RZ/A2M Group

JPEG Codec Unit "JCU" Driver Example

AN4456EJ0102
Rev.1.02
Dec.28, 2018

Introduction

This application note describes the sample driver which is decoded from the JPEG image data and encoded to the JPEG image data.

The JPEG Codec Unit(JCU) driver example offers the following features:

- The JPEG image data is converted to a raw image data of the RGB565, ARGB8888, and YCbCr422 formats.
- The raw image data of the YCbCr format is converted to a JPEG image data.

Target Device

RZ/A2M

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

Limitations

The count mode (division process) of must not be used. The mode must be conducted an extensive evaluation, if the mode is used.

Contents

1. Specifications

Table 1.1 Peripheral device used lists the Peripheral Functions and Their Applications, and Figure 1.1 Operation check conditions shows the Operation Overview.

Table 1.1 Peripheral device used

Peripheral device	Usage
JPEG Codec Unit(JCU)	Converts image data.
Interrupt controller(INTC)	The processor will receive interrupts when decoding or encoding is completed, failed, or paused.
Serial Communication Interface with FIFO(SCIF) Ch2	Output sample code message.

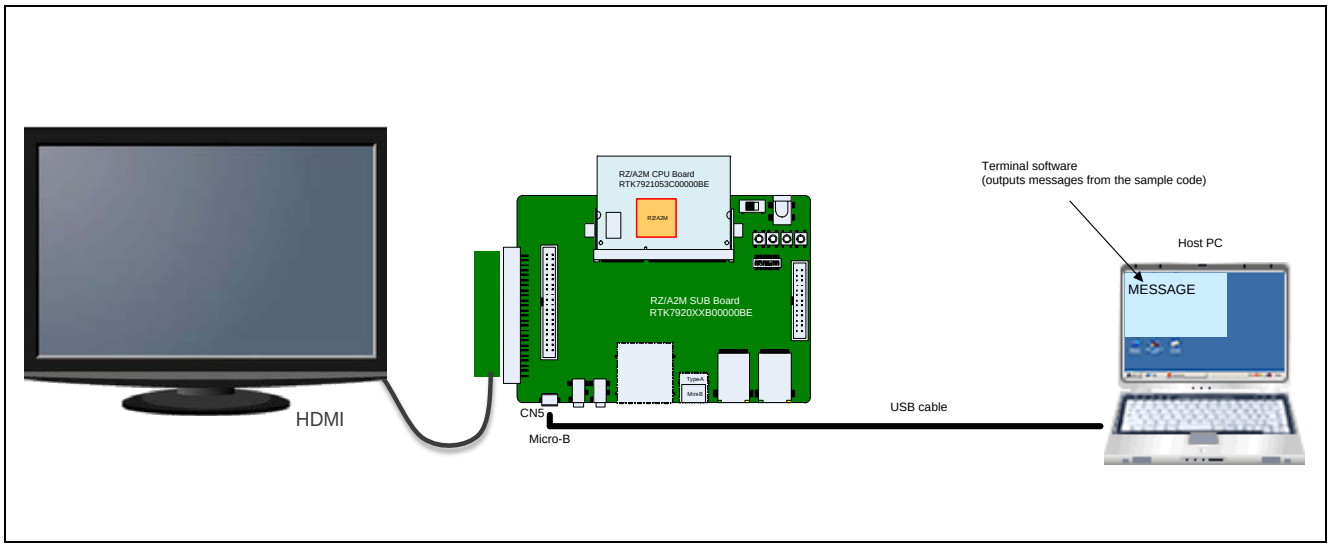


Figure 1.1 Operation check conditions

2. Operation Confirmation Conditions

Table 2.1 Operation Confirmation Conditions(1/2)

item	Contents
MCU used	RZ/A2M
Operating frequency (Note)	CPU Clock (Iϕ) : 528MHz Image processing clock (Gϕ) : 264MHz Internal Bus Clock (Bϕ) : 132MHz Peripheral Clock 1 (P1ϕ) : 66MHz Peripheral Clock 0 (P0ϕ) : 33MHz QSPI0_SPCLK : 66MHz CKIO : 132MHz
Operating voltage	Power supply voltage (I/O): 3.3 V Power supply voltage (either 1.8V or 3.3V I/O (PVcc SPI)) : 3.3V Power supply voltage (internal): 1.2 V
Integrated development environment	e2 studio V7.3.0
C compiler	"GNU Arm Embedded Tool chain 6-2017-q2-update" compiler options(except directory path) Release: -mcpu=cortex-a9 -march=armv7-a -marm -mthumb-interwork -mlittle-endian -mfloat-abi=hard -mfpu=neon -mno-unaligned-access -Os -ffunction-sections -fdata-sections -Wunused -Wuninitialized -Wall -Wextra -Wmissing-declarations -Wconversion -Wpointer-arith -Wpadded -Wshadow -Wlogical-op -Waggregate-return -Wfloat-equal -Wnull-dereference -Wmaybe-uninitialized -Wstack-usage=100 -fabi-version=0 Hardware Debug: -mcpu=cortex-a9 -march=armv7-a -marm -mthumb-interwork -mlittle-endian -mfloat-abi=hard -mfpu=neon -mno-unaligned-access -Og -ffunction-sections -fdata-sections -Wunused -Wuninitialized -Wall -Wextra -Wmissing-declarations -Wconversion -Wpointer-arith -Wpadded -Wshadow -Wlogical-op -Waggregate-return -Wfloat-equal -Wnull-dereference -Wmaybe-uninitialized -g3 -Wstack-usage=100 -fabi-version=0

Note: The operating frequency used in clock mode 1 (Clock input of 24MHz from EXTAL pin)

Table 2.2 Operation Confirmation Conditions(2/2)

Operation mode	Boot mode 3 (Serial Flash boot 3.3V)
Terminal software communication settings	• Communication speed: 115200bps • Data length: 8 bits • Parity: None • Stop bits: 1 bit • Flow control: None
Board to be used	RZ/A2M CPU board RTK7921053C00000BE RZ/A2M SUB board RTK79210XXB00000BE
Device (functionality to be used on the board)	• Serial flash memory allocated to SPI multi-I/O bus space (channel 0) Manufacturer : Macronix Inc. Model Name : MX25L51245GXD • RL78/G1C (Convert between USB communication and serial communication to communicate with the host PC.) • LED1

3. Description of Software

3.1 Operation Outline

Figure 3-1 shows the sequence of image data converted using the synchronous function.

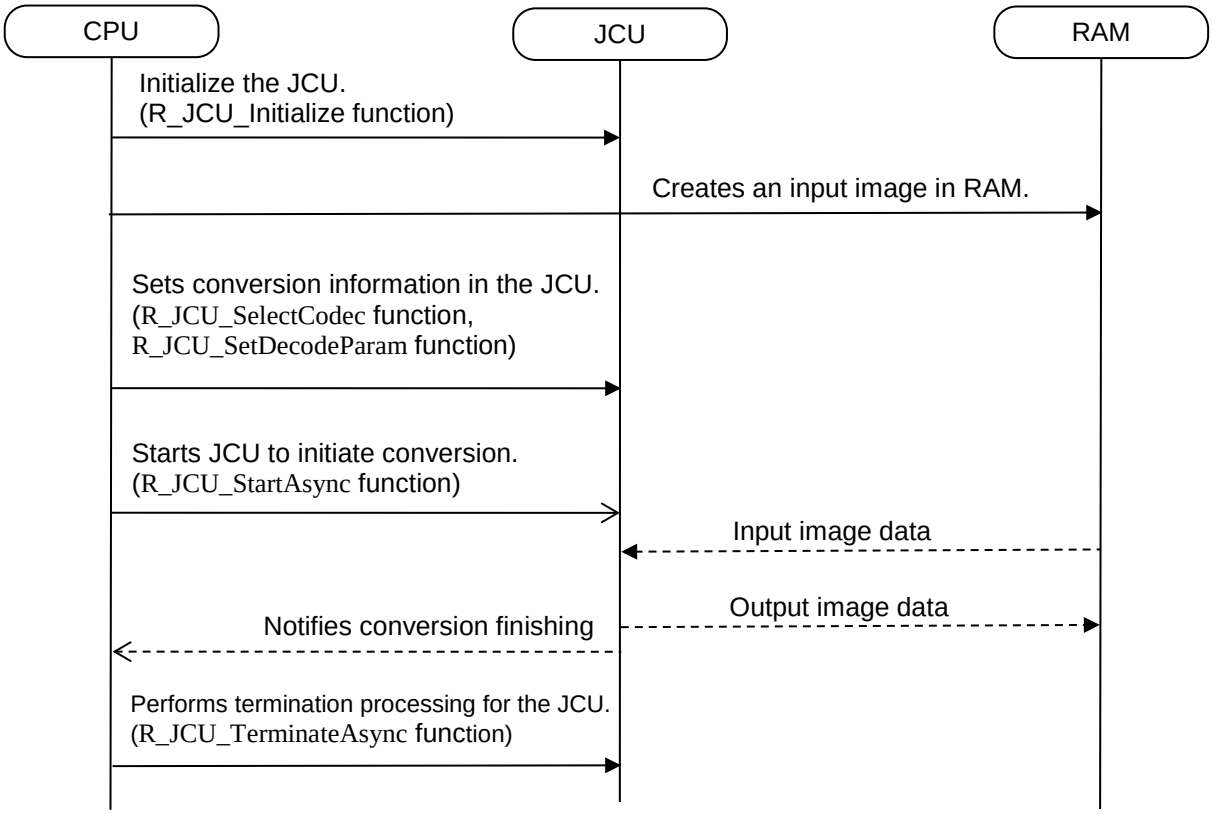


Figure 3-1 Sequence of image data conversion

This sample program has processing of 3 kinds, "decoding processing of a JPEG picture"(R_JCU_SampleDecode function), "decoding and encoding processing of a JPEG picture"(R_JCU_SampleDecodeEncode function) and "the processing indicated after decoding of a JPEG picture"(R_JCU_SampleDecodeAndShow function).

3.2 Interrupt

Table 3.1 shows Interrupts using by sample code.

Table 3.1 Interrupts using by sample code

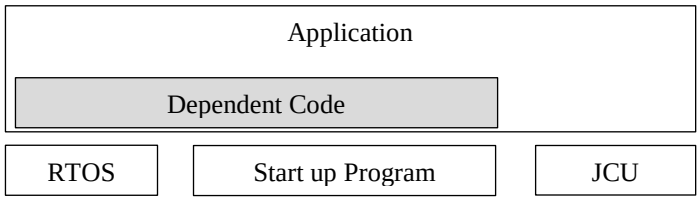
Interrupt (Source ID)	Priority	Summary
JEDI	JCU_INT_PRIORITY(=30)	Compression/Decompression process.
JDTI	JCU_INT_PRIORITY(=30)	Data transfer process

3.3 Constants, Type, Classes and Functions

Refer the HTML file attached the project.

3.4 Porting Guide

If you changed RTOS or start up program from a JCU running environment, the code in the application that depends on RTOS or start up program must be changed.



Callback function that specified to JCU API function that starts asynchronous operation depends on RTOS. Attached example application do the polling at while statement then CPU load will be 100%.

```
is_event = false;

e= R_JCU_StartAsync( (r_co_function_t) gs_SetTrue,  &is_event );
    if(e){goto fin;}
while ( ! is_event )
    { }  /* Pooling */
e= R_JCU_GetAsyncError(); if(e){goto fin;}
```

In order to avoid the CPU load becoming 100%, the following code must change to use RTOS binary semaphore, event group, thread attached event and so on.

- Specify API function to stop waiting instead of "gs_SetTrue" specified to the argument of asynchronous operating function. This function will be called from interrupt or thread
- Specify synchronizing object of RTOS instead of "&is_event" specified to the argument of asynchronous operating function
- Change polling "while" statement to calling RTOS API to wait
- Specify a different synchronization object for the synchronization object specified for "R_JCU_StartAsync" and the synchronization object for "R_JCU_TerminateAsync". "R_JCU_StartAsync" and "R_JCU_ContinueAsync" can be specified with the same synchronization object.

When calling from middleware, change the following code to use the porting layer of middleware.

- Instead of specifying "gs_SetTrue" as the argument of the asynchronous processing function, specify the function of the porting layer which abstracts to cancel the wait of RTOS. This function is called from an interrupt or thread
- Instead of specifying "&is_event" as an argument of the asynchronous processing function, specify a pointer indicating an instance of the middleware
- Change the "while" statement that does polling to the code of calling the function of porting layer which abstracts to wait in RTOS. Specify a pointer indicating an instance of the middleware to the argument
- Specify a different function for the function specified for "R_JCU_StartAsync" and for the function specified for "R_JCU_TerminateAsync". AAAA and CCCC can be specified with the same function

Mirror area and physical address of RAM depends on the setting of MMU that defined in start up program. The address to store in the pointer is virtual address. The address to access from JCU hardware is physical address. These addresses relationship depends on the setting of MMU.

4. Functions

Table 4.1 API functions

Section	Function Name	Outline
4.1	R_JCU_Initialize	Initializes the JCU driver.
0	R_JCU_TerminateAsync	Performs termination processing for the JCU driver (asynchronous process).
0	R_JCU_SelectCodec	Sets the JCU mode.
0	R_JCU_SetPauseForImageInfo	When the image information can be acquired, it's made the setting which is paused.
0	R_JCU_SetErrorFilter	The particular error code(jcu_int_detail_error_t) was set to valid.
0	R_JCU_StartAsync	Starts JCU process (asynchronous process).
0	R_JCU_GetAsyncError	Returns result of asynchronous process.
4.8	R_JCU_ContinueAsync	Resume the JCU process (asynchronous process).
0	R_JCU_SetDecodeParam	Sets decoding parameter.
0	R_JCU_GetImageInfo	Gets information on the JPEG data.
0	R_JCU_SetEncodeParam	Sets encoding parameter.
0	R_JCU_SetQuantizationTable	Sets the Quantization table.
0	R_JCU_SetHuffmanTable	Sets the Huffman table.
0	R_JCU_GetEncodedSize	Gets the size of data to be compressed.
0	R_JCU_OnInterrupting	Inerrupt service routine (ISR)

Table 4.2 User defined functions

Section	Function Name	Outline
0	R_JCU_OnInitialize	Initializes the user defined process.
0	R_JCU_OnFinalize	Finalizes the user defined process.
0	R_JCU_EnableInterrupt	Callbacks on request of interrupt enabling.
0	R_JCU_DisableInterrupt	Callbacks on request of interrupt disabling.

4.1 R_JCU_Initialize

Outline	Initializes the JCU driver.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_Initialize (jcu_config_t* in_out_Config);	
Description	The state will be in the initialized status. Initializes the internal status(gs_jcu_internal_information). The user defined function(R_JCU_OnInitialize) is called. Perform the following processing in the user defined function. 1. Clock supply to JCU. 2. Sets the priority of interrupt. 3. Sets the environment-depend process.	
Arguments	jcu_config_t* in_out_Config	NULL
Return value	Error code.	

4.2 R_JCU_TerminateAsync

Outline	Performs termination processing for the JCU driver.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_TerminateAsync(r_co_function_t in_OnFinalized, volatile void* in_OnFinalizedArgument);	
Description	The processing which finishes a JCU driver. This function is asynchronous function that will return before processing ends. The state will be changed. Perform the following processing in the user defined function. - Clock stopped to JCU. - Clear the priority of interrupt. - Sets the environment-depend process. 0 R_JCU_GetAsyncError must be called after finishing this asynchronous operation.	
Arguments	r_co_function_t in_OnFinalized	Callback function called when interrupt was signaled. This function will be called from interrupt or thread. If any error was raised, this function will be not called.
	volatile void* in_OnFinalizedArgument	Argument of callback function called when interrupt was signaled
Return value	Error code.	

4.3 R_JCU_SelectCodec

Outline	Sets the JCU mode.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_SelectCodec(const jcu_codec_t codec);	
Description	This function selects the JCU mode(Compression or De-compression). All parameters of decode, encode and count mode must be set again. Because when this function was called, these parameters were initialized.	
Arguments	const jcu_codec_t codec	JCU mode(Compression or De-compression)
Return value	Error code.	

4.4 R_JCU_SetPauseForImageInfo

Outline	When the image information can be acquired, it's made the setting which is paused.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_SetPauseForImageInfo(const bool_t is_pause)	
Description	When the image information can be acquired, it's made the setting which is paused by the R_JCU_GetImageInfo function.	
Arguments	const bool_t is_pause	TRUE: It's made the setting which is paused. FALSE: It's made the setting which isn't paused.
Return value	Error code.	

4.5 R_JCU_SetErrorFilter

Outline	The particular error code(jcu_int_detail_error_t) was set to valid.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_SetErrorFilter(jcu_int_detail_errors_t filter);	
Description	The particular error code was set to valid. When the valid decoding error occurred, interrupt occurs.	
Arguments	jcu_int_detail_errors_t filter	The valid decoding error code(jcu_int_detail_error_t) as the bit flag value.
Return value	Error code.	

4.6 R_JCU_StartAsync

Outline	Starts JCU process.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_StartAsync(r_co_function_t in_OnFinished, volatile void* in_OnFinishedArgument);	
Description	Starts JCU process. The function is asynchronous function that will return before decoding or encoding ends or pauses. Using the R_JCU_SetDecoderParam API function or the R_JCU_SetEncoderParamSet API function, set the parameters before the JCU process starts You cannot stop the JCU process, after the JCU process starts. 0 R_JCU_GetAsyncError must be called after finishing this asynchronous operation.	
Arguments	r_co_function_t in_OnFinalized	Callback function called after interrupt handling. This function will be called from interrupt or thread. If any error was raised, this function will be not called.
	volatile void* in_OnFinalizedArgument	Argument of callback function called after interrupt handling
Return value	Error code.	

4.7 R_JCU_GetAsyncError

Outline	Returns error raised in asynchronized process.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_GetAsyncError(void);	
Description		
Arguments	None	
Return value	Error code.	

4.8 R_JCU_ContinuetAsync

Outline	Resumes the JCU process (asynchronous process).
Header	r_jcu.h
Declaration	jcu_errorcode_t R_JCU_ContinueAsync(const jcu_continue_type_t type, r_co_function_t in_OnFinished, volatile void* in_OnFinishedArgument);
Description	Processing of JCU which paused is resumed. The function is asynchronous function that will return before decoding or encoding ends or pauses. 0 R_JCU_GetAsyncError must be called after finishing this asynchronous operation.

Arguments		
	jcu_continue_type_t type	Mode of restarting JCU
	r_co_function_t in_OnFinalized	Callback function called after interrupt handling. This function will be called from interrupt or thread. If any error was raised, this function will be not called.
	volatile void* in_OnFinalizedArgument	Argument of callback function called after interrupt handling
Return value	Error code.	

4.9 R_JCU_SetDecodeParam

Outline	Sets decoding parameter.
Header	r_jcu.h
Declaration	jcu_errorcode_t R_JCU_SetDecodeParam(const jcu_decode_param_t *const decode, const jcu_buffer_param_t *const buffer, const uint32_t interruptKind);
Description	Sets decoding parameter. If the pixel format isn't ARGB8888, the alpha value must be zero. If the pixel format isn't YCbCr, the Cb/Cr value must be JCU_CBCR_OFFSET_0.

Arguments		
	const jcu_decode_param_t *const decode	Pointer to variable of decode parameter information.
	const jcu_buffer_param_t *const buffer	Pointer to variable of buffer.
Return value	Error code.	

4.10 R_JCU_GetImageInfo

Outline	Gets information on the JPEG data.
Header	r_jcu.h
Declaration	jcu_errorcode_t R_JCU_GetImageInfo(jcu_image_info_t *const buffer);
Description	Gets the image information (width, height, pixel format) of the decoded JPEG data. If data is read before the request which reads the image information, the data is not guaranteed. If the pixel format of the decoded JPEG data is outside of the jcu_jpeg_format_t, it's the error, so JCU can't decode.

Arguments		
	jcu_image_info_t *const buffer	Pointer to variable of image information.
Return value	Error code.	

4.11 R_JCU_SetEncodeParam

Outline	Sets encoding parameter.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_SetEncodeParam(const jcu_encode_param_t *const encode, const jcu_buffer_param_t *const buffer, const uint32_t interruptKind);	
Description	Sets Encoding parameter.	
Arguments	const jcu_encode_param_t *const encode	Pointer to variable of encode parameter information.
	const jcu_buffer_param_t *const buffer	Pointer to variable of buffer.
Return value	Error code.	

4.12 R_JCU_SetQuantizationTable

Outline	Sets the Quantization table.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_SetQuantizationTable(const jcu_table_no_t tableNo, const uint8_t *const table);	
Description	Quantization table data. For the setting value of the quantization table data, see "RZ/A2M Group User's Manual: Hardware" section 45.3.1 (4), (a) Quantization Table Specification. Attached "QuantizationTable_Generator.html" file can calculate an example of quantization table.	
Arguments	const jcu_table_no_t tableNo	Quantization table number.
	const uint8_t *const table	Quantization table.
Return value	Error code.	

4.13 R_JCU_SetHuffmanTable

Outline	Sets the Huffman table.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_SetHuffmanTable(const jcu_table_no_t tableNo, const jcu_huff_t type, const uint8_t *const table);	
Description	Huffman table data. For the setting value of the Huffman table data, see "RZ/A2M Group User's Manual: Hardware" section 45.3.1 (4), (b) Huffman Table Specification.	
Arguments	const jcu_table_no_t tableNo	Huffman table number.
	const jcu_huff_t type	Type of Huffman table (AC or DC).
	const uint8_t *const table	Huffman table
Return value	Error code.	

4.14 R_JCU_GetEncodedSize

Outline	Gets the size of data to be compressed.	
Header	r_jcu.h	
Declaration	jcu_errorcode_t R_JCU_GetEncodedSize(size_t *const out_Size);	
Description	Gets the size of data to be compressed. If data is read before interrupt of encoding complete, the data is not guaranteed.	
Arguments	size_t *const out_Size	Pointer to variable of the data size.
Return value	Error code.	

4.15 R_JCU_OnInterrupting

Outline	Inerrupt service routine (ISR)	
Header	r_jcu.h	
Declaration	errnum_t R_JCU_OnInterrupting(void);	
Description	All JCU interrupt callback functions registered by R_JCU_OnInitialize function must call this ISR.	
Arguments	None.	
Return value	Error code.	

4.16 R_JCU_OnInitialize

Outline	Initializes the user defined process.	
Header	r_jcu_pl.h	
Declaration	errnum_t R_JCU_OnInitialize(void);	
Description	This user-defined function is callbacked from an initializing process of the JCU driver. If necessary, execute the following processing. - Clock control - Set interrupt priority - Environment-depend process	
Arguments	None.	
Return value	Error code.	

4.17 R_JCU_OnFinalize

Outline	Finalizes the user defined process.	
Header	r_jcu_pl.h	
Declaration	errnum_t R_JCU_OnFinalize(void);	
Description	This user-defined function is callbacked from a finalizing process of the JCU driver. If necessary, execute the following processing. - Clock stops - Clear interrupt priority - Environment-depend process	
Arguments	None.	
Return value	Error code.	

4.18 R_JCU_EnableInterrupt

Outline	Callbacks on request of interrupt enabling.	
Header	r_jcu_pl.h	
Declaration	void R_JCU_EnableInterrupt(void);	
Description	This user-defined function is callbacked from the JCU driver.	
Arguments	None.	
Return value	None.	

4.19 R_JCU_DisableInterrupt

Outline	Callbacks on request of interrupt disabling.	
Header	r_jcu_pl.h	
Declaration	bool_t R_JCU_DisableInterrupt(void);	
Description	This user-defined function is callbacked from the JCU driver.	
Arguments	None.	
Return value	Whether all JCU interrupts have been enabled before calling this function.	

5. Reference Documents

User's Manual: Hardware

RZ/A2M Group User's Manual: Hardware

The latest version can be downloaded from the Renesas Electronics website.

RTK7921053C00000BE (RZ/A2M CPU board) User's Manual

The latest version can be downloaded from the Renesas Electronics website.

RTK79210XXB00000BE (RZ/A2M SUB board) User's Manual

The latest version can be downloaded from the Renesas Electronics website.

ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition Issue C

The latest version can be downloaded from the ARM website.

ARM Cortex™-A9 Technical Reference Manual Revision: r4p1

The latest version can be downloaded from the ARM website.

ARM Generic Interrupt Controller Architecture Specification - Architecture version 2.0

The latest version can be downloaded from the ARM website.

ARM CoreLink™ Level 2 Cache Controller L2C-310 Technical Reference Manual Revision: r3p3

The latest version can be downloaded from the ARM website.

Technical Update/Technical News

The latest information can be downloaded from the Renesas Electronics website.

User's Manual: Development Tools

Integrated development environment e2studio User’s Manual can be downloaded from the Renesas Electronics website.

The latest version can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.02	Dec. 28, 2018	-	Modify standby control. Modify to checking STBACK register in R_JCU_OnInitialize function and R_JCU_OnFinalize function.
1.00	Sep. 14, 2018	-	First edition issued

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

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)

RENESAS

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 7.2