**Document Type: Detail Design**

**Document Name:**

**ADSP FRAMEWORK: SSIU DRIVER**

Renesas Electronics Corporation

Renesas Design Vietnam Co. Ltd

R-Car Software Solution 2 Group

Middleware 1 Team

Table of Contents

[1. Overview 3](#_Toc529967471)

[2. Function list 4](#_Toc529967472)

[3. Detail information 5](#_Toc529967473)

[3.1 Data type definition 5](#_Toc529967474)

[3.1.1 SSIU\_SSI\_MODULE 5](#_Toc529967475)

[3.1.2 SSIU\_SSI\_INFO 5](#_Toc529967476)

[3.1.3 SSIU\_PIN\_MODE 5](#_Toc529967477)

[3.1.4 SSIU\_SWAP\_FUNC 6](#_Toc529967478)

[3.1.5 SSIU\_OPERATION\_MODE 6](#_Toc529967479)

[3.1.6 SSIU\_PARAMS 7](#_Toc529967480)

[3.1.7 SSIU\_ERR\_CODE 7](#_Toc529967481)

[3.2 Macro definition 8](#_Toc529967482)

[3.3 Function definition 9](#_Toc529967483)

[3.3.1 xa\_ssiu\_get\_register\_base 9](#_Toc529967484)

[3.3.2 xa\_ssiu\_get\_pin\_mode 10](#_Toc529967485)

[3.3.3 xa\_ssiu\_check\_valid\_index 11](#_Toc529967486)

[3.3.4 xa\_ssiu\_setup 12](#_Toc529967487)

[3.3.5 xa\_ssiu\_start 14](#_Toc529967488)

[3.3.6 xa\_ssiu\_stop 15](#_Toc529967489)

[4. Revision history 16](#_Toc529967490)

List of Figures

[Figure 1‑1 The software architecture 3](#_Toc529967491)

[Figure 3‑1 xa\_ssiu\_get\_register\_base flowchart 9](#_Toc529967492)

[Figure 3‑2 xa\_ssiu\_get\_pin\_mode flowchart 10](#_Toc529967493)

[Figure 3‑3 xa\_ssiu\_check\_valid\_index flowchart 11](#_Toc529967494)

[Figure 3‑4 xa\_ssiu\_setup flowchart 13](#_Toc529967495)

[Figure 3‑5 xa\_ssiu\_start flowchart 14](#_Toc529967496)

[Figure 3‑6 xa\_ssiu\_stop flowchart 15](#_Toc529967497)

List of Table

[Table 2‑1 Function list 4](#_Toc529967498)

[Table 3‑1 SSIU\_PIN\_MODE type information 5](#_Toc529967499)

[Table 3‑2 SSIU\_SSI\_INFO type information 5](#_Toc529967500)

[Table 3‑3 SSIU\_PIN\_MODE type information 5](#_Toc529967501)

[Table 3‑4 SSIU\_SWAP\_FUNC type information 6](#_Toc529967502)

[Table 3‑5 SSIU\_OPERATION\_MODE type information 6](#_Toc529967503)

[Table 3‑6 SSIU\_PARAMS type information 7](#_Toc529967504)

[Table 3‑7 SSIU\_ERR\_CODE type information 7](#_Toc529967505)

[Table 2‑1 Function list 4](#_Toc534286086)

[Table 3‑1 SSIU\_SSI\_MODULE type information 5](#_Toc534286087)

[Table 3‑2 SSIU\_SSI\_INFO type information 5](#_Toc534286088)

[Table 3‑3 SSIU\_PIN\_MODE type information 5](#_Toc534286089)

[Table 3‑4 SSIU\_SWAP\_FUNC type information 6](#_Toc534286090)

[Table 3‑5 SSIU\_OPERATION\_MODE type information 6](#_Toc534286091)

[Table 3‑6 SSIU\_PARAMS type information 7](#_Toc534286092)

[Table 3‑7 SSIU\_ERR\_CODE type information 7](#_Toc534286093)

# Overview

This section provides an overview of software architecture.

User Application

ADSP Interface

Kernel Space

User Space

ADSP Driver

ARM

Audio HW

**ADSP Framework**

Renderer class

TDM class

Equalizer class

Capture class

ADSP

Plugin

Equalizer Plugin\*

TDM Plugin

Capture Plugin

Renderer Plugin

This document’s target is in side of red square

\* not connect to hardware modules

DAC/

ADC

SCU

PDMA

FIFO

SSI

SSIUU

ADMA

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\_ssiu\_setup | This API is to set up registers necessary for SSIU module execution |
| xa\_ssiu\_get\_pin\_mode | This API is to get the pin mode |
| xa\_ssiu\_check\_valid\_index | This API is to check the validity of SSIU interface index |
| xa\_ssiu\_start | This API is to start SSIU module |
| xa\_ssiu\_stop | This API is to stop SSIU module |
| Internal function | xa\_ssiu\_get\_register\_base | This function is to get register base of SSIU BUSIFn |

# Detail information

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

## Data type definition

### SSIU\_SSI\_MODULE

The data type SSIU\_SSI\_MODULE is a type-defined enumeration that lists all supported pin modes for SSIU.

Table 3‑1 SSIU\_SSI\_MODULE type information

|  |  |  |
| --- | --- | --- |
| **Member name** | **Value** | **Outline** |
| SSIU\_SSI0 | 0 | SSI0 |
| SSIU\_SSI1 | 1 | SSI1 |
| SSIU\_SSI2 | 2 | SSI2 |
| SSIU\_SSI3 | 3 | SSI3 |
| SSIU\_SSI4 | 4 | SSI4 |
| SSIU\_SSI5 | 5 | SSI5 |
| SSIU\_SSI6 | 6 | SSI6 |
| SSIU\_SSI7 | 7 | SSI7 |
| SSIU\_SSI8 | 8 | SSI8 |
| SSIU\_SSI9 | 9 | SSI9 |
| SSIU\_SSIMAX | 10 | Maximum number of SSI modules |

### SSIU\_SSI\_INFO

The data type SSIU\_SSI\_INFO is a type-defined structure that lists SSIU’s properties: module index and bus index.

Table 3‑2 SSIU\_SSI\_INFO type information

|  |  |  |
| --- | --- | --- |
| **Member name** | **Data type** | **Outline** |
| module\_index | UWORD32 | SSIU module index |
| bus\_index | UWORD32 | SSIU bus index |

### SSIU\_PIN\_MODE

The data type SSIU\_PIN\_MODE is a type-defined enumeration that lists all supported pin modes for SSIU.

Table 3‑3 SSIU\_PIN\_MODE type information

|  |  |  |
| --- | --- | --- |
| **Member name** | **Value** | **Outline** |
| SSIU\_NON\_PIN\_MODE | 0 | No setting pin mode for SSIU |
| SSIU\_S3\_S4\_PIN\_MODE | 1 | SSI3 works as slave and SSI4 works as slave |
| SSIU\_M3\_S4\_PIN\_MODE | 2 | SSI3 works as master and SSI4 works as slave |
| SSIU\_S0\_S2\_PIN\_MODE | 3 | SSI0 works as slave and SSI2 works as slave |
| SSIU\_M0\_S2\_PIN\_MODE | 4 | SSI0 works as master and SSI2 works as slave |
| SSIU\_S0\_S1\_PIN\_MODE | 5 | SSI0 works as slave and SSI1 works as slave |
| SSIU\_M0\_S1\_PIN\_MODE | 6 | SSI0 works as master and SSI1 works as slave |
| SSIU\_S0\_S9\_PIN\_MODE | 7 | SSI0 works as slave and SSI9 works as slave |
| SSIU\_M0\_S9\_PIN\_MODE | 8 | SSI0 works as master and SSI9 works as slave |
| SSIU\_S3\_S9\_PIN\_MODE | 9 | SSI3 works as slave and SSI9 works as slave |
| SSIU\_M3\_S9\_PIN\_MODE | 10 | SSI3 works as master and SSI9 works as slave |
| SSIU\_S0\_S3\_PIN\_MODE | 11 | SSI0 works as slave and SSI3 works as slave |
| SSIU\_M0\_S3\_PIN\_MODE | 12 | SSI0 works as master and SSI3 works as slave |

### SSIU\_SWAP\_FUNC

The data type SSIU\_SWAP\_FUNC is a type-defined enumeration of indicators of whether swap function is enabled or not.

Table 3‑4 SSIU\_SWAP\_FUNC type information

|  |  |  |
| --- | --- | --- |
| **Member name** | **Value** | **Outline** |
| SSIU\_WORD\_SWAP\_DISABLE | 0 | Swap function is disabled |
| SSIU\_WORD\_SWAP\_ENABLE | 1 | Swap function is enabled |

### SSIU\_OPERATION\_MODE

The data type SSIU\_OPERATION\_MODE is a type-defined enumeration of indicators of whether swap function is enabled or not.

Table 3‑5 SSIU\_OPERATION\_MODE type information

|  |  |  |
| --- | --- | --- |
| **Member name** | **Value** | **Outline** |
| SSIU\_BASIC\_MODE | 0 | SSIU basic mode |
| SSIU\_TDM\_EXT\_MODE | 1 | SSIU TDM extension mode |
| SSIU\_TDM\_SPLIT\_MODE | 2 | SSIU TDM split mode |
| SSIU\_TDM\_16CH\_MODE | 3 | SSIU TDM 16-channel mode |

### SSIU\_PARAMS

The data type SSIU\_PARAMS is a type-defined structure that possesses necessary parameters for SSIU module.

Table 3‑6 SSIU\_PARAMS type information

|  |  |  |
| --- | --- | --- |
| **Member name** | **Data type** | **Outline** |
| channels | UWORD32 | channel of data used for SSIU module |
| pcm\_width | UWORD32 | PCM width of data used for SSIU module |
| pin\_mode | SSIU\_PIN\_MODE | pin mode for SSIU module |
| swap\_func | SSIU\_SWAP\_FUNC | Swap function indicator |
| operation\_mode | SSIU\_OPERATION\_MODE | SSIU operation mode |

### SSIU\_ERR\_CODE

The data type SSIU\_ERR\_CODE is a type-defined enumeration that lists all error codes for SSIU module.

Table 3‑7 SSIU\_ERR\_CODE type information

|  |  |
| --- | --- |
| **Member name** | **Value** |
| SSIU\_ERR\_NONE | 0 |
| SSIU\_ERR\_OUT\_RANGE | -1 |
| SSIU\_ERR\_BUSY | -2 |

## Macro definition

|  |  |  |
| --- | --- | --- |
| **Macros** | **Value** | **Outline** |
| SSI\_MODE0 | XF\_RCAR\_REG\_SSI\_MODE0 | SSI Mode Register 0 |
| SSI\_MODE1 | XF\_RCAR\_REG\_SSI\_MODE1 | SSI Mode Register 1 |
| SSI\_MODE2 | XF\_RCAR\_REG\_SSI\_MODE2 | SSI Mode Register 2 |
| SSI\_MODE3 | XF\_RCAR\_REG\_SSI\_MODE3 | SSI Mode Register 3 |
| SSIU\_BUSMAX | 8 | Maximum number of bus |
| SSIU\_PIN\_MODE\_S\_S | 0 | Connection between two SSIs is slave-slave |
| SSIU\_PIN\_MODE\_M\_S | 1 | Connection between two SSIs is master-slave |
| BASE\_REG(n) | XF\_RCAR\_REG\_SSI\_BUSIF\_MODE(n) | Base register of SSI module |

Note: XF\_RCAR\_REG\_SSI\_MODEn (n = 0, 1, 2, 3) and XF\_RCAR\_REG\_SSI\_BUSIF\_MODE(n) are in repository s492d/include/sys/xt-shmem/board-rcar/xf-registers.h

## Function definition

### xa\_ssiu\_get\_register\_base

DD\_PLG\_TDM\_08\_001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | static inline SSIU\_CONTROL \*xa\_ssiu\_get\_register\_base(SSIU\_SSI\_INFO module) | | | |
| **Function** | This function is to get register base of SSIU BUSIFn. | | | |
| **Arguments** | Type | Name | I/O | Description |
| SSIU\_SSI\_INFO | module | I | SSIU module information |
| **Return value** | Base register of SSIU | | | |
| **Description** | * xa\_ssiu\_get\_register\_base command processing:   - Get register base of SSIU BUSIFn | | | |

[Covers: FD\_PLG\_TDM\_005, FD\_PLG\_TDM\_017]

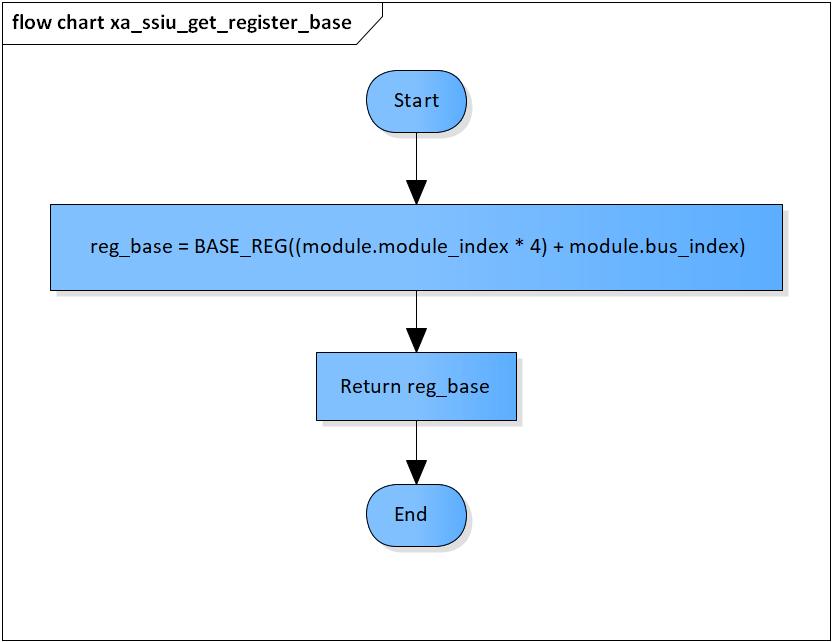


Figure 3‑1 xa\_ssiu\_get\_register\_base flowchart

### xa\_ssiu\_get\_pin\_mode

DD\_PLG\_TDM\_08\_002

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | SSIU\_PIN\_MODE xa\_ssiu\_get\_pin\_mode(SSIU\_SSI\_INFO com\_module, SSIU\_SSI\_INFO module, UWORD32 mode\_type) | | | |
| **Function** | This function is to get pin mode. | | | |
| **Arguments** | Type | Name | I/O | Description |
| SSIU\_SSI\_INFO | com\_module | I | Common SSIU module information |
| SSIU\_SSI\_INFO | module | I | SSIU module information |
| UWORD32 | mode\_type | I | Operation mode type |
| **Return value** | Pin mode | | | |
| **Description** | * xa\_ssiu\_get\_pin\_mode command processing:   - Return pin mode based on com\_module | | | |

[Covers: FD\_PLG\_TDM\_005]

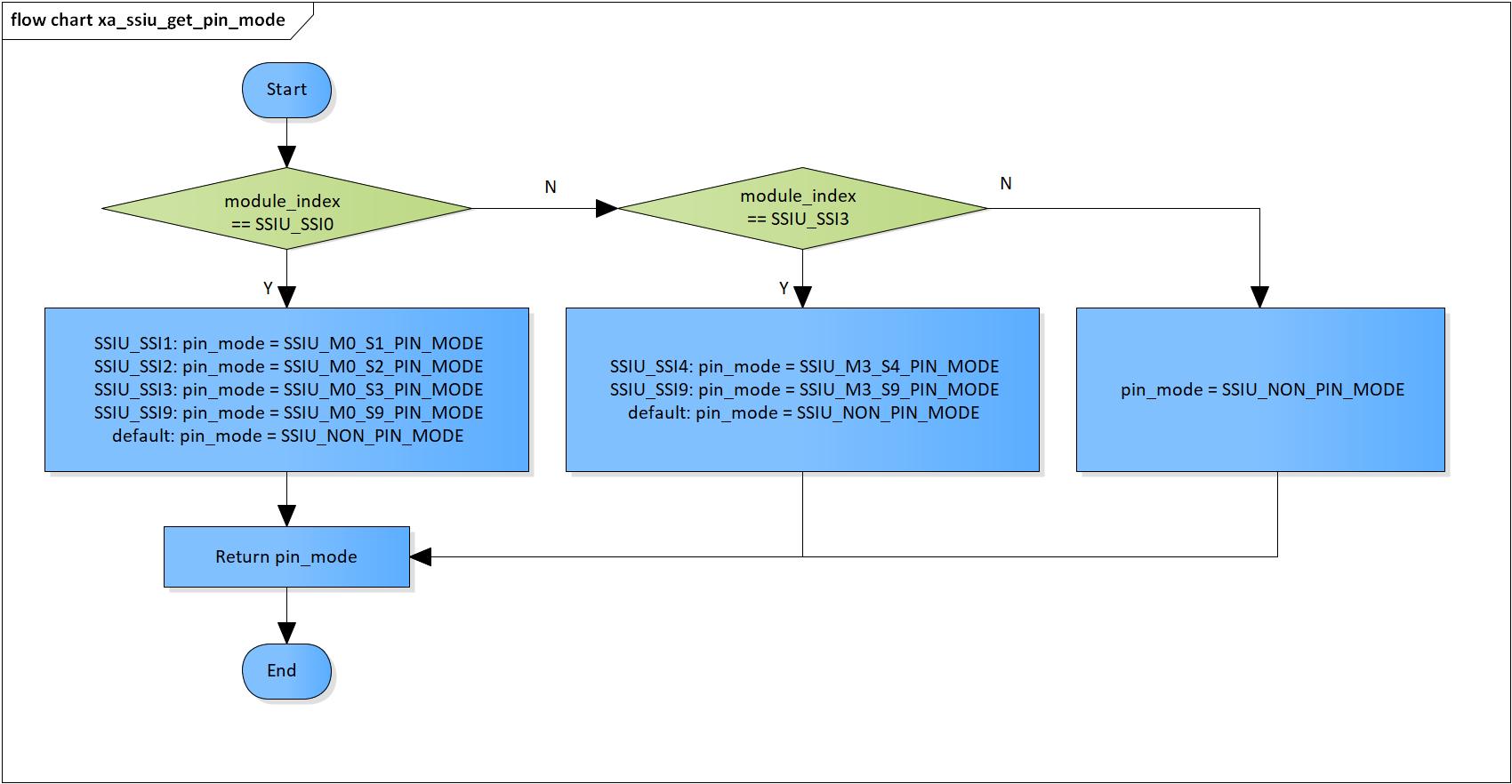


Figure 3‑2 xa\_ssiu\_get\_pin\_mode flowchart

### xa\_ssiu\_check\_valid\_index

DD\_PLG\_TDM\_08\_003

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | SSIU\_ERROR\_CODE xa\_ssiu\_check\_valid\_index(UWORD32 bus\_index) | | | |
| **Function** | This function is to check the validity of SSIU interface index. | | | |
| **Arguments** | Type | Name | I/O | Description |
| UWORD32 | bus\_index | I | Bus index  Valid value: 0 |
| **Return value** | SSIU\_ERR\_OUT\_RANGE | | Bus index is invalid | |
| SSIU\_ERR\_NONE | | Normal end | |
| **Description** | * xa\_ssiu\_check\_valid\_index command processing:   - Check the bus index | | | |

[Covers: FD\_PLG\_TDM\_024, FD\_PLG\_TDM\_026, FD\_PLG\_TDM\_034, FD\_PLG\_TDM\_036]

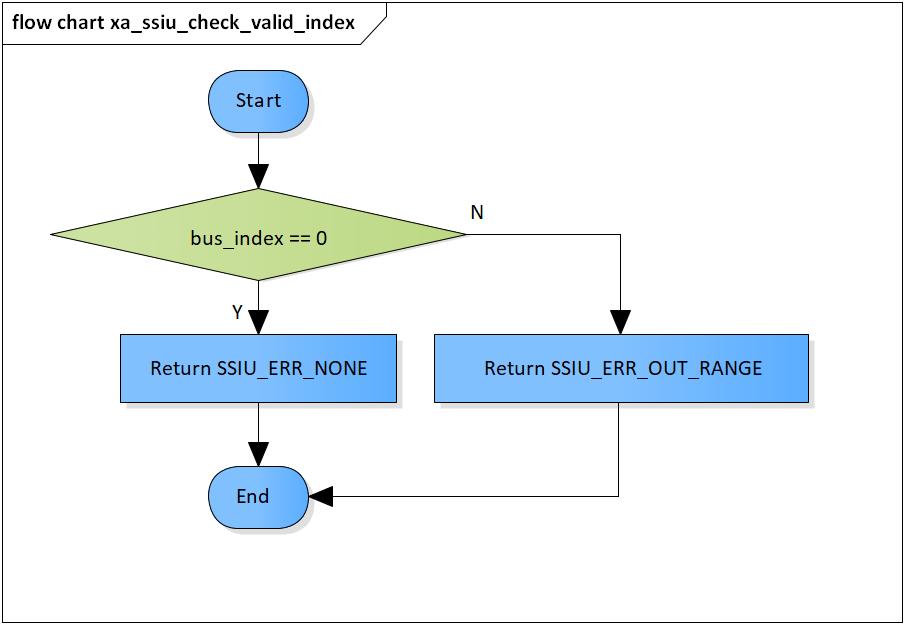


Figure 3‑3 xa\_ssiu\_check\_valid\_index flowchart

### xa\_ssiu\_setup

DD\_PLG\_TDM\_08\_004

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | SSIU\_ERR\_CODE xa\_ssiu\_setup(SSIU\_SSI\_MODULE module, SSIU\_PARAMS params) | | | |
| **Function** | This function is to set up registers necessary for SSIU module execution. | | | |
| **Arguments** | Type | Name | I/O | Description |
| SSIU\_SSI\_MODULE | module | I | SSIU module |
| SSIU\_PARAMS | params | I | Struct of parameters to set |
| **Return value** | SSIU\_ERR\_OUT\_RANGE | | Module index is invalid  Bus index is invalid  Channel is invalid  Pin mode is invalid  PCM width is invalid | |
| SSIU\_ERR\_NONE | | Normal end | |
| **Description** | * xa\_ssiu\_setup command processing:   - Set registers for SSIU execution | | | |

[Covers: FD\_PLG\_TDM\_005]

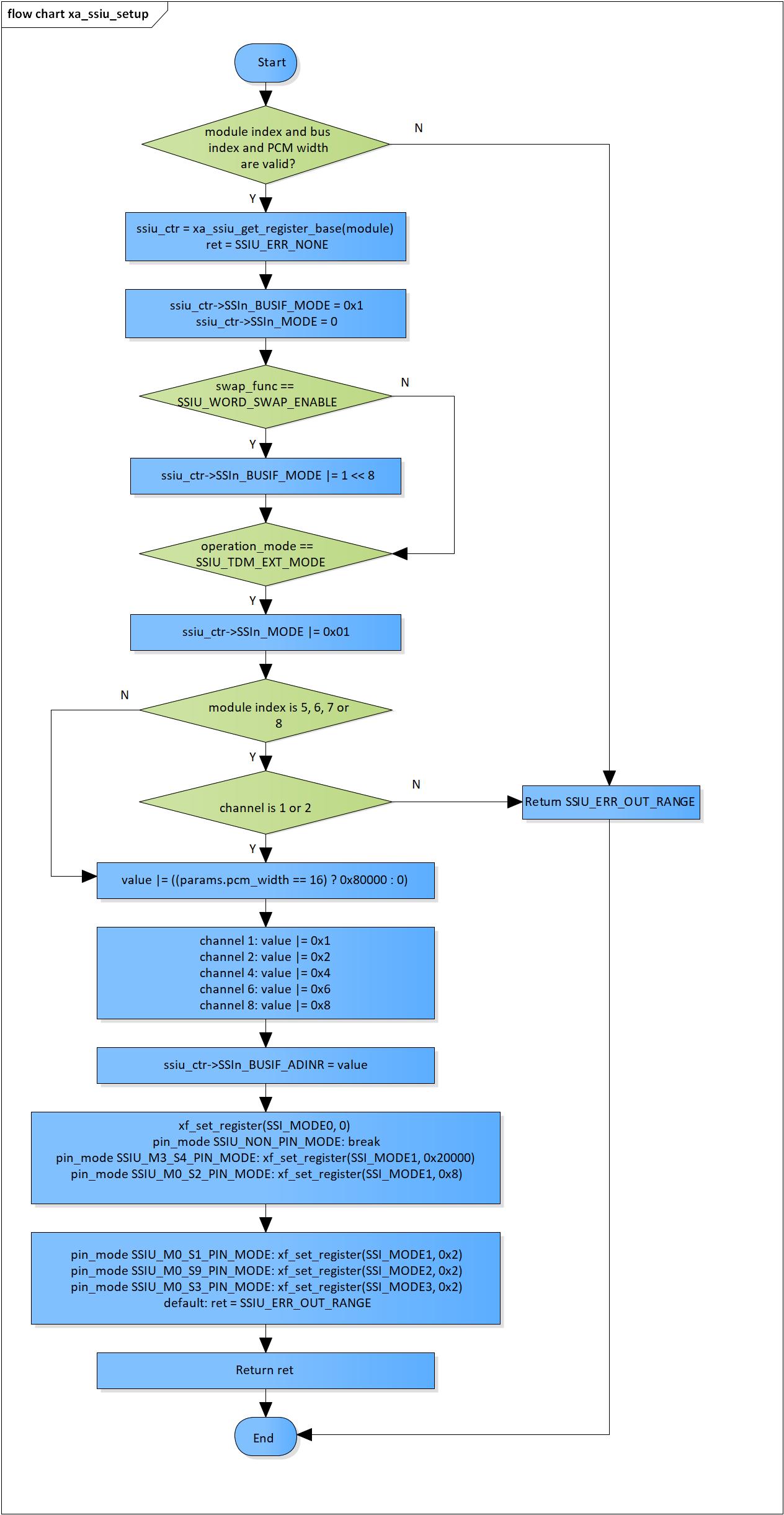


Figure 3‑4 xa\_ssiu\_setup flowchart

### xa\_ssiu\_start

DD\_PLG\_TDM\_08\_005

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | SSIU\_ERROR\_CODE xa\_ssiu\_start(SSIU\_SSI\_INFO module) | | | |
| **Function** | This function is to start SSIU module. | | | |
| **Arguments** | Type | Name | I/O | Description |
| SSIU\_SSI\_INFO | module | I | SSIU module information |
| **Return value** | SSIU\_ERR\_OUT\_RANGE | | Module index or bus index is invalid | |
| SSIU\_ERR\_BUSY | | SSIU is running | |
| SSIU\_ERR\_NONE | | Normal end | |
| **Description** | * xa\_ssiu\_start command processing:   - Set SSIn\_CONTROL (n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9) register based on the bus index | | | |

[Covers: FD\_PLG\_TDM\_005]

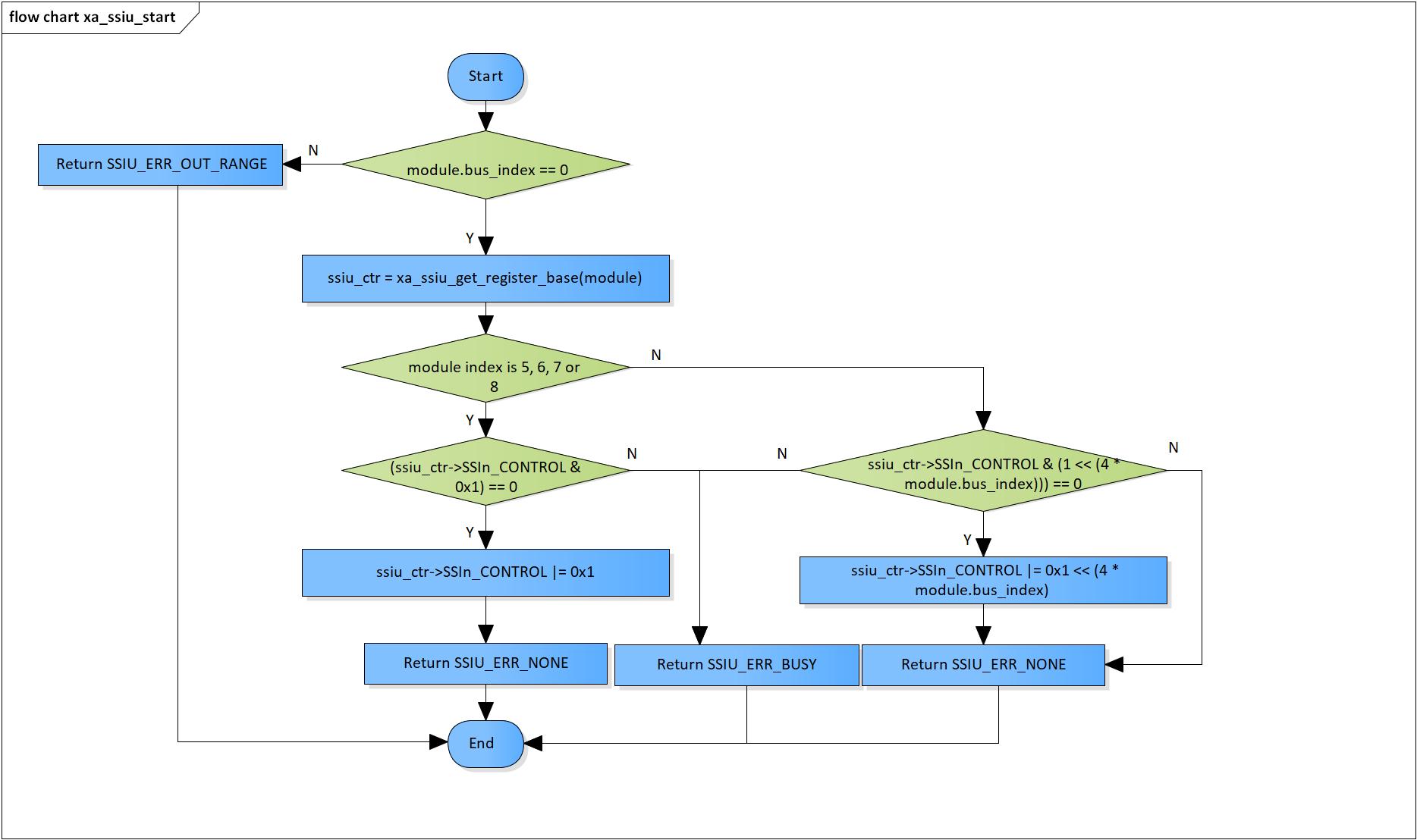


Figure 3‑5 xa\_ssiu\_start flowchart

### xa\_ssiu\_stop

DD\_PLG\_TDM\_08\_006

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Syntax** | SSIU\_ERROR\_CODE xa\_ssiu\_stop(SSIU\_SSI\_INFO module) | | | |
| **Function** | This function is to stop SSIU module. | | | |
| **Arguments** | Type | Name | I/O | Description |
| SSIU\_SSI\_MODULE | module | I | SSIU module |
| **Return value** | SSIU\_ERR\_OUT\_RANGE | | Module index or bus index is invalid | |
| SSIU\_ERR\_NONE | | Normal end | |
| **Description** | * xa\_ssiu\_stop command processing:   - Set ‘0’ to SSIn\_CONTROL (n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9) register’s bit corresponding with the bus index | | | |

[Covers: FD\_PLG\_TDM\_017]

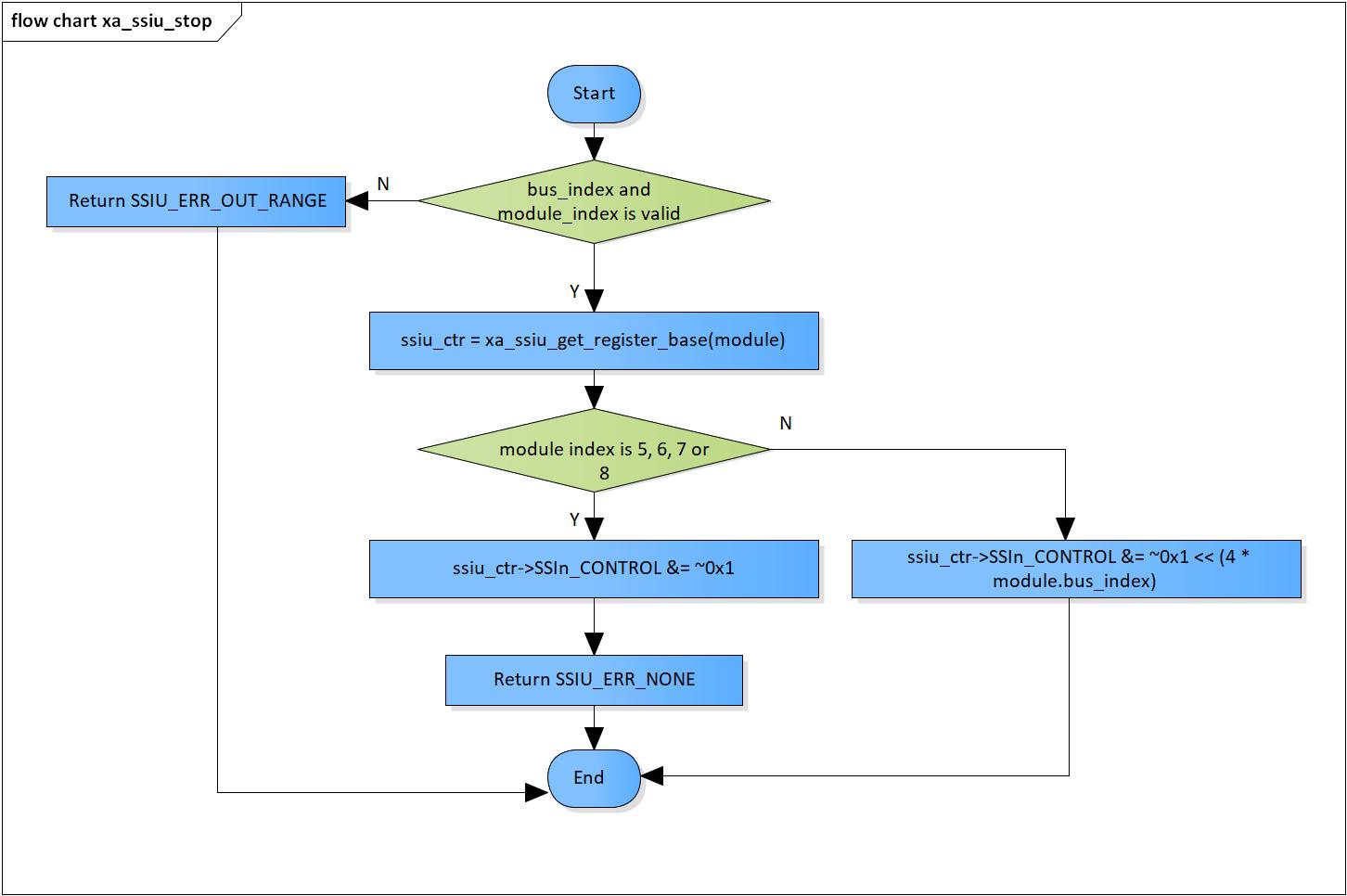


Figure 3‑6 xa\_ssiu\_stop flowchart

# Revision history

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Date** | **Page** | **Content** | **Approved** | **Changed** |
| 1.0.0 | Nov 14 2018 | - | First Edition issued | Vu Phan | Nguyen Dang |
| 1.1.0 | Dec 10 2018 | - | Add traceability ID | Vu Phan | Nguyen Dang |
| 1.2.0 | Jan 03, 2019 | - | Add range for input parameters | Vu Phan | Tien Tran |