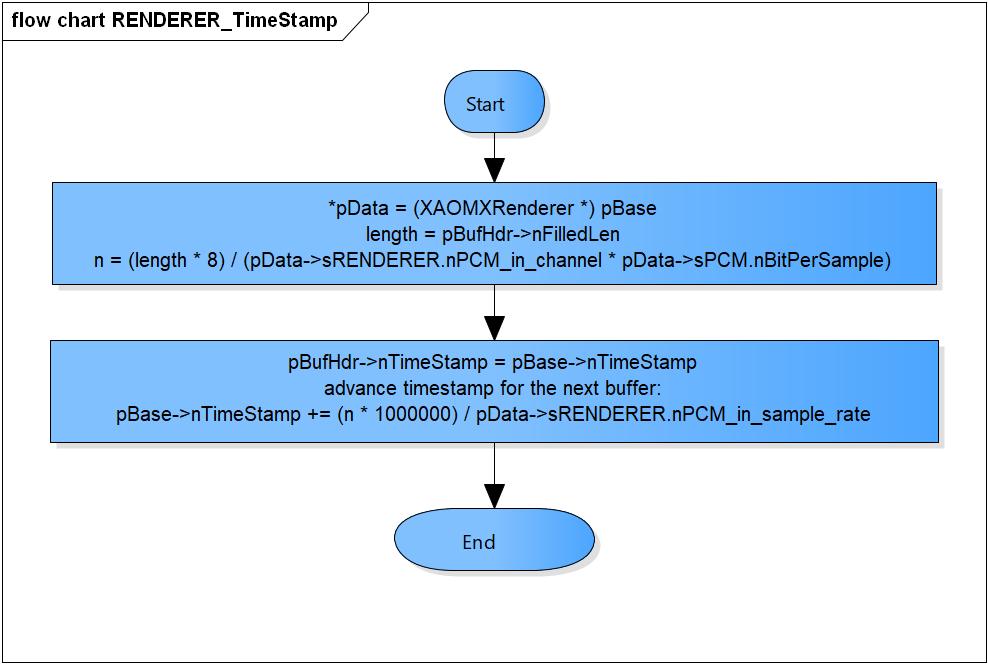


Figure 3‑16 RENDERER\_TimeStamp flowchart

#### RENDERER\_GetParameter

DD\_API\_RDR\_01\_018

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE RENDERER\_GetParameter(XAOMXCodecBase \*pBase,  OMX\_INDEXTYPE nIndex,  OMX\_PTR pParam) | | | |
| **Function** | Get parameter and save to pParam. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I | Pointer to base codec structure of Renderer component. |
| OMX\_INDEXTYPE | nIndex | I | Index of parameter structure.  Supported value:  OMX\_IndexParamAudioPcm  XAOMX\_IndexParamAudioRenderer  OMX\_IndexParamCompBufferSupplier |
| OMX\_PTR | pParam | O | Pointer to parameter structure want to save. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorUnsupportedIndex | | Wrong Index of structure. | |
| **Description** | * RENDERER\_GetParameter processing:   - Get parameters of OMX\_AUDIO\_PARAM\_PCMMODETYPE, XAOMX\_AUDIO\_PARAM\_RENDERER, OMX\_PARAM\_BUFFERSUPPLIERTYPE structure base on nIndex. | | | |

[Covers: FD\_API\_RDR\_004]

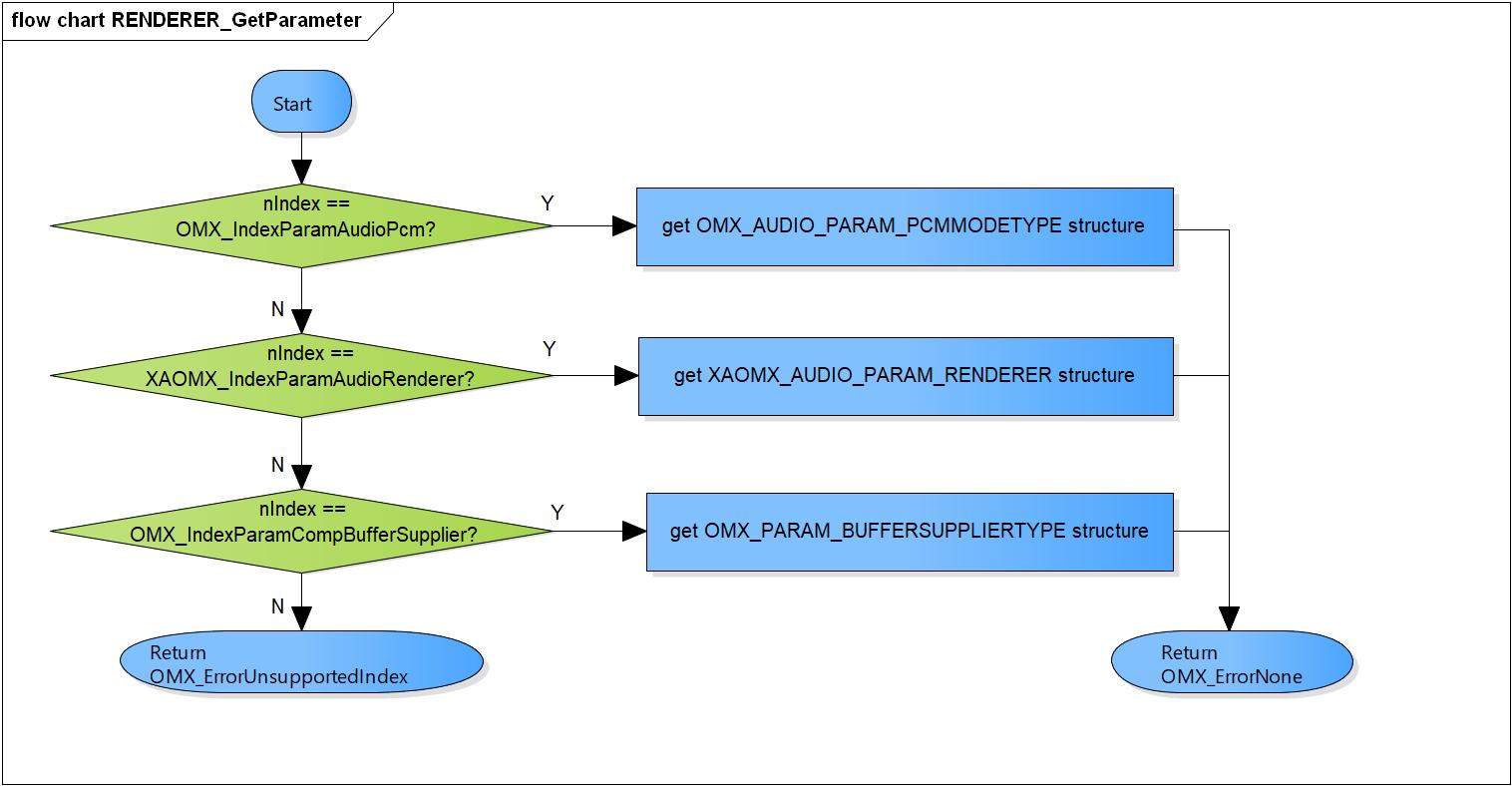


Figure 3‑17 RENDERER\_GetParameter flowchart

#### RENDERER\_SetParameter

DD\_API\_RDR\_01\_019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE RENDERER\_SetParameter(XAOMXCodecBase \*pBase,  OMX\_INDEXTYPE nIndex,  OMX\_PTR pParam) | | | |
| **Function** | Set parameter from pParam and save to parameter structure. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAOMXCodecBase \* | pBase | I/O | Pointer to base codec structure of Renderer component. |
| OMX\_INDEXTYPE | nIndex | I | Index of parameter structure.  Supported range:  OMX\_IndexParamAudioPcm  XAOMX\_IndexParamAudioRenderer  OMX\_IndexParamCompBufferSupplier |
| OMX\_PTR | pParam | I | Pointer to parameter structure want to set. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorUnsupportedIndex | | Wrong Index of structure | |
| OMX\_ErrorBadPortIndex | | Port index is invalid. | |
| **Description** | * RENDERER\_SetParameter processing:   - Set parameters of OMX\_AUDIO\_PARAM\_PCMMODETYPE, XAOMX\_AUDIO\_PARAM\_RENDERER, OMX\_PARAM\_BUFFERSUPPLIERTYPE structure base on nIndex. | | | |

[Covers: FD\_API\_RDR\_005]

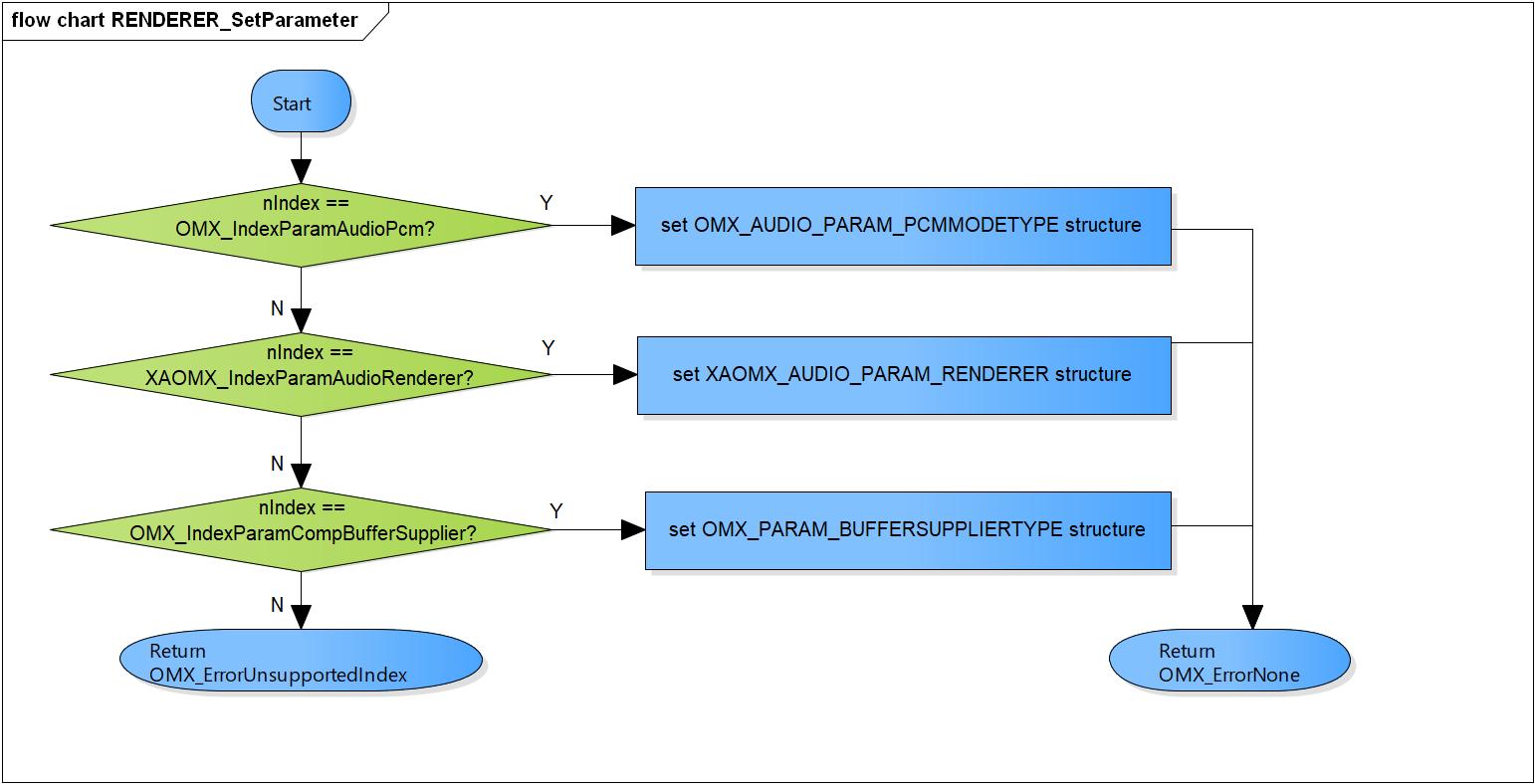


Figure 3‑18 RENDERER\_SetParameter flowchart

#### RENDERER\_ComponentInit

DD\_API\_RDR\_01\_020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE RENDERER\_ComponentInit(xf\_proxy\_t \*proxy,  OMX\_HANDLETYPE hComponent,  OMX\_PTR pAppData,  OMX\_CALLBACKTYPE \*pCallbacks) | | | |
| **Function** | Initialize base codec interface, parameters, PCM and Renderer format. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_proxy\_t \* | proxy | O | OpenMAX interface proxy. |
| OMX\_HANDLETYPE | hComponent | O | Pointer to Renderer component handle. |
| OMX\_PTR | pAppData | I | Pointer to pAppData of Renderer component. |
| OMX\_CALLBACKTYPE | pCallbacks | I | Pointer to Callbacks of Renderer component. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorInsufficientResources | | Initialize base codec interface is fail. | |
| **Description** | * RENDERER\_ComponentInit processing:   - Initialize base codec interface, parameters, PCM and Renderer format. | | | |

[Covers: FD\_API\_RDR\_001]

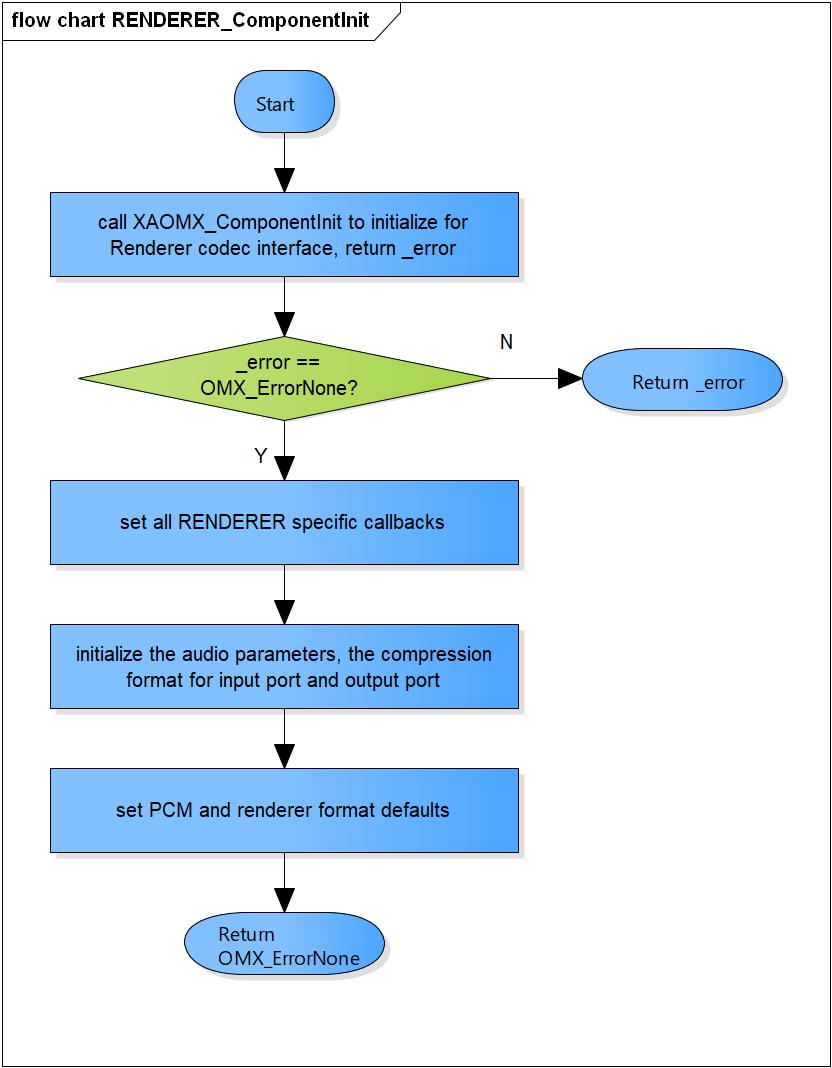


Figure 3‑19 RENDERER\_ComponentInit flowchart

#### RENDERER\_ComponentDeInit

DD\_API\_RDR\_01\_021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static OMX\_ERRORTYPE RENDERER\_ComponentDeInit(OMX\_HANDLETYPE hComponent) | | | |
| **Function** | Destroy Renderer component and private data. | | | |
| **Arguments** | Type | Name | I/O | Description |
| OMX\_HANDLETYPE | hComponent | I | Pointer to Renderer component handle. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorBadParameter | | Private data is NULL | |
| **Description** | * RENDERER\_ComponentDeInit processing:   - Destroy Renderer component and private data. | | | |

[Covers: FD\_API\_RDR\_002]

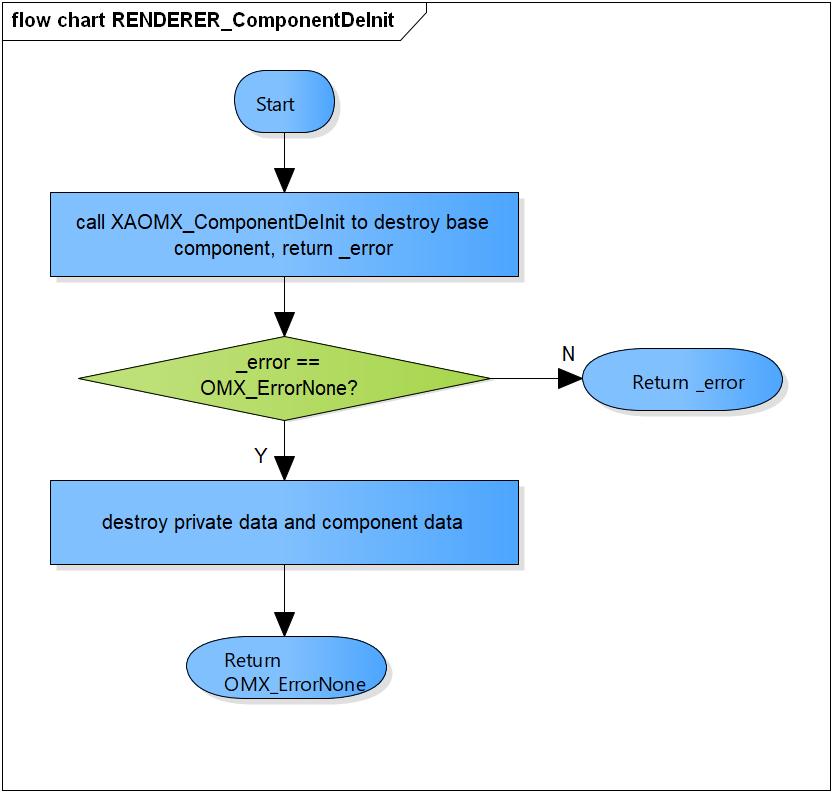


Figure 3‑20 RENDERER\_ComponentDeInit flowchart

#### RENDERER\_ComponentCreate

DD\_API\_RDR\_01\_022

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | OMX\_ERRORTYPE RENDERER\_ComponentCreate(xf\_proxy\_t \*proxy, OMX\_HANDLETYPE \*hComponent, OMX\_PTR pAppData, OMX\_CALLBACKTYPE \*pCallbacks) | | | |
| **Function** | Create base component and initialize the Renderer component. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_proxy\_t\* | proxy | O | OpenMAX interface proxy. |
| OMX\_HANDLETYPE\* | hComponent | O | Pointer to Renderer component handle. |
| OMX\_PTR | pAppData | I | Pointer to pAppData of Renderer component. |
| OMX\_CALLBACKTYPE \* | pCallbacks | I | Pointer to Callbacks of Renderer component. |
| **Return value** | OMX\_ErrorNone | | Normal return. | |
| OMX\_ErrorInsufficientResources | | Cannot create the base component.  Cannot allocate private memory.  Initialize the Renderer component is fail. | |
| **Description** | * RENDERER\_ComponentCreate processing:   - Create the base component.  - Allocate the private memory.  - Initialize the Renderer component. | | | |

[Covers: FD\_API\_RDR\_001]

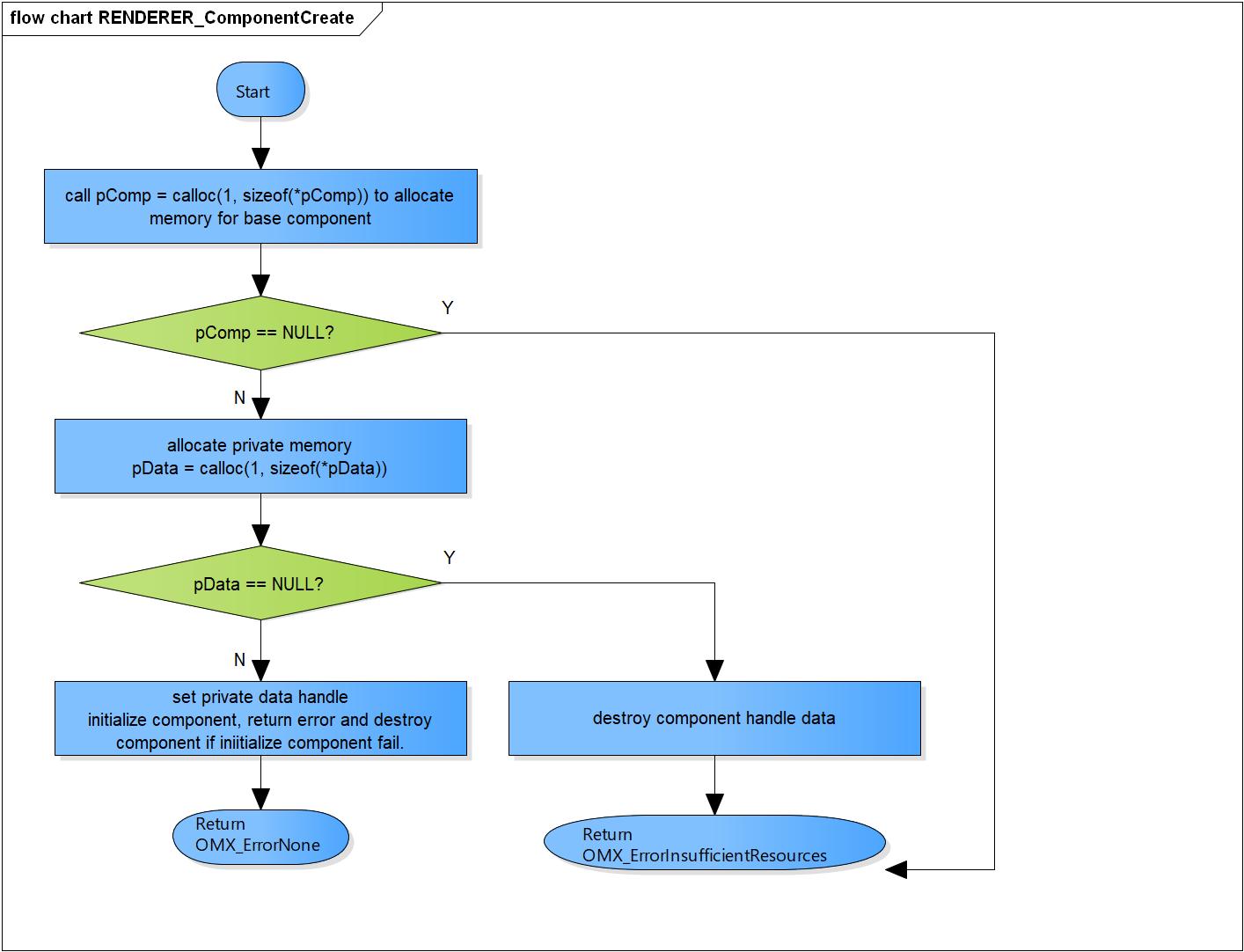


Figure 3‑21 RENDERER\_ComponentCreate flowchart

# Revision history

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Date** | **Page** | **Content** | **Approved** | **Changed** |
| 1.0.0 | Nov 14 2018 | - | First Edition issued | Vu Phan | Vu Phan |
| 1.1.0 | Dec 10 2018 | - | Add traceability ID | Vu Phan | Ngu Pham |
| 1.2.0 | Jan 03, 2019 | - | Add range for input parameters | Vu Phan | Tien Tran |
| 1.3.0 | Apr 16, 2019 | 8, 20 | Update for Android |  |  |