User's Manual

RENESAS

**CONFIDENTIAL**

RH850/D1x Device Family

# Renesas Graphics Library
# JPEG Codec Unit A (JCUA) Driver

## User's Manual: Software



Renesas Electronics
www.renesas.com

Rev. 2.0 May 2020

# 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)

# How to Use This Manual

## 1. Purpose and Target Readers

This manual is designed to provide the user with an understanding the functions of JPEG Codec Unit A (JCUA). This manual is written for engineers who use JCUA.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

Please refer to documents of drivers and hardware for a target system implementing JCUA as necessary.

The following documents are related documents. Make sure to refer to the latest versions of these documents.

| Document Type | Description | Document Title | Document No. |
|---|---|---|---|
| User's manual for Hardware | Hardware specifications (pin assignments, memory maps, peripheral function specifications, electrical characteristics, timing charts) and operation description | RH850/D1L/D1M Group<br>User's Manual: Hardware | R01UH0451EJ0220 |
| User's manual for Software | Description of RGL overview | Renesas Graphics Library<br>User's Manual: Software | R01US0181ED0400 |
| | Description of WM | Renesas Graphics Library<br>Window Manager (WM) Driver<br>User's Manual: Software | LLWEB-10035990 |
| | Description of SPEA | Renesas Graphics Library<br>Sprite Engine A (SPEA) Driver<br>User's Manual: Software | LLWEB-10035991 |
| | Description of VDCE | Renesas Graphics Library<br>Video Data Controller E (VDCE) Driver<br>User's Manual: Software | LLWEB-10035992 |
| | Description of VOWE | Renesas Graphics Library<br>Video Output Warping Engine (VOWE) Driver<br>User's Manual: Software | LLWEB-10035993 |
| | Description of JCUA | Renesas Graphics Library<br>JPEG Codec Unit A (JCUA) Driver<br>User's Manual: Software | LLWEB-10035994<br>(This manual) |
| | Description of SFMA | Renesas Graphics Library<br>Serial Flash Memory Interface A (SFMA) Driver<br>User's Manual: Software | LLWEB-10064753 |
| | Description of HYPB | Renesas Graphics Library<br>HyperBus Controller (HYPB) Driver<br>User's Manual: Software | LLWEB-10064754 |
| | Description of OCTA | Renesas Graphics Library<br>OctaBus Controller (OCTA) Driver<br>User's Manual: Software | LLWEB-10064755 |
| | Description of VOCA | Renesas Graphics Library<br>Video Output Checker (VOCA) Driver<br>User's Manual: Software | LLWEB-10063801 |

| | | | |
|---|---|---|---|
| | Description of DISCOM | Renesas Graphics Library<br>Display Output Comparator (DISCOM) Driver<br>User's Manual: Software | LLWEB-10063802 |
| | Description of DRW2D | Renesas Graphics Library<br>2D Graphics (DRW2D) Driver<br>User's Manual: Software | LLWEB-10059472 |
| Porting Layer Guide | Description of porting layer of RGL | Renesas Graphics Library<br>Porting Layer Guide | LLWEB-10035995 |

## 2. Notation of Numbers and Symbols

This manual uses the following notation.

| | | |
|---|---|---|
| Binary | 0bXXXXXXXX | (X=0 or 1) |
| Decimal | XXX | (X=0-9) |
| Hex | 0xXXXXXXXX | (X=0-9, A-F) |

## 3. List of Abbreviations and Acronyms

| Abbreviation | Full Form |
|---|---|
| API | Application Programming Interface |
| bpp | bit per pixel |
| DHT | Define Huffman Tables |
| DQT | Define Quantization Tables |
| EOI | End of image |
| Frame buffer | A region in the RAM memory where the decompressed image result from JCUA is stored. |
| H/W | Hardware |
| JCUA | JPEG Codec Unit A. This is H/W which controls JPEG decoder. |
| JPEG | Joint Photographic Experts Group |
| MCU | Minimum Coded Unit |
| Stride | Distance in pixels between two adjacent pixel rows of the frame buffer in the memory. |
| SOF0~ SOFF | Start of Frame Type 0 ~ Type F |
| SOF | Start of Frame |
| SOI | Start of image |
| SOS | Start of Scan |
| S/W | Software |
| VDCE | Video Data Controller E. This is H/W, which controls video input, image synthesis and video output. |

# Table of Contents

# 1.Overview

## 1.1 Feature and Scope

The JCUA driver handles decompression of JPEG data which has been compressed in the JPEG baseline standard format. The JCUA driver is only available for the RH850/D1Mx RGL package.

## 1.2 Component Structure

The component structure of JCUA is shown in ***Figure 1-1***.

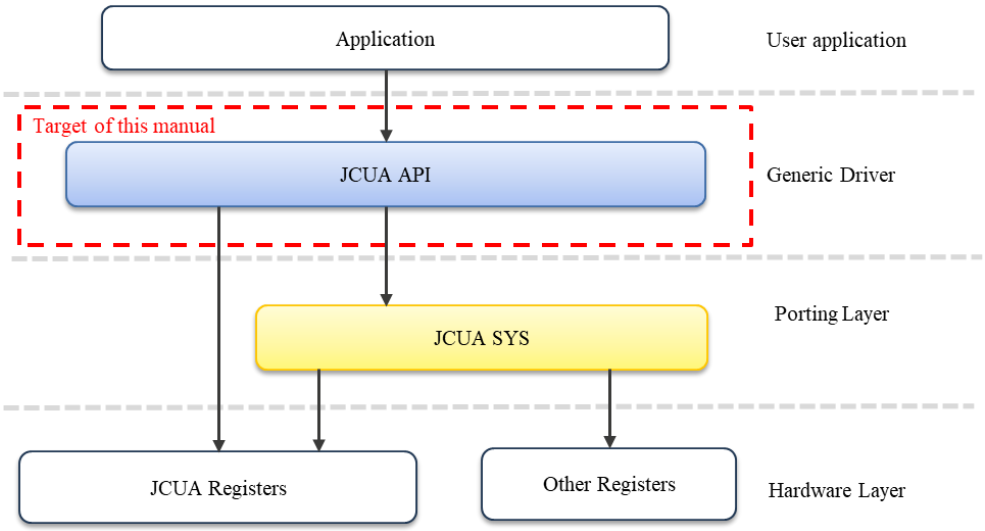


**Figure 1-1 Component Structure**

For the details of API, please refer to ***Chapter 4***.

# 2.Basic Specification

## 2.1 Summary Specification

The summary of specification is described in *Table 2-1*.

**Table 2-1 Summary Specifications**

| Items | Description |
|---|---|
| Target LSI | RH850/D1M1(H), RH850/D1M1-V2, RH850/D1M1A, RH850/D1M2(H) |
| Main Feature | • Specifications<br>○ ISO / IEC 10918-1: 1994 JPEG baseline standard.<br>○ The JCUA driver does not support the following basic features:<br>▪ Scanning with two elements<br>▪ Non-interleave scanning with multiple elements<br>▪ Unsupported pixel formats. (e.g. monochrome, CMYK, YCCK)<br>• Pixel Format<br>○ Decompression:<br>▪ YCbCr444 (H = 1:1:1, V = 1:1:1) [8 lines by 8 pixels]<br>▪ YCbCr422 (H = 2:1:1, V = 1:1:1) [8 lines by 16 pixels]<br>▪ YCbCr411 (H = 4:1:1, V = 1:1:1) [8 lines by 32 pixels]<br>▪ YCbCr420 (H = 2:1:1, V = 2:1:1) [16 lines by 16 pixels]<br>○ Output pixel format to the buffer:<br>▪ YCbCr422<br>▪ ARGB8888<br>▪ RGB565<br>• Image Size<br>○ Decompression image size: 1 x 1 - 65535 x 65535<br>○ Output image size:<br>▪ YCbCr422 or RGB565: 1 x 1- 16380 x 65535<br>▪ ARGB8888: 1 x 1- 8190 x 65535 |
| Semaphore/ Mutex | N/A for JCUA. This can be implemented with porting layer. |
| Interrupts | Used in JCUA Driver. For more details please see *section 2.3* |

Note: The JCUA driver only provides the decode function.

## 2.2 Reserved Word

JCUA uses the following prefixes for avoiding confusion from other software. Prefixes of JCUA is described in *Table 2-2*.

**Table 2-2 Prefixes**

| Prefix | Description |
|---|---|
| R_JCUA_* | Prefix for JCUA Module |
| r_jcua_* | |

## 2.3 Interrupt Handler List

**Table 2-3 Interrupt Handler List**

| No. | Interrupt Name | Interrupt Handler Name | Description |
|---|---|---|---|
| (1) | JEDI | R_JCUA_IsrStop | Decompression process interrupt request. |
| (2) | JDTI | R_JCUA_IsrFinish | Data transfer interrupt request. |

## 2.4 Error Handling

### 2.4.1 Return Code

JCUA driver returns 4 types of error codes.

#### 2.4.1.1 Parameter level

Following errors occur by a cause such as abnormality of parameter. In this case, please set valid parameter again.

- R_JCUA_ERR_PARAM_INCORRECT
- R_JCUA_ERR_RANGE_UNIT
- R_JCUA_ERR_RANGE_PARAM

#### 2.4.1.2 Timing level

Following errors occur by a cause such as abnormality of execution timing. In this case, please call again after changing to valid state or timing.

- R_JCUA_ERR_NOT_ACCEPTABLE

#### 2.4.1.3 Hardware level

Following errors occur when unexpected error occurs internally. In this case, please reset the RH850/D1x device.

- R_JCUA_ERR_NG
- R_JCUA_ERR_FATAL_HW
- R_JCUA_ERR_BUS_TIMEOUT

#### 2.4.1.4 System level

Following errors occur by a cause such as OS dependent error (e.g. system call error, resource shortage). In this case, please do recovery processing from a system layer, because this status cannot be restored only in this library.

- R_JCUA_ERR_FATAL_OS
- R_JCUA_ERR_TIMER_CTRL

### 2.4.2 Callback

JCUA driver notifies following error events.

#### 2.4.2.1 Parameter level

JCUA driver notifies following event when frame buffer is too small. In this case, please re-execute the decoding with correction of parameters such as enlarging the frame buffer.

- R_JCUA_CB_FACTOR_DECODE_SIZEOVER

#### 2.4.2.2 Hardware level

JCUA driver notifies following event when H/W error or timeout occurs. In this case, please reset (Close and de-Initialize) the JCUA driver and re-do from Initialization from R_JCUA_Init. The data cannot be decoded.

- R_JCUA_CB_FACTOR_DECODE_ERRORED
- R_JCUA_CB_FACTOR_HEADER_TIMEOUT
- R_JCUA_CB_FACTOR_DECODE_TIMEOUT

#### 2.4.2.3 Fatal level

JCUA driver notifies following event when unexpected error occurs internally. In this case, please reset the RH850/D1x device.

- R_JCUA_CB_FACTOR_FATAL_ERROR

## 2.5 State Transition

**Table 2-4: State Details**

| No. | State Name | Description |
|---|---|---|
| (1) | Uninitialized | It shows that JCUA driver is not initialized. |
| (2) | Initialized | It shows that decompression mode of JCUA driver is not set after the JCUA driver is initialized. |
| (3) | Idle (decode-ready) | It shows that Decompression mode of JCUA driver is set to a decoding mode. The decoding is not performed yet. |
| (4) | Executing (decoding) | It shows that JCUA driver is performing a decoding. |
| (5) | Executing (decode-paused) | It shows that pause caused by input division or output division is occurring after JCUA driver performed a decoding. When a decoding is restarted, the status transition to a decoding status occurs. |

The image describes state transition.

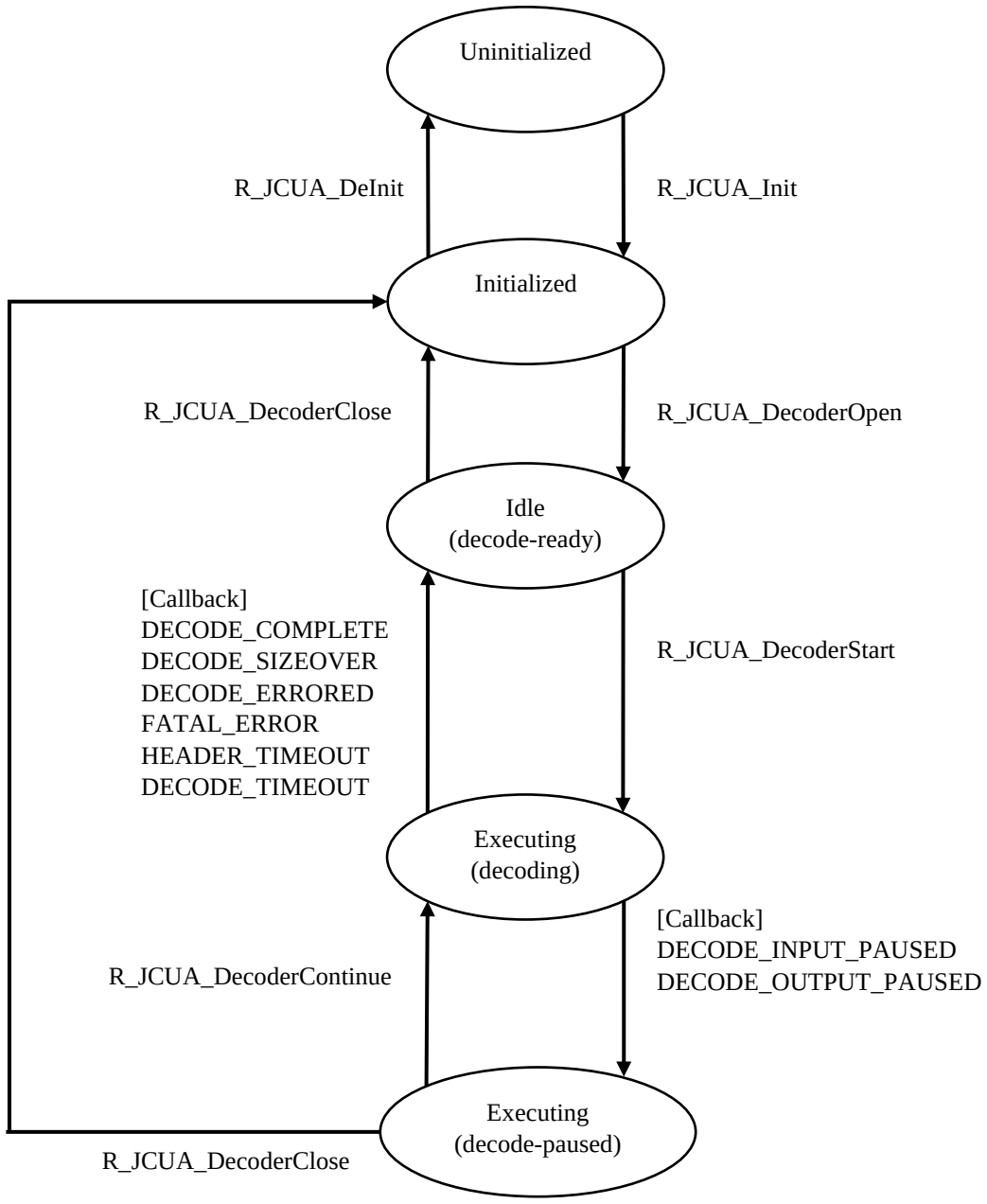


**Figure 2-1 State Transition Diagram of JCUA driver**

**Table 2-5 State Transition Table of JCUA driver**

| Function Name | State | | | | |
|---|---|---|---|---|---|
| | Uninitialized | Initialized | Idle (decode-ready) | Executing (decoding) | Executing (decode-paused) |
| R_JCUA_Init | OK | NG | NG | NG | NG |
| R_JCUA_DeInit | NG | OK | NG | NG | NG |
| R_JCUA_DecoderOpen | NG | OK | NG | NG | NG |
| R_JCUA_DecoderClose | NG | NG | OK | NG | OK |
| R_JCUA_DecoderStart | NG | NG | OK | NG | NG |
| R_JCUA_DecoderContinue | NG | NG | NG | NG | OK |
| R_JCUA_ErrorInfoGet | NG | NG | OK | NG | NG |
| R_JCUA_VersionStringGet | OK | OK | OK | OK | OK |
| R_JCUA_MacroVersionGet | OK | OK | OK | OK | OK |
| R_JCUA_IsrStop | OK | OK | OK | OK | OK |
| R_JCUA_IsrFinish | OK | OK | OK | OK | OK |
| R_JCUA_IsrTimeOut | OK | OK | OK | OK | OK |

# 3.Function Description

## 3.1 Fundamental Concepts

### 3.1.1 JCUA unit

RH850/D1x device has the following number of units of the JCUA.

**Table 3-1 Number of units**

| Feature | RH850/D1x Device Name | |
|---|---|---|
| | D1L2(H) | D1M1(H) / D1M1-V2 / D1M1A / D1M2(H) |
| Number of units | 0 | 1 |

Almost JCUA API functions have the argument “Unit”.
User specifies the JCUA H/W unit number to be controlled. The range is only 0.

### 3.1.2 Basic concept

Compressed JPEG data is input from the memory to is JCUA.
The input data is processed in the image data are transferred in MCUs from JCUA to external buffer.
The basic working concept of JPEG Codec Unit shown in *Figure 3-1*.

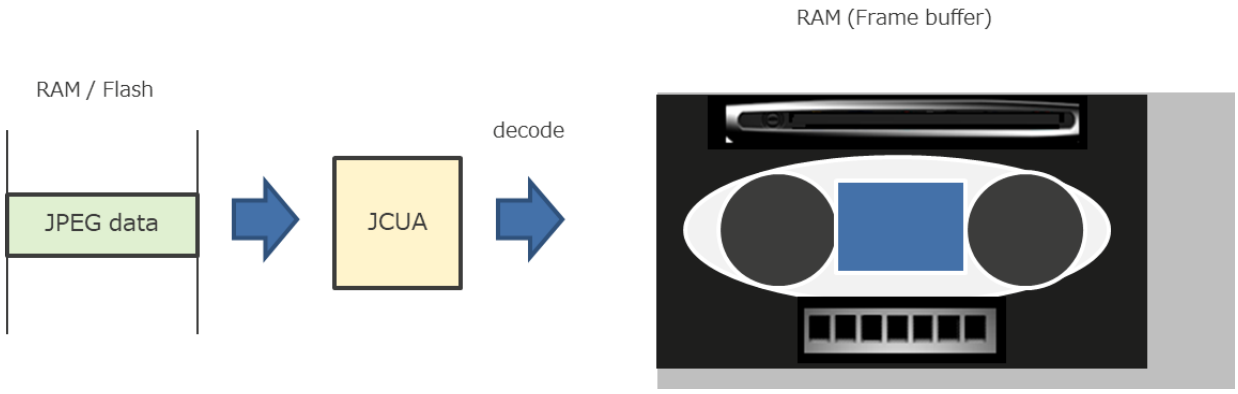


**Figure 3-1 Basic Action Processing**

### 3.1.3 Size extension of output image data

The unit for decoding of JPEG-compressed data from JCUA driver is an MCU (minimum coded unit). If compressed image size isn't a multiple of MCU, JCUA driver expands decoding image to a unit of MCU automatically by padding. The padded portion of data becomes undefined. MCU is the unit of processing of JPEG-compressed data. MCU size is decided by pixel format (YCbCr444, YCbCr422 or etc.).

Example: 4 x 10 pixel image compressed by YCbCr422 (MCU: 16 x 8 pixel) expands decoding image to 16 x 16 pixel which is a unit of MCU.

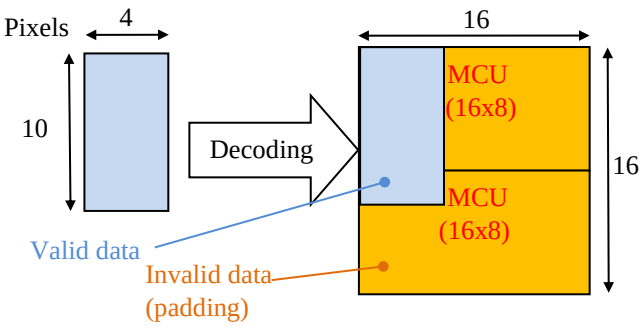


**Figure 3-2 Size expansion of output image data**

### 3.1.4 Automatic calculation of stride

When user set R_JCUA_STRIDE_AUTO to stride member of r_jcua_FrameBuffer_t structure, JCUA driver calculates necessary size of stride automatically to output a decoding image. A case where the image data (horizontal size is 65 pixels) are JPEG-compressed from YCbCr444 (MCU: 8 x 8 pixels), the image is scaled-down (see r_jcua_SubSampling_t) to 1/8 horizontally, and is then decoded and output to a frame buffer (pixel format: RGB565, stride: 0), is described below as an example.

1. JCUA driver decodes JPEG compressed data and expands a decoding image to width 72 pixel (*1) which is a unit of MCU.
2. Next, JPEG driver resizes decoding image in 1/8, so decoding image is converted from width 72 pixel to width 9 pixel (*2).
3. In case of output format is RGB565, 1 pixel data is displayed at 2 bytes (= 2 bytes per pixel). So at least 18 bytes are need to display 9 pixels. The JCUA driver rounds stride to a multiple of 128 bytes according to VDCE H/W specification. Therefore, the stride size needed for displaying image becomes 128 bytes (=64 pixels) (*3).

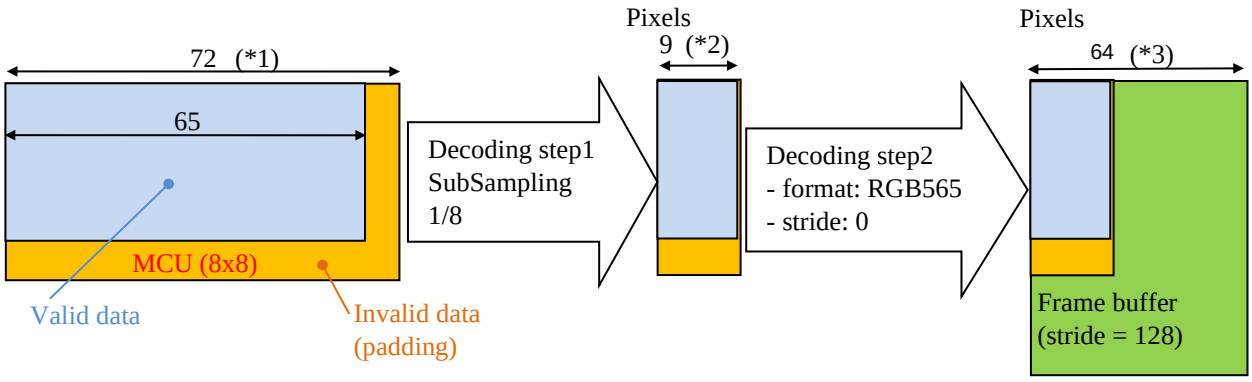


**Figure 3-3 Automatic calculation of stride**

### 3.1.5 Timeout

JCUA driver has timeout process. Because JCUA H/W continues searching the start code (SOI) until it is found. Timeout occurs following termination.

- Header timeout is detected if header decoding completion or error interrupt is not notified from H/W after user starts decoding.
- Data timeout is detected if decoding completion or error interrupt is not notified from H/W after JCUA driver starts data decoding internally.

## 3.2 Using the API

### 3.2.1 Initialization / De-Initialization

To decode JPEG-compressed data, start by executing R_JCUA_Init function and R_JCUA_DecoderOpen function. The R_JCUA_Init function initializes JCUA H/W (applies a reset, starts clock supply, and the like) and enables JCUA driver. The R_JCUA_DecoderOpen function registers a callback function, prepares for decoding, and selects the executable state for decoding of JPEG-compressed data.

To shut down JCUA driver, execute R_JCUA_DecoderClose function and R_JCUA_DeInit function. The R_JCUA_DecoderClose function can only be executed while processing for decoding is not in progress. The function releases the registration of callback function and return the driver to state in which JPEG-compressed data cannot be decoded. R_JCUA_DeInit function returns the JCUA driver to the disabled state.

### 3.2.2 Color Format

When JCUA driver is used with VDCE to output image data to display, Format and Swap members of r_jcua_FrameBuffer_t structure are set to the values defined in the following table for given pixel format.

**Table 3-2 Decoded image format vs. setting of Format and Swap members**

| Decoded image format | Format | Swap |
|---|---|---|
| YCbCr422 | R_JCUA_OUTPUT_FORMAT_YCBCR422 | R_JCUA_SWAP_LONG_AND_WORD_AND_BYTE |
| ARGB8888 | R_JCUA_OUTPUT_FORMAT_ARGB8888 | R_JCUA_SWAP_LONG |
| RGB565 | R_JCUA_OUTPUT_FORMAT_RGB565 | R_JCUA_SWAP_LONG_AND_WORD |

### 3.2.3 Frame buffer

To specify the output position, specify coordinates as the Position member of r_jcua_DecodeSetting_t structure.
To specify decoding option, set sub-sampling (VertlSubSampling/HorizSubSampling), and alpha value (Alpha) in Parameter member of r_jcua_DecodeSetting_t structure.
The information on the decoded JPEG-compressed data are set in ImgInfo member of r_jcua_DecodeSetting_t structure.
The values to set for ImgInfo are indicated by red letters in following figure.

Note: Specifying Local RAM area for the frame buffer is prohibited.

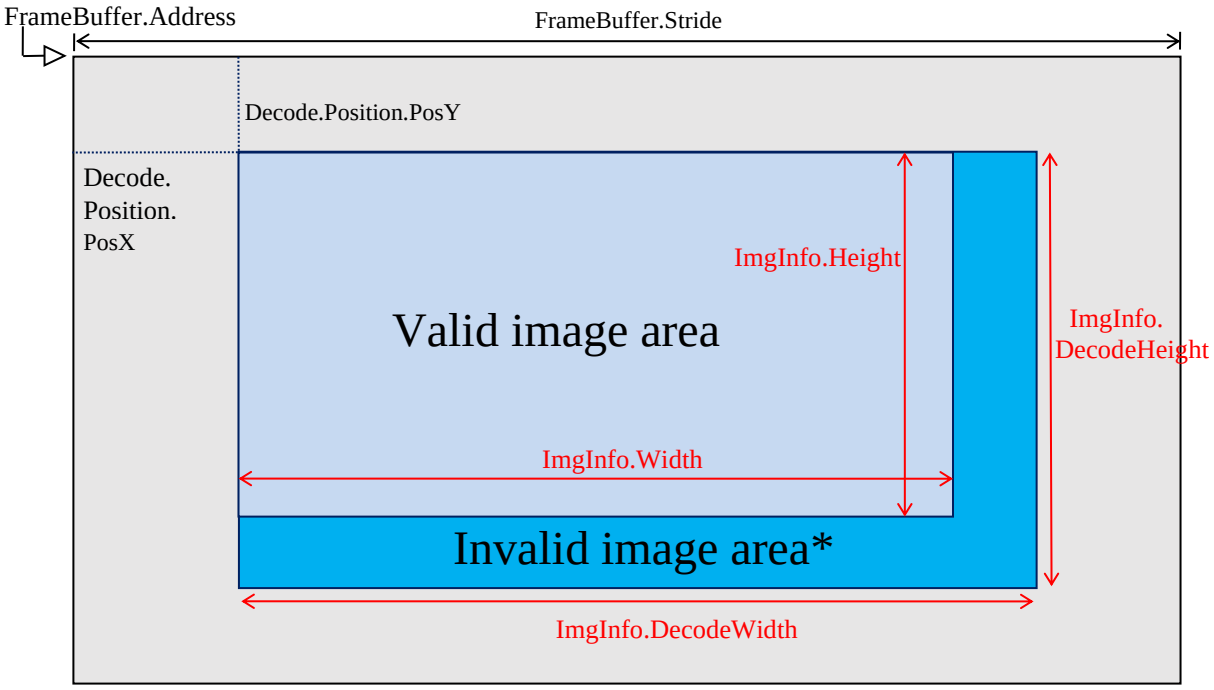


*Image data which isn't the MCU size is automatically extended to be the size of a full MCU.

**Figure 3-4 Decoding Parameters and Output Image**

### 3.2.4 Normal decoding method

To proceed with normal decoding of JPEG-compressed data, execute R_JCUA_DecoderOpen function to set the state in which decoding of JPEG-compressed data can proceed, and then execute R_JCUA_DecoderStart function.

Examples of the code and descriptions of the parameters for decoding by the R_JCUA_DecoderStart function are shown below.

**Example**

```
r_jcua_FrameBuffer_t     fb;
r_jcua_DecodeSetting_t   ds;

/* frame buffer */
fb.Address = LOC_OUTPUT_DATA_ADDRESS;
fb.Size    = 320 * 256 * 4; /* 320(pixel) * 256(pixel) * 4(byte per pixel) */
fb.Stride  = 320 * 4;       /* 320(pixel) * 4(byte per pixel ) */
fb.Format  = R_JCUA_OUTPUT_FORMAT_ARGB8888;
fb.Swap    = R_JCUA_SWAP_LONG;

/* Decode setting */
ds.OptionFlag = R_JCUA_DECODE_OPTION_POSITION | R_JCUA_DECODE_OPTION_PARAM;
ds.Position.PosX = 24;    /* (pixel) */
ds.Position.PosY = 15;    /* (pixel) */
ds.Parameter.VertlSubSampling = R_JCUA_SUB_SAMPLING_1_4;
ds.Parameter.HorizSubSampling = R_JCUA_SUB_SAMPLING_1_2;
ds.Parameter.Alpha            = 192;   /* Alpha value */

error = R_JCUA_DecoderStart(LOC_JCUA_UNIT_NUM, LOC_JPEG_DATA_ADDRESS, &fb, &ds);
```

After R_JCUA_DecoderStart function starts decoding, callback function that R_JCUA_DecoderOpen function registered is called on completion of decoding or when an error occurs. If decoding is completed, vector argument for the callback function is set to value R_JCUA_CB_FACTOR_DECODE_COMPLETE, which indicates completion of decoding.

### 3.2.5 Splitting decoding into two or more rounds

Splitting the decoding of JPEG-compressed data into two or more rounds is possible. This is useful if the available buffer size is small or when user need to interrupt decoding that is in progress.

An example of the code and descriptions of parameters for decoding by the R_JCUA_DecoderStart function are shown below.

**Example**

```
r_jcua_FrameBuffer_t     fb;
r_jcua_DecodeSetting_t   ds;
/* frame buffer */
fb.Address = LOC_OUTPUT_DATA_ADDRESS;
fb.Size    = 320 * 256 * 4; /* 320(pixel) * 256(pixel) * 4(byte per pixel ) */
fb.Stride  = 320 * 4;       /* 320(pixel) * 4(byte per pixel ) */
fb.Format  = R_JCUA_OUTPUT_FORMAT_ARGB8888;
fb.Swap    = R_JCUA_SWAP_LONG;

ds.OptionFlag = R_JCUA_DECODE_OPTION_DIVISION_MODE;
/* position */
ds.DivisionMode.InputBuffer.IsEnable        = R_TRUE;
ds.DivisionMode.InputBuffer.IsInitAddress   = R_TRUE;
ds.DivisionMode.InputBuffer.RestartAddress  = LOC_JPEG_DATA_ADDRESS;
ds.DivisionMode.InputBuffer.DataCount       = 2048;        /* bytes */
ds.DivisionMode.OutputBuffer.IsEnable       = R_FALSE;
ds.DivisionMode.OutputBuffer.IsInitAddress  = R_FALSE;
ds.DivisionMode.OutputBuffer.RestartAddress = R_NULL;
ds.DivisionMode.OutputBuffer.DataCount      = 0;

error = R_JCUA_DecoderStart(LOC_JCUA_UNIT, LOC_JPEG_DATA_ADDRESS, &fb, &ds);
```

Set the OptionFlag member of r_jcua_DecodeSetting_t structure to the value R_JCUA_DECODE_OPTION_DIVISION_MODE. Set the logical OR of this and other values as required.

The InputBuffer and OutputBuffer members of the r_jcua_DecodeSetting_t structure are same as the corresponding members of r_jcua_DivisionBufferInfo_t type structure. Make settings for splitting of data input to the JCUA driver for InputBuffer and make settings for splitting of data output from JCUA driver for OutputBuffer. The following table shows the setting values.

For information on settings of InputBuffer and OutputBuffer refer r_jcua_DivisionBufferInfo_t

If IsEnable is R_TRUE, JCUA H/W macro processes the amount of data specified by DataCount, after which callback function registered by R_JCUA_DecoderOpen function is called. The Factor argument of the callback function is set to parameter R_JCUA_CB_FACTOR_DECODE_INPUT_PAUSED or R_JCUA_CB_FACTOR_DECODE_OUTPUT_PAUSED, indicating completion of input or output of the amount of data specified by DataCount and that JCUA H/W macro is in a paused state. After updating the buffers and other required tasks, call the R_JCUA_DecoderContinue function to resume the decoding. If IsEnable is R_FALSE, the callback above does not occur.

Regardless of IsEnable setting, the same callback function is called after all decoding is completed. Factor argument of the callback function is set to the value R_JCUA_CB_FACTOR_DECODE_COMPLETE to indicate that decoding has been completed.

If decoding is completed, Factor argument for the callback function is set to the value R_JCUA_CB_FACTOR_DECODE_COMPLETE or logical sum.

### 3.2.6 Decoding errors

Errors in the decoding of JPEG compressed data are due to following three factors.

1. The image being decoded takes up more memory than the frame buffer.
2. The image being decoded has an incompatible format or is incorrect data.
3. An error that was not anticipated occurs.

If these errors occur, a callback function r_jcua_CallbackFunction_t is called.

In case 1, Factor argument of the callback function is set to the value R_JCUA_CB_FACTOR_DECODE_SIZEOVER, indicating that frame buffer was too small. If this error occurs, re-execute the decoding with correction of parameters such as enlarging the frame buffer.

In case 2, Factor argument of the callback function is set to the value R_JCUA_CB_FACTOR_DECODE_ERRORED, indicating that decoding was not correctly completed. If this error occurs, decoding of the JPEG-compressed data is not possible.

In case 3, Factor argument of the callback function is set to the value R_JCUA_CB_FACTOR_FATAL_ERROR, indicating that an error which was not anticipated has occurred. In this state, the JCUA driver becomes incapable of correct operation. Reset the RH850/D1x device.

## 3.3 Device difference

The following table shows the function differences depending on the device.

**Table 3-3 APIs supported by JCUA driver**

| Function | RH850/D1x Device Name | |
|---|---|---|
| | D1L2(H) | D1M1(H), D1M1-V2, D1M1A, D1M2(H) |
| All API of JCUA driver | No | Full |

## 3.4 Header File List

**Table 3-4 Header File List**

| No. | Header File Name | Description |
|---|---|---|
| (1) | r_jcua_api.h | Header file for JCUA API |
| (2) | r_typedefs.h | Header file for predefined data types |

# 4.Functions

## 4.1 Function List

This section shows the JCUA API functions in *Table 4-1* and executable state of each function is described in the specification of each function.

**Table 4-1 List of JCUA API Functions**

| Function Name | Purpose |
|---|---|
| *R_JCUA_Init* | This function initializes the JCUA driver. |
| *R_JCUA_DeInit* | This function de-initializes the JCUA driver. |
| *R_JCUA_DecoderOpen* | This function opens the JCUA driver by specified operation mode and registers the callback function. |
| *R_JCUA_DecoderClose* | This function closes the JCUA driver by specified operation mode and unregisters the callback function. |
| *R_JCUA_DecoderStart* | This function begins decoding the compressed JPEG data. |
| *R_JCUA_DecoderContinue* | This function continues the decode processing. |
| *R_JCUA_ErrorInfoGet* | This function gets the error information. |
| *R_JCUA_VersionStringGet* | This function returns the version string of this JCUA driver. |
| *R_JCUA_MacroVersionGet* | This function returns the major and minor version of the macro. |
| *R_JCUA_IsrTimeOut* | This function is executed in response to the timeout interrupt. |
| *R_JCUA_IsrStop* | This function is executed in response to decompression processing interrupt (JEDI interrupt). |
| *R_JCUA_IsrFinish* | This function is executed in response to the data transfer processing interrupt (JDTI interrupt). |

## 4.2 JCUA API Functions

This chapter describes the application interface functions, which are required for general use of the driver, but which are related to a specific functionality of the macro itself.

### 4.2.1 R_JCUA_Init

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_Init(const uint32_t Unit)
```

**Input Parameter**

**Table 4-2 Input parameter of R_JCUA_Init**

| Parameter | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return Codes**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range. |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |
| R_JCUA_ERR_FATAL_OS | - Fatal error has occurred at OS interface. |
| R_JCUA_ERR_FATAL_HW | - Fatal error has occurred at H/W. |
| R_JCUA_ERR_BUS_TIMEOUT | - Timeout error has occurred at bus reset. |
| R_JCUA_ERR_TIMER_CTRL | - An error has occurred at Timer unit. |

**Description**

This function initializes the JCUA driver.
This function calls the system function R_JCUA_Sys_Init. The initialization code of environment-depend (e.g. interrupt priority, power control or clock control) is implemented by function R_JCUA_Sys_Init.
If function gets successfully executed, the return code will be R_JCUA_ERR_OK and the driver will be in Initialized state.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Synchronous.

**Call from Interrupt**

Prohibited.

**Precondition**

See ***Table 2-5*** about status conditions.

**See also**

r_jcua_Error_t

### 4.2.2 R_JCUA_DeInit

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_DeInit(const uint32_t Unit)
```

**Input Parameter**

**Table 4-3 Input parameter of R_JCUA_DeInit**

| Parameter In | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return code**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range. |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |
| R_JCUA_ERR_FATAL_OS | - Fatal error has occurred at OS interface. |
| R_JCUA_ERR_TIMER_CTRL | - An error has occurred at Timer unit. |

**Description**

This function de-initializes the JCUA driver.
This function calls system function R_JCUA_Sys_DeInit. R_JCUA_Sys_DeInit function clears priority of interrupt and conducts processing for system environment.
If the function gets successfully executed, return code will be R_JCUA_ERR_OK and driver will be in Uninitialized state.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Synchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

r_jcua_Error_t

### 4.2.3 R_JCUA_DecoderOpen

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_DecoderOpen(const uint32_t                  Unit,
                                  const r_jcua_CallbackFunction_t CallbackFunction,
                                  const uint32_t                  CallbackParam)
```

**Input Parameter**

**Table 4-4 Input parameter of R_JCUA_DecoderOpen**

| Parameter In | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |
| CallbackFunction | Specifies the pointer to callback function.<br>The callback function is called when decoding is completed, failed and paused. |
| CallbackParam | Specifies the value which is passed to callback function. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_PARAM_INCORRECT | - An error has occurred when CallbackFunction is R_NULL. |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range. |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |
| R_JCUA_ERR_FATAL_OS | - Fatal error has occurred at OS interface. |

**Description**

This function opens JCUA driver by decompression mode and registers callback function.
If function gets successfully executed, the return code will be R_JCUA_ERR_OK and the driver will be in Idle (decode-ready) state.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Synchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

r_jcua_Error_t
r_jcua_CallbackFunction_t
r_jcua_CallbackReason_t

### 4.2.4 R_JCUA_DecoderClose

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_DecoderClose(const uint32_t Unit)
```

**Input Parameter**

**Table 4-5 Input Parameter of R_JCUA_DecoderClose**

| Parameter In | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range. |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |
| R_JCUA_ERR_FATAL_OS | - Fatal error has occurred at OS interface. |

**Description**

This function closes driver and unregisters callback functions.
If the function gets successfully executed, the return code will be R_JCUA_ERR_OK and the driver will be in Initialized state.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Synchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

r_jcua_Error_t

### 4.2.5 R_JCUA_DecoderStart

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_DecoderStart(const uint32_t                   Unit,
                                   const uint8_t* const             SourceAddress,
                                   const r_jcua_FrameBuffer_t* const FrameBuffer,
                                   r_jcua_DecodeSetting_t* const    Decode)
```

**Input Parameter**

**Table 4-6 Input parameter of R_JCUA_DecoderStart**

| Parameter In | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |
| SourceAddress | Specifies the address of compressed JPEG data.<br>This address must be a multiple of 8 bytes. R_NULL is error. |
| FrameBuffer | Specifies the address of frame buffer information. |

**Input-Output Parameter**

**Table 4-7 Input - Output parameter of R_JCUA_DecoderStart**

| Parameter In-Out | Description |
|---|---|
| Decode | Decode setting parameters.<br>Following members are input parameters.<br>Decode->OptionFlag<br>Decode->Position<br>Decode->Parameter<br>Decode->DivisionMode<br>Following member is output parameters.<br>Decode->ImgInfo |

**Output Parameter**

None

**Return codes**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_PARAM_INCORRECT | - An error has occurred when value passed to argument is incorrect |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |
| R_JCUA_ERR_FATAL_OS | - Fatal error has occurred at OS interface. |
| R_JCUA_ERR_TIMER_CTRL | - An error has occurred at Timer unit. |

**Description**

This function begins decoding the compressed JPEG data. This function sets decode parameters and starts decode processing. If the function gets successfully executed, return code will be R_JCUA_ERR_OK and driver will be in the Executing (decoding) state.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Asynchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See ***Table 2-5*** about status conditions.

**See also**

r_jcua_Error_t
r_jcua_FrameBuffer_t
r_jcua_DecodeSetting_t

### 4.2.6 R_JCUA_DecoderContinue

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_DecoderContinue(const uint32_t              Unit,
                                      const r_jcua_PauseReason_t  Factor)
```

**Input Parameter**

**Table 4-8 Input parameter of R_JCUA_DecoderContinue**

| Parameter In | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |
| Factor | Specifies the pause reason.<br>R_JCUA_PAUSE_FACTOR_INPUT_PAUSED<br>R_JCUA_PAUSE_FACTOR_OUTPUT_PAUSED |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_NG | - An error has occurred, but no specific error code is defined for it. |
| R_JCUA_ERR_PARAM_INCORRECT | - An error has occurred when value passed to argument is invalid. |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range. |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |
| R_JCUA_ERR_FATAL_OS | - Fatal error has occurred at OS interface. |
| R_JCUA_ERR_TIMER_CTRL | - An error has occurred at Timer unit. |

**Description**

This function resumes the decode processing.
The JCUA driver enters to Executing (decode-paused) state by input division or output division after JCUA driver performed a decoding. By calling this function, driver can resume the decode processing which in paused state.

If function gets successfully executed, the return code will be R_JCUA_ERR_OK and the driver will be in Executing (decoding) state.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Asynchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

r_jcua_Error_t
r_jcua_PauseReason_t

### 4.2.7 R_JCUA_ErrorInfoGet

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_ErrorInfoGet(const uint32_t              Unit,
                                   r_jcua_ErrorDetail_t *const ErrorCode)
```

**Input Parameter**

**Table 4-9 Input parameter of R_JCUA_ErrorInfoGet**

| Parameter In | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

**Table 4-10 Output parameter of R_JCUA_ErrorInfoGet**

| Parameter Out | Description |
|---|---|
| ErrorCode | Specifies the error information. |

**Return codes**

| | |
|---|---|
| R_JCUA_ERR_OK | - No error has occurred. |
| R_JCUA_ERR_PARAM_INCORRECT | - An error has occurred when value passed to argument is incorrect. |
| R_JCUA_ERR_RANGE_UNIT | - Unit number was outside the range |
| R_JCUA_ERR_NOT_ACCEPTABLE | - An error has occurred when function called in an incorrect state. |

**Description:**

This function gets the detail error information.
When an error occurred after call the R_JCUA_DecoderStart function or R_JCUA_DecoderContinue function, it's possible to confirm error cause.
If function gets successfully executed, the return code will be R_JCUA_ERR_OK.

**Reentrancy**

Non-reentrant.
If user implements following functions to prevent multiple executions, this function will become re-entrant.

- R_JCUA_Sys_Lock
- R_JCUA_Sys_Unlock

**Sync/Async**

Synchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

r_jcua_Error_t
r_jcua_ErrorDetail_t

### 4.2.8 R_JCUA_VersionStringGet

**Function Prototypes**

```
const uint8_t* R_JCUA_VersionStringGet(void)
```

**Input Parameter**

None

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

Pointer of string.

**Description**

This function returns the version string of this JCUA driver.

**Reentrancy**

Reentrant

**Sync/Async**

Synchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See ***Table 2-5*** about status conditions.

**See also**

None

### 4.2.9 R_JCUA_MacroVersionGet

**Function Prototypes**

```
r_jcua_Error_t R_JCUA_MacroVersionGet(uint32_t *const Major,
                                      uint32_t *const Minor)
```

**Input Parameter**

None

**Input-Output Parameter**

None

**Output Parameter**

**Table 4-11 Input parameter of R_JCUA_MacroVersionGet**

| Parameter In | Description |
|---|---|
| Major | Specifies the major version. |
| Minor | Specifies the minor version. |

**Return codes**

R_JCUA_ERR_OK - No error has occurred.
R_JCUA_ERR_PARAM_INCORRECT - An error has occurred when value passed to argument is incorrect.

**Description**

This function returns the major and minor version of H/W macro.

**Reentrancy**

Reentrant

**Sync/Async**

Synchronous

**Call from Interrupt**

Prohibited.

**Precondition**

See ***Table 2-5*** about status conditions.

**See also**

r_jcua_Error_t

### 4.2.10 R_JCUA_IsrTimeOut

**Function Prototypes**

```
void R_JCUA_IsrTimeOut(const uint32_t Unit)
```

**Input Parameter**

**Table 4-12 Input parameter of R_JCUA_IsrTimeOut**

| Parameter in | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

None

**Description**

This function terminates decoding because of a timeout.
This function is called from JCUA porting layer. As the default implementation, this function is called from ISR of OSTM1TINT interrupt. See Porting layer guide for the detail.
If timeout occurs, driver will be in the Idle (decode-ready) state.

**Reentrancy**

Non-reentrant.

**Sync/Async**

Synchronous

**Call from Interrupt**

Permitted.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

None

## 4.3 Interrupt Functions

### 4.3.1 R_JCUA_IsrStop

**Function Prototypes**

```
void R_JCUA_IsrStop(const uint32_t  Unit)
```

**Input Parameter**

**Table 4-13 Input parameter of R_JCUA_IsrStop**

| Parameter | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

None

**Description**

This function is executed in response to the data transfer processing interrupt (JEDI interrupt).
This function will be called from the JEDI interrupt handler.
Regarding JEDI interrupt, refer H/W user's manual.

**Reentrancy**

Non-reentrant.

**Sync/Async**

Synchronous

**Call from Interrupt**

Permitted.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

None

### 4.3.2 R_JCUA_IsrFinish

**Function Prototypes**

```
void R_JCUA_IsrFinish(const uint32_t  Unit)
```

**Input Parameter**

**Table 4-14 Input parameter of R_JCUA_IsrFinish**

| Parameter | Description |
|---|---|
| Unit | Specifies the JCUA Unit number. |

**Input-Output Parameter**

None

**Output Parameter**

None

**Return codes**

None

**Description**

This function is executed in response to the data transfer processing interrupt (JDTI interrupt).
This function will be called from the JDTI interrupt handler.
Regarding JDTI interrupt, refer H/W user's manual.

**Reentrancy**

Non-reentrant.

**Sync/Async**

Synchronous

**Call from Interrupt**

Permitted.

**Precondition**

See *Table 2-5* about status conditions.

**See also**

None

# 5.Types

## 5.1 Basic Types

This section shows the basic types used in this library.

**Table 5-1 Basic Types**

| Types | Definition | | Basic types |
|---|---|---|---|
| char_t | typedef char | char_t | signed char |
| int8_t | typedef signed char | int8_t | signed char |
| int16_t | typedef signed short | int16_t | signed short |
| int32_t | typedef signed int | int32_t | signed int |
| int64_t | typedef signed long long | int64_t | signed long long |
| uint8_t | typedef unsigned char | uint8_t | unsigned char |
| uint16_t | typedef unsigned short | uint16_t | unsigned short |
| uint32_t | typedef unsigned int | uint32_t | unsigned int |
| uint64_t | typedef unsigned long long | uint64_t | unsigned long long |
| float32_t | typedef float | float32_t | float |
| float64_t | typedef double | float64_t | double |

## 5.2 JCUA API Types

This section shows the JCUA API function types used in this library

### 5.2.1 r_jcua_CallbackFunction_t

**Description**

This type describes the callback function.

**Definition**

```
typedef void (*r_jcua_CallbackFunction_t)(uint32_t                Unit,
                                          r_jcua_CallbackReason_t Factor,
                                          uint32_t                Param)
```

**Table 5-2 Parameter of r_jcua_CallbackFunction_t**

| Parameter | Description |
|---|---|
| Unit | It shows the JCUA Unit number. |
| Factor | It shows the interrupt factor. |
| Param | It shows CallbackParam which is specified the R_JCUA_DecoderOpen function. |

**See also**

r_jcua_CallbackReason_t
R_JCUA_DecoderOpen

## 5.3 Definition

This section shows the definition values used in JCUA API.

### 5.3.1 API version

**Description**

This constant is the value which shows version information of JCUA driver.

**Table 5-3 Definition of JCUA API version**

| Name | Description |
|---|---|
| R_JCUA_VERSION_HI | MSB byte of the version information. It is major version information. This value is changed with release version. |
| R_JCUA_VERSION_LO | LSB byte of the version information. It is minor version information. This value is changed with release version. |

**See Also**

None

### 5.3.2 Decode options

**Description**

This type describes the validity of optional decode functions. Value of logical sum is used.

**Table 5-4 Decode options**

| Name | Values | Description |
|---|---|---|
| R_JCUA_DECODE_OPTION_NONE | 0x00uL | All optional decode functions are ineffective. |
| R_JCUA_DECODE_OPTION_POSITION | 0x01uL | Coordinate designation is effective. |
| R_JCUA_DECODE_OPTION_PARAM | 0x02uL | Additional decoding function is effective. |
| R_JCUA_DECODE_OPTION_DIVISION_MODE | 0x04uL | Division decoding function is effective. |

**See Also**

r_jcua_Coordinate_t
r_jcua_DecodeParam_t
r_jcua_DivisionModeParam_t

### 5.3.3 Stride calculation

**Description**

If decoding proceeds while setting Stride member of the r_jcua_FrameBuffer_t structure is R_JCUA_STRIDE_AUTO (=0), JCUA driver automatically calculates the size of the stride which is required for output of the image (128 bytes unit).

**Table 5-5 Stride calculation**

| Name | Value | Description |
|---|---|---|
| R_JCUA_STRIDE_AUTO | 0 | Stride is calculated automatically. |

**See Also**

r_jcua_FrameBuffer_t

## 5.4 Enumerated Type

This section shows the enumerated types used in JCUA API Functions.

### 5.4.1 r_jcua_Error_t

**Description**

This type describes the error code of JCUA driver functions.

**Definition**

```
typedef enum
{
    R_JCUA_ERR_OK                   = 0x00,
    R_JCUA_ERR_NG                   = 0x01,
    R_JCUA_ERR_PARAM_INCORRECT      = 0x02,
    R_JCUA_ERR_RANGE_UNIT           = 0x03,
    R_JCUA_ERR_RANGE_PARAM          = 0x04,
    R_JCUA_ERR_NOT_ACCEPTABLE       = 0x05,
    R_JCUA_ERR_FATAL_OS             = 0x06,
    R_JCUA_ERR_FATAL_HW             = 0x07,
    R_JCUA_ERR_BUS_TIMEOUT          = 0x08,
    R_JCUA_ERR_TIMER_CTRL           = 0x09
} r_jcua_Error_t
```

**Table 5-6 Enumerator of r_jcua_Error_t**

| Name | Description |
|---|---|
| R_JCUA_ERR_OK | No error has occurred. |
| R_JCUA_ERR_NG | An error has occurred, but no specific error code is defined for it. |
| R_JCUA_ERR_PARAM_INCORRECT | A parameter provided to a function is incorrect. |
| R_JCUA_ERR_RANGE_UNIT | A JCUA Unit number is outside the range. |
| R_JCUA_ERR_RANGE_PARAM | A parameter is outside the range. |
| R_JCUA_ERR_NOT_ACCEPTABLE | An error has occurred when function called from an incorrect state. |
| R_JCUA_ERR_FATAL_OS | Fatal error has occurred at OS interface. |
| R_JCUA_ERR_FATAL_HW | Fatal error has occurred at H/W. |
| R_JCUA_ERR_BUS_TIMEOUT | Timeout error has occurred at bus reset. |
| R_JCUA_ERR_TIMER_CTRL | An error has occurred at Timer unit. |

**See also**

None

### 5.4.2 r_jcua_CallbackReason_t

**Description**

This type describes the callback reason. Value of logical sum is used.

**Definition**

```
typedef enum
{
    R_JCUA_CB_FACTOR_NONE                   = 0x00000000,
    R_JCUA_CB_FACTOR_DECODE_COMPLETE        = 0x00000001,
    R_JCUA_CB_FACTOR_DECODE_INPUT_PAUSED    = 0x00000002,
    R_JCUA_CB_FACTOR_DECODE_OUTPUT_PAUSED   = 0x00000004,
    R_JCUA_CB_FACTOR_DECODE_SIZEOVER        = 0x01000008,
    R_JCUA_CB_FACTOR_DECODE_ERRORED         = 0x01000010,
    R_JCUA_CB_FACTOR_FATAL_ERRORED          = 0x01000020,
    R_JCUA_CB_FACTOR_HEADER_TIMEOUT         = 0x01000040,
    R_JCUA_CB_FACTOR_DECODE_TIMEOUT         = 0x01000080
} r_jcua_CallbackReason_t
```

**Table 5-7 Enumerator of r_jcua_CallbackReason_t**

| Name | Description |
|---|---|
| R_JCUA_CB_FACTOR_NONE | None. |
| R_JCUA_CB_FACTOR_DECODE_COMPLETE | Decode processing completed. |
| R_JCUA_CB_FACTOR_DECODE_INPUT_PAUSED | Decode processing paused by input side division decoding. |
| R_JCUA_CB_FACTOR_DECODE_OUTPUT_PAUSED | Decode processing paused by output side division decoding. |
| R_JCUA_CB_FACTOR_DECODE_SIZEOVER | Decode processing failed (size of the compressed JPEG data is too large). |
| R_JCUA_CB_FACTOR_DECODE_ERRORED | Decode processing was failed (By the interrupt which decode error occurred). |
| R_JCUA_CB_FACTOR_FATAL_ERROR | Fatal error occurred. |
| R_JCUA_CB_FACTOR_HEADER_TIMEOUT | Decode processing was failed (SOI or SOS not detected until timeout occurs). |
| R_JCUA_CB_FACTOR_DECODE_TIMEOUT | Decode processing was failed (EOI not detected until timeout occurs). |

**See also**

None

### 5.4.3 r_jcua_PauseReason_t

**Description**

This type describes the pause reason.

**Definition**

```
typedef enum
{
    R_JCUA_PAUSE_FACTOR_INPUT_PAUSED  = 0x00000001,
    R_JCUA_PAUSE_FACTOR_OUTPUT_PAUSED = 0x00000002
} r_jcua_PauseReason_t
```

**Table 5-8 Enumerator of r_jcua_PauseReason_t**

| Name | Description |
|---|---|
| R_JCUA_PAUSE_FACTOR_INPUT_PAUSED | Decode processing paused (By the interrupt which input side division decoding). |
| R_JCUA_PAUSE_FACTOR_OUTPUT_PAUSED | Decode processing paused (By the interrupt which output side division decoding). |

**See also**

None

### 5.4.4 r_jcua_JpegFormat_t

**Description**

This type describes the pixel format of compressed JPEG data that is decoded.

**Definition**

```
typedef enum
{
    R_JCUA_JPEG_FORMAT_YCBCR444     = 0x00,
    R_JCUA_JPEG_FORMAT_YCBCR422     = 0x01,
    R_JCUA_JPEG_FORMAT_YCBCR420     = 0x02,
    R_JCUA_JPEG_FORMAT_YCBCR411     = 0x06
} r_jcua_JpegFormat_t
```

**Table 5-9 Enumerator of r_jcua_JpegFormat_t**

| Name | Description |
|---|---|
| R_JCUA_JPEG_FORMAT_YCBCR444 | YCbCr444 |
| R_JCUA_JPEG_FORMAT_YCBCR422 | YCbCr422 |
| R_JCUA_JPEG_FORMAT_YCBCR420 | YCbCr420 |
| R_JCUA_JPEG_FORMAT_YCBCR411 | YCbCr411 |

**See also**

None

### 5.4.5 r_jcua_OutputFormat_t

**Description**

This type describes the output pixel format of output image data.

**Definition**

```
typedef enum
{
    R_JCUA_OUTPUT_FORMAT_YCBCR422        = 0x00,
    R_JCUA_OUTPUT_FORMAT_ARGB8888        = 0x01,
    R_JCUA_OUTPUT_FORMAT_RGB565          = 0x02
} r_jcua_OutputFormat_t
```

**Table 5-10 Enumerator of r_jcua_OutputFormat_t**

| Name | Description |
|---|---|
| R_JCUA_OUTPUT_FORMAT_YCBCR422 | YCbCr422 |
| R_JCUA_OUTPUT_FORMAT_ARGB8888 | ARGB8888 |
| R_JCUA_OUTPUT_FORMAT_RGB565 | RGB565 |

**See also**

None

### 5.4.6 r_jcua_SubSampling_t

**Description**

This type describes the sub samples of the decoded image data.

**Definition**

```
typedef enum
{
    R_JCUA_SUB_SAMPLING_1_1   = 0x00,
    R_JCUA_SUB_SAMPLING_1_2   = 0x01,
    R_JCUA_SUB_SAMPLING_1_4   = 0x02,
    R_JCUA_SUB_SAMPLING_1_8   = 0x03
} r_jcua_SubSampling_t
```

**Table 5-11 Enumerator of r_jcua_SubSampling_t**

| Name | Description |
|---|---|
| R_JCUA_SUB_SAMPLING_1_1 | No subsampling. |
| R_JCUA_SUB_SAMPLING_1_2 | Subsamples output data into 1/2. |
| R_JCUA_SUB_SAMPLING_1_4 | Subsamples output data into 1/4. |
| R_JCUA_SUB_SAMPLING_1_8 | Subsamples output data into 1/8. |

**See also**

None

### 5.4.7 r_jcua_ErrorDetail_t

**Description**

This type describes the error classification of the JCUA driver decoding failure.

**Definition**

```
typedef enum
{
    R_JCUA_ERR_JDC_OK                       = 0x00,
    R_JCUA_ERR_JDC_SOI_NOT_FOUND            = 0x01,
    R_JCUA_ERR_JDC_INVALID_SOF              = 0x02,
    R_JCUA_ERR_JDC_UNPROVIDED_SOF           = 0x03,
    R_JCUA_ERR_JDC_SOF_ACCURACY             = 0x04,
    R_JCUA_ERR_JDC_DQT_ACCURACY             = 0x05,
    R_JCUA_ERR_JDC_COMPONENT_1              = 0x06,
    R_JCUA_ERR_JDC_COMPONENT_2              = 0x07,
    R_JCUA_ERR_JDC_NO_SOF0_DQT_DHT          = 0x08,
    R_JCUA_ERR_JDC_SOS_NOT_FOUND            = 0x09,
    R_JCUA_ERR_JDC_EOI_NOT_FOUND            = 0x0A,
    R_JCUA_ERR_JDC_RESTART_INTERVAL_NUM     = 0x0B,
    R_JCUA_ERR_JDC_IMAGE_SIZE               = 0x0C,
    R_JCUA_ERR_JDC_LAST_MCU_NUM             = 0x0D,
    R_JCUA_ERR_JDC_BLOCK_NUM                = 0x0E,
    R_JCUA_ERR_JDC_UNKNOWN_FATAL_ERROR      = 0x0F
} r_jcua_ErrorDetail_t
```

**Table 5-12 Enumerator of r_jcua_ErrorDetail_t**

| Name | Description |
|---|---|
| R_JCUA_ERR_JDC_OK | No error has occurred. |
| R_JCUA_ERR_JDC_SOI_NOT_FOUND | SOI not detected: SOI not detected until EOI detected. |
| R_JCUA_ERR_JDC_INVALID_SOF | SOF1 to SOFF detected. |
| R_JCUA_ERR_JDC_UNPROVIDED_SOF | Un provided pixel format detected. |
| R_JCUA_ERR_JDC_SOF_ACCURACY | SOF accuracy error: Other than 8 detected. |
| R_JCUA_ERR_JDC_DQT_ACCURACY | DQT accuracy error: Other than 0 detected. |
| R_JCUA_ERR_JDC_COMPONENT_1 | The number of SOF0 header components detected is other than 1, 3, or 4. |
| R_JCUA_ERR_JDC_COMPONENT_2 | The number of components differs between SOF0 header and SOS. |
| R_JCUA_ERR_JDC_NO_SOF0_DQT_DHT | SOF0, DQT, and DHT not detected when SOS detected. |
| R_JCUA_ERR_JDC_SOS_NOT_FOUND | SOS not detected: SOS not detected until EOI detected. |
| R_JCUA_ERR_JDC_EOI_NOT_FOUND | EOI not detected (default). |
| R_JCUA_ERR_JDC_RESTART_INTERVAL_NUM | Restart interval data number error detected. |
| R_JCUA_ERR_JDC_IMAGE_SIZE | Image size error detected.<br>There is a possibility that the stride value which is set to Stride member of r_jcua_FrameBuffer_t structure is wrong, or the size of frame buffer is smaller than decoding image size.<br>If this error occurs, sets a right value to Stride member of r_jcua_FrameBuffer_t structure or re-allocate the frame buffer, then calls the R_JCUA_DecoderStart function. |
| R_JCUA_ERR_JDC_LAST_MCU_NUM | Last MCU data number error detected. |
| R_JCUA_ERR_JDC_BLOCK_NUM | Block data number error detected. |
| R_JCUA_ERR_JDC_UNKNOWN_FATAL_ERROR | Fatal error occurred.<br>There is a possibility that a memory is broken, or the LSI is hung-up. If this error occurs, reset the LSI. |

**See also**

R_JCUA_ErrorInfoGet
r_jcua_FrameBuffer_t
R_JCUA_DecoderStart

### 5.4.8 r_jcua_Swap_t

**Description**

This type describes the data swap setting in memory. The data is swapped every 8 bytes. The data image in memory assume "(1)-(2)-(3)-(4)-(5)-(6)-(7)-(8)" before swapping data.

**Definition**

```
typedef enum
{
    R_JCUA_SWAP_NONE                    = 0x00,
    R_JCUA_SWAP_BYTE                    = 0x01,
    R_JCUA_SWAP_WORD                    = 0x02,
    R_JCUA_SWAP_WORD_AND_BYTE           = 0x03,
    R_JCUA_SWAP_LONG                    = 0x04,
    R_JCUA_SWAP_LONG_AND_BYTE           = 0x05,
    R_JCUA_SWAP_LONG_AND_WORD           = 0x06,
    R_JCUA_SWAP_LONG_AND_WORD_AND_BYTE  = 0x07
} r_jcua_Swap_t
```

**Table 5-13 Enumerator of r_jcua_Swap_t**

| Name | Description |
|---|---|
| R_JCUA_SWAP_NONE | No swap. (1)-(2)-(3)-(4)-(5)-(6)-(7)-(8) |
| R_JCUA_SWAP_BYTE | Byte swap. (2)-(1)-(4)-(3)-(6)-(5)-(8)-(7) |
| R_JCUA_SWAP_WORD | Word swap. (3)-(4)-(1)-(2)-(7)-(8)-(5)-(6) |
| R_JCUA_SWAP_WORD_AND_BYTE | Word-byte swap. (4)-(3)-(2)-(1)-(8)-(7)-(6)-(5) |
| R_JCUA_SWAP_LONG | Long word swap. (5)-(6)-(7)-(8)-(1)-(2)-(3)-(4) |
| R_JCUA_SWAP_LONG_AND_BYTE | Long word-byte swap. (6)-(5)-(8)-(7)-(2)-(1)-(4)-(3) |
| R_JCUA_SWAP_LONG_AND_WORD | Long word-word swap. (7)-(8)-(5)-(6)-(3)-(4)-(1)-(2) |
| R_JCUA_SWAP_LONG_AND_WORD_AND_BYTE | Long word-word-byte swap. (8)-(7)-(6)-(5)-(4)-(3)-(2)-(1) |

**See also**

None

## 5.5 Structure

This section shows the structure used in JCUA API function.

### 5.5.1 r_jcua_Coordinate_t

**Description**

This type describes the position.

**Definition**

```
typedef struct
{
    uint32_t      PosX;
    uint32_t      PosY;
} r_jcua_Coordinate_t
```

**Table 5-14 Member of r_jcua_Coordinate_t structure**

| Member | Description |
|---|---|
| PosX | Horizontal position. Represented in pixel unit. Default is 0.<br>It should be 8 bytes aligned.<br>If the output format is YCbCr422 or RGB565, then PosX must be a multiple of 4 pixels.<br>If the output format is ARGB8888, then PosX must be even number. |
| PosY | Vertical position. Represented in pixel unit. Default is 0. |

**See also**

None

### 5.5.2 r_jcua_FrameBuffer_t

**Description**

This type describes the frame buffer information that is an output destination of data which decoded compressed JPEG data.

**Definition**

```
typedef struct
{
    void*                   Address;
    uint32_t                Size;
    int16_t                 Stride;
    r_jcua_OutputFormat_t   Format;
    r_jcua_Swap_t           Swap;
} r_jcua_FrameBuffer_t
```

**Table 5-15 Member of r_jcua_FrameBuffer_t structure**

| Member | Description |
|---|---|
| Address | Address of frame buffer.<br>This address must be a multiple of 8 bytes.<br>Note: Specifying Local RAM area is prohibited. |
| Size | Size of frame buffer in bytes.<br>This size must be a multiple of 8 bytes.<br>This value must be bigger than the complete decoded image size, also if output division mode is used. |
| Stride | Stride of frame buffer in bytes.<br>This stride must be a multiple of 8 bytes.<br>If the stride sets to R_JCUA_STRIDE_AUTO, JCUA driver calculates the stride which is just fits for VDCE. (multiple of 128 bytes) |
| Format | Output pixel format of frame buffer.<br>R_JCUA_OUTPUT_FORMAT_YCBCR422<br>R_JCUA_OUTPUT_FORMAT_ARGB8888<br>R_JCUA_OUTPUT_FORMAT_RGB565 |
| Swap | Data swap setting in memory.<br>R_JCUA_SWAP_NONE<br>R_JCUA_SWAP_BYTE<br>R_JCUA_SWAP_WORD<br>R_JCUA_SWAP_WORD_AND_BYTE<br>R_JCUA_SWAP_LONG<br>R_JCUA_SWAP_LONG_AND_BYTE<br>R_JCUA_SWAP_LONG_AND_WORD<br>R_JCUA_SWAP_LONG_AND_WORD_AND_BYTE |

**See also**

r_jcua_Swap_t
r_jcua_OutputFormat_t

### 5.5.3 r_jcua_DecodeParam_t

**Description**

This type describes decode with subsample parameter information.

**Definition**

```
typedef struct
{
    r_jcua_SubSampling_t   VertlSubSampling;
    r_jcua_SubSampling_t   HorizSubSampling;
    uint8_t                Alpha;
} r_jcua_DecodeParam_t
```

**Table 5-16 Member of r_jcua_DecodeParam_t structure**

| Member | Description |
|---|---|
| VertlSubSampling | Horizontal Subsampling.<br>R_JCUA_SUB_SAMPLING_1_1 (default)<br>R_JCUA_SUB_SAMPLING_1_2<br>R_JCUA_SUB_SAMPLING_1_4<br>R_JCUA_SUB_SAMPLING_1_8 |
| HorizSubSampling | Vertical Subsampling.<br>R_JCUA_SUB_SAMPLING_1_1 (default)<br>R_JCUA_SUB_SAMPLING_1_2<br>R_JCUA_SUB_SAMPLING_1_4<br>R_JCUA_SUB_SAMPLING_1_8 |
| Alpha | Alpha value (0 to 255). Default is 255.<br>If the output pixel format of output image data isn't ARGB8888, the alpha value has to be zero. Alpha value specifies the per-pixel transparency. |

**See also**

r_jcua_SubSampling_t

### 5.5.4 r_jcua_DivisionBufferInfo_t

**Description**

This type describes the division decoding buffer information.

**Definition**

```
typedef struct
{
    unt8_t        IsEnable;
    unt8_t        IsInitAddress;
    uint32_t      *RestartAddress;
    uint32_t      DataCount;
} r_jcua_DivisionBufferInfo_t
```

**Table 5-17 Member of r_jcua_DivisionBufferInfo_t structure**

| Member | Description |
|---|---|
| IsEnable | R_FALSE: Disable the input/output buffer division decoding. (default)<br>R_TRUE: Enable the input/output buffer division decoding. Following parameters are valid when IsEnable is R_TRUE. |
| IsInitAddress | R_FALSE: When decoding paused, the input/output buffer address isn't initialized.<br>R_TRUE: When decoding paused, the input/output buffer address is initialized by RestartAddress. |
| RestartAddress | If IsInitAddress is R_TRUE, the input/output buffer address is initialized by this RestartAddress. |
| DataCount | Division decoding count in bytes. Ranges from 8 to 65528.<br>For input buffer, the DataCount must be a multiple of 8 and its unit is byte.<br>For output buffer, the DataCount must be a multiple of 16 and its unit is line. |

**See also**

None

### 5.5.5 r_jcua_DivisionModeParam_t

**Description**

This type describes the division decoding parameter information.

**Definition**

```
typedef struct
{
    r_jcua_DivisionBufferInfo_t  InputBuffer;
    r_jcua_DivisionBufferInfo_t  OutputBuffer;
} r_jcua_DivisionModeParam_t
```

**Table 5-18 Member of r_jcua_DivisionModeParam_t structure**

| Member | Description |
|---|---|
| InputBuffer | Input buffer information of division decoding. |
| OutputBuffer | Output buffer information of division decoding. |

**See also**

None

### 5.5.6 r_jcua_ImageInfo_t

**Description**

This type describes the decoded JPEG image information

**Definition**

```
typedef struct
{
    uint32_t               Width;
    uint32_t               Height;
    uint32_t               DecodeWidth;
    uint32_t               DecodeHeight;
    uint32_t               Stride;
    uint32_t               StridePixel;
    r_jcua_JpegFormat_t    EncodedFormat;
} r_jcua_ImageInfo_t
```

**Table 5-19 Member of r_jcua_ImageInfo_t structure**

| Member | Description |
|---|---|
| Width | Width (pixel) of the original compressed JPEG data is stored. |
| Height | Height (pixel) of the original compressed JPEG data is stored. |
| DecodeWidth | Width of the decoded RAW image is stored.<br>The value is multiple of MCU. (e.g. MCU size of YCbCr422 is 16x8). |
| DecodeHeight | Height of the decoded RAW image is stored.<br>The value is multiple of MCU. |
| Stride | Stride (byte) of the frame buffer. |
| StridePixel | Stride (pixel) of the frame buffer. |
| EncodedFormat | Pixel format of compressed JPEG data.<br>R_JCUA_JPEG_FORMAT_YCBCR444<br>R_JCUA_JPEG_FORMAT_YCBCR422<br>R_JCUA_JPEG_FORMAT_YCBCR420<br>R_JCUA_JPEG_FORMAT_YCBCR411 |

**See also**

r_jcua_JpegFormat_t

### 5.5.7 r_jcua_DecodeSetting_t

**Description**

This type describes the decode parameter information.

**Definition**

```
typedef struct
{
    uint32_t                    OptionFlag;
    r_jcua_Coordinate_t         Position;
    r_jcua_DecodeParam_t        Parameter;
    r_jcua_DivisionModeParam_t  DivisionMode;
    r_jcua_ImageInfo_t *        ImgInfo;
} r_jcua_DecodeSetting_t
```

**Table 5-20 Member of r_jcua_DecodeSetting_t structure**

| Member | Description |
|---|---|
| OptionFlag | Flag indicates the validity of optional decode functions.<br>This flag can be set the value of logical sum of Decode options.<br>R_JCUA_DECODE_OPTION_NONE<br>R_JCUA_DECODE_OPTION_POSITION<br>R_JCUA_DECODE_OPTION_PARAM<br>R_JCUA_DECODE_OPTION_DIVISION_MODE |
| Position | The coordinate of output position information of decompressed JPEG data.<br>When decompressed JPEG data overflowed the frame buffer, an error has occurred.<br>This parameter is valid when R_JCUA_DECODE_OPTION_POSITION flag is on. |
| Parameter | Additional decoding function information.<br>This parameter is valid when R_JCUA_DECODE_OPTION_PARAM flag is on. |
| DivisionMode | Division decoding function information.<br>This parameter is valid when R_JCUA_DECODE_OPTION_DIVISION_MODE flag is on. |
| ImgInfo | The pointer to variable of compressed JPEG data information.<br>The entity of pointer must be kept until decode is finished.<br>If compressed JPEG data information is unnecessary, set R_NULL. |

**See also**

r_jcua_Coordinate_t
r_jcua_DecodeParam_t
r_jcua_DivisionModeParam_t
r_jcua_ImageInfo_t
Decode options
R_JCUA_DecoderStart

| Revision History | Renesas Graphics Library<br>JPEG Codec Unit A (JCUA) Driver<br><br>User's Manual: Software |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 0.1 | Nov 29, 2018 | - | First edition. |
| 0.2 | Mar 28, 2019 | 27 | Added the description to parameter. |
| | | 29, 35, 36, 37 | Added the "const" to argument.<br>R_JCUA_DecoderContinue<br>R_JCUA_IsrTimeOut<br>R_JCUA_IsrStop<br>R_JCUA_IsrFinish |
| | | 34, 42 | Fixed typo. |
| 1.0 | June 12, 2019 | 5, 6 | Improved description of Error handling |
| | | 28 | Removed description of Error handling |
| 2.0 | May 13, 2020 | 14, 53 | Add the restriction of Local RAM. |

Renesas Graphics Library
JPEG Codec Unit A (JCUA) Driver
User's Manual: Software
Publication Date: Rev.0.1 Nov 29, 2018
Rev.2.0 May 13, 2020

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

**Renesas Electronics America Inc. San Jose Campus**
6024 Silver Creek Valley Road, San Jose, CA 95138, USA
Tel: +1-408-284-8200, Fax: +1-408-284-2775

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

**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 101-T01, Floor 1, Building 7, Yard No. 7, 8th Street, Shangdi, Haidian District, Beijing 100085, 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, 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, #06-02 Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit No 3A-1 Level 3A Tower 8 UOA Business Park, No 1 Jalan Pengaturcara U1/51A, Seksyen U1, 40150 Shah Alam, Selangor, Malaysia
Tel: +60-3-5022-1288, Fax: +60-3-5022-1290

**Renesas Electronics India Pvt. Ltd.**
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700

**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 7.0

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

# Renesas Graphics Library
# JPEG Codec Unit A (JCUA) Driver

Renesas Electronics Corporation