User's Manual

CONFIDENTIAL

# ADSP Framework
# RCG3AHFWN0203ZDP

User's Manual

RCG3AHFWN0203ZDPE

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).

Rev. 1.00 May, 2019

Renesas Electronics
www.renesas.com

# 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended 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; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, 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.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

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

(Rev.4.0-1 November 2017)

Trademarks

- Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.
- Android™ is a trademark of Google LLC. Use of this trademark is subject to Google Permissions.
- Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
- Windows and Windows Media are registered trademarks of Microsoft Corporation in the United States and other countries.
- Android is a trademark of Google Inc. Use of this trademark is subject to Google permissions.
- All other company names and product names mentioned in this manual are registered trademarks or trademarks of their respective companies.
- The registered trademark symbol (®) and trademark symbol (™) are omitted in this manual.

# How to Use This Manual

## 1. Purpose and Target Reader

This manual is designed to provide the user with an understanding of the interface specifications of the Software product. It is intended for users designing application systems incorporating the Software product. Please refer to the related documents with this product.

> Use this Software after carefully reading the precautions. The precautions are stated in the main text of each section, at the end of each section, and in the usage precaution section.

> The revision history summarizes major corrections and additions to the previous version. It does not cover all the changes. For details, refer to this manual.

## 2. Restrictions on the Use of this Middleware

Any customer who wishes to use this Software must obtain a software license from Renesas Electronics.

## 3. Related Manuals

## 4. Technical Terms and Abbreviation

# - Table of Contents -

# - List of Figures -

# - List of Tables -

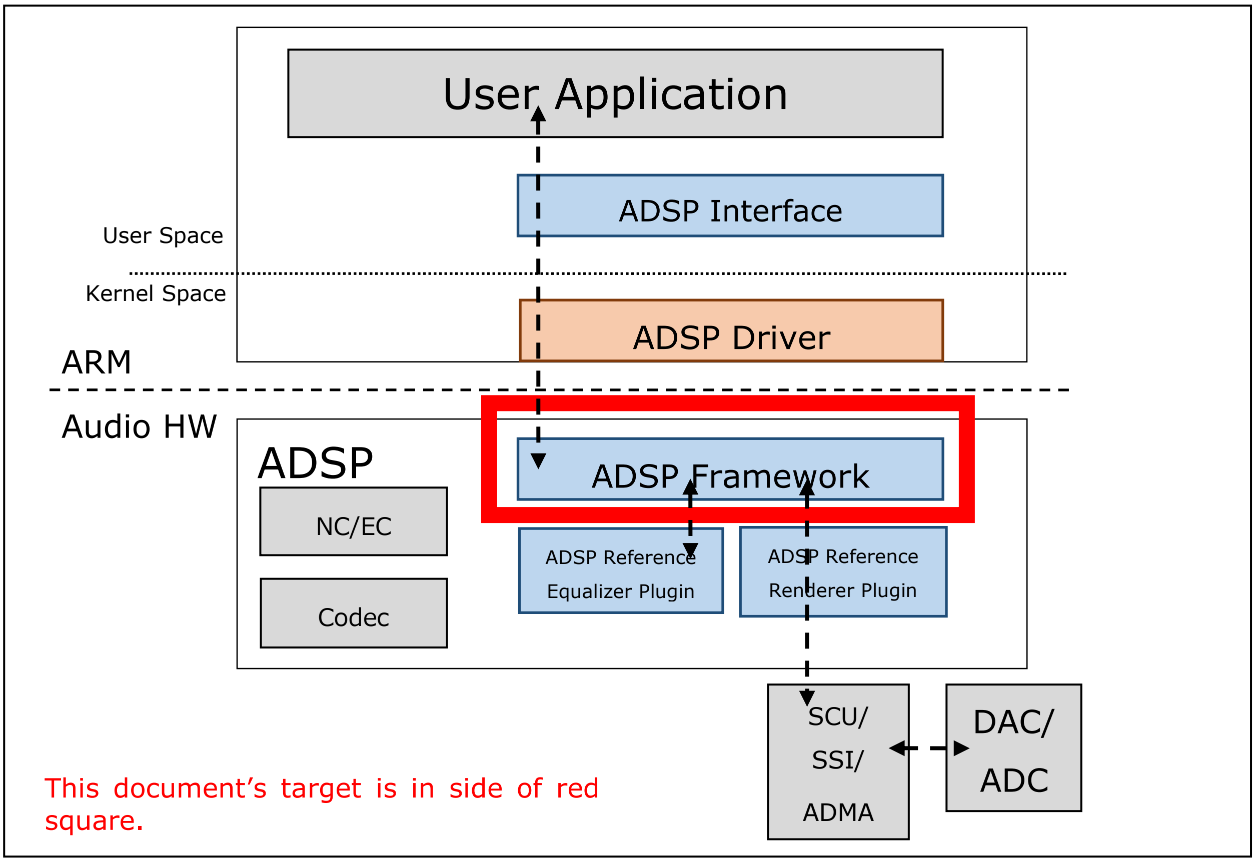
# 1. Overview

## 1.1. Overview

In this chapter, overview of ADSP Framework is explained. All expression in this manual is based on default parameters. If the software is customized from original source code, make the appropriate change according to the customization.

## 1.2. The architecture of the Software and scope of this document

The architecture of ADSP Framework is shown in Figure 1-1. ADSP Framework is a software which operates on ADSP and control ADSP Plugin according to messages from ADSP Interface.



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

Figure 1-1 The software architecture

## 1.3. Specification overview

Table 1-1 shows the specification overview of ADSP Framework.

Table 1-1 Specification overview

| Item | Outline |
|---|---|
| DSP | HiFi2 by Cadence Design Systems, Inc. |
| IDE | Xtensa Xplorer 7.0.4 (RG-2016.4) |
| Configuration | hifi2_rcar_rg20164c |
| Memory map | adspfwk_r001c |
| Interrupt | call0 ABI |
| Compiler | Xtensa C and C++ Compiler (Version 12.0.4) |
| Endian | Little endian |
| OS | XOS by Cadence Design Systems, Inc. |

[note1] This software adopts call0 ABI. Windowed ABI is not supported.
[note2] This software adopts XOS. XTOS is not supported. XTHAL can also be used.

## 1.4. Memory specification

Table 1-2 Memory specification of ADSP Framework. See 2.2 for the structure of shared buffer area.

Table 1-2 Memory map

| Allocation | Item | Cache type | start address | size |
|---|---|---|---|---|
| Internal memory | I-TCM0 | — | 0xECE8_0000 | 0x0001_0000 |
| | I-TCM1 | — | 0xECE9_0000 | 0x0001_0000 |
| | D-TCM0 | — | 0xECE7_8000 | 0x0000_8000 |
| | D-TCM1 | — | 0xECE6_0000 | 0x0001_0000 |
| External memory | Debug area | Read Cache / Writeback Cache | 0x5700_0000 | 0x0010_0000 |
| | Code area | Read Cache / Writeback Cache | 0x5710_0000 | 0x0030_0000 |
| | Shared buffer area | Read Cache / Writeback Cache | 0x5740_0000 | 0x00C0_0000 |

## 1.5. Interrupt specification

Table 1-3 Interrupt specification shows the interrupts used by ADSP Framework.

Table 1-3 Interrupt specification

| No. | type | source | direction | usage |
|---|---|---|---|---|
| 15 | edge | IRQIN_SET[0] | CPU to ADSP | send command |
| — | Level | IRQOUT_SET | ADSP to CPU | send response |
| 0 | Level | Output FIFO 0 | FIFO→ADSP | Output FIFO 0 Interrupt |
| 1 | Level | Input FIFO 0 | FIFO→ADSP | Input FIFO 0 Interrupt |
| 2 | Level | Output FIFO 1 | FIFO→ADSP | Output FIFO 1 Interrupt |
| 3 | Level | Input FIFO 1 | FIFO→ADSP | Input FIFO 1 Interrupt |
| 4 | Level | Output FIFO 2 | FIFO→ADSP | Output FIFO2 Interrupt |
| 5 | Level | Input FIFO 2 | FIFO→ADSP | Input FIFO 2 Interrupt |
| 8 | Level | Output FIFO 3 | FIFO→ADSP | Output FIFO 3 Interrupt |
| 9 | Level | Input FIFO 3 | FIFO→ADSP | Input FIFO3 Interrupt |

## 1.6. Related documents

Table 1-4 shows related documents and references.

Table 1-4 Related documents

| No. | Name | Published by |
|---|---|---|
| [1] | R-Car Series, 3rd Generation User’s Manual: Hardware | Renesas Electronics Corporation |
| [2] | Xtensa® LX4 Microprocessor Data Book | Tensilica, Inc. |
| [3] | Xtensa® XOS Reference Manual | Cadence Design Systems, Inc. |
| [4] | Xtensa® System Software Reference Manual | Cadence Design Systems, Inc. |

## 1.7. Type definitions

Table 1-5 shows the type definitions used by ADSP Framework.

Table 1-5 Type definitions

| Type | Size [byte] | Description | | | |
|---|---|---|---|---|---|
| s8 | 1 | signed 8bit integer | -128 | to | 127 |
| s16 | 2 | signed 16bit integer | -32768 | to | 32767 |
| s32 | 4 | signed 32bit integer | -2147483648 | to | 2147483647 |
| u8 | 1 | unsigned 8bit integer | 0 | to | 255 |
| u16 | 2 | unsigned 16bit integer | 0 | to | 65535 |
| u32 | 4 | unsigned 32bit integer | 0 | to | 4294967295 |

[note] The size of pointer depends on the architecture this software is used.

# 2. Software specification

ADSP Framework is an event-driven framework. Commands are stored in shared buffer, and their execution is triggered by interruption to ADSP Framework. When ADSP Framework notifies information, the information is stored in shared buffer and then is performed by storing information to shared buffer and ADSP Framework issues interrupt.

## 2.1. State transition

Figure 2-1 shows state transition of each ADSP Plugin. See 2.1.1, 2.1.2 or 2.1.3 for the flows in each stage.

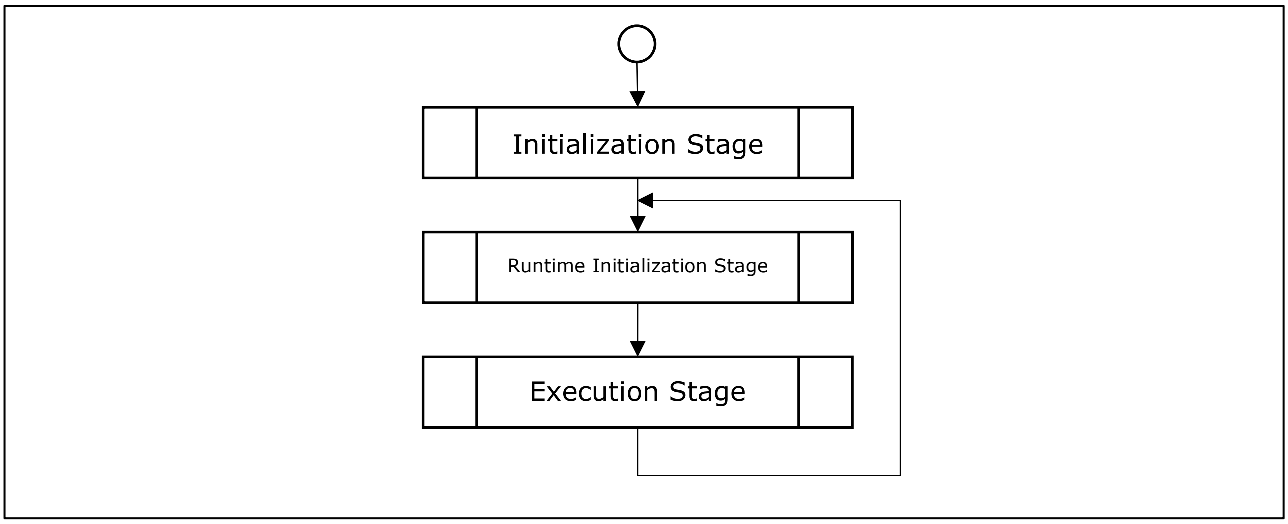


Figure 2-1 ADSP Plugin state transition

### 2.1.1. Initialization Stage

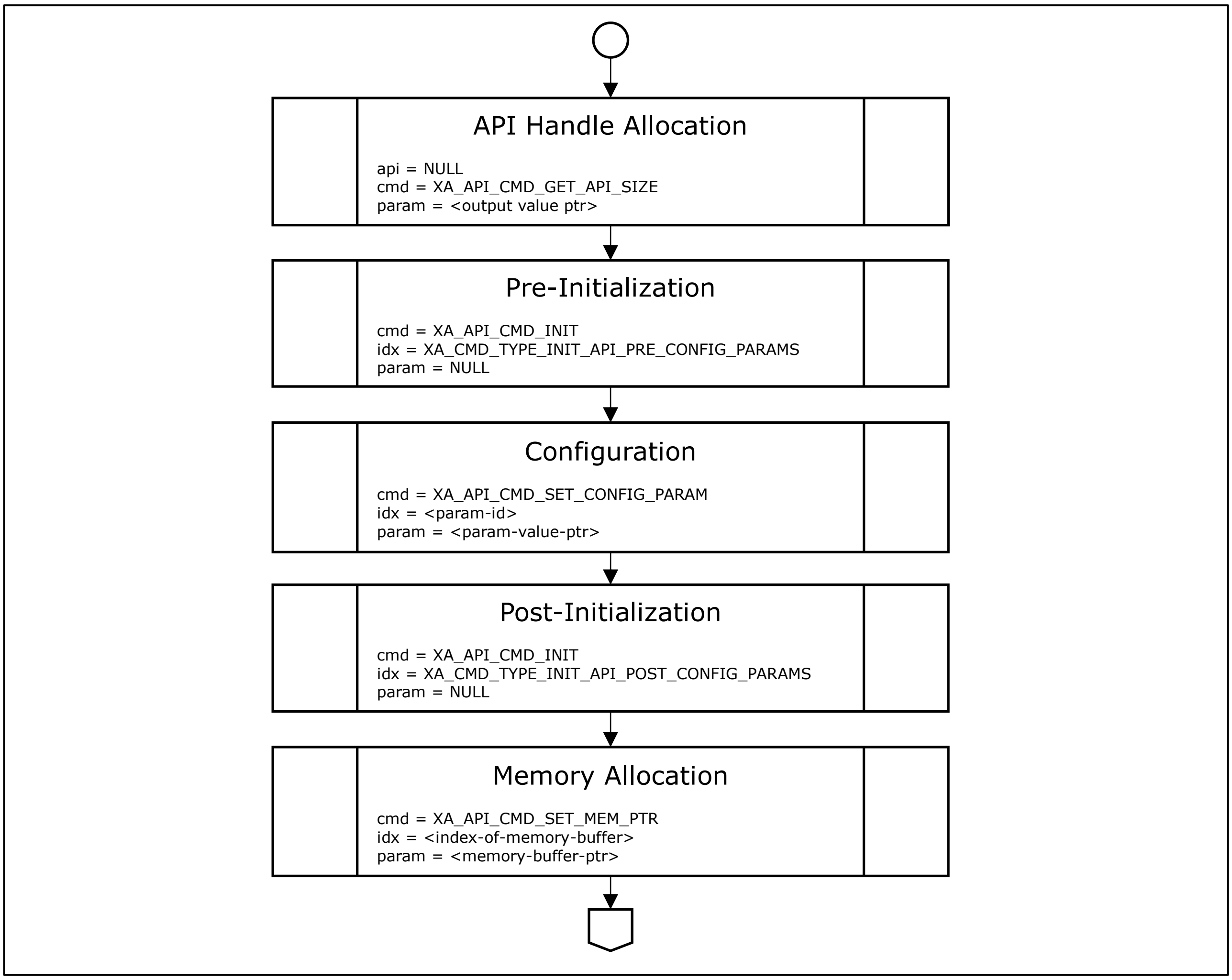

API Handle Allocation
api = NULL
cmd = XA_API_CMD_GET_API_SIZE
param = <output value ptr>
Pre-Initialization
cmd = XA_API_CMD_INIT
idx = XA_CMD_TYPE_INIT_API_PRE_CONFIG_PARAMS
param = NULL
Configuration
cmd = XA_API_CMD_SET_CONFIG_PARAM
idx = <param-id>
param = <param-value-ptr>
Post-Initialization
cmd = XA_API_CMD_INIT
idx = XA_CMD_TYPE_INIT_API_POST_CONFIG_PARAMS
param = NULL
Memory Allocation
cmd = XA_API_CMD_SET_MEM_PTR
idx = <index-of-memory-buffer>
param = <memory-buffer-ptr>

Figure 2-2 Initialization Stage flow chart

Initialization Stage performs allocation of resources for ADSP Plugin and configures static parameters. This stage consists of the following steps.

(1) API Handle Allocation
It is the start of ADSP Plugin lifecycle. It requires the size of ADSP Plugin API structure.

(2) Pre-Initialization
It is the entry point for ADSP Plugin. It initializes the API structure allocated by ADSP Framework and configures the default value.

(3) Configuration
It configures the static parameters of ADSP Plugin.

(4) Post-Initialization
It means the configure of the static parameters is finished. It fixes persistent area, scratch area and input buffer area. The size of each buffer can’t be increased after this step.

(5) Memory Allocation
ADSP Framework allocates ADSP Plugin buffer and registers it to ADSP Plugin. It is possible to exchange data with ADSP Plugin after this step.

## 2.1.2. Runtime Initialization Stage

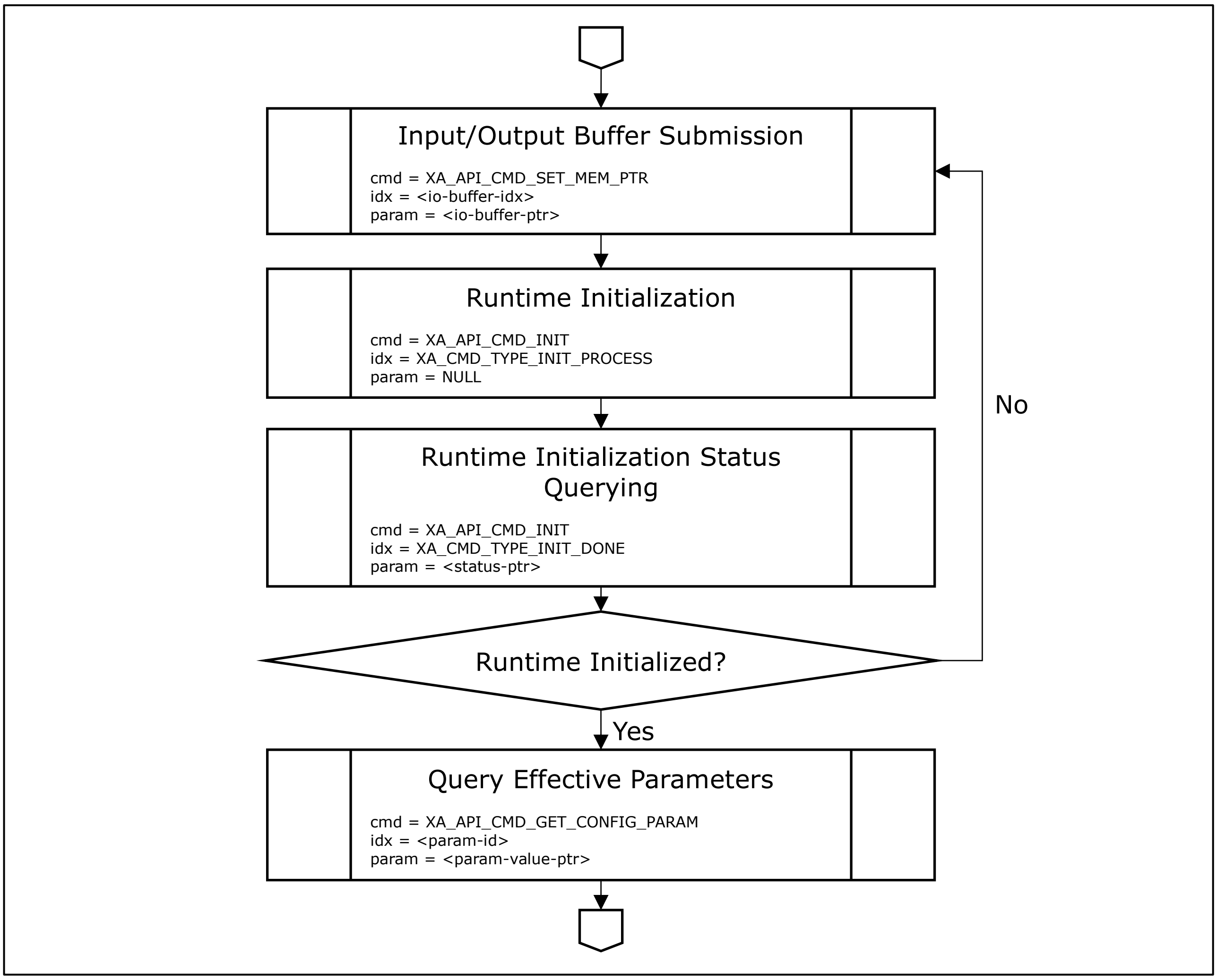

Input/Output Buffer Submission
cmd = XA_API_CMD_SET_MEM_PTR
idx = <io-buffer-idx>
param = <io-buffer-ptr>
Runtime Initialization
cmd = XA_API_CMD_INIT
idx = XA_CMD_TYPE_INIT_PROCESS
param = NULL
Runtime Initialization Status Querying
cmd = XA_API_CMD_INIT
idx = XA_CMD_TYPE_INIT_DONE
param = <status-ptr>
Runtime Initialized?
No
Yes
Query Effective Parameters
cmd = XA_API_CMD_GET_CONFIG_PARAM
idx = <param-id>
param = <param-value-ptr>

Figure 2-3 Runtime Initialization Stage flow chart

Runtime Initialization Stage performs configuration of parameter which depends on input data and so on. It checks input data but never make output data.

### 2.1.3. Execution Stage

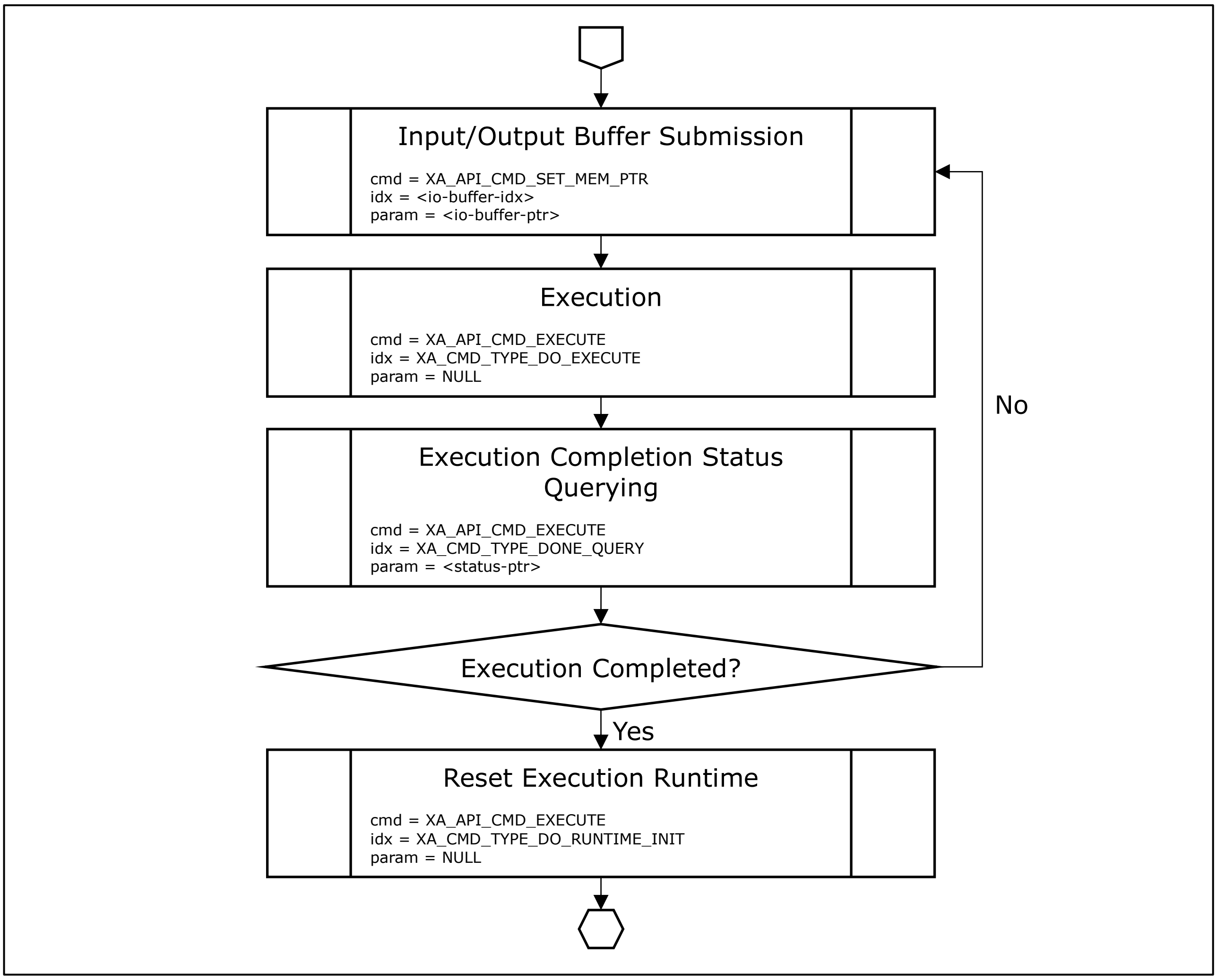


Figure 2-4 Execution Stage  flow chart

Execution Stage performs ADSP Plugin processing. The way to process input and output data in ADSP Plugin should be determined at the beginning of this stage.

ADSP Framework usually processes input and output data successively except when a physical cause inputs or outputs data e.g. Renderer plugin and Capturer plugin.

When ADSP Framework receives notification that input data is ended (see 2.6.9), it provides all input data, output all output data, and then finishes this stage. After this stage is finished, it is necessary to shift the state to Runtime Initialization Stage to run ADSP Plugin process.

It is possible to shift the state to Runtime Initialization Stage (see 2.6.10) at any time in this stage.

## 2.2. Memory structure

Figure 2-5 shows the memory structure of shared buffer in ADSP Framework. Table 2-1 shows the default configuration of the shared buffer. It is possible to extend the size of control area and data area in total up to 12 MB. Table 2-2 shows the macro which defines buffer.

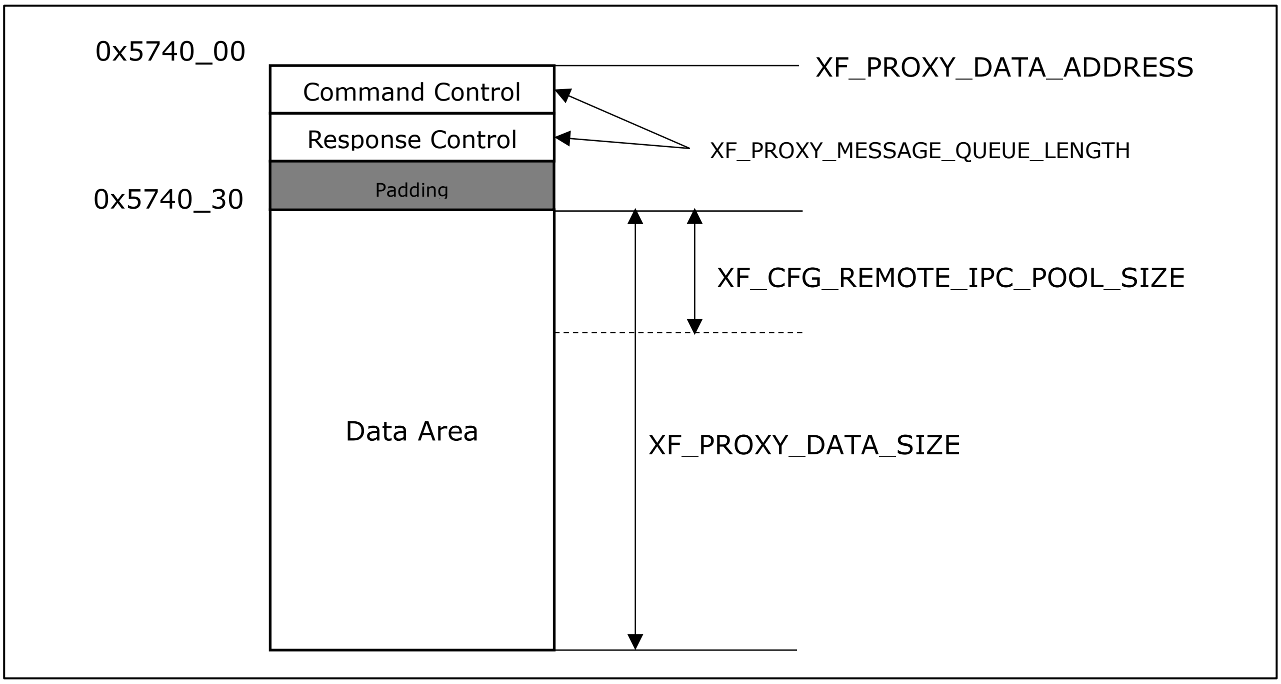


Figure 2-5 Shared buffer structure

Table 2-1 Shared buffer information

| type | buffer size | number of buffers | total size | remarks |
|---|---|---|---|---|
| command control area | 16 | 256 | 4,096 | shared buffer information |
| | 8 | 1 | 8 | index information |
| response control area | 16 | 256 | 4,096 | shared buffer information |
| | 8 | 1 | 8 | index information |
| data area | — | — | 262,144 | shared buffer information |

[note] There may be padding area between each area though not shown in Figure 2-5.

Table 2-2 The list of macros to configure shared buffer

| macro | initial value | remarks |
|---|---|---|
| XF_PROXY_DATA_ADDRESS | 0x57400000 | The start address of shared buffer |
| XF_PROXY_DATA_SIZE | 0xC00000 | The size of shared buffer |
| XF_PROXY_MESSAGE_QUEUE_LENGTH | 256 | The number of communication control area |
| XF_CFG_REMOTE_IPC_POOL_SIZE | 262,144 | The effective size of data area |

## 2.3. Message structure

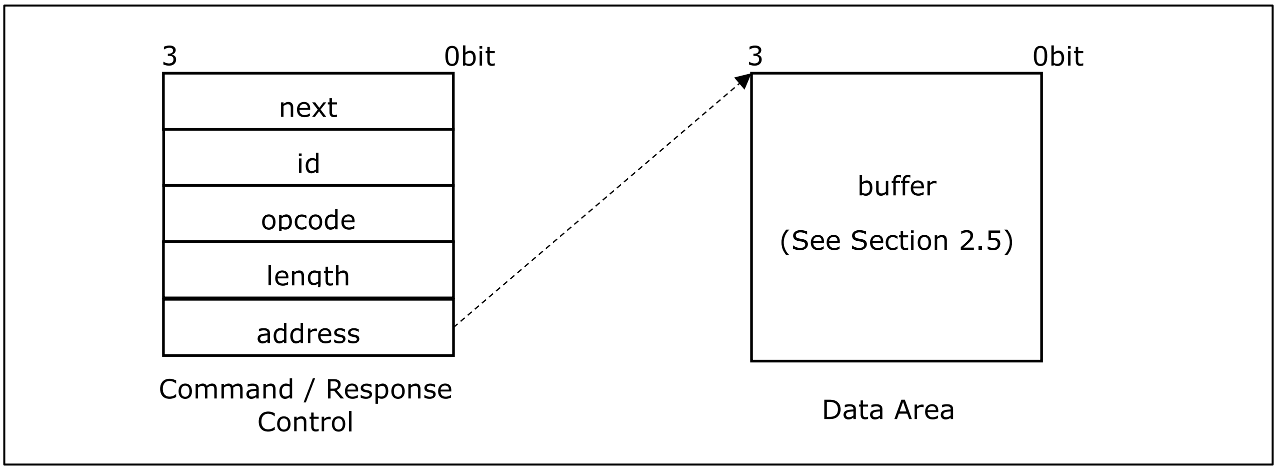


Figure 2-6 The structure of message buffer

Table 2-3 Explanation of message structure

| Member | Outline |
|---|---|
| next | This is the pointer to point to next item in the list. |
| id | This is the ID for internal administration. Users need not take care of it. |
| opcode | This is message opcode. See 2.4 for details.<br>It is configured according to "opcode" parameter when xf_command in ADSP Interface is executed. |
| length | It is configured according to "length" parameter when xf_command in ADSP Interface is executed. |
| address | It is configured according to "buffer" parameter when xf_command in ADSP Interface is executed. |
| buffer | This is the buffer used for communication. The structure changes according to opcode. |

## 2.4. Opcodes

Table 2-4 shows the opcodes provided by this software. See 2.6 for details.

Table 2-4 Opcodes

| Opcodes | Outline |
|---|---|
| XF_UNREGISTER | This opcode unregisters client. |
| XF_REGISTER | This opcode registers client. |
| XF_ROUTE | This opcode connects the ports between 2 ADSP plugin. |
| XF_UNROUTE | This opcode disconnects the connection between the ports of 2 ADSP plugin. |
| XF_ALLOC | This opcode allocates shared memory buffer. |
| XF_FREE | This opcode deallocates the shared memory buffer. |
| XF_SET_PARAM | This opcode sets parameters. |
| XF_GET_PARAM | This opcode gets parameters. |
| XF_EMPTY_THIS_BUFFER | This opcode sends input buffer. It also notifies the end of file by sending buffer with 0 byte data. |
| XF_FILL_THIS_BUFFER | This opcode sends output buffer. It also shifts the state of ADSP Plugin to Runtime Initialization Stage by sending buffer with 0 byte data. |
| XF_FLUSH | This opcode clears buffers for specific port. |
| XF_PAUSE | This opcode pause component operation. |
| XF_RESUME | This opcode resumes component operation. |
| XF_MMAP_THIS_BUFFER | This opcode maps buffer in shared memory/local memory to data buffer in ADSP plugin |

## 2.5. Structures

Table 2-5 shows the structures to define the structure of buffer. See 2.6 for details.

Table 2-5 Structures

| Name | Outline | Remarks |
|---|---|---|
| xf_set_param_msg_t | Structure for XF_SET_PARAM | |
| xf_get_param_msg_t | Structure for XF_GET_PARAM | |
| xf_start_msg_t | Structure for Runtime Initialization Stage | It is used for response transmit when Runtime Initialization Stage is finished. |
| xf_user_msg_t | Structure for user message | Used for response transmit. |

## 2.6. Messages

Below show the expression of message specification in this chapter.

Template for this chapter:

| Command code for opcode parameter of xf_command in ADSP Interface. | |
|---|---|
| Synopsis | Explanation of command code |
| Buffer structure | Buffer structure for buffer parameter of xf_command in ADSP Interface. |
| Parameters | Explanation of each parameter in buffer structure |

### 2.6.1. XF_UNREGISTER

| XF_UNREGISTER | |
|---|---|
| Synopsis | This opcode sends XF_UNREGISTER message to ADSP Plugin.<br>It will terminate component and used to generate error-response message for originator. |
| Buffer structure | None |
| Parameters | None |

### 2.6.2. XF_REGISTER

| XF_REGISTER | | |
|---|---|---|
| Synopsis | This opcode sends XF_REGISTER message to ADSP Plugin.<br>It will allocate new client id and create new component along with its port specification. | |
| Buffer structure | `void *buffer` | |
| Parameters | buffer | ID string of component. |

### 2.6.3. XF_ROUTE

| XF_ROUTE | | |
|---|---|---|
| Synopsis | This opcode sends XF_ROUTE message to ADSP Plugin.<br>It routes output port and create message to communicate between 2 ADSP plugins. | |
| Buffer structure | `typedef struct xf_route_port_msg`<br>`{`<br>`    u32 src;`<br>`    u32 dst;`<br>`    u32 alloc_number;`<br>`    u32 alloc_size;`<br>`    u32 alloc_align;`<br>` } xf_route_port_msg_t;` | |
| Parameters | src | Source port specification. |
| | dst | Destination port specification. |
| | alloc_number | Number of buffers to allocate. |
| | alloc_size | Length of buffers to allocate. |
| | alloc_align | Alignment restriction for a buffer. |

### 2.6.4. XF_UNROUTE

| XF_UNROUTE | | |
|---|---|---|
| Synopsis | This opcode sends XF_UNROUTE message to ADSP Plugin.<br>It un-routes output port and destroy all memory buffers allocated between 2 ADSP plugin. | |
| Buffer structure | `typedef struct xf_unroute_port_msg`<br>`{`<br>`    u32 src;`<br>`    u32 dst;`<br>`} xf_unroute_port_msg_t;` | |
| Parameters | src | Source port specification. |
| | dst | Destination port specification. |

### 2.6.5. XF_ALLOC

| XF_ALLOC | |
|---|---|
| Synopsis | This opcode sends XF_ALLOC message to ADSP Plugin.<br>It allocates shared memory buffer. Both CPU and ADSP can read and write in this area. |
| Buffer structure | None |
| Parameters | None |

### 2.6.6. XF_FREE

| XF_FREE | | |
|---|---|---|
| Synopsis | This opcode sends XF_FREE message to ADSP Plugin.<br>It will free shared memory area. | |
| Buffer structure | `void *buffer` | |
| Parameters | buffer | Pointer to point to the shared memory. |

### 2.6.7. XF_SET_PARAM

| XF_SET_PARAM | | |
|---|---|---|
| Synopsis | This opcode sends XF_SET_PARAM message to ADSP Plugin.<br>It can send multiple sub-commands and parameters at a time by adding the number of items. | |
| Buffer structure | `typedef struct xf_set_param_item`<br>`{`<br>`    u32 id;`<br>`    u32 value;`<br>`} xf_set_param_item_t;`<br><br>`typedef struct xf_set_param_msg`<br>`{`<br>`    xf_set_param_item_t item[0];`<br>`} xf_set_param_msg_t;` | |
| Parameters | id | Sub-command to ADSP Plugin |
| | value | Parameters to ADSP Plugin |

### 2.6.8. XF_GET_PARAM

| XF_GET_PARAM | | |
|---|---|---|
| Synopsis | This opcode sends XF_GET_PARAM message to ADSP Plugin.<br>It can send / receive multiple sub-commands and parameters at a time by adding the number of ids. | |
| Buffer structure | `typedef union xf_get_param_msg`<br>`{`<br>`    struct`<br>`    {`<br>`        u32 id[0];`<br>`    } c;`<br>`    struct`<br>`    {`<br>`        u32 value[0];`<br>`    } r;`<br>`} xf_get_param_msg_t;` | |
| Parameters | id | Sub-command to ADSP Plugin |
| | value | Parameters received from ADSP Plugin by callback. |

## 2.6.9. XF_EMPTY_THIS_BUFFER

| XF_EMPTY_THIS_BUFFER | | |
|---|---|---|
| Synopsis | This opcode sends XF_EMPTY_THIS_BUFFER message to ADSP Plugin.<br>There are 2 usages for this message.<br><br>(1) It sends the data in buffer as input buffer. The size is defined by length parameter.<br>(2) It notifies the end of data when buffer is set to NULL and length parameter in xf_command is set to 0. | |
| Buffer structure | `void *buffer;` | |
| Parameters | buffer | Buffer to send to ADSP Plugin |

## 2.6.10. XF_FILL_THIS_BUFFER

| XF_FILL_THIS_BUFFER | | |
|---|---|---|
| Synopsis | This opcode sends XF_FILL_THIS_BUFFER message to ADSP Plugin.<br>There are 2 usages for this message.<br><br>(1) It sends the data in buffer as output buffer. The size is defined by length parameter.<br>(2) It shifts the state of ADSP Plugin to Runtime Initialization Stage and notifies initial information by callback function when length parameter in xf_command is set to 0. | |
| Buffer structure | `void *buffer;` | |
| Parameters | buffer | Buffer to send to ADSP Plugin |

### 2.6.11. XF_FLUSH

| XF_FLUSH | |
|---|---|
| Synopsis | This opcode sends XF_FLUSH message to ADSP Plugin.<br>It is mainly used for trick-play.<br><br>It deletes the buffer send by XF_EMPTY_THIS_BUFFER immediately and call callback functions of each buffer.<br><br>When output port of ADSP Plugin is connected to the input port of another plugin, the command propagates to other plugins successively. If output port is not connected to other plugins, it calls callback function in this message. |
| Buffer structure | None |
| Parameters | None |

### 2.6.12. XF_PAUSE

| XF_PAUSE | |
|---|---|
| Synopsis | This opcode sends XF_PAUSE message to ADSP Plugin.<br><br>It stops consumption of input buffer and transmitting of output buffer in ADSP Plugin. While pausing, it is possible to send new input buffer and output buffer to ADSP Plugin. |
| Buffer structure | None |
| Parameters | None |

### 2.6.13. XF_RESUME

| XF_RESUME | |
|---|---|
| Synopsis | This opcode sends XF_RESUME message to ADSP Plugin.<br>It resumes ADSP Plugin pause. |
| Buffer structure | None |
| Parameters | None |

### 2.6.14. XF_FILL_THIS_BUFFER

| XF_FILL_THIS_BUFFER | | |
|---|---|---|
| Synopsis | This opcode sends XF_FILL_THIS_BUFFER message to ADSP Plugin.<br>This sends a buffer’s address to plugin so that plugin will store that address for use. | |
| Buffer structure | `void *buffer;` | |
| Parameters | buffer | Buffer to send to ADSP Plugin |

# 3. Notes

This section describes the notice of developing user programs.

## 3.1. Function Call

User programs which call the functions in this specification should obey the calling rules of compiler.

## 3.2. Other notes

### 3.2.1. Allocation of memory

Before calling the functions in this specification, allocate necessary memory area and each structure used for the parameters of each function.

### 3.2.2. Out of range memory access

The functions in this specification never access out of allocated memory or related I/O.

### 3.2.3. Combination with other applications

Take care not to duplicate symbol names when other applications are combined with other programs.

### 3.2.4. Monitoring on Performance

The products embedding this Software shall observe performance of the Software periodically with Watch Dog timer or such functions in order not to damage system performance.

| Revision History | ADSP Framework User's Manual |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | May. 24, 2019 | - | New Create |

ADSP Framework User's Manual

Publication Date: May. 24, 2019 Rev. 1.00

Published by: Renesas Electronics Corporation

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

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

**Renesas Electronics Corporation**
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

**Renesas Electronics America Inc.**
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

**Renesas Electronics Canada Limited**
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

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

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

**Renesas Electronics Hong Kong Limited**
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

**Renesas Electronics Taiwan Co., Ltd.**
13F, No. 363, Fu Shing North Road, Taipei 10543, 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 1207, 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 India Pvt. Ltd.**
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

**Renesas Electronics Korea Co., Ltd.**
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 5.2

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：https://www.renesas.com/contact/

Colophon 3.1

ADSP Framework

RCG3AHFWN0203ZDP

Renesas Electronics Corporation