**Document Type: Detail Design**

**Document Name:**

**ADSP FRAMEWORK: EQUALIZER CLASS**

Renesas Design Vietnam Co. Ltd

R-Car Software Solution 2 Group

Middleware 1 Team

Renesas Electronics Corporation

Table of Contents

[1. Overview 3](#_Toc530052950)

[2. Function list 4](#_Toc530052951)

[3. Detail information 5](#_Toc530052952)

[3.1 Data type and Macro definition 5](#_Toc530052953)

[3.2 Globle variable 6](#_Toc530052954)

[3.3 Function definition 7](#_Toc530052955)

[3.3.1 xa\_equalizer\_prepare\_runtime 7](#_Toc530052956)

[3.3.2 xa\_equalizer\_empty\_this\_buffer 9](#_Toc530052957)

[3.3.3 xa\_equalizer\_fill\_this\_buffer 11](#_Toc530052958)

[3.3.4 xa\_equalizer\_memtab 13](#_Toc530052959)

[3.3.5 xa\_equalizer\_port\_route 14](#_Toc530052960)

[3.3.6 xa\_equalizer\_port\_unroute 16](#_Toc530052961)

[3.3.7 xa\_equalizer\_preprocess 17](#_Toc530052962)

[3.3.8 xa\_equalizer\_postprocess 19](#_Toc530052963)

[3.3.9 xa\_equalizer\_flush 21](#_Toc530052964)

[3.3.10 xa\_equalizer\_cleanup 23](#_Toc530052965)

[3.3.11 xa\_equalizer\_terminate 24](#_Toc530052966)

[3.3.12 xa\_equalizer\_destroy 25](#_Toc530052967)

[3.3.13 xa\_equalizer\_factory 26](#_Toc530052968)

[4. Revision history 27](#_Toc530052969)

List of Figures

[Figure 1‑1 The software architecture 3](#_Toc530052970)

[Figure 3‑1 xa\_equalizer\_prepare\_runtime flowchart 8](#_Toc530052971)

[Figure 3‑2 xa\_equalizer\_empty\_this\_buffer flowchart 10](#_Toc530052972)

[Figure 3‑3 xa\_equalizer\_fill\_this\_buffer flowchart 12](#_Toc530052973)

[Figure 3‑4 xa\_equalizer\_memtab flowchart 13](#_Toc530052974)

[Figure 3‑5 xa\_equalizer\_port\_route flowchart 15](#_Toc530052975)

[Figure 3‑6 xa\_equalizer\_port\_unroute flowchart 16](#_Toc530052976)

[Figure 3‑7 xa\_equalizer\_preprocess flowchart 18](#_Toc530052977)

[Figure 3‑8 xa\_equalizer\_postprocess flowchart 20](#_Toc530052978)

[Figure 3‑9 xa\_equalizer\_flush flowchart 22](#_Toc530052979)

[Figure 3‑10 xa\_equalizer\_cleanup flowchart 23](#_Toc530052980)

[Figure 3‑11 xa\_equalizer\_terminate flowchart 24](#_Toc530052981)

[Figure 3‑12 xa\_equalizer\_destroy flowchart 25](#_Toc530052982)

[Figure 3‑13 xa\_equalizer\_factory flowchart 26](#_Toc530052983)

List of Table

[Table 2‑1 Function list 4](#_Toc530044254)

[Table 3‑1 XAEqualizer type structure information 5](#_Toc530044255)

[Table 3‑2 Macro definitions 5](#_Toc530044256)

[Table 3‑3 Globle variable xa\_equalizer\_cmd 6](#_Toc530044257)

# Overview

This section provides an overview of the software architecture.

User Application

ADSP Interface

Kernel Space

User Space

ADSP Driver

ARM

Audio HW

**ADSP Framework**

TDM class

Equalizer class

Capture class

Renderer class

ADSP

Plugin

Equalizer Plugin\*

TDM Plugin

Capture Plugin

Renderer Plugin

This document’s target is in side of red square

\* not connect to SCU/SSI ADMA block

SCU/ SSI/

ADMA

DAC/

ADC

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

# Function list

The following is list of functions:

Table 2‑1 Function list

|  |  |  |
| --- | --- | --- |
| **Type** | **Function Name** | **Outline** |
| API | xa\_equalizer\_factory | This API is to construct generic audio component and set functions pointer for internal functions. |
| Internal functions | xa\_equalizer\_prepare\_runtime | This function is to prepare equalizer for steady operation. |
| xa\_equalizer\_empty\_this\_buffer | This function is to put message into input port queue. |
| xa\_equalizer\_fill\_this\_buffer | This function is to place message into output port. |
| xa\_equalizer\_flush | This function is to purge input port queue. |
| xa\_equalizer\_memtab | This function is to initialize output/input port. |
| xa\_equalizer\_port\_route | This function is to route output port - allocate buffers. |
| xa\_equalizer\_port\_unroute | This function is to unroute output port and destroy all buffers allocated. |
| xa\_equalizer\_preprocess | This function is to prepare data for input/output buffer. |
| xa\_equalizer\_postprocess | This function is to consume bytes from input buffer and reschedule processing. |
| xa\_equalizer\_terminate | This is equalizer termination-state command processor function. |
| xa\_equalizer\_destroy | This function is to destroy component. |
| xa\_equalizer\_cleanup | This function is to purge input port queue and cancel internal scheduling. |

# Detail information

This section describes detail information of data types, macro definitions, implemented APIs and internal function units, globle variable.

## Data type and Macro definition

The XAEqualizer type structure is the work area used by the equalizer class. When using this class, secure the area with the application program. It’s not necessary to refer to this area because it only contains the internal variables and working buffers of the class. Make sure not to change the value of this area with the application program.

Table 3‑1 XAEqualizer type structure information

|  |  |
| --- | --- |
| Member name | Outline |
| XACodecBase base | Generic audio codec data |
| xf\_input\_port\_t input | Input port queue |
| u32 in\_idx | The index of input |
| xf\_output\_port\_t output | Output port queue |
| u32 out\_idx | The index of output |
| u32 factor | Time conversion factor (input byte "duration" in timebase units) |
| u32 sample\_size | Sample size in bytes |
| u32 produced | Total number of produced audio frames since last reset |

Table 3‑2 Macro definitions

|  |  |  |
| --- | --- | --- |
| Macro | Value | Outline |
| XA\_EQZ\_FLAG\_INPUT\_SETUP | \_\_XA\_BASE\_FLAG(1 << 0) | Input port setup condition |
| XA\_EQZ\_FLAG\_OUTPUT\_SETUP | \_\_XA\_BASE\_FLAG(1 << 1) | Output port setup condition |

Note: Macro \_\_XA\_BASE\_FLAG(f) ((f) << 6) is in xa-class-base.h file.

## Globle variable

Table 3‑3 Globle variable xa\_equalizer\_cmd

|  |  |
| --- | --- |
| static XA\_ERRORCODE (\* const xa\_equalizer\_cmd[])(XACodecBase \*base, xf\_message\_t \*m) | |
| Description: variable stores function pointers arcoding to opcode index to run in runtime operation. | |
| Array index | Value (function pointer) |
| XF\_OPCODE\_TYPE(XF\_SET\_PARAM) | xa\_base\_set\_param |
| XF\_OPCODE\_TYPE(XF\_GET\_PARAM) | xa\_base\_get\_param |
| XF\_OPCODE\_TYPE(XF\_ROUTE) | xa\_equalizer\_port\_route |
| XF\_OPCODE\_TYPE(XF\_UNROUTE) | xa\_equalizer\_port\_unroute |
| XF\_OPCODE\_TYPE(XF\_EMPTY\_THIS\_BUFFER) | xa\_equalizer\_empty\_this\_buffer |
| XF\_OPCODE\_TYPE(XF\_FILL\_THIS\_BUFFER) | xa\_equalizer\_fill\_this\_buffer |
| XF\_OPCODE\_TYPE(XF\_FLUSH) | xa\_equalizer\_flush |

Note: Marco XF\_OPCODE\_TYPE(opcode) : ((opcode) & (0x3F)) in xf-opcode.h file.

## Function definition

### xa\_equalizer\_prepare\_runtime

DD\_FWK\_EQZ\_01\_001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static inline XA\_ERRORCODE xa\_equalizer\_prepare\_runtime(XAEqualizer \*equalizer) | | | |
| **Function** | This function is to prepare equalizer for steady operation. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XAEqualizer | Equalizer | I/O | Pointer to codec instance structure (struct XARenderer) |
| **Return value** | XA\_API\_FATAL\_MEM\_ALLOC | | API structure is NULL (error from plugin). | |
| XA\_API\_FATAL\_MEM\_ALIGN | | API structure is not aligned to 4 bytes (error from plugin). | |
| XA\_EQZ\_CONFIG\_FATAL\_STATE | | Pre-initialization is not completed yet (error from plugin). | |
| XA\_API\_FATAL\_INVALID\_CMD\_TYPE | | Sample rate is invalid. | |
| XA\_NO\_ERROR | | Nomally end. | |
| **Description** | * xa\_euqualizer\_prepare\_runtime command processing:   - Get config parameters: sample rate, pcm width, channels, size of input/output buffer.  - Set equalizer timestamp factor | | | |

[Covers: FD\_FWK\_CMN\_002]

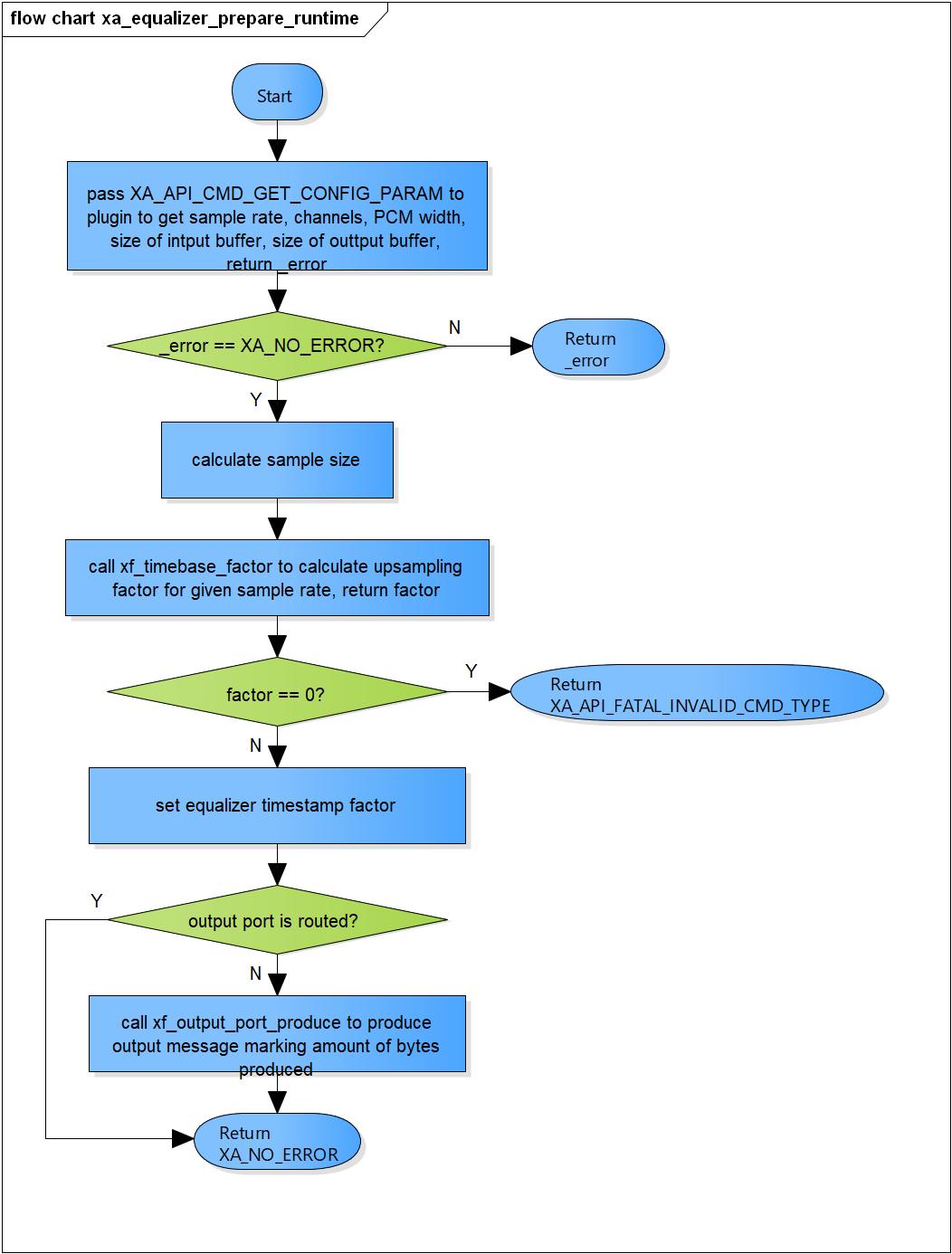


Figure 3‑1 xa\_equalizer\_prepare\_runtime flowchart

### xa\_equalizer\_empty\_this\_buffer

DD\_FWK\_EQZ\_01\_002

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_empty\_this\_buffer(XACodecBase \*base,  xf\_message\_t \*m); | | | |
| **Function** | This function is to put message into input port queue. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure (struct XACodecBase). |
| xf\_message\_t | m | I | Pointer to audio message (struct xf\_message). |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_INVALID\_CMD | | Port is not a input port.  Post initialization state is not completed yet.  State of equalizer is not runtime initialization or execution. | |
| **Description** | * xa\_equalizer\_empty\_this\_buffer command processing:   - Put message into input queue. | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_010]

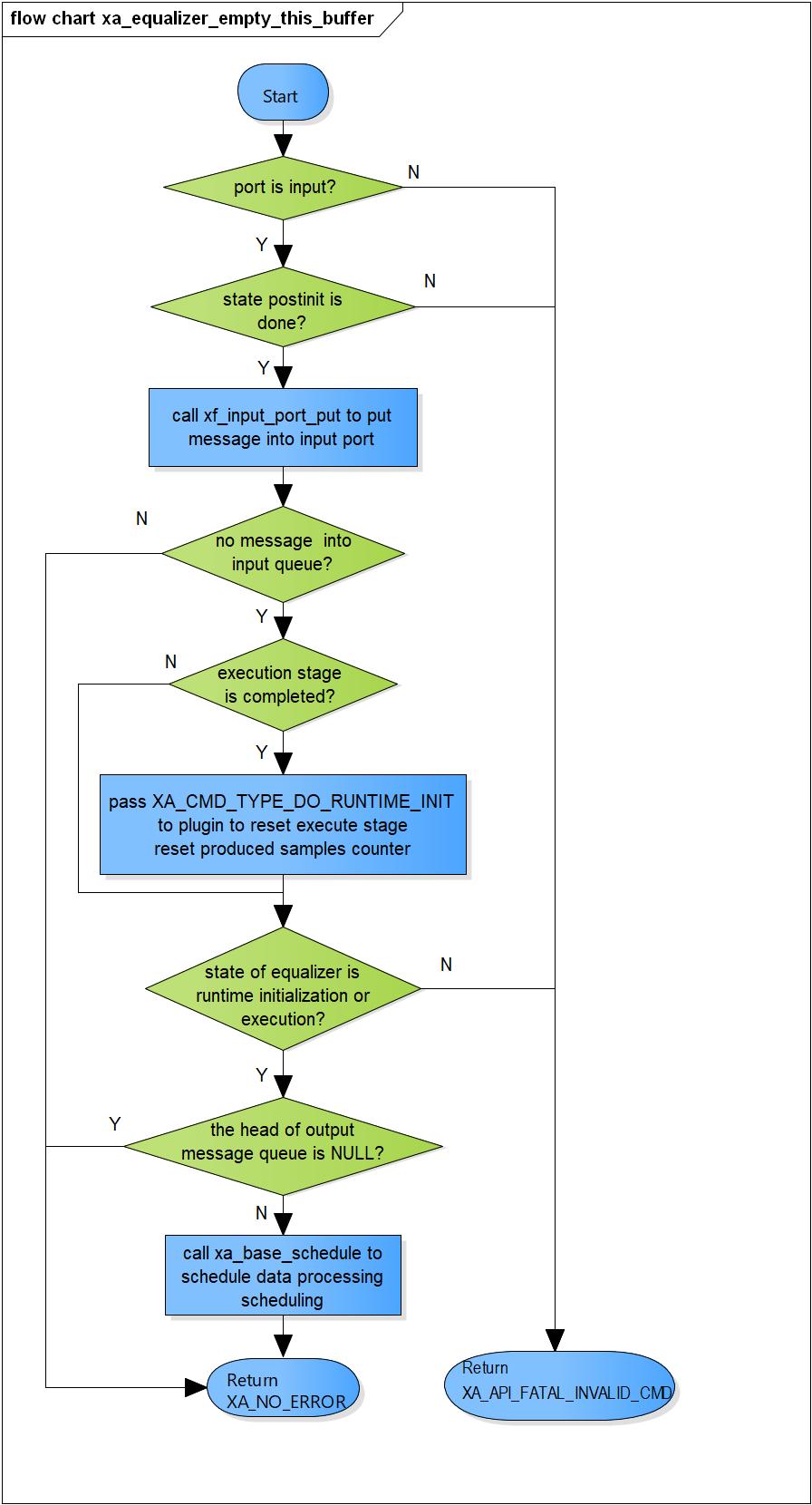


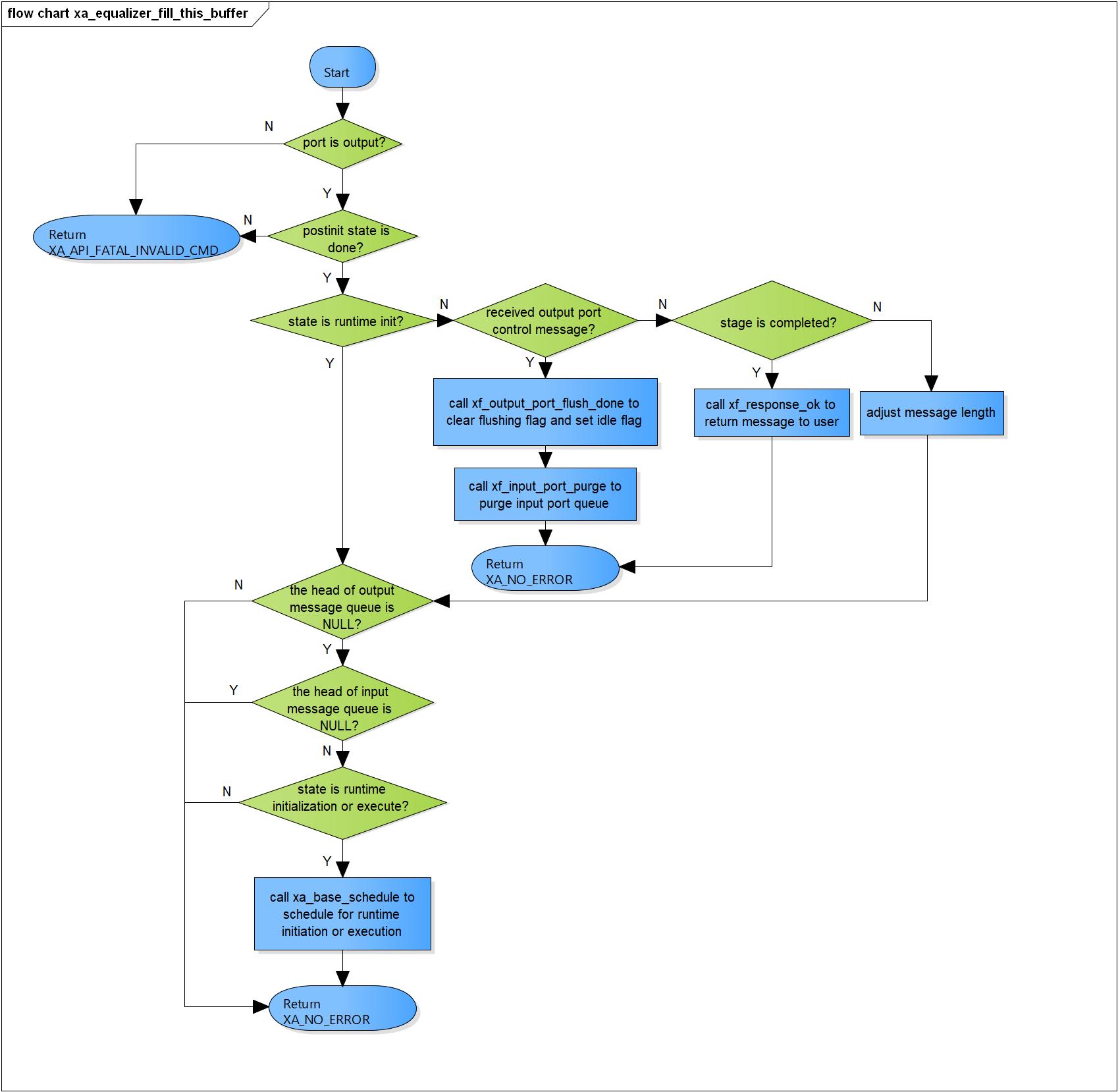
Figure 3‑2 xa\_equalizer\_empty\_this\_buffer flowchart

### xa\_equalizer\_fill\_this\_buffer

DD\_FWK\_EQZ\_01\_003

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_fill\_this\_buffer(XACodecBase \*base,  xf\_message\_t \*m); | | | |
| **Function** | This function is to place message into output port. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure (struct XACodecBase). |
| xf\_message\_t | m | I/O | Pointer to audio message (struct xf\_message). |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_INVALID\_CMD | | Port is not an output port.  Post initialization state is not completed yet. | |
| **Description** | * xa\_equalizer\_fill\_this\_buffer command processing:   - Adjust message length.  - Place message into output port. | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_011]

Figure 3‑3 xa\_equalizer\_fill\_this\_buffer flowchart

### xa\_equalizer\_memtab

DD\_FWK\_EQZ\_01\_004

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_memtab(XACodecBase \*base, WORD32 idx, WORD32 type, WORD32 size, WORD32 align, u32 core) | | | |
| **Function** | This function is to initialize input/output port. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure (struct XACodecBase). |
| WORD32 | idx | I | Port index. |
| WORD32 | type | I | Port type (input or output port)  Valid values:  XA\_MEMTYPE\_INPUT  XA\_MEMTYPE\_OUTPUT |
| WORD32 | size | I | Buffer size of port. |
| WORD32 | align | I | Port alignment. |
| u32 | core | I | Core index of ADSP framework. |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_INVALID\_CMD\_TYPE | | Port is invalid. | |
| XA\_API\_FATAL\_MEM\_ALLOC | | Initialize input/ouput port is fail | |
| **Description** | * Memtab command processing:   - Initialize Input/Output port specification and allocate internal buffer. | | | |

[Covers: FD\_FWK\_CMN\_008, FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_001]

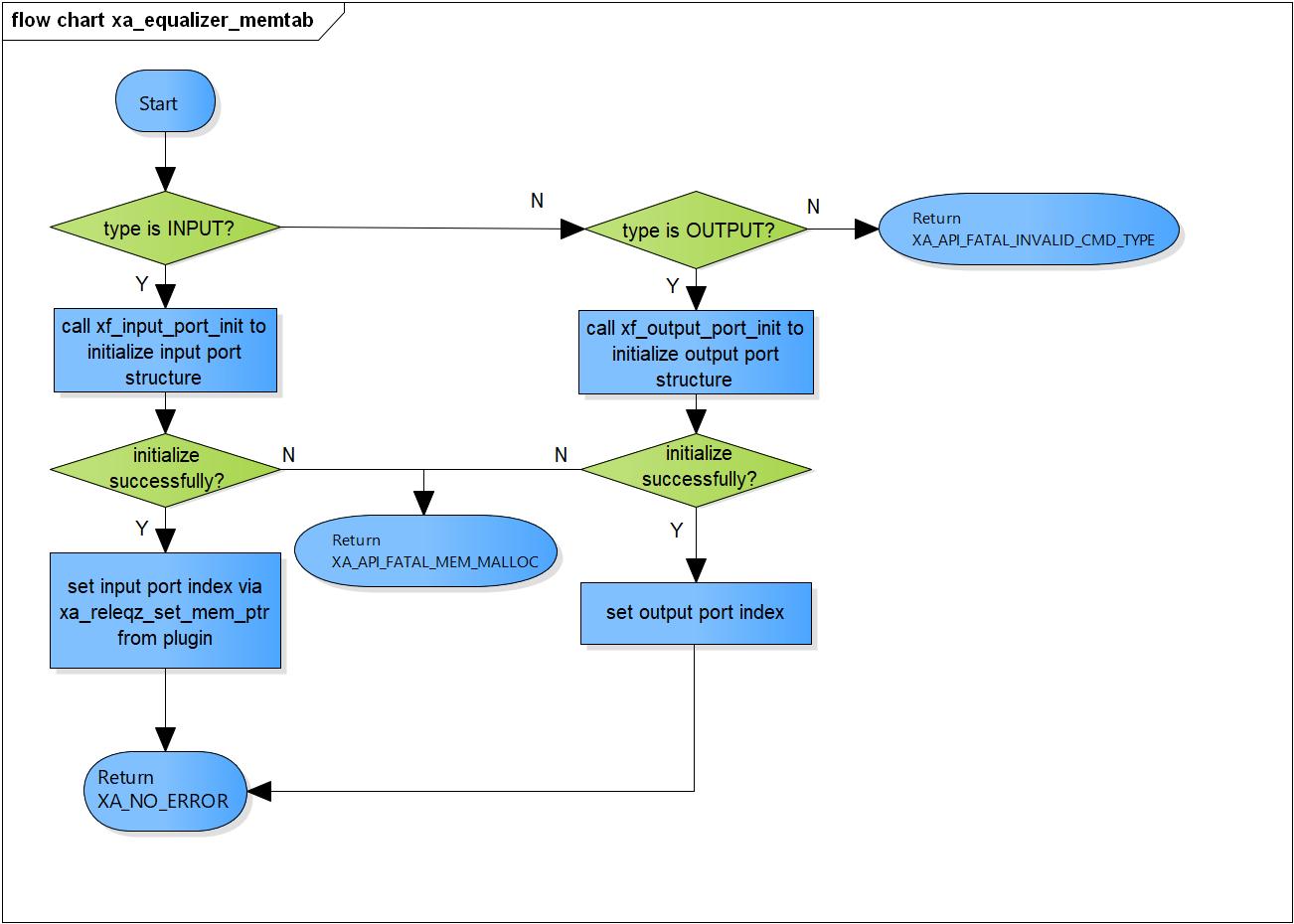


Figure 3‑4 xa\_equalizer\_memtab flowchart

### xa\_equalizer\_port\_route

DD\_FWK\_EQZ\_01\_005

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_port\_route(XACodecBase \*base,  xf\_message\_t \*m) | | | |
| **Function** | This function is to route output port - allocate memory buffers. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I | Pointer to codec instance structure (struct XACodecBase). |
| xf\_message\_t | m | I | Pointer to audio message (struct xf\_message). |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_INVALID\_CMD\_TYPE | | Ports have been routed before yet.  Port is not an output. | |
| XA\_API\_FATAL\_INVALID\_CMD | | Post initialization state is not completed yet. | |
|  | XA\_API\_FATAL\_MEM\_ALLOC | | Route output port is failed. | |
| **Description** | * Output port routing command processing:   - Make sure port is not routed yet.  - Route output port - allocate buffers.  - Schedule processing. | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_006]

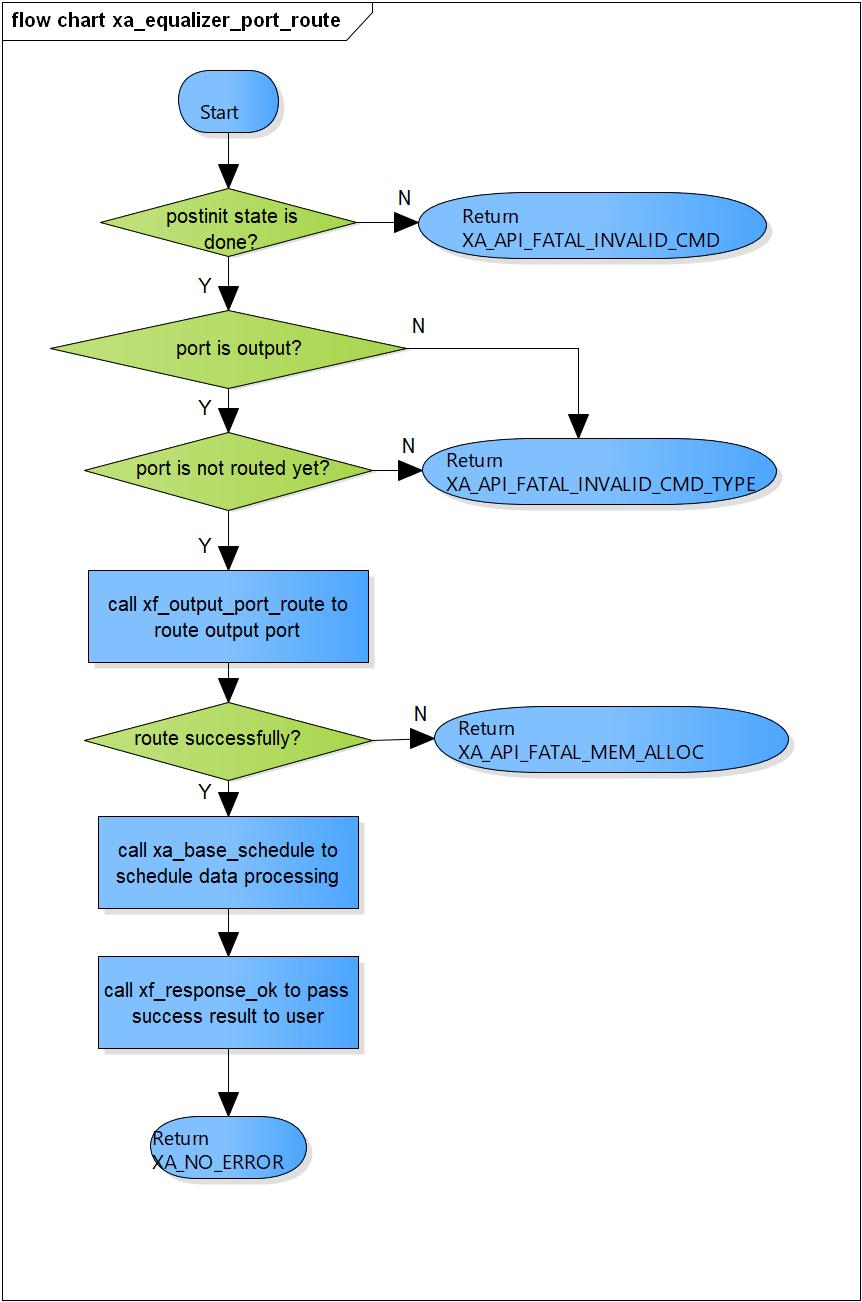


Figure 3‑5 xa\_equalizer\_port\_route flowchart

### xa\_equalizer\_port\_unroute

DD\_FWK\_EQZ\_01\_006

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_port\_unroute(XACodecBase \*base, xf\_message\_t \*m) | | | |
| **Function** | This function is to unroute output port and destroy all memory buffers allocated. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure  (struct XACodecBase). |
| xf\_message\_t | m | I | Pointer to audio message (struct xf\_message). |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_INVALID\_CMD\_TYPE | | Port is not an output. | |
| XA\_API\_FATAL\_INVALID\_CMD | | Post initialization state is not completed yet. | |
| **Description** | * Port unroute command command processing:   - Cancel any pending processing  - Clear output-port-setup condition  - Unroute output port | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_007]

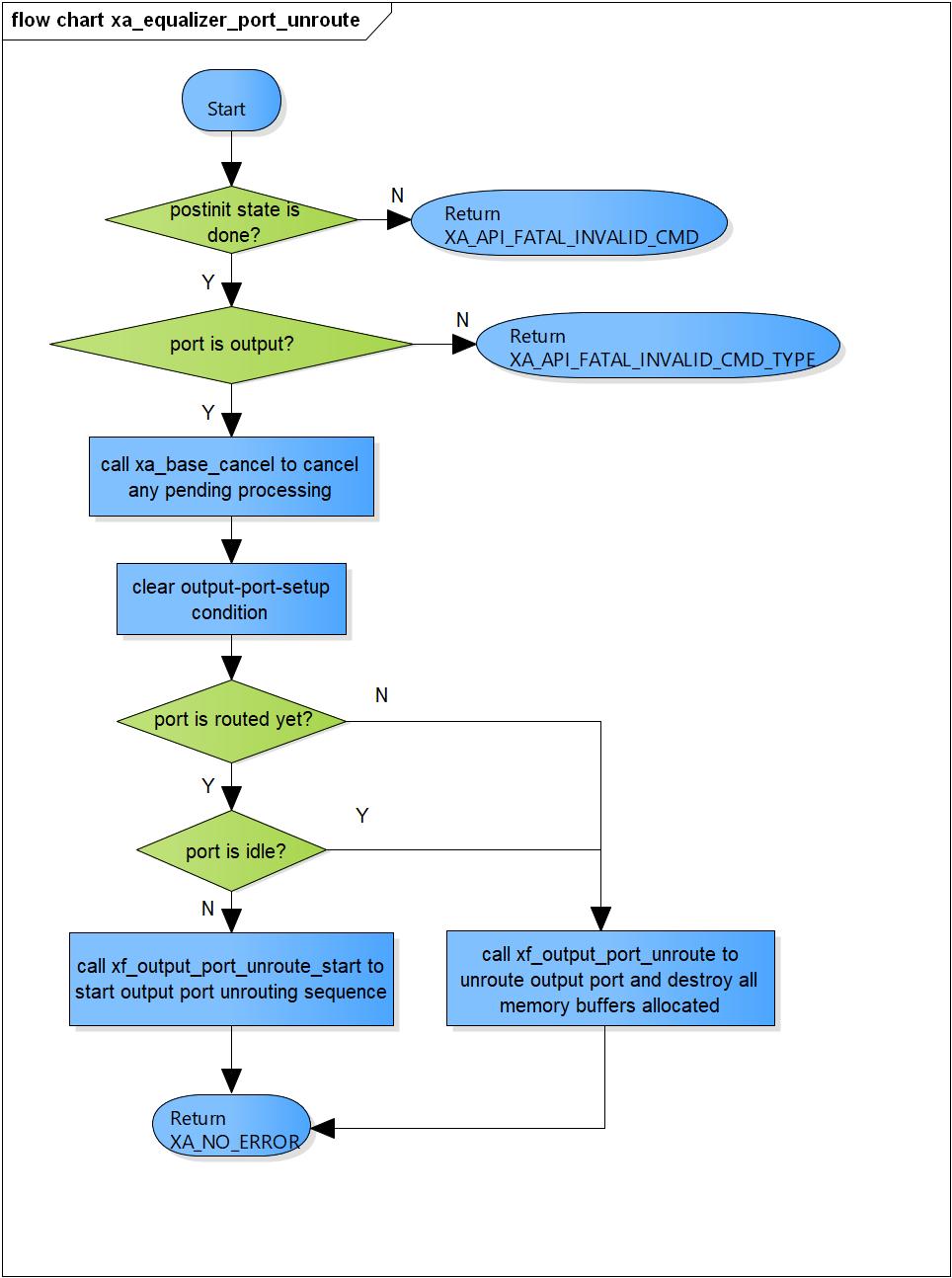


Figure 3‑6 xa\_equalizer\_port\_unroute flowchart

### xa\_equalizer\_preprocess

DD\_FWK\_EQZ\_01\_007

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_preprocess(XACodecBase \*base) | | | |
| **Function** | This function is to prepare data for input/output buffer. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure (struct XACodecBase). |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_CODEC\_EXEC\_NO\_DATA | | No output buffer available | |
| XA\_API\_FATAL\_MEM\_ALLOC | | API structure or output buffer is NULL (error from plugin). | |
| XA\_API\_FATAL\_MEM\_ALIGN | | API structure or output buffer is not aligned 4 bytes (error from plugin). | |
| XA\_API\_FATAL\_INVALID\_CMD\_TYPE | | The index of output is wrong (error from plugin). | |
| XA\_EQZ\_CONFIG\_FATAL\_STATE | | Post initialization state is not completed yet (error from plugin). | |
| **Description** | * Preprocess command processing:   - Prepare output buffer  - Set output buffer pointer  - Mark output port is setup  - Prepare input buffer  - Check if input stream is over | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_002, FD\_FWK\_CMN\_003]

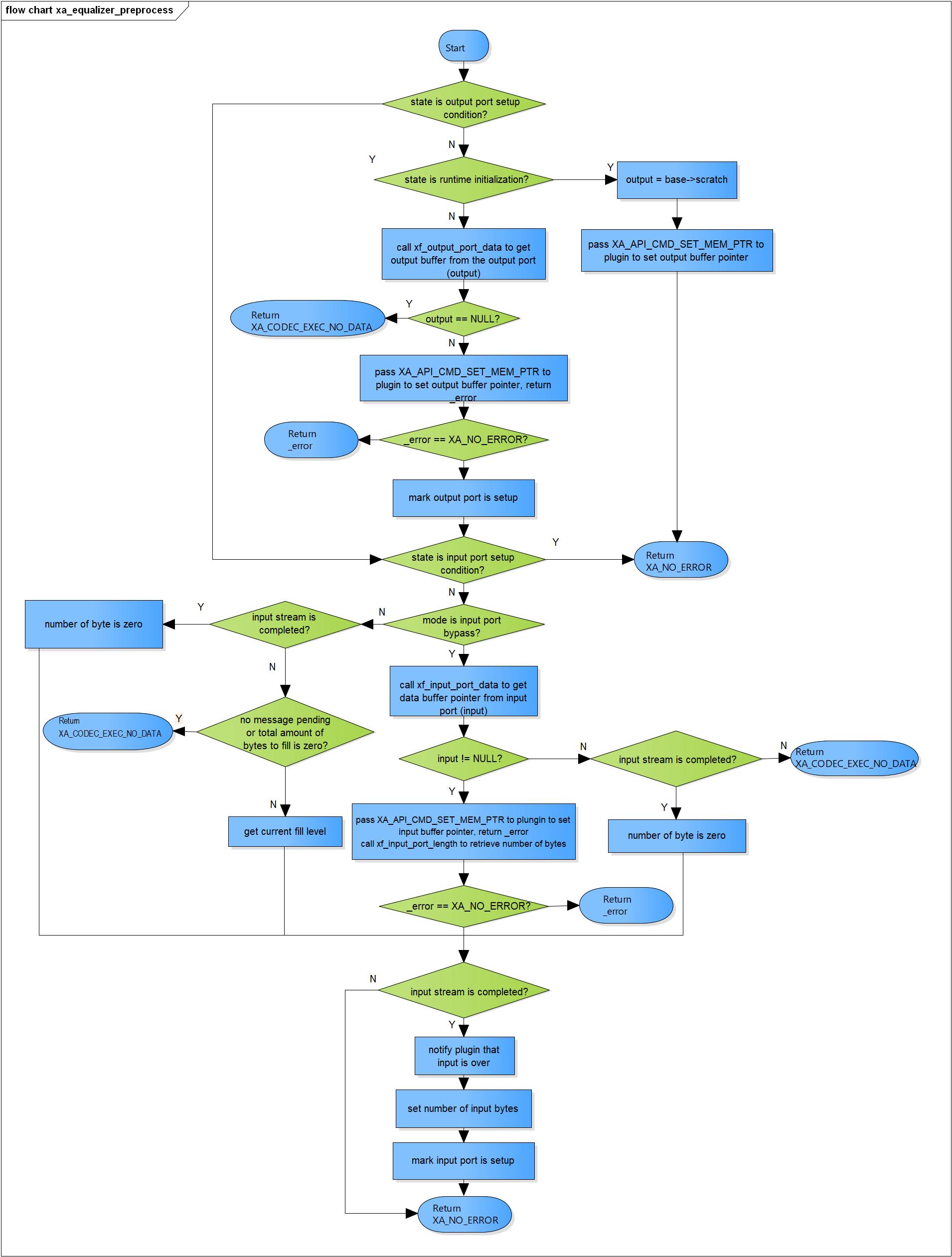


Figure 3‑7 xa\_equalizer\_preprocess flowchart

### xa\_equalizer\_postprocess

DD\_FWK\_EQZ\_01\_008

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_postprocess(XACodecBase \*base,  WORD32 i\_done) | | | |
| **Function** | This function is to consume bytes from input buffer and reschedule processing. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure (struct XACodecBase). |
| WORD32 | i\_done | I | State of processing of plugin |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_MEM\_ALLOC | | API structure is NULL (error from plugin). | |
| XA\_API\_FATAL\_MEM\_ALIGN | | API structure is not aligned 4 bytes (error from plugin). | |
| **Description** | * Postprocess command processing:   - Get number of produced samples from plugin, consume specified number of bytes from input port.  - Check state if process is done  - Get number of output bytes  - Reschedule execution if we have pending output message | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_002, FD\_FWK\_CMN\_003]

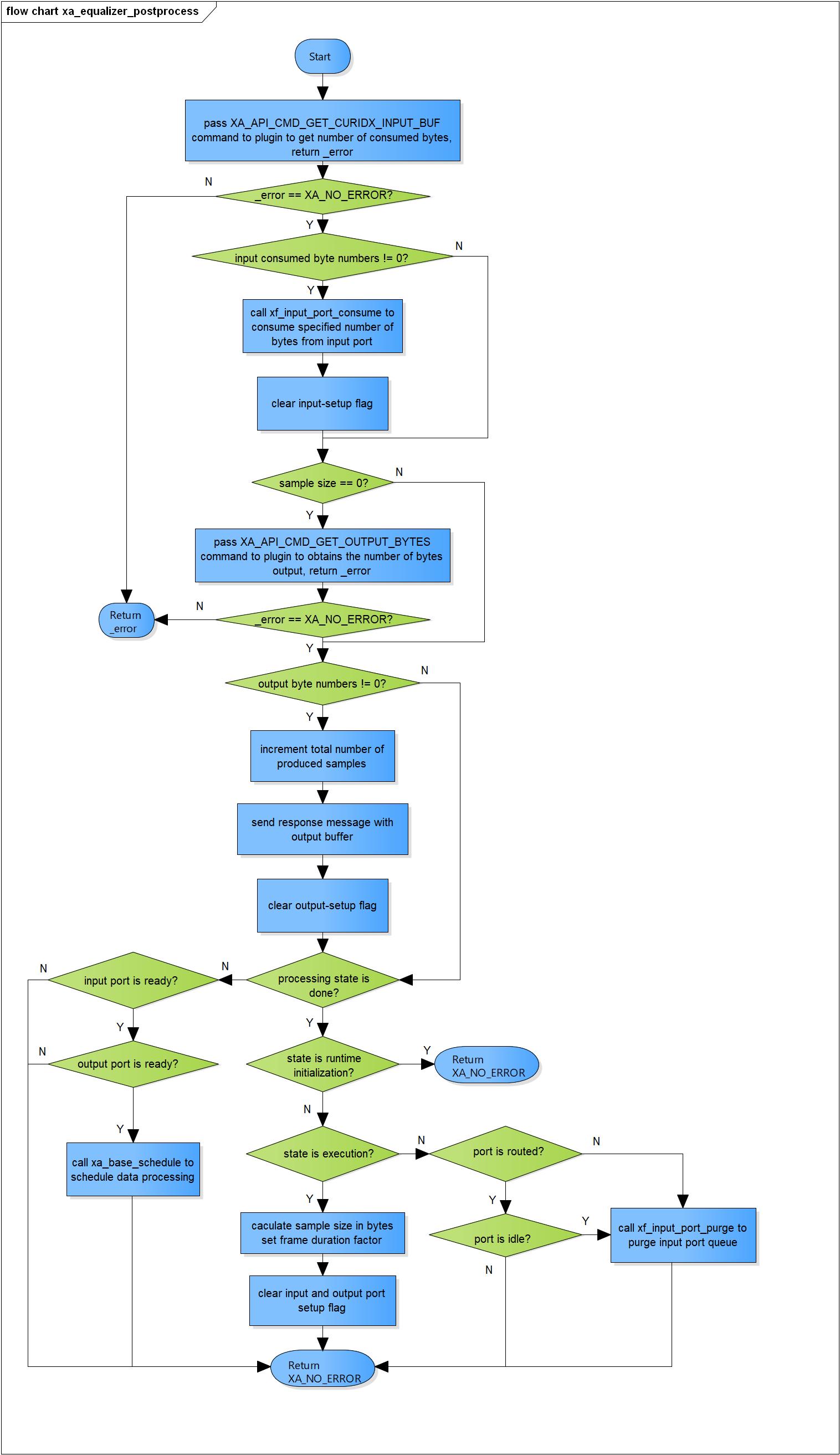


Figure 3‑8 xa\_equalizer\_postprocess flowchart

### xa\_equalizer\_flush

DD\_FWK\_EQZ\_01\_009

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static XA\_ERRORCODE xa\_equalizer\_flush(XACodecBase \*base, xf\_message\_t \*m) | | | |
| **Function** | This function is to purge input port queue. | | | |
| **Arguments** | Type | Name | I/O | Description |
| XACodecBase | base | I/O | Pointer to codec instance structure (struct XACodecBase). |
| xf\_message\_t | m | I | Pointer to audio message (struct xf\_message). |
| **Return value** | XA\_NO\_ERROR | | Nomally end. | |
| XA\_API\_FATAL\_INVALID\_CMD | | Post initialization state is not completed yet | |
| XA\_API\_FATAL\_INVALID\_CMD\_TYPE | | Message length is zero. | |
| **Description** | * xa\_equalizer\_flush command processing:   - Check if post initialization state is not completed yet  - Ensure input parameter length is zero  - Flush ouput port | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_012]

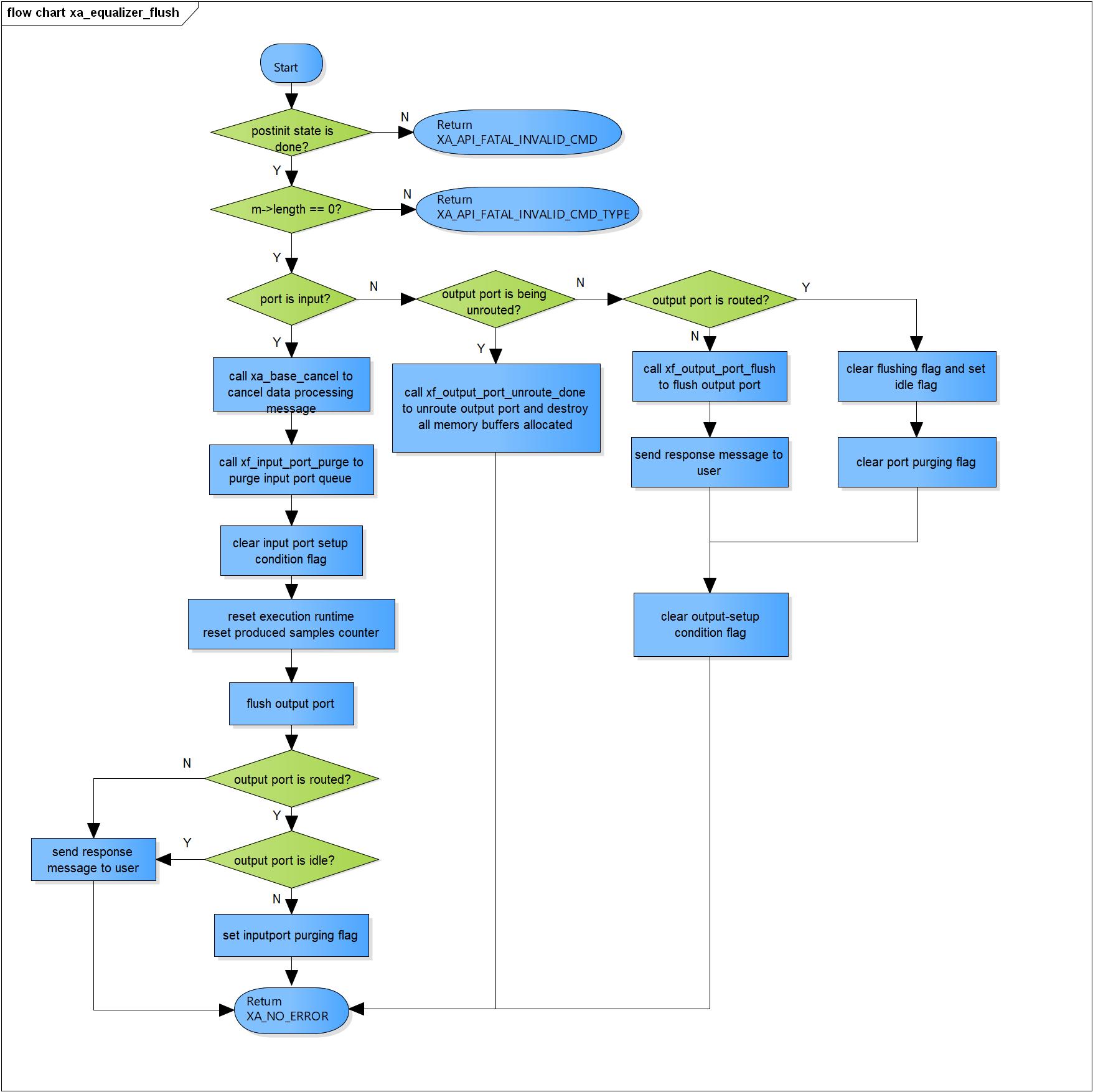


Figure 3‑9 xa\_equalizer\_flush flowchart

### xa\_equalizer\_cleanup

DD\_FWK\_EQZ\_01\_010

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 xa\_equalizer\_cleanup(xf\_component\_t \*component, xf\_message\_t \*m) | | | |
| **Function** | This function is to purge input port queue and cancel internal scheduling. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_component\_t | component | I/O | Pointer to codec instance structure (struct xf\_component ). |
| xf\_message\_t | m | I/O | Pointer to audio message (struct xf\_message). |
| **Return value** | 1 | Equalizer component cannot be destroyed. | | |
| 0 | Equalizer component has been destroyed totally. | | |
| **Description** | * xa\_equalizer\_cleanup command processing:   - Complete message with error response.  - Cancel internal scheduling message if needed.  - Purge input port queue.  - Flush output port then destroy component. | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_004]

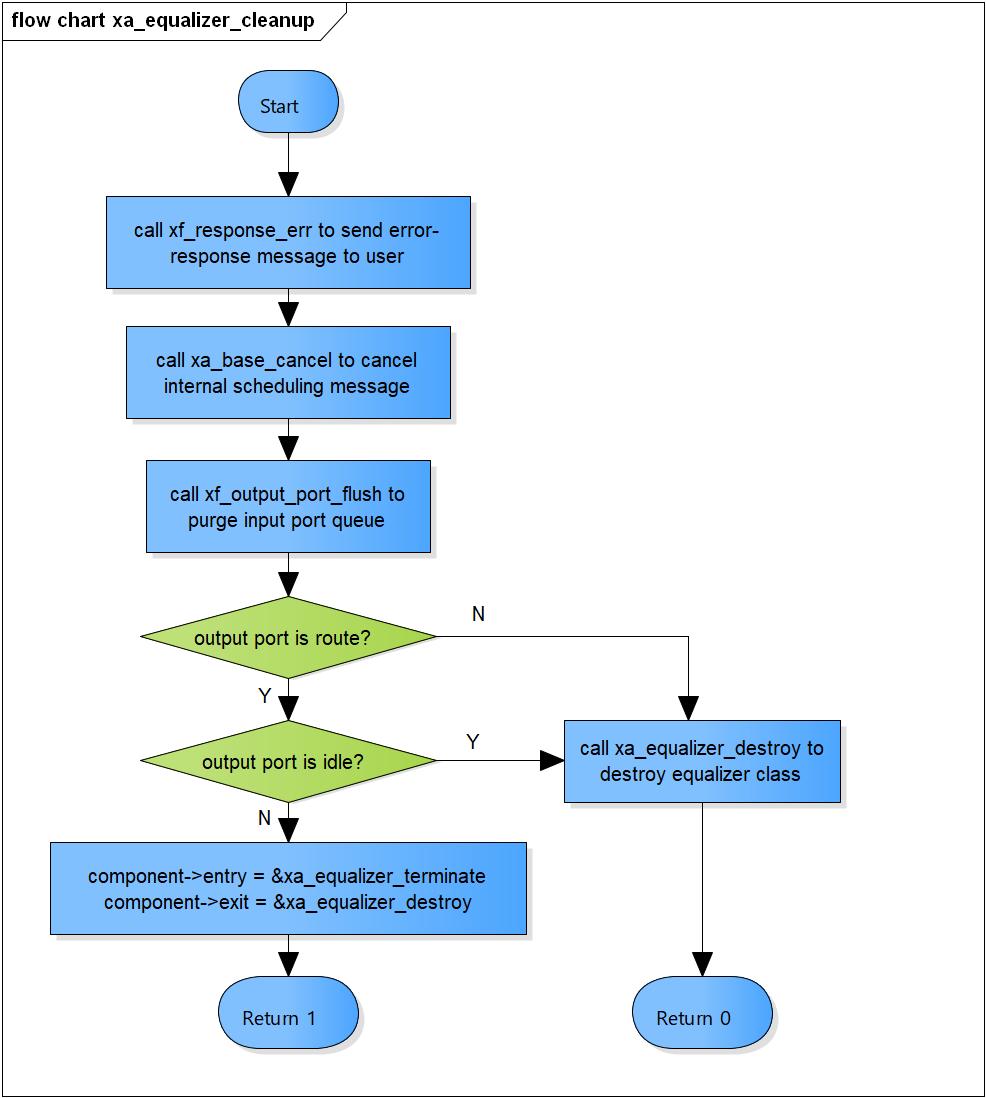


Figure 3‑10 xa\_equalizer\_cleanup flowchart

### xa\_equalizer\_terminate

DD\_FWK\_EQZ\_01\_011

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 xa\_equalizer\_terminate(xf\_component\_t \*component,  xf\_message\_t \*m) | | | |
| **Function** | This is equalizer termination-state command processor function. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_component\_t | component | I/O | Pointer to codec instance structure (struct xf\_component ). |
| xf\_message\_t | m | I/O | Pointer to audio message (struct xf\_message). |
| **Return value** | -1 | Output port flushing is completed, component ready to destroy. | | |
| 0 | Normal end | | |
| **Description** | * xa\_equalizer\_terminate command processing:   - Check if we received output port control message. | | | |

[Covers: FD\_FWK\_CMN\_004]

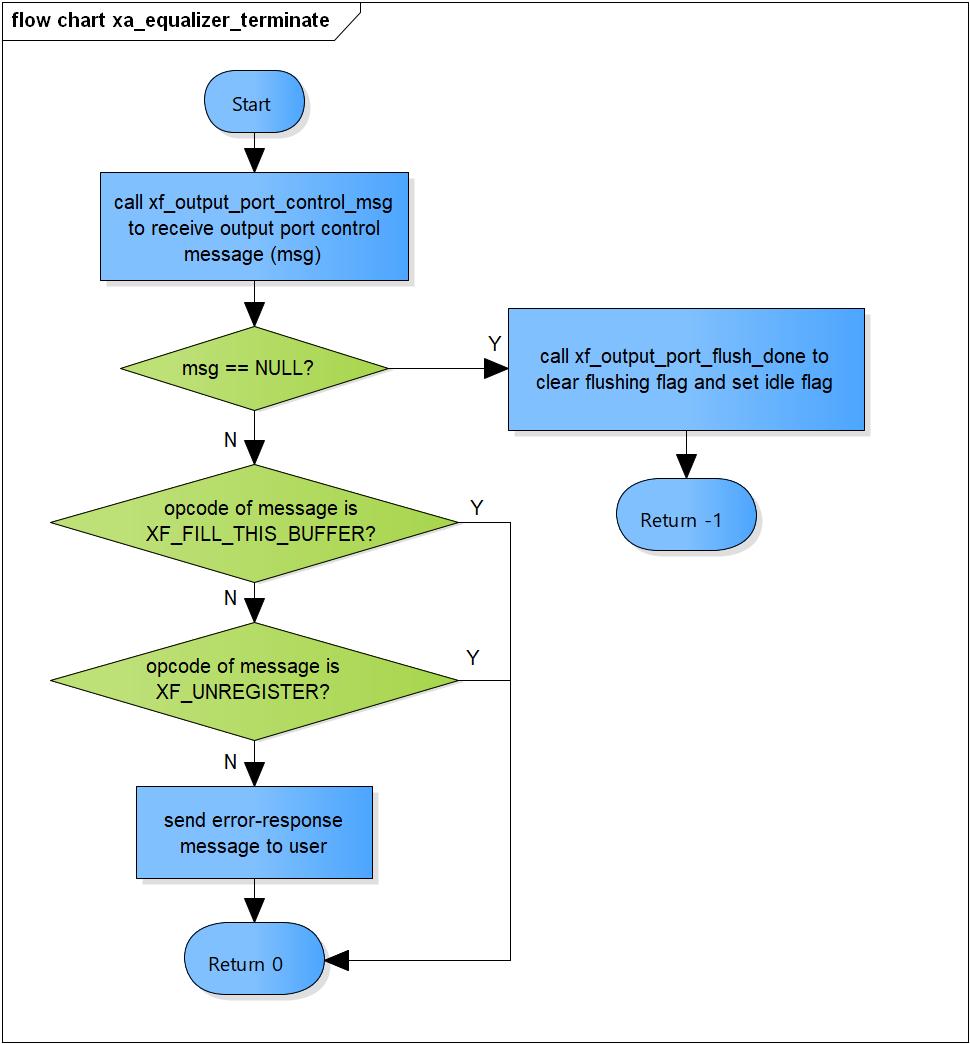


Figure 3‑11 xa\_equalizer\_terminate flowchart

### xa\_equalizer\_destroy

DD\_FWK\_EQZ\_01\_012

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static s32 xa\_equalizer\_destroy(xf\_component\_t \*component, xf\_message\_t \*m) | | | |
| **Function** | This function is to destroy component. | | | |
| **Arguments** | Type | Name | I/O | Description |
| xf\_component\_t | component | I/O | Pointer to codec instance structure (struct xf\_component ). |
| xf\_message\_t | m | x | Pointer to audio message (struct xf\_message). |
| 0 | | Indicate the component is destroyed. | |
| **Description** | * xa\_equalizer\_destrocommand processing:   - Destroy input port.  - Destroy output port.  - Destroy base object. | | | |

[Covers: FD\_FWK\_CMN\_004]

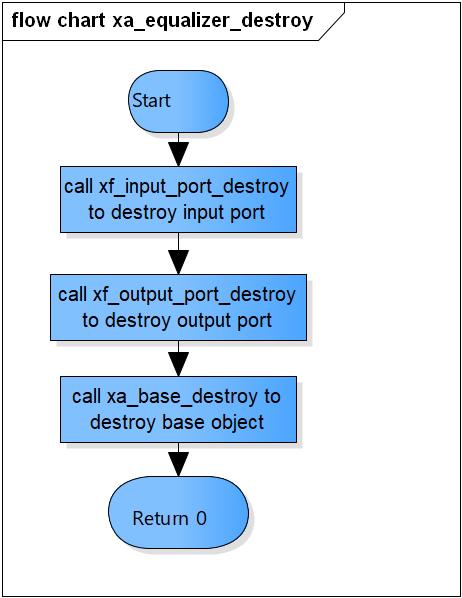


Figure 3‑12 xa\_equalizer\_destroy flowchart

### xa\_equalizer\_factory

DD\_FWK\_EQZ\_01\_013

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | xf\_component\_t \* xa\_equalizer\_factory(u32 core, xa\_codec\_func\_t process) | | | |
| **Function** | This function is to initialize Equalizer component. | | | |
| **Arguments** | Type | Name | I/O | Description |
| u32 | core | I | Core index of ADSP framework. |
| xa\_codec\_func\_t | process | I | Codec API entry point (function). |
| **Return Value** | Return handle to component. | | | |
| **Description** | * xa\_equalizer\_factory command processing:   - Initialize Equalizer component. | | | |

[Covers: FD\_FWK\_CMN\_005, FD\_FWK\_CMN\_008, FD\_FWK\_CMN\_009]

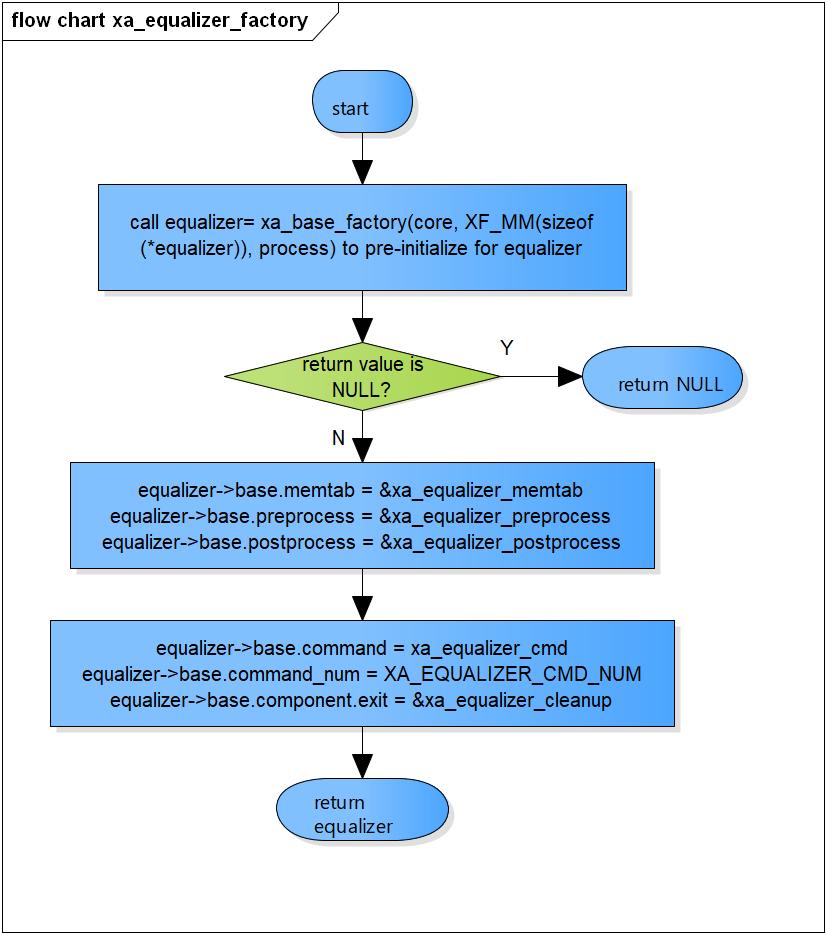


Figure 3‑13 xa\_equalizer\_factory flowchart

# 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 |