RX Family

MPC Module Using Firmware Integration Technology

Introduction

This application note describes the Multi-Function Pin Controller (MPC) module which uses Firmware Integration Technology (FIT). This module uses MPC to control the routing of a function to a pin. In this document, this module is referred to as the MPC FIT module.

Target Devices

- RX110, RX111, RX113 Groups
- RX130 Group
- RX13T Group
- RX230 Group
- RX231 Group
- RX23T Group
- RX23W Group
- RX24T Group
- RX24U Group
- RX64M Group
- RX651, RX65N Groups
- RX66T Group
- RX66N Group
- RX71M Group
- RX72T Group
- RX72M Group
- RX72N Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Target Compilers

- Renesas Electronics C/C++ Compiler Package for RX Family
- GCC for Renesas RX
- IAR C/C++ Compiler for Renesas RX

For details of the confirmed operation contents of each compiler, refer to “6.1 Confirmed Operation Environment".

Contents

1. Overview

1.1 MPC FIT Module

The MPC FIT module can be used by being implemented in a project as an API. See section 2.13, Adding the FIT Module to Your Project for details on methods to implement this FIT module into a project.

1.2 Overview of the MPC FIT Module

Modern MCUs have continued to add more peripherals while still maintaining relatively low pin counts. When this occurs, each pin will have multiple functions allocated to it. On RX MCUs, the routing of a function to a pin is controlled by the Multi-Function Pin Controller. This driver abstracts this functionality allowing use of the same pin definitions from the r_gpio_rx module.

1.3 API Overview

Table 1.1 lists the API functions included in this module.

Table 1.1 API Functions

Function	Description
R_MPC_Write()	Sets the function of a pin.
R_MPC_Read()	Reads the function configuration of a pin.
R_MPC_GetVersion()	Returns the current version of this module.

2. API Information

This FIT module has been confirmed to operate under the following conditions.

2.1 Hardware Requirements

The MCU used must support the following functions:

- MPC

2.2 Software Requirements

This driver is dependent upon the following FIT modules:

- Renesas Board Support Package (r_bsp) v5.20 or higher
- General Purpose Input/Output Driver (r_gpio_rx)

2.3 Limitations

2.3.1 RAM Location Limitations

In FIT, if a value equivalent to NULL is set as the pointer argument of an API function, error might be returned due to parameter check. Therefore, do not pass a NULL equivalent value as pointer argument to an API function.
The NULL value is defined as 0 because of the library function specifications. Therefore, the above phenomenon would occur when the variable or function passed to the API function pointer argument is located at the start address of RAM (address 0x0). In this case, change the section settings or prepare a dummy variable at the top of the RAM so that the variable or function passed to the API function pointer argument is not located at address 0x0.
In the case of the CCRX project (e2 studio V7.5.0), the RAM start address is set as 0x4 to prevent the variable from being located at address 0x0. In the case of the GCC project (e2 studio V7.5.0) and IAR project (EWRX V4.12.1), the start address of RAM is 0x0, so the above measures are necessary.
The default settings of the section may be changed due to the IDE version upgrade. Please check the section settings when using the latest IDE.

2.4 Supported Toolchain

This driver has been confirmed to work with the toolchain listed in 6.1, Confirmed Operation Environment.

2.5 Interrupt Vector

None.

2.6 Header Files

All API calls and their supporting interface definitions are located in “r_mpc_rx_if.h”.

2.7 Integer Types

This project uses ANSI C99. These types are defined in stdint.h.

2.8 Configuration Overview

The configuration option settings of this module are located in r_mpc_rx_config.h. The option names and setting values are listed in the table below:

Configuration options in r_mpc_rx_config.h	
MPC_CFG_PARAM_CHECKING_ENABLE 1	= 1: Include parameter checking in the build. = 0: Omit parameter checking from the build. = BSP_CFG_PARAM_CHECKING_ENABLE: Use the system default setting. Note: Code size can be reduced by excluding parameter checking from the build.

2.9 Code Size

Typical code sizes associated with this module are listed below.

The ROM (code and constants) and RAM (global data) sizes are determined by the build-time configuration options described in 2.8, Configuration Overview. The table lists reference values when the C compiler's compile options are set to their default values, as described in 2.4, Supported Toolchain. The compile option default values are optimization level: 2, optimization type: for size, and data endianness: little-endian. The code size varies depending on the C compiler version and compile options.

ROM, RAM and Stack Code Sizes							
Device	Category	Memory Used					
		Renesas Compiler		GCC		IAR	
		With Parameter Checking	Without Parameter Checking	With Parameter Checking	Without Parameter Checking	With Parameter Checking	Without Parameter Checking
All devices	ROM	128 bytes	110 bytes	320 bytes	280 bytes	437 bytes	405 bytes
	RAM	0 bytes	0 bytes	0 bytes	0 bytes	0 bytes	0 bytes
	Maximum stack usage	32 bytes	32 bytes	-	-	56 bytes	56 bytes

2.10 Parameters

This section describes the parameter structure used by the API functions in this module. The structure is located in r_mpc_rx_if.h as are the prototype declarations of API functions.

2.10.1 MPC Pin Configuration

This data structure is used for configuring a pin's function. To find valid settings for *pin_function*, refer to the Multi-Function Pin Controller (MPC) section of your MCU's hardware manual. Select the Pin Function Control Register for the port that your pin is on. On this page you will find a table with available functions for each pin on the selected port.

```
/* Options for configuring the MPC register of a pin. */
typedef struct
{
    uint8_t pin_function;  //Which peripheral function is assigned to this pin
    bool    irq_enable;    //This pin is used as IRQ pin
```

```
    bool    analog_enable; //This pin is used as ADC input, DAC output, or for
                           //LVD (CMPA2)
} mpc_config_t;
```

2.11 Return Values

This section describes return values of API functions. This enumeration is located in r_mpc_rx_if.h as are the prototype declarations of API functions.

Below are the available return values for the R_MPC_Write() function.

```
/* Function return type. */
typedef enum
{
    MPC_SUCCESS = 0,
    MPC_ERR_INVALID_CFG, // The configuration specified cannot be applied to
                         // this pin
} mpc_err_t;
```

2.12 Callback Function

None.

2.13 Adding the FIT Module to Your Project

This module must be added to each project in which it is used. Renesas recommends the method using the Smart Configurator described in (1) or (3) below. However, the Smart Configurator only supports some RX devices. Please use the methods of (2) or (4) for RX devices that are not supported by the Smart Configurator.

(1) Adding the FIT module to your project using the Smart Configurator in e^2 studio
By using the Smart Configurator in e^2 studio, the FIT module is automatically added to your project. Refer to “Renesas e^2 studio Smart Configurator User Guide (R20AN0451)” for details.

(2) Adding the FIT module to your project using the FIT Configurator in e^2 studio
By using the FIT Configurator in e^2 studio, the FIT module is automatically added to your project. Refer to “Adding Firmware Integration Technology Modules to Projects (R01AN1723)” for details.

(3) Adding the FIT module to your project using the Smart Configurator in CS+
By using the Smart Configurator Standalone version in CS+, the FIT module is automatically added to your project. Refer to “Renesas e^2 studio Smart Configurator User Guide (R20AN0451)” for details.

(4) Adding the FIT module to your project in CS+
In CS+, please manually add the FIT module to your project. Refer to “Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)” for details.

2.14 “for”, “while” and “do while” statements

In this module, “for”, “while” and “do while” statements (loop processing) are used in processing to wait for register to be reflected and so on. For these loop processing, comments with “WAIT_LOOP” as a keyword are described. Therefore, if user incorporates fail-safe processing into loop processing, user can search the corresponding processing with “WAIT_LOOP”.

The following shows example of description.

```
while statement example :
/* WAIT_LOOP */
while(0 == SYSTEM.OSCOVFSR.BIT.PLOVF)
{
    /* The delay period needed is to make sure that the PLL has stabilized. */
}

for statement example :
/* Initialize reference counters to 0. */
/* WAIT_LOOP */
for (i = 0; i < BSP_REG_PROTECT_TOTAL_ITEMS; i++)
{
    g_protect_counters[i] = 0;
}

do while statement example :
/* Reset completion waiting */
do
{
    reg = phy_read(ether_channel, PHY_REG_CONTROL);
    count++;
} while ((reg & PHY_CONTROL_RESET) && (count < ETHER_CFG_PHY_DELAY_RESET)); /* WAIT_LOOP */
```

3. API Functions

R_MPC_Write

This function sets the function of a pin.

Format

```
mpc_err_t       R_MPC_Write (
        gpio_port_pin_t         pin,
        mpc_config_t            * pconfig
)
```

Parameters

gpio_port_pin_t pin

Which pin to configure.

*mpc_config_t *pconfig*

Pointer to structure with pin configuration information. See section 2.10.1, MPC Pin Configuration.

Return Values

```
[MPC_SUCCESS]           /* Successful; pin configured.          */
[MPC_ERR_INVALID_CFG]   /* Error; invalid configuration input   */
```

Properties

Prototyped in file “r_mpc_rx_if.h”

Description

This function will configure a pin based on the information in the mpc_config_t structure. Not all pins support the same functionality. For example, not all pins are able to be configured as analog pins for ADC or DAC use. Also, not all combinations of functionality are capable. For example, a pin cannot be configured as an analog pin and for peripheral use at the same time.

To see what functions are available for a pin, refer to the Multi-Function Pin Controller (MPC) section of your MCU’s hardware manual. Select the Pin Function Control Register for the port that your pin is on. On this page you will find a table with available functions for each pin on the selected port.

Which pin is to be configured by this function is defined using the gpio_port_pin_t type from the r_gpio_rx module.

Example

```
mpc_config_t      config;
gpio_port_pin_t   pin;

/* Set PE0 to be used as analog pin for ADC. */
pin = GPIO_PORT_E_PIN_0;

config.analog_enable = true;
config.irq_enable = false;
config.pin_function = 0;

if (MPC_SUCCESS != R_MPC_Write(pin, &config))
{
```

```
    /* Error, pin does not support this configuration. Handle error. */
    …
}

/* Set P27 to be used as IRQ pin and for SCI operations. */
pin = GPIO_PORT_2_PIN_7;

config.analog_enable = false;
config.irq_enable = true;
config.pin_function = 0xA;

if (MPC_SUCCESS != R_MPC_Write(pin, &config))
{
    /* Error, pin does not support this configuration. Handle error. */
    …
}
```

Special Notes:
None.

R_MPC_Read

This function reads the function configuration of a pin.

Format

```
void    R_MPC_Read (
        gpio_port_pin_t         pin,
        mpc_config_t            * pconfig
)
```

Parameters

gpio_port_pin_t pin

Which pin to read configuration information for.

*mpc_config_t *pconfig*

Pointer to structure where pin configuration information will be stored. See Section 2.10.1, MPC Pin Configuration.

Return Values

None.

Properties

Prototyped in file “r_mpc_rx_if.h”

Description

This function will read the configuration information for a pin and store it in a structure supplied by the user.

Example

```
mpc_config_t config;

/* See if P03 has already been configured as analog pin for DAC use. */
R_MPC_Read(GPIO_PORT_0_PIN_3, &config);

if (config.analog_enable == true)
{
    /* P03 has already been set as analog pin. */
    …
}
else
{
    /* P03 has not been configured yet. Configure it now. */
    …
}
```

Special Notes:

None.

R_MPC_GetVersion

Returns the current version of this API.

Format
uint32_t R_MPC_GetVersion (void)

Parameters
None.

Return Values
Version of this API.

Properties
Prototyped in file “r_mpc_rx_if.h”

Description
This function will return the version of the currently running API. The version number is encoded where the top 2 bytes are the major version number and the bottom 2 bytes are the minor version number. For example, Version 4.25 would be returned as 0x00040019.

Example
```
uint32_t cur_version;

/* Get version of running r_mpc_rx API. */
cur_version = R_MPC_GetVersion();

/* Check to make sure version is new enough for this application’s use. */
if (MIN_VERSION > cur_version)
{
    /* This r_mpc_rx version is not new enough and does not have XXX feature
       that is needed by this application. Alert user. */
    …
}
```

Special Notes:
None.

4. Pin Setting

MPC FIT module don’t use pin setting.

5. Demo Projects

Demo projects include function main() that utilizes the FIT module and its dependent modules (e.g. r_bsp). This FIT module includes the following demo projects.

5.1 mpc_demo_rskrx113, mpc_demo_rskrx64m, mpc_demo_rskrx71m, mpc_demo_rskrx65n and mpc_demo_rskrx65n_2m

These are the demo programs for the RX113, RX64M, RX71M, RX65N and RX65N-2MB MPC FIT modules. The programs demonstrate how to use the MPC to configure a port bit as an interrupt input. IRQ2 is chosen as the interrupt for these demos and is used to detect key-presses on SW2. Once the code is compiled and down-loaded to the target board and running, SW2 can be pressed to cause IRQ2 (IRQ9 on the RX65N, IRQ13 on the RX65N-2MB) interrupts to occur. The IRQ2 (IRQ9 on the RX65N, IRQ13 on the RX65N-2MB) interrupt handler in turn toggles the state of LED3.

5.2 mpc_demo_rskrx231

This is the demo program for the RX231 FIT module. The description for this program is the same as for the mpc_demo_rskrx113 demo except that IRQ4 is used to detect SW2 key-presses.

5.3 Adding a Demo to a Workspace

Demo projects are found in the FITDemos subdirectory of the distribution file for this application note. To add a demo
project to a workspace, select *File* >> *Import* >> *General* >> *Existing Projects into Workspace*, then click "Next". From the Import Projects dialog, choose the "Select archive file" radio button. "Browse" to the FITDemos subdirectory, select the desired demo zip file, then click "Finish".

5.4 Downloading Demo Projects

Demo projects are not included in the RX Driver Package. When using the demo project, the FIT module needs to be downloaded. To download the FIT module, right click on this application note and select "Sample Code (download)" from the context menu in the *Smart Browser* >> *Application Notes* tab.

6. Appendices

6.1 Confirmed Operation Environment

This section describes confirmed operation environment for the MPC FIT module.

Table 6.1 Confirmed Operation Environment (Rev.3.40)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.7.0 IAR Embedded Workbench for Renesas RX 4.12.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
	GCC for Renesas RX 4.8.4.201902 Compiler option: The following option is added to the default settings of the integrated development environment. -std=gnu99 Linker option: The following user defined option should be added to the default settings of the integrated development environment, if “Optimize size (-Os)” is used: -Wl,--no-gc-sections This is to work around a GCC linker issue whereby the linker erroneously discard interrupt functions declared in FIT peripheral module
	IAR C/C++ Compiler for Renesas RX version 4.12.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.3.40
Board used	Renesas Starter Kit+ for RX72N (product No.: RTK5572Nxxxxxxxxxx)

Table 6.2 Confirmed Operation Environment (Rev.3.30)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.7.0 IAR Embedded Workbench for Renesas RX 4.12.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
	GCC for Renesas RX 4.8.4.201902 Compiler option: The following option is added to the default settings of the integrated development environment. -std=gnu99 Linker option: The following user defined option should be added to the default settings of the integrated development environment, if “Optimize size (-Os)” is used: -Wl,--no-gc-sections This is to work around a GCC linker issue whereby the linker erroneously discard interrupt functions declared in FIT peripheral module
	IAR C/C++ Compiler for Renesas RX version 4.12.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.3.30
Board used	RX13T CPU Card (product No.: RTK0EMXA10C00000BJ)

Table 6.3 Confirmed Operation Environment (Rev.3.20)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.5.0 IAR Embedded Workbench for Renesas RX 4.12.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
	GCC for Renesas RX 4.8.4.201902 Compiler option: The following option is added to the default settings of the integrated development environment. -std=gnu99 Linker option: The following user defined option should be added to the default settings of the integrated development environment, if “Optimize size (-Os)” is used: -Wl,--no-gc-sections This is to work around a GCC linker issue whereby the linker erroneously discard interrupt functions declared in FIT peripheral module
	IAR C/C++ Compiler for Renesas RX version 4.12.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.3.20
Board used	Renesas Starter Kit+ for RX72M (product No.: RTK5572Mxxxxxxxxxx)

Table 6.4 Confirmed Operation Environment (Rev.3.10)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.5.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
Endian	Big endian/little endian
Revision of the module	Rev.3.10
Board used	Renesas Solution Starter Kit for RX23W (product No.: RTK5523Wxxxxxxxxx)

Table 6.5 Confirmed Operation Environment (Rev.3.00)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.4.0 IAR Embedded Workbench for Renesas RX 4.10.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
	GCC for Renesas RX 4.8.4.201803 Compiler option: The following option is added to the default settings of the integrated development environment. -std=gnu99 Linker option: The following user defined option should be added to the default settings of the integrated development environment, if “Optimize size (-Os)” is used: -Wl,--no-gc-sections This is to work around a GCC linker issue whereby the linker erroneously discard interrupt functions declared in FIT peripheral module
Endian	IAR C/C++ Compiler for Renesas RX version 4.10.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.3.00
Board used	Renesas Starter Kit+ for RX65N-2MB (product No.: RTK50565Nxxxxxxxxx)

Table 6.6 Confirmed Operation Environment (Rev.2.50)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.3.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
Endian	Big endian/little endian
Revision of the module	Rev.2.50
Board used	Renesas Starter Kit for RX72T (product No.: RTK5572Txxxxxxxxx)

Table 6.7 Confirmed Operation Environment (Rev.2.41)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.3.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
Endian	Big endian/little endian
Revision of the module	Rev.2.41
Board used	Renesas Starter Kit for RX66T (product No.: RTK50566T0SxxxxxBE) Renesas Starter Kit+ for RX 65N-2MB (product No.: RTK50565N2CxxxxxBR) Renesas Starter Kit+ for RX130-512KB (product No.: RTK5051308CxxxxxBR)

Table 6.8 Confirmed Operation Environment (Rev.2.40)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.0.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.00.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
Endian	Big endian/little endian
Revision of the module	Rev.2.40
Board used	Renesas Starter Kit for RX66T (product No.: RTK50566T0SxxxxxBE) Renesas Starter Kit+ for RX 65N-2MB (product No.: RTK50565N2CxxxxxBR) Renesas Starter Kit+ for RX130-512KB (product No.: RTK5051308CxxxxxBR)

Table 6.9 Confirmed Operation Environment (Rev.2.31)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 6.0.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V2.07.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
Endian	Big endian/little endian
Revision of the module	Rev.2.31
Board used	Renesas Starter Kit+ for RX 65N-2MB (product No.: RTK50565N2CxxxxxBR) Renesas Starter Kit+ for RX130-512KB (product No.: RTK5051308CxxxxxBR)

Table 6.10 Confirmed Operation Environment (Rev.2.30)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 6.0.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V2.07.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
Endian	Big endian/little endian
Revision of the module	Rev.2.30
Board used	Renesas Starter Kit+ for RX 65N-2MB (product No.: RTK50565N2CxxxxxBR) Renesas Starter Kit+ for RX130-512KB (product No.: RTK5051308CxxxxxBR)

6.2 Troubleshooting

(1) Q: I have added the FIT module to the project and built it. Then I got the error: Could not open source file "platform.h".

A: The FIT module may not be added to the project properly. Check if the method for adding FIT modules is correct with the following documents:

- Using CS+:

  Application note "Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)"
- Using e^2 studio:

  Application note "Adding Firmware Integration Technology Modules to Projects (R01AN1723)"

When using a FIT module, the board support package FIT module (BSP module) must also be added to the project. Refer to the application note "Board Support Package Module Using Firmware Integration Technology (R01AN1685)".

(2) Q: I have added the FIT module to the project and built it. Then I got the error: This MCU is not supported by the current r_mpc_rx module.

A: The FIT module you added may not support the target device chosen in your project. Check the supported devices of added FIT modules.

(3) Q: I have added the FIT module to the project and built it. Then I got an error for when the configuration setting is wrong.

A: The setting in the file "r_mpc_rx_config.h" may be wrong. Check the file "r_mpc_rx_config.h". If there is a wrong setting, set the correct value for that. Refer to 2.8 Configuration Overview for details.

7. Reference Documents

User's Manual: Hardware
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

User's Manual: Development Tools
RX Family Compiler CC-RX User's Manual (R20UT3248)
The latest versions can be downloaded from the Renesas Electronics website.

Related Technical Updates

This module reflects the content of the following technical updates.

None

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Nov15, 2013	—	Initial release.
1.20	Apr 17, 2014	1,3	Added support for the RX64M Group.
1.30	Jul 02, 2014	—	Fixed RX63N receive bug. Used latest Colophon.
1.40	Dec 12,2014	Various 1 5	Updated to current app-note template Added RX113 to the list of supported MCUs Added a Code Size section.
1.50	Jan 17,2015	—	Added support for the RX71M Group.
1.60	Jun 30,2015	—	Added support for the RX231 Group.
1.70	Sep 30,2015	—	Added support for the RX23T Group.
1.80	Oct 1,2015	—	Added support for the RX130 Group.
1.90	Dec 1,2015	— 1, 6 4 11	Added support for the RX24T Group. Changed the document number for the “Board Support Package Firmware Integration Technology Module” application note. Changed the description in section 2. Added “4. Demo Projects”.
2.00	Feb 1,2016	— 12	Added support for the RX230 Group. Added “Related Technical Updates”.
2.01	Jul 29, 2016	11	Added RSKRX64M to “4. Demo Projects”.
2.10	Oct 1,2016	— 5	Added support for the RX65N Group. Changed 2.9 Code Size for the tabular format of Code Size. Updated 2.9 Code Size for the RX65N Group.
2.20	Feb 28, 2017	— 4	Added support for the RX24U Group. Added RXC v2.06.00 to “2.5 Supported Toolchains”.
2.30	Jul 21, 2017	— 4 6	Added support for the RX130-512KB and RX65N-2MB. Added RXC v2.07.00 to “2.5 Supported Toolchains”. Updated “2.12 Adding the FIT Module to Your Project”.
2.31	Oct 31, 2017	11 12	Added RSKRX65N, RSKRX65N-2MB to “4. Demo Projects” Added 4.4 Downloading Demo Project Added 5. Appendices
2.40	Sep 28, 2018	1 5 14	Added support for the RX66T. Added code size corresponding to RX66T 6.1 Confirmed Operation Environment: Added table for Rev.2.40
2.41	Nov 16, 2018	— 14	Added document number in XML Changed Renesas Starter Kit Product No for RX66T. Added table for Rev.2.41
2.50	Feb 01, 2019	Program 1 8-11 14	Added support for RX72T. Added support for RX72T. Removed ‘Reentrant’ description in each API function. 6.1 Confirmed Operation Environment: Added Table for Rev 2.50

3.00	May.20.19	—	Supported the following compilers: - GCC for Renesas RX - IAR C/C++ Compiler for Renesas RX
		1	Added the section of Target compilers. Deleted related documents.
		4	2.2 Software Requirements Requires r_bsp v5.20 or higher
		5	Updated the section of 2.8 Code Size
		14	Table 6.1 Confirmed Operation Environment: Added table for Rev.3.00
		18	Deleted the section of Website and Support.
		Program	Changed below for support GCC and IAR compiler: Deleted the inline expansion of the R_MPC_GetVersion function.
3.10	Jun.28.19	1	Added support for RX23W.
		14	6.1 Confirmed Operation Environment: Added Table for Rev.3.10
		Program	Added support for RX23W.
3.20	Aug.15.19	1	Added support for RX72M.
		14	6.1 Confirmed Operation Environment: Added Table for Rev.3.20 Table 6.2: Corrected board name for RX23W
		Program	Added support for RX72M.
3.30	Nov.25.19	1	Added support for RX13T.
		4	2.3 Limitations Added Limitations
		5	Updated the section of 2.9 Code Size
		14	6.1 Confirmed Operation Environment: Added Table for Rev.3.30
		Program	Added support for RX13T. Change the comment of API function to the Doxygen style.
3.40	Dec.30.19	1	Added support for RX72N, RX66N.
		5	Updated the section of 2.9 Code Size
		14	6.1 Confirmed Operation Environment: Added Table for Rev.3.40
		Program	Added support for RX72N, RX66N.

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. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on

   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. 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 the 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.

5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.

6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).

7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.