

RX100/RX200 Series

Updating Firmware Using Start-Up Program Protection and Serial Communication

Introduction

This application note describes updating the on-chip code flash memory using the start-up program protection in the RX100/RX200 Series. Serial communication is used for sample programs control and data transfer.

In this application note, "firmware program" and "firmware update program" are defined as follows:

- Firmware program: Program to be written in the user area of the code flash memory
- Firmware update program: Program to rewrite the firmware program

This application note includes the following sample programs: the firmware update program and the firmware to verify the firmware update program operation.

In this application note, the firmware update program is placed in the default area of the start-up program protection. If you plan to place the firmware update program in the alternate area, translate "default area" into "alternate area" in this document and vice versa.

Target Device

- RX230 Group, RX231 Group, RX260 Group, and RX261 Group
- RX110 Group, RX111 Group, RX113 Group, RX130 Group, and RX140 Group

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

Related Documents

- Firmware Integration Technology User's Manual (R01AN1833)
- RX Family Adding Firmware Integration Technology Modules to Projects (R01AN1723)
- RX Family Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)
- RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685)
- RX Family Flash Module Using Firmware Integration Technology (R01AN2184)
- RX Family SCI Multi-Mode Module Using Firmware Integration Technology (R01AN1815)
- RX Family BYTEQ Module Using Firmware Integration Technology (R01AN1683)



Contents

| 1. C |)verview | 4 |
|------|--|-----|
| 1.1 | About This Application Note | 4 |
| 1.2 | Operation Confirmation Environment | 5 |
| 1.3 | Module Configuration | 6 |
| 1.4 | File Structure | 7 |
| 1.5 | Project | 9 |
| 2. 0 | Obtaining the Development Environment | 10 |
| 2.1 | e ² studio | 10 |
| 2.2 | CS+ | 10 |
| 2.3 | Compiler Package | 10 |
| 2.4 | Renesas Flash Programmer | 10 |
| 3. S | Setting Up the Project | 11 |
| 3.1 | Creating a Workspace | 11 |
| 3.2 | Creating a Project | 13 |
| 3.3 | Importing the Project | 16 |
| 3.4 | Changed Setting Information | 19 |
| 3.4 | 4.1 Configuration Option | 19 |
| 3.4 | 4.2 Modifying the Project Setting | 21 |
| 4. C | Operation Confirmation | 26 |
| 4.1 | Building the Project | 26 |
| 4.2 | Preparing Debugging | 27 |
| 4.2 | 2.1 Preparing Devices | 27 |
| 4.2 | 2.2 Setting the Host PC | 28 |
| 4.3 | Debugging the Project | 29 |
| 5. A | Application Overview | 37 |
| 5.1 | Configuration of the Firmware Update Program | 37 |
| 5.2 | Operation Overview | |
| 5.2 | 2.1 Programming the Firmware Update Program | 38 |
| 5.2 | 2.2 Programming the Firmware Program | 39 |
| 5.2 | 2.3 Updating the Firmware Program | |
| 5.2 | 2.4 Updating the Firmware Update Program | |
| 5.2 | | |
| | 2.6 Recovery when Programming Failed | |
| 5.2 | 2.7 How to Create a Firmware Update Program for Updating | |
| 5.3 | Process Flowchart and Screen Output: Firmware Update Program | |
| 5.3 | · · · · | |
| | 3.2 Firmware Update Processing | ~~~ |

RX100/RX200 Series Updating Firmware Using Start-Up Program Protection and Serial Communication

| 5.3 | 3.3 | Firmware Start-up Processing7 | /1 |
|--------------|-----|--|------------|
| 5.4 | Pro | bcess Flowchart and Screen Output: Firmware Program7 | |
| 5.5 | | tailed Information of the Firmware Update Program7 | |
| 5.5 | 5.1 | File Composition | /5 |
| 5.5 | 5.2 | Constants7 | ' 6 |
| 5.5 | 5.3 | Type Definitions | 32 |
| 5.5 | 5.4 | Variables | |
| 5.5 | 5.5 | Functions | 39 |
| 5.6 | De | tailed Information of the Firmware9 |)2 |
| 5.6 | 6.1 | File Composition |)2 |
| 5.6 | 6.2 | Constants9 |)2 |
| 5.6 | 6.3 | Type Definitions |)2 |
| 5.6 | 6.4 | Variables9 |)3 |
| 5.6 | 6.5 | Functions |)4 |
| 6. Ir 6.1 | • | ort a Project9 porting a Project into CS+9 | |
| | | rence Documents9 | |



1. Overview

1.1 About This Application Note

This application note describes the method to safely update the code flash memory using the start-up protection.

The firmware update program is placed into the default area of the start-up program protection and the constants data area. The firmware update program is controlled from the host PC through serial communication to update the code flash memory. The MCU operates in single-chip mode and the Motorola S-record data is used as data for reprogramming. The XMODEM/SUM is used as the data transfer protocol. Therefore, the terminal software on the host PC must be capable of XMODEM/SUM transfer.

Table 1.1 lists the Peripheral Functions Used and Their Applications, and Figure 1.1 shows the Operation Overview.

| Table 1.1 | Peripheral Functions | Used and Their Applications |
|-----------|----------------------|-----------------------------|
|-----------|----------------------|-----------------------------|

| Peripheral Function | Application |
|--------------------------------|--|
| Flash memory | Reprogramming the code flash memory |
| Serial communication interface | Asynchronous serial communication with the host PC |



Figure 1.1 Operation Overview



The sample programs in this application note use the Firmware Integration Technology (FIT) modules to control peripheral modules. The FIT modules used in this application note are as follows:

- Board Support Package Module Using Firmware Integration Technology (BSP)
- Flash Module Using Firmware Integration Technology (Flash FIT module)
- SCI Multi-Mode Module Using Firmware Integration Technology (SCI FIT module)
- BYTEQ Module Using Firmware Integration Technology

1.2 Operation Confirmation Environment

The operation of the sample programs in this application note have been confirmed under the following conditions.

| ltem | Contents |
|------------------------|--|
| MCU used | R5F51305ADFN (RX130 group) |
| | R5F51406BDFN (RX140 group) |
| | R5F52318ADFP (RX231 group) |
| | R5F52618BGFP (RX261 group) |
| Board used | Renesas Starter Kit for RX130 (product No.: RTK5005130C00000BE) |
| | Renesas Starter Kit for RX140 (product No.: RTK551406BS00000BE) |
| | Renesas Starter Kit for RX231 (product No.: R0K505231C000BE) |
| | Evaluation Kit for RX261 MCU Group (product No.: RTK5EK2610S00001BE) |
| Integrated development | Renesas Electronics |
| environment | e ² studio Version 2024-07 |
| | Renesas Electronics |
| | CS+ V.8.12.00 |
| C compiler | Renesas Electronics |
| | C/C++ Compiler Package for RX Family V.3.06.00 |
| | Compiler option |
| | -lang = c99 |
| Flash programmer | Renesas Flash Programmer V.3.16.00 |
| Emulator | E2 Lite |
| Endian | Little endian |

Table 1.2 Operation Confirmation Conditions



1.3 Module Configuration

Figure 1.2 shows the module configuration of the sample program and Table 1.3 lists the FIT modules implemented in the sample program.



Figure 1.2 Module Configuration

Table 1.3 Module List

| Category | Application Note (Document No.) | FIT Module Name |
|---------------|---|-----------------|
| BSP | RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685) | r_bsp |
| Device driver | RX Family Flash Module Using Firmware Integration Technology (R01AN2184) | r_flash_rx |
| Device driver | RX Family SCI Multi-Mode Module Using Firmware Integration Technology (R01AN1815) | r_sci_rx |
| Device driver | RX Family BYTEQ Module Using Firmware Integration Technology (R01AN1683) | r_byteq |
| Application | Main program | src |



1.4 File Structure

Figure 1.3 shows the file structure of this application note.



Figure 1.3 File Structure

When the ZIP file provided by this application note is unzipped, the folder is created with the same name as the ZIP, containing associated folders and files.



"Firmware update project

(update_firmware_rx130/update_firmware_rx140/update_firmware_rx231/update_firmware_rx261)" and "Firmware project (firmware_rx130/firmware_rx140/firmware_rx231/firmware_rx261)" under the "r01an3740_rx130/r01an3740_rx140/r01an3740_rx231/r01an3740_rx261" folder are the projects that set up the sample programs in this application note. The operation of the application note can be confirmed by importing these projects into the workspace in the e² studio.



1.5 Project

This application note includes the e² studio projects for building and evaluating this application note. The projects have the build configuration and the debug configuration which store the build setting and the debug setting, respectively.

Table 1.4 lists the build configuration and the debug configuration that are registered in the project, and Table 1.5 shows the Target Specific Settings.

| | Configuration Example | Description |
|------------------------|--------------------------------------|---|
| Build configuration | HardwareDebug (Debug on hardware) | Configuration to generate a load module with debug information |
| Debug configuration | HardwareDebug (E2 Lite) | Performs hardware debugging via E2 Lite emulator using the load module generated with HardwareDebug (Debug on hardware) |

Table 1.4 Project Configuration

| Item | Setting |
|--------------------------------|-----------------------------------|
| Toolchain version | V3.04.00 |
| Debug hardware | E2 Lite (RX) |
| Endianness | Little-endian data |
| Target selection (RX130 group) | R5F51305ADFN (RX130 LFQFP 80pin) |
| Target selection (RX140 group) | R5F51406BDFN (RX140 LFQFP 80pin) |
| Target selection (RX231 group) | R5F52318ADFP (RX231 LQFP 100pin) |
| Target selection (RX261 group) | R5F52618BDFP (RX261 LFQFP 100pin) |
| Renesas RTOS support | None |

Table 1.5 Target Specific Settings



2. Obtaining the Development Environment

2.1 e² studio

Visit the following URL and download the e² studio.

https://www.renesas.com/en-us/products/software-tools/tools/ide/e2studio.html

This document assumes that V2022-04 or later version of e^2 studio is used. If a version earlier than V2022-04 is used, some features of e^2 studio may not be supported. Make sure to download the latest version of e^2 studio on the website.

2.2 CS+

Visit the following URL and download the CS+.

https://www.renesas.com/us/en/software-tool/cs.html

2.3 Compiler Package

Visit the following URL and download the RX Family C/C++ Compiler Package. https://www.renesas.com/us/en/software-tool/cc-compiler-package-rx-family.html

2.4 Renesas Flash Programmer

Visit the following URL and download the Renesas Flash Programmer.

https://www.renesas.com/us/en/software-tool/renesas-flash-programmer-programming-gui.html



3. Setting Up the Project

This application note includes the projects which have been configured the environment. The procedure to import the projects using the smart browser is described here. For importing the projects into CS+, refer to 6.1. Importing a Project into CS+.

3.1 Creating a Workspace

- 1. Start the e^2 studio.
- 2. The dialog to select a workspace opens. Enter a workspace and click Launch.

| 📴 e² studio Launcher | | | × |
|---|--------------------------|------------------------------|--------|
| Select a directory as workspace | e | | |
| e ² studio uses the workspace direct | ory to store its prefere | nces and development artifac | ts. |
| Workspace: C:¥workspace | | ~ Bro | wse |
| | | | |
| Ente | a workspace folder. | | |
| Use this as the default and do no | t ask again | Click La | inch |
| <u>R</u> ecent Workspaces | | CHICK La | |
| | | Launch | Cancel |
| | | | |



3. When the Welcome dialog opens, click "Hide".





3.2 Creating a Project

When using the Smart Brower features, a project or a file used has to be selected first. Thus, first create a project with the required MCU selected as the target device. This is a dummy project for using the Smart Browser. For configuring the imported project, refer to 3.4. Changed Setting Information.

1. Select *File >> New >> C/C++ Project*. The Wizard for creating a new project opens.

| | | - | | nesas Views Run Window Help | |
|---|--|------------------------|----------------|--|--------|
| 2 | New Open File Open Projects from File System Recent Files | Alt+Shift+N> | 2 | Renesas C/C++ Project Synergy C/C++ Project Makefile Project with Existing Code C/C++ Project | > |
| | Close Editor Close All Editors | Ctrl+W Ctrl+Shift+W | | | |
| | Save Save As Save All Revert | Ctrl+S Ctrl+Shift+S | C ^o | Folder Click "C/C++ Project". Source File Header File File from Template | |
| Ð | Move Rename Refresh | F2 F5 | 3 | Class Code Generator Example | |
| | Convert Line Delimiters To | > | -3 | Other | Ctrl+N |
| 8 | Print | Ctrl+P | | | |
| | Import Export | | | | |
| | Properties | Alt+Enter | | | |
| | Switch Workspace Restart Exit | > | | | |



2. Select "Renesas CC-RX C/C++ Executable Project" and click Next.



Enter a project name in the Project name field and click Next.

| 8 | | | _ | |
|-------------------------|---|-------------------------------|-------------------------------------|-------------|
| | CC-RX Executable Project C-RX Executable Project | Enter a projec | ct name. | Ď |
| <u>P</u> roject name: s | ample | | | |
| Use <u>d</u> efault I | location | | | |
| Location: | C:¥workspace¥sample | | В | rowse |
| | Create Directory for Project | ct | | |
| Choose file syste | em: default 🖂 | | | |
| Working sets | | | | |
| Add project | to working sets | | 1 | Ve <u>w</u> |
| Working sets: | | | ✓ S₅ | elect |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | Click N | lext. | |
| | | | | |
| | | <u>×</u> | | |
| ? | < <u>B</u> ack | <u>N</u> ext > <u>F</u> inisl | h | Cancel |
| | | | | |



Updating Firmware Using Start-Up Program Protection and Serial Communication

3. In the Target Device field, choose "R5F51305AxFN" for the RX130 group, "R5F51406BxFN" for the RX140 group, "R5F52318AxFP" for the RX231 group, and "R5F52618BxFP" for the RX261 group. Specify other settings as required. Click the Finish button. The following screenshots illustrate an example configuration for the RX231 group.

| | | | Γ | | | | | | | - | , | | | | |
|--|--|--|---|---|--|---|---|--|---|---|--|---|--|--|---------------------|
| ew Renesas CC-RX | Executable Proje | ct | | 8 | | | | | - [| | < | <u></u> | P - | () - i - | 0 - |
| elect toolchain, device | & debug settings | | | Device Selection | | | | | | | | • | | / 🥌 🔻 🕲 | |
| Toolchain Settings | | | | You can filter devices by reg | ular exp | ression | | | | | | - Outli | | . 🖻 🔤 | C/C++ |
| Toolchain Settings .anguage: | ○C++ | | - | | | | | | | | _ | Outli | | ive editor t | |
| | esas CCRX | ~ | | Search Device | | | | | | | _ | provide | | | nat |
| | | | | Device | RAM | ROM | Pin | RTOS | Smar | Perip | | | | | |
| oolchain Version: v3.04 | | ~ | | > RX100 ~ RX200 | | | | | | | | | | | |
| | Manage To | oolchains | | > RX210 | | | | | | | | | | | |
| RTOS: Non | e | ~ | | > RX21A | | | | | | | | | | | |
| TOS Version: | | \sim | | > RX220 | | | | | | | | | | | |
| Oliele " | button are | | t the t- | > RX230 | | | | | | | | | | | |
| | bullon and | , selec | | rget device. | | | | | | | | | | | |
| rget Board: Custom | | ~ | Create | > RX231 - 64pin | | | | | | | | | | | |
| | Download additiona | al boards | E2 Lit | ✓ RX231 - 100pin R5F52315AxFP | 32 KB | 128 KB | 100 | | | ~ | | | | | |
| rget Device: R5F5110 |)1AxLM | | Create | R5F52315AxLA | 32 KB | 120 KB | | | ~ | ¥ | | | | | |
| | Unloc | k Devices | RX Sir | R5F52315CxFP | 32 KB | 128 KB | | | ~ | ¥ | | | | | |
| Endian: Little | | \sim | | R5F52315CxLA | 32 KB | 128 KB | | | ✓ | ✓ | | | | | |
| Project Type: Default | | \sim | Create | R5F52316AxFP | 32 KB | 256 KB | | | ~ | ~ | | | | | |
| | | | | R5F52316AxLA R5F52316CxFP | 32 KB 32 KB | 256 KB 256 KB | 100 100 | | 3 | \$ | | | | | |
| | | | | R5F52316CxLA | 32 KB | 256 KB | 100 | | 2 | ~ | | | | | |
| | | | | R5F52317AxFP | 64 KB | 384 KB | 100 | | ✓ | Image: A set of the set of the | | | | | |
| | | | | R5F52317AxLA | 64 KB | 384 KB | | | ~ | ~ | | | | | |
| | | | | R5F52317BxFP R5F52317BxLA | 64 KB 64 KB | 384 KB | 100 100 | | 3 | \$ | | | | | |
| 0 | < <u>B</u> ack | <u>N</u> ext > | | R5F52317BXLA R5F52318AxFP | 64 KB | 384 KB 512 KB | 100 | | ž | ž | | | | 7 8 | - 0 |
| | | | | R5F52318AxLA | 64 KB | 512 KB | 100 | | 1 | ¥ | | | | | |
| | | Descript | tion | R5F52318BxFP | 64 KB | 512 KB | 100 | | ✓ | × | | ion | Ţ | ype | |
| | | | | | | | | | | | | | | | |
| | | | | R5F52318BxLA | 64 KB | 512 KB | 100 | | ~ | ~ | ~ | | | | |
| | | | | | 64 KB | 512 KB | | | ~ | ~ | ~ | | | | |
| | | | | R5F52318BxLA | 64 KB | 512 KB | | DK | • | ✓ | ~ | | | | |
| ms selected | | | | | 64 KB | 512 KB | | ж | ✓ | ancel | ~ | | | | • 7 3 |
| | | | | 0 | | 512 KB | | ж | | ancel | ~ | | | : 🕫 🛄 🕫 | |
| ems selected | | | | | 64 KB | 512 KB | | DK | ✓ | ancel | ~ | | | | - 2 0 |
|) | xecutable Project | | | 0 | 8 | 512 KB | | | | | ~ | | | ··· 🕫 📖 🖻 | |
|) ew Renesas CC-RX E | 111 | | | 0 | () New I | | CC-RX I | Executat | | | ~ | | | n ⊡ € ~ :: - | |
| ew Renesas CC-RX Es elect toolchain, device 8 | 111 | | | 0 | Select | Renesas t t Coding A | CC-RX I | Executal: settings | | | • | | | a ⊡ 6 ∿ ∺ — | |
| ew Renesas CC-RX Es ielect toolchain, device 8 foolchain Settings | 111 | | | 0 | Selec | Renesas t t Coding A e Smart Co | CC-RX I Assistant | Executal: settings | ole Proje | | • | | | a ⊡ 6 ∿ ∺ — | |
| ew Renesas CC-RX E ielect toolchain, device 8 foolchain Settings anguage: © C (| ያ debug settings | | | 0 | New I Select Us Us | Renesas t Coding A e Smart Ct e Peripher | CC-RX I Assistant | Executal: settings for Generator | ole Proje ≜ | ct | ↓ ↓ | the for | | _ | |
| ew Renesas CC-RX Ex elect toolchain, device 8 oolchain Settings anguage: © C (oolchain: Renes | & debug settings | | | 0 | New I Selec Us Us Smart FIT Co | Renesas t Coding A e Smart Ce e Peripher Configurao figurator | CC-RX I Assistant onfigurat al Code tor is a s | Executal: settings cor Generator ingle User | e Interface | ct that com | | | ctional | ities of Code | C General |
| ew Renesas CC-RX E elect toolchain, device 8 oolchain Settings anguage: © C (| & debug settings | > | | 0 | Select Select Smart FIT Cou | Renesas t Coding A e Smart Ce e Peripher Configura nfigurator les. | CC-RX I Assistant al Code tor is a s which in | Executal: settings for Generator ingle User nports, co | a le Proje | ct that com | ates o | lifferent | ictionali t types o | ities of Code | Genera d middle |
| ew Renesas CC-RX Ex elect toolchain, device 8 oolchain Settings anguage: © C (oolchain: Renes oolchain Version: v3.04.1 | & debug settings | > | | 0 | New I Select Us Us Smart FIT Co modul Smart pin co | Renesas t Coding A e Smart Ce e Peripher Configura nfigurator ies. Configura | CC-RX I Assistant onfigurat al Code tor is a s which in tor enco n view. | Executal: settings for Generator ingle User inports, co mpasses u | De Proje Interface Interface Interface Interface Interface | ct that com nd genera ck configu | ates o uratio | lifferent on view, | ictionali t types (interru | ities of Code of drivers an ot configura | Generat d middle |
| ew Renesas CC-RX Ex elect toolchain, device 8 oolchain Settings anguage: © C (polchain: Reness polchain Version: v3.04.1 TOS: None | & debug settings | > | | 0 | New I Select Us Smart FIT Co Modul Smart pin co Hardw | Renesas t Coding A e Smart Cc e Peripher Configura nfigurator les. Configura nfiguratio rare resou | CCC-RX II Assistant al Code 1 tor is a s which in tor enco n view. | Executal: settings cor Generator ingle User nports, co mpasses u lict in peri | Interface nfigures a unified clo ipheral me | ct that com nd genera ck config odules, inf | ates o uratio | lifferent on view, | ictionali t types (interru | ities of Code | Generat d middle |
| Aw Renesas CC-RX Ex alect toolchain, device 8 anguage: © C (polchain: Reness polchain: Version: v3.04.1 TOS: None | k debug settings C++ as CCRX 00 <u>Manage Tool</u> | v v Ichains v | | ? - - × × | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configura nfigurator ies. Configura | CC-RX I Assistant tor is a s which in tor enco. reces confi | Executal settings cor Generator ingle User nports, co mpasses u lict in perimodules w | Interface | ct that com nd genera ck config odules, int | ates o uratio terruj | different on view, ots and j | ictionali t types (interru | ities of Code of drivers an ot configura | Generat d middle |
| w Renesas CC-RX Ex elect toolchain, device 8 solchain Settings anguage: © C (solchain: Renes solchain Version: v3.04.1 TOS: None TOS Version: | k debug settings C++ as CCRX 00 <u>Manage Tool</u> | v v Ichains v | -52318 | ? - - × × | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX I Assistant tor is a s which in tor enco. reces confi | Executate settings for Generator ingle User nports, co mpasses u lict in perimodules v aialable or | a ple Proje a linerface a nified clo ipheral m iil be not aly for the | ct that com nd genera ck configi odules, int ified. supported | ates d uratio terruj d dev | different on view, ots and p rices) | interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| w Renesas CC-RX Es elect toolchain, device 8 polchain Settings nguage: @ C (polchain: Renes polchain: Renes polchain: V3.04.1 TOS: None TOS Version: v3.04.1 version: v3.04.1 | k debug settings C++ as CCRX 00 <u>Manage Tool</u> | v khains se "R5F | | | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX I Assistant tor is a s which in tor enco. reces confi | Executate settings for Generator ingle User nports, co mpasses u lict in perimodules v aialable or | Interface | ct that com nd genera ck configi odules, int ified. supported | ates d uratio terruj d dev | different on view, ots and p rices) | interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| w Renesas CC-RX Es dect toolchain, device 8 olchain Settings nguage: © C (olchain: Renes olchain Version: v3.04.1 rOS: None rOS Version: evice Settings rget Board: Custom | k debug settings | ichains Se "R5F | Create Han | ? | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcces confileware i lieware i lieware i ator is av | Executate settings for Generator ingle User nports, co mpasses u lict in perimodules v aialable or | Interface interface infigures a unified cloc ippheral m ill be not ill be not illl be not ill be not illl be n | ct that com nd genera ck config field. supporte: | ates d uratio terruj d dev | different on view, ots and p rices) | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| w Renesas CC-RX E) elect toolchain, device 8 bolchain Settings inguage: C (lolchain: Renes bolchain Version: v3.04.1 TOS: None TOS Version: evice Settings irget Board: Custom | k debug settings C++ ass CCRX Manage Tool Chooss ownlog - additional b | ichains v se "R5F | Create Han E2 Lite (RX | ? . <td< td=""><td>Smart FIT Co Market Smar</td><td>Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd</td><td>CC-RX II Assistant tor is a s which in tor enco n view. Tcces confileware i lieware i lieware i ator is av</td><td>Executal settings or Generator mpasses tu licit in peri modules v aulable or U Driver ar</td><td>Interface</td><td>ct that com nd genera ck configgi odules, ini fifed. supporte pplica ware</td><td>uration terrup d dev tion</td><td>different on view, ots and p vices) N</td><td>ictional t types of interrup pins occ</td><td>ities of Code of drivers an ot configura curred in diff</td><td>Generat d middle</td></td<> | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcces confileware i lieware i lieware i ator is av | Executal settings or Generator mpasses tu licit in peri modules v aulable or U Driver ar | Interface | ct that com nd genera ck configgi odules, ini fifed. supporte pplica ware | uration terrup d dev tion | different on view, ots and p vices) N | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| w Renesas CC-RX E) elect toolchain, device 8 bolchain Settings unguage: © C (bolchain: Renes bolchain Version: v3.04.1 TOS: None TOS Version: evice Settings urget Board: Custom | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | ichains. Chains. See "R5F Soards. | Create Han E2 Lite (RX | ? | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcces confileware i lieware i lieware i ator is av | Executal settings or Generator mpasses tu licit in peri modules v aulable or U Driver ar | Interface | ct that com nd genera ck configgi odules, ini fifed. supporte pplica ware | uration terrup d dev tion | different on view, ots and p vices) | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| ew Renesas CC-RX E) elect toolchain, device 8 oolchain Settings anguage: ① C (oolchain: Renes oolchain Version: v3.04.1 TOS: None TOS Version: V3.04.1 TOS: None TOS Version: V3.04.1 TOS: None arget Board: Custom Device Settings arget Device: RSF52318 | k debug settings C++ ass CCRX Manage Tool Chooss ownlog - additional b | ichains. Chains. See "R5F Soards. | Create Han E2 Lite (RX | ? | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcces confileware i lieware i lieware i ator is av | Executal settings for Generator ingle User mpasses (lict in per anilable or Univer an Configur and Se | Interface Int | ct that com d genera ck config ified. supporter pplica ware vare set set ann | terru d dev tion Mc ected | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| w Renesas CC-RX E) elect toolchain, device 8 oolchain Settings anguage: © C (oolchain: Renes oolchain: Renes oolchain Version: v3.04.1 TOS: None TOS Version: v3.04.1 TOS: None TOS Version: v3.04.1 TOS: Custom roget Board: Custom parget Device: RSF52318 Endian: Little | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | se "R5F | Create Ham E2 Lite (RX Create Deb RX Simula | ? | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcces confileware i lieware i lieware i ator is av | Executal settings for Generator ingle User mpasses (lict in per anilable or Univer an Configur and Se | Interface | ct that com d genera ck config ified. supporter pplica ware vare set set ann | terru d dev tion Mc ected | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| ew Renesas CC-RX E) elect toolchain, device 8 oolchain Settings anguage: © C (oolchain: Renes oolchain Version: v3.04.1 tTOS: None tTOS Version: Device Settings arget Board: Custom Date Custom Custom Custom | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | se "R5F | Create Ham E2 Lite (RX Create Deb RX Simula | The second se | Smart FIT Co Market Smar | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcres confi leware to ator is av | Executal settings for Generator ingle User mpasses (lict in per anilable or Univer an Configur and Se | Interface Int | ct that com d genera ck config ified. supporter pplica ware vare set set ann | terru d dev tion Mc ected | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generat d middle |
| w Renesas CC-RX E) elect toolchain, device 8 oolchain Settings anguage: © C (oolchain: Renes oolchain: Renes oolchain Version: v3.04.1 TOS: None TOS Version: v3.04.1 TOS: None TOS Version: v3.04.1 TOS: Custom roget Board: Custom parget Device: RSF52318 Endian: Little | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | v ichains. v se "R5F v soards. v bevices v | Create Han E2 Lite (RX Create Deb RX Simula Create Rele | Image: Configuration of the second guarding of the second gu | Select Vew I Select Smart FIT Co Mardw Smart Pin Co Hardw drivers | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcres confi leware to ator is av | Executal settings for Generator mpasses to ilicit in peri ailable or Driver ar Configur and Se | Interface Int | ct that com d genera ck config ified. supporter pplica ware vare set set ann | terru d dev tion Mc ected | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generation view |
| w Renesas CC-RX E) lect toolchain, device 8 olchain Settings nguage: C (olchain: Renes olchain Version: v3.04.1 COS: None COS Version: evice Settings rget Board: Custom prget Device: R5F52318 Endiar: Little | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | v ichains. v se "R5F v soards. v bevices v | Create Ham E2 Lite (RX Create Deb RX Simula | Image: Configuration of the second guarding of the second gu | Select Vew I Select Smart FIT Co Mardw Smart Pin Co Hardw drivers | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcres confi leware to ator is av | Executal settings for Generator mpasses to ilicit in peri ailable or Driver ar Configur and Se | Interface Int | ct that com d genera ck config ified. supporter pplica ware vare set set ann | terru d dev tion Mc ected | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generation view |
| w Renesas CC-RX E) lect toolchain, device 8 olchain Settings nguage: C (olchain: Renes olchain Version: v3.04.1 OS: None OS Version: v3.04.1 OS: None Svice Settings rget Board: Custom prget Device: R5F52318 Endian: Little | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | v ichains. v se "R5F v soards. v bevices v | Create Han E2 Lite (RX Create Deb RX Simula Create Rele | Image: Configuration of the second guarding of the second gu | Select Vew I Select Smart FIT Co Mardw Smart Pin Co Hardw drivers | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcres confi leware to ator is av | Executal settings for Generator mpasses to ilicit in peri ailable or Driver ar Configur and Se | Interface Int | ct that com d genera ck config ified. supporter pplica ware vare set set ann | terru d dev tion Mc ected | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generation view |
| v Renesas CC-RX Eb ect toolchain, device 8 Jolchain Settings Iguage: ① C (Jolchain: Renes Dolchain Version: v3.04.1 DS: None DS Version: v3.04.1 DS: None DS Version: v3.04.1 DS: Custom get Board: Custom Dget Device: R5F52318 Endian: Little | k debug settings C++ ass CCRX Manage Tool Chooss ownloar additional t AxFP | v ichains. v se "R5F v soards. v bevices v | Create Harr E2 Lite (RX Create Deb RX Simulai Create Rele | Image: Configuration of the second guarding of the second gu | Select Vew I Select Smart FIT Co Mardw Smart Pin Co Hardw drivers | Renesas t Coding A e Smart Ce e Peripher Configurator les. Configuratio rare resou and midd | CC-RX II Assistant tor is a s which in tor enco n view. Tcres confi leware to ator is av | Executal settings for Generator ingle User mpasses L Lict in per modules w nodules w Driver ar Driver and Ge | Interface Int | ct that com d genera ck config ified. supporter pplica ware vare set set ann | ates of uration terrup d dev tion tion are | different on view, ots and p vices) n odules in GUI ported | ictional t types of interrup pins occ | ities of Code of drivers an ot configura curred in diff | Generation view |



3.3 Importing the Project

This section describes the procedure to import the sample program project into the created workspace.

- 1. In the Project Explorer, select the project created in 3.2 "Creating a Project".
- 2. Open the Smart Browser.

| Select the project. e Hardw C/C++ C/C++ Code Generator Debug Project Explore E S 7 8 = 0 Sample [HardwareDebug] Partner OS Solution Toolkit Click "Smart Browser". Solution Toolkit Configurator Fitzing Fitz onfigurator | File Edit Source Refactor Navigate Search | Project | Renesas Views Run Window | Help |
|---|---|-------------|--------------------------|------------------------|
| Image: Second | Select the project. | e Hardw | C/C++ | > 🖢 🕶 🗞 🕶 📾 🕤 🥔 👩 🕶 🚳 |
| Project Explore Partner OS Sample [HardwareDebug] Partner OS Smart Configurator Click "Smart Browser". Solution Toolkit Partner OS | | 185 E Di . | Code Generator | > |
| > is sample [HardwareDebug] Renesas QE Click "Smart Browser". Smart Configurator Solution Toolkit > & Conflicts View | | 998 I III / | Debug | > |
| Smart Configurator Solution Toolkit Sview | | | Partner OS | > |
| | > 🗁 sample [HardwareDebug] | | | Click "Smart Browser". |
| Tracing > 💯 FIT onfigurator | | | Solution Toolkit | > 🤌 Conflicts View |
| 🕷 Renesas Software Installer 📃 🔍 Optimization Assistant | | | | |
| Smart Browser | | | | 🦗 Smart Browser |

- 3. Click the Application Notes tab in the Smart Browser tab.
- 4. Click the Refresh icon.

| levice: - Context Help User's Manual Technical Update Application Notes Tool News Notifications | | Ŧ |
|--|---|------------------|
| Click the Application Notes tab. | 3 show. d click "Refresh" button to get information of project's device, itton to get information of any devices. | Click "Refresh". |

5. When the Region Setting dialog appears, select the region where you are working and click OK.





6. Select this application note from the list and right click on it. Then select "Sample Code (import projects)" from the context menu.^{*1}

| Open Sample Code (download) | Click. |
|--------------------------------|--------|
| Sample Code (import projects) | Click. |
| Property | |

Note: 1. If verification by My Renesas has never been done, "My Renesas" dialog opens before downloading the file. Enter the mail address and the password you registered on the Renesas website.

| 🗿 My Renesas Logiı | | MyF | × Renesas |
|---|--|--|----------------------|
| Enter the details us Email address: Password: | d to register your My Renesas account. | <u>Forqot pa</u> <u>Proxy</u> | assword? Settings |
| samples for sel | ccount? <u>Registering at My Renesas</u> is ease ected products, submit technical support wnloads, document update notification: | requests and get access t s and more. | |

7. Click the Accept button.

| - | Code) | × |
|--|--|---|
| acknowledge agreement to the (the "Agreement"), or by down using all or any portion of the I "Licensed Software"), (a) you a whom you are authorized to ac Licensee is legally bound by the you have the right, power, and | atton or other button or mechanism designed to e terms of an electronic copy of the <u>Disclaimer002</u> hloading, installing, accessing, or otherwise copying or licensed software described in the Agreement (the accept the Agreement on behalf of the licensee for ct (the "Licensee"), and acknowledge that the e Agreement, and (b) you represent and warrant that d authority to act on behalf of and bind the Licensee. AGREE TO THE TERMS CONTAINED IN THIS OT HAVE THE RIGHT, POWER, AND AUTHORITY TO | |
| AGREEMENT, OR IF YOU DO N | | |



- 8. Save the application note.
- 9. RX130 group: Select "update_firmware_rx130" and "firmware_rx130" from the Project section and click the Finish button

RX140 group: Select "update_firmware_rx140" and "firmware_rx140" from the Project section and click the Finish button

RX231 group: Select "update_firmware_rx231" and "firmware_rx231" from the Project section and click the Finish button

RX261 group: Select "update_firmware_rx261" and "firmware_rx261" from the Project section and click the Finish button

10. After the project is imported, delete the project created for using the smart browser ("sample" in this document) as it is not necessary anymore.



3.4 Changed Setting Information

In this application note, settings in the configuration file and the project have been changed for each FIT module to set up the sample programs. The details are described in the sub-sections.

3.4.1 Configuration Option

The configuration options to set up the sample programs have been changed for each FIT module.

For items of the configuration options and their settings, refer to the document in the doc folder of each FIT module.

The following describes changes in the configuration options of Smart Configurator.

(1) Modifying the Flash FIT module

In the software component configuration screen, the following setting is changed to allow the Flash FIT module to reprogram the code flash memory.

| Property | Value |
|---|--|
| ✓ | |
| # Parameter check | Enable parameter checks |
| # Enable code flash programming | Includes code to program ROM area |
| # Enable BGO/Non-blocking data flash operations | Forces data flash API function to block until completed. |
| # Enable BGO/Non-blocking code flash operations | Forces ROM API function to block until completed. |
| # Enable code flash self-programming | Programming code flash while executing in RAM. |

(2) Modifying the SCI FIT module

In the software component configuration screen, the transmit data empty interrupt is set to be enabled.

| [| Property | Value |
|---|--------------------------|--------|
| | # Transmit end interrupt | Enable |
| 1 | | - |

In the software component configuration screen, the pins of RXD1 and TXD1 are set to be enabled.

| Property | Value | |
|---|--------|---|
| ~ 回 SCI | | _ |
| SCI1 | | |
| 🛰 SCK1 Pin | 🖾 Used | |
| RXD1/SMISO1/SSCL1 Pin | Used | |
| TXD1/SMOSI1/SSDA1 Pin | ✓ Used | |
| - CTS1#/RTS1#/SS1# Pin | 🖾 Used | |



(3) Modifying the SCI FIT module (RX231 group, RX261 group)

In the software component configuration screen, the SCI channel used in the SCI FIT module is changed from CH1 to CH5 for the RX231 group and CH6 for the RX261 group. * The figure below shows an example of using the RX231 group.

| Property | Value |
|--|---------|
| # Include software support for channel 0 | Not |
| # Include software support for channel 1 | Not |
| # Include software support for channel 2 | Not |
| # Include software support for channel 3 | Not |
| # Include software support for channel 4 | Not |
| # Include software support for channel 5 | Include |

In the software component configuration screen, the pins of RXD5 and TXD5 are set to be enabled for the RX231 group and the pins of RXD6 and TXD6 for the RX261 group. * The figure below shows an example of using the RX231 group.

| Property | Value | |
|---|--------|--|
| 🗸 🔲 SCI | | |
| 🗸 🔟 SCIO | | |
| 🛰 SCK0 Pin | Used | |
| RXD0/SMISO0/SSCL0 Pin | Used | |
| TXD0/SMOSI0/SSDA0 Pin | Used 🗌 | |
| CTS0#/RTS0#/SS0# Pin | Used | |
| 🗸 🖽 SCI1 | | |
| 🛰 SCK1 Pin | 🔲 Used | |
| NRXD1/SMISO1/SSCL1 Pin | Used | |
| TXD1/SMOSI1/SSDA1 Pin | 🔲 Used | |
| ► CTS1#/RTS1#/SS1# Pin | Used | |
| ~ 🗉 SCI5 | | |
| 🛰 SCK5 Pin | 🖾 Used | |
| ~ RXD5/SMISO5/SSCL5 Pin | Used | |
| TXD5/SMOSI5/SSDA5 Pin | ✓ Used | |
| CTS5#/RTS5#/SS5# Pin | 🖾 Used | |

In the pin configuration screen, the assignment is changed to PA3 for RXD5 and PA4 for TXD5 for the RX231 group; for the RX261 group, RXD6 is changed to PB0 and TXD6 is changed to PB1. * The figure below shows an example of using the RX231 group.

| Enabl | Function | Assignment | Pin Number | Direction | Remarks |
|--------------|----------|------------------------------------|-------------|-----------|---------|
| | CTS5# | Not assigned | Not assigne | None | |
| | RTS5# | Not assigned | Not assigne | None | |
| \checkmark | RXD5 | PA3/MTIOC0D/MTCLKD/TIOCD0/TCLKB/R | / 67 | 1 | |
| | SCK5 | Not assigned | Not assigne | None | |
| | SMISO5 | Not assigned | Not assigne | None | |
| | SMOSI5 | Not assigned | Not assigne | None | |
| | SS5# | Not assigned | Not assigne | None | |
| | SSCL5 | Not assigned | Not assigne | None | |
| | SSDA5 | Not assigned | Not assigne | None | |
| \checkmark | TXD5 | PA4/MTIC5U/MTCLKA/TMRI0/TIOCA1/TXI | / 66 | 0 | |



3.4.2 Modifying the Project Setting

The default build-time settings in the firmware update project have been changed to values described in Table 3.1 and Table 3.2, respectively. The default build-time settings in the firmware project have been changed to values described in Table 3.3, respectively.

You can confirm the settings changed with the following procedure:

 Right click on the target project (update_firmware_rx130 or firmware_rx130 for the RX130 group, update_firmware_rx140 or firmware_rx140 for the RX140 group, update_firmware_rx231 or firmware_rx231 for the RX231 group, and update_firmware_rx261 or firmware_rx261 for the RX261 group) in the Project Explorer and select Properties from the context menu. * The following screenshots illustrate an example configuration for the RX231 group.

| late_firmware_rx231 | [HardwareDebug] | |
|---------------------|---|---|
| | New Go Into | > |
| | Open in New Window Show In | Alt+Shift+W> |
| | Copy Paste | Ctrl+C Ctrl+V |
| | Celete Source | Delete > |
| | Move Rename | F2 |
| | a Import Export | |
| ł | Build Project Clean Project Refresh Close Project Close Unrelated Project | F5 |
| | Build Targets Index Build Configurations | > > > |
| | Run As Debug As Team Compare With Restore from Local History MISRA-C | >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> |
| * | C/C++ Project Settings Renesas C/C++ Project Settings Run C/C++ Code Analysis | Ctrl+Alt+P |
| | System Explorer Command Prompt Validate | Click. |
| | Configure Source | > > |
| | Properties | Alt+Enter |



2. Select "C/C++ Build" and then "Settings". * The following screenshots illustrate an example configuration for the RX231 group.





3. In the Tool Settings tab, confirm that settings have been changed to values described in Table 3.1, Table 3.2 and Table 3.3 for the firmware update project and the firmware project, respectively.

| ltem | Changed Item | Description | | |
|----------------------|--|---|--|--|
| Compiler - Source | Include path is added to the "Include file directories" section. | Add include paths each FIT module needs to specify. | | |
| Course | | When using the Smart Configurator to incorporate the FIT module, the include path is specified automatically. | | |
| | | Add include paths for the sample program. In this project, | | |
| | | "src/src_update_firmware", "src/src_update_firmware/r_fw_up_rx", and "src/src_update_firmware/r_xmodem" are added. | | |
| | | Setting example: | | |
| | | Include file directories (-include) | | |
| | | "\${workspace_loc:/\${ProjName}/src/src_update_firmware}" ^ "\${workspace_loc:/\${ProjName}/src/src_update_firmware/r_fw_up_rx}" "\${workspace_loc:/\${ProjName}/src/src_update_firmware/r_xmodem}" | | |
| Linker - Section | Sections RPFRAM and RPFW_UP_RAM are added to the RAM area. | Specify the RAM area to be used by the sample program. | | |
| | | Setting example: | | |
| | | Section Viewer | | |
| | | Address Section Name 0x00000004 SU | | |
| | | SI | | |
| | | B 1 | | |
| | | R 1 | | |
| | | B_2 | | |
| | | R_2 | | |
| | | B | | |
| | | R | | |
| | | RPFRAM | | |
| | | RPFW_UP_RAM | | |

 Table 3.1
 Changed Build Settings of the Project (Firmware Update Project) (1/2)



| ltem | Changed Item | Descripti | on | |
|---|--|--|---|-------------|
| - Section FW_UP_COMPLETE are a ROM area. The start address of the FV | Sections FW_UP_VER and FW_UP_COMPLETE are added to the ROM area. The start address of the FW_UP_VER section is set as follows: | alternate area of the start-up progra protection and place the P* section | | |
| | 0xFFFF6800 for RX130 group 0xFFFF6000 for RX140, RX231, and RX261 groups The start address of the C_1 section is set | Address 0xFFFF6000 0xFFFF6008 | Section Name FW_UP_VER C_1 C_2 C C\$* D* W* | |
| | as follows: 0xFFFF6808 for RX130 group 0xFFFF6008 for RX140, RX231, and RX261 groups | 0xFFFF6FF0 0xFFFFC000 0xFFFFFF80 0xFFFFFFFC | FW_UP_COMPLETE P* EXCEPTVECT RESETVECT | |
| | The start address of the FW_UP_COMPLETE section is set as follows: 0xFFFF73F0 for RX130 group 0xFFFF6FF0 for RX140, RX231, and RX261 groups | | | |
| | The start address of the P* section is set to 0xFFFFC000. | | | |
| Linker - Section - Symbol file | 'PFRAM=RPFRAM' and 'PFW_UP_RAM=RPFW_UP_RAM' are added to the "ROM to RAM mapped section" section. | sample p | ROM to RAM mapping rogram executes the p the code flash memor | program for |

Table 3.2 Changed Build Settings of the Project (Firmware Update Project) (2/2)



| Item | Changed Item | Description | | |
|-----------------------------|---|---|---|--|
| Compiler | Include path is added to the | Add include paths eacl | h FIT module needs to specify. | |
| - Source | "Include file directories" section. | When using the Smart Configurator to incorporate the | | |
| - Source | | - | • | |
| | | FIT module, the include | e path is specified | |
| | | automatically. | | |
| | | Add include paths for t | he sample program. | |
| | | | | |
| | | In this project, | <i>"</i> , , , , , , , , , , , , , , , , , , , | |
| | | "src/src_firmware" and | "src/src_firmware/r_fw_up_rx" | |
| | | are added. | | |
| | | Setting example: | | |
| | | | | |
| | | Include file directories (-include) | n 🗐 🍋 😨 🏹 🏂 | |
| | | "\${workspace_loc:/\${ProjName}/src/src_firm "\${workspace_loc:/\${ProjName}/src/src_firm | | |
| Linker | Sections RPFRAM and | | | |
| | | | to be used by the sample | |
| Section | RPFW_UP_RAM are added to the | program. | | |
| | RAM area. | Setting example: | | |
| | | Section Viewer | | |
| | | | | |
| | | Address Section Name | | |
| | | 0x0000004 SU | | |
| | | SI | | |
| | | B_1 | | |
| | | R_1 | | |
| | | B_2 | | |
| | | R_2 | | |
| | | В | | |
| | | R | | |
| | | RPFRAM | | |
| | | RPFW_UP_RAM | | |
| 0xFFFE0000 for RX130 group | code flash memory. | ram at the start address of the | | |
| | 0xFFF80000 for RX231 group and | Setting example: 0xFFF80000 C_1 C_2 C C\$* D* W* W* | | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C \$* D* | | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C \$* D* W* | | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C \$* D* W* L | | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C C * D* W* L L P* OxFFFFF80 EXCEPTVECT | | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C\$* D* W* L P* 0xFFFFFF80 0xFFFFFFFC RESETVECT 0xFFFFFFFC RESETVECT | ta in the following areas since | |
| | 0xFFF80000 for RX231 group and | OxFFF80000 C_1 C_2 C C C D* W* L P* OxFFFFFF80 EXCEPTVECT OxFFFFFFFC RESETVECT Note: Do not place da the firmware pro For RX130 grou | ogram cannot use these areas: p: | |
| | 0xFFF80000 for RX231 group and | OxFFF80000 C_1 C_2 C C C S* D* W* L P* OxFFFFFF80 EXCEPTVECT OxFFFFFFF80 EXCEPTVECT OxFFFFFFFC RESETVECT Note: Do not place da the firmware pro For RX130 grou Addresses FFFI | ogram cannot use these areas: ip: F_6800h to FFFF_BFFFh | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C\$* D* W* L P* 0xFFFFFF80 EXCEPTVECT 0xFFFFFFFC RESETVECT 0xFFFFFFFC Note: Do not place da the firmware profile For RX130 group Addresses FFFI For RX140, RX2 For RX140, RX2 | ogram cannot use these areas: ip: F_6800h to FFFF_BFFFh 231, and RX261 groups: | |
| | 0xFFF80000 for RX231 group and | 0xFFF80000 C_1 C_2 C C\$* D* W* L P* 0xFFFFFF80 EXCEPTVECT 0xFFFFFFFC RESETVECT 0xFFFFFFFC Note: Do not place da the firmware profile For RX130 group Addresses FFFI For RX140, RX2 For RX140, RX2 | ogram cannot use these areas: ip: F_6800h to FFFF_BFFFh | |
| Linker | 0xFFF80000 for RX231 group and | OxFFF80000 C_1 C_2 C C C C C C C C C C C C C C C C C C | ogram cannot use these areas: ip: F_6800h to FFFF_BFFFh 231, and RX261 groups: F_6000h to FFFF_BFFFh | |
| - | 0xFFF80000 for RX231 group and RX261 group 'PFRAM=RPFRAM' and | 0xFFF80000 C_1 C_2 C CS* D* W* L P* 0xFFFFFF80 EXCEPTVECT 0xFFFFFFFC RESETVECT 0xFFFFFFFC Note: Do not place da the firmware proposed to the firmwa | ogram cannot use these areas: ip: F_6800h to FFFF_BFFFh 231, and RX261 groups: F_6000h to FFFF_BFFFh mapping since the sample | |
| - Section | 0xFFF80000 for RX231 group and RX261 group 'PFRAM=RPFRAM' and 'PFW_UP_RAM=RPFW_UP_RAM' | OxFFF80000 C_1 C_2 C C C C C C C C C C C C C C C C C C | ogram cannot use these areas: p: F_6800h to FFFF_BFFFh 231, and RX261 groups: F_6000h to FFFF_BFFFh mapping since the sample program for switching the start- | |
| - | 0xFFF80000 for RX231 group and RX261 group 'PFRAM=RPFRAM' and | 0xFFF80000 C_1 C_2 C CS* D* W* L P* 0xFFFFFF80 EXCEPTVECT 0xFFFFFFFC RESETVECT 0xFFFFFFFC Note: Do not place da the firmware proposed to the firmwa | ogram cannot use these areas: p: F_6800h to FFFF_BFFFh 231, and RX261 groups: F_6000h to FFFF_BFFFh mapping since the sample program for switching the start- | |

| | | | / · · · |
|-----------|-------------------------------|----------------|--------------------|
| Table 3.3 | Changed Build Settings | of the Project | (Firmware Project) |



4. Operation Confirmation

4.1 Building the Project

Follow the procedure below to build the project and create the load module.

 Click the project to be built (update_firmware_rx130 or firmware_rx130 for RX130 group, update_firmware_rx140 or firmware_rx140 for RX140 group, update_firmware_rx231 or firmware_rx231 for RX231 group, and update_firmware_rx261 or firmware_rx261 for RX261 group). * The following screenshots illustrate an example configuration for the RX231 group.



2. Select Project >> Build Project.



3. The build is completed when the message "Build complete." is displayed in the Console panel.

| Renesas Optimizing Linker Completed 'Finished building target:' |
|--|
| 'Build complete.' |
| 14:31:32 Build Finished. 0 errors, 0 warnings. (took 1m:26s.888ms) |



4.2 Preparing Debugging

4.2.1 Preparing Devices

The evaluation board needs to be prepared before debugging.

Table 4.1 lists Devices and Configurations and Figure 4.1 shows the Debug Configuration (RX130/RX140/RX231) and Figure 4.2 shows the Debug Configuration (RX261).

| Table 4.1 | Devices and | Configurations |
|-----------|-------------|----------------|
|-----------|-------------|----------------|

| No. | Device | Remarks |
|-----|--|--|
| 1 | Development PC | PC used for development |
| 2 | Evaluation board (Renesas Starter Kit for RX130/RX140/RX231, Evaluation Kit for RX261 MCU Group) | The RX261 has a built-in E2 Lite, so an external E2 Lite is not required. |
| 3 | Host PC Serial communication software which is capable of XMODEM/SUM transfer | Development PC can be used as the host PC. |
| 4 | USB cable (Mini Type-B) | The serial I/O signals of the Renesas Starter Kit for RX130/RX140/RX231 are USB serial converted and can be used as a virtual COM port by connecting to a host PC via USB. |
| | USB cable (Micro Type-B) | The serial I/O signals of the Evaluation Kit for RX261 MCU Group are USB serial converted and can be used as a virtual COM port by connecting to a host PC via USB. |



Figure 4.1 Debug Configuration (RX130/RX140/RX231)





Figure 4.2 Debug Configuration (RX261)

4.2.2 Setting the Host PC

Table 4.2 lists the serial communication specification for the device and the host PC. For configuration of the terminal software, refer to the document for the terminal software.

| Description |
|----------------------------|
| Asynchronous communication |
| 115200 bps |
| 8 bits |
| None |
| 1 bit |
| None |
| |



4.3 Debugging the Project

Follow the procedure below to start debugging the firmware update project. The procedure can be used for the firmware project in the same manner.

1. Select Run >> Debug Configurations in the e^2 studio.





2. RX130 group:

Select "update_firmware_rx130 HardwareDebug" under "Renesas GDB Hardware Debugging". RX140 group:

Select "update_firmware_rx140 HardwareDebug" under "Renesas GDB Hardware Debugging". RX231 group:

Select "update_firmware_rx231 HardwareDebug" under "Renesas GDB Hardware Debugging". RX261 group:

Select "update_firmware_rx261 HardwareDebug" under "Renesas GDB Hardware Debugging".

Click the Debugger tab and then the Connection Settings tab. Change "EXTAL frequency value" to '8.0000' and "Power Target From The Emulator" to 'No'. * The following screenshots illustrate an example configuration for the RX231 group.

| | Click the Debugger tab. | | J. |
|---|---|---------------------|----------------|
| | | | 2 |
| 3 💀 🗎 🗶 🖻 🍸 👻 | Name: update_firmwaye_rx231 HardwareDebug | | |
| e filter text | 📄 Main 🔯 Debugger ⊳ Startup 🗖 Com 🛛 Click the Conne | ection Settings tab | |
| C/C++ Application | | cuon octungs tab. | |
| C/C++ Remote Application | Debug hardware: E2 Lite (RX) V Target Device: R5F52318 | | |
| EASE Script | | | |
| GDB Hardware Debugging | GDB Settings Connection Settings Debug Tool Settings | Change to '8.00 | <i>.</i> '0 |
| GDB OpenOCD Debugging | ✓ Clock | | \sim |
| GDB Simulator Debugging (RH850) | Main Clock Source | EXTAL | - |
| Java Applet | Extal Frequency[MHz] | 8.0000 | |
| Java Application | Operating Frequency [MHz] | | |
| Launch Group | Permit Clock Source Change On Writing Internal Flash Memory | y Yes | ~ |
| Remote Java Application | ✓ Connection with Target Board | | |
| Renesas GDB Hardware Debugging | Emulator | (Auto) | |
| firmware_rx231 HardwareDebug | Connection Type | Fine | 2 |
| update_firmware_rx231 HardwareDebug | JTag Clock Frequency[MHz] | 6.00 Change to 'No | ~ [,] |
| Renesas Simulative Debugging (RX, RL78) | Fine Baud Rate[Mbps] | 1.50 Change to No | J. |
| | Hot Plug | No | v |
| | ✓ Power | | |
| | Power Target From The Emulator (MAX 200mA) | No | ~ |
| | Supply Voltage (V) | 3.3 | ~ |
| Click "update_firmware_rx231 | ✓ CPU Operating Mode | | |
| HardwareDebug". | Register Setting | Single Chip | ~ |
| Taldwalebebug. | Mode pin | Single-chip mode | ~ |
| | Change startup bank | No | ~ |
| | Startup bank | Bank 0 | ~ |
| | ✓ Communication Mode | | |
| | Mode | Debug Mode | ~ _~ |
| | | | |
| | | | |
| r matched 14 of 16 items | | Revert Apply | ¥ |



Updating Firmware Using Start-Up Program Protection and Serial Communication

3. Click the Debug Tool Settings tab. Change "Debug the program re-writing the on-chip PROGRAM ROM" to 'Yes'. * The following screenshots illustrate an example configuration for the RX231 group.

| Debug Configurations eate, manage, and run configurations | Click the Debug Too | ol Sett | ings tab. | Ŕ |
|---|--|-----------|-----------------|-------|
| 🖻 🕫 🗎 🗶 🖻 🏹 🕶 | Name: update_firmware_rx231 HardwareDebug | | | |
| pe filter text | Main Debugger Startup Common Source | | | |
| C C/C++ Application C C/C++ Remote Application E EASE Script GDB Hardware Debugging | Debug hardware: E2 Lite (RX) V Target Device: R5F523 | 18 | | |
| GDB OpenOCD Debugging | ~ IO | | | ^ |
| GDB Simulator Debugging (RH850) | Use Default IO Filename | Yes | | ~ |
| 🗾 Java Applet | IO Filename | \${suppo | ort_area_loc} | |
| Java Application | ✓ General Debug | | | |
| 🖶 Launch Group | Reset After Reload | Yes | | ~ |
| 🖳 Remote Java Application | ✓ Memory | | | |
| Renesas GDB Hardware Debugging | Endian | Little Er | ndian | ~ |
| firmware_rx231 HardwareDebug | Verify On Writing To Memory | No | | ~ |
| update_firmware_rx231 HardwareDebug | Internal Flash Memory Overwrite | [0] | | |
| Renesas Simulator Debugging (RX, RL78) | External Memory Areas | [0] | | |
| | Work RAM Start Address | 0x1000 | Change to | 'Yes' |
| | Work RAM Size (Bytes) | 0x500 | | 100. |
| | ✓ System | | | |
| | Debug the program re-writing the on-chip PROGRAM RON | Yes | | ~ |
| | Debug the program re-writing the on-chip DATA FLASH | No | | ~ |
| | ✓ Start/Stop Function Setting | | | |
| | Execute function before running user program | No | | ~ |
| | Address for start function | 0x0 | | |
| | Execute function after stopping user program | No | | ~ |
| | Address for stop function | 0x0 | | |
| | Work RAM Start Address | 0xfdd0 | | |
| | | 0.000 | | * |
| | | | Re <u>v</u> ert | Apply |



Updating Firmware Using Start-Up Program Protection and Serial Communication

4. Click "..." button on the right side of "Internal Flash Memory Overwrite". * The following screenshots illustrate an example configuration for the RX231 group.

|) 🖻 🏚 🗎 🗮 🏳 🏹 👻 | Name: update_firmware_rx231 HardwareDebug | | |
|---|--|----------------------|-----|
| ype filter text | 🖹 Main 🌾 Debugger 🕨 Startup 🗔 Common 🦤 Source | | |
| C C/C++ Application C C/C++ Remote Application EASE Script GDB Hardware Debugging | Debug hardware: E2 Lite (RX) Target Device: R5F5231 GDB Settings Connection Settings Debug Tool Settings | 8 | |
| GDB GDB OpenOCD Debugging | ✓ IO | | ^ |
| GDB Simulator Debugging (RH850) | Use Default IO Filename | Yes | ~ |
| Java Applet | IO Filename | \${support area loc} | |
| Java Application | ✓ General Debug | | |
| Launch Group | Reset After Reload | Yes | ~ |
| Remote Java Application | ✓ Memory | Click. | |
| Renesas GDB Hardware Debugging | Endian | Little Endian | ~ |
| firmware_rx231 HardwareDebug | Verify On Writing To Memory | No | ~ ~ |
| update_firmware_rx231 HardwareDebug | Internal Flash Memory Overwrite | [0] | |
| Renesas Simulator Debugging (RX, RL78) | External Memory Areas | [0] | |
| Encloses simulator bebugging (IN, NETO) | Work RAM Start Address | 0x1000 | |
| | Work RAM Size (Bytes) | 0x500 | |
| | ✓ System | | |
| | Debug the program re-writing the on-chip PROGRAM ROM | Yes | ~ |
| | Debug the program re-writing the on-chip DATA FLASH | No | ~ |
| | Start/Stop Function Setting | | |
| | Execute function before running user program | No | ~ |
| | Address for start function | 0x0 | |
| | Execute function after stopping user program | No | ~ |
| | Address for stop function | 0x0 | |
| | Work RAM Start Address | 0xfdd0 | |
| | | 0.000 | ~ |



5. Specify to execute overwrite operation after erasing all blocks in the code flash memory. Click the Deselect All button and click OK.

Г



6. Click the Debug button. * The following screenshots illustrate an example configuration for the RX231 group.

| | | | 1 |
|--|--|----------------------|-------|
| 2 🕫 🗎 🗶 🖻 🏹 🗸 | Name: update_firmware_rx231 HardwareDebug | | |
| pe filter text | 📄 Main 🏷 Debugger 🕨 Startup 💷 Common 🦻 Source | | |
| E C/C++ Application E C/C++ Remote Application E EASE Script | Debug hardware: E2 Lite (RX) | 8 | |
| © GDB Hardware Debugging | GDB Settings Connection Settings Debug Tool Settings | | |
| GDB OpenOCD Debugging | × 10 | | ^ |
| GDB Simulator Debugging (RH850) | Use Default IO Filename | Yes | ~ |
| Java Applet | IO Filename | \${support_area_loc} | |
| Java Application | ✓ General Debug | | |
| Launch Group | Reset After Reload | Yes | ~ |
| Remote Java Application | ✓ Memory | | |
| Renesas GDB Hardware Debugging | Endian Little Endian | | ~ |
| firmware_rx231 HardwareDebug | Verify On Writing To Memory | No | ~ |
| update_firmware_rx231 HardwareDebug | Internal Flash Memory Overwrite | [0] | |
| Renesas Simulator Debugging (RX, RL78) | External Memory Areas | [0] | |
| | Work RAM Start Address | 0x1000 | |
| | Work RAM Size (Bytes) | 0x500 | |
| | ✓ System | | |
| | Debug the program re-writing the on-chip PROGRAM ROM | Yes | ~ |
| | Debug the program re-writing the on-chip DATA FLASH No | | ~ |
| | ✓ Start/Stop Function Setting | | |
| | Execute function before running user program | No | ~ |
| | Address for start function | 0x0 | |
| | Execute function after stopping user program | No | ~ |
| | Address for stop function | 0x0 | |
| | Work RAM Start Address | 0xfdd0 | ~ |
| | | 0.000 | |
| | Click "Debug". | Revert | Annly |
| ter matched 14 of 16 items | Chick Dobdy . | Revert | Apply |

When the following message appears, click Switch.

| © C | onfirm Perspective Switch X |] |
|------------|---|---------------|
| ? | This kind of launch is configured to open the Debug perspective when it suspends. | |
| | This Debug perspective supports application debugging by providing views for displaying the debug stack, variables and breakpoints. | |
| | Switch to this perspective? | |
| <u> </u> | member my decision | Click Switch. |
| | <u>Switch</u> | |



Updating Firmware Using Start-Up Program Protection and Serial Communication

When the load module has been downloaded, the Debug perspective opens. * The following screenshots illustrate an example configuration for the RX231 group.

| 🔹 🔳 🕈 Debug 🗸 🖻 update_firmware_rx231 Har 🗸 👌 📑 🔹 | 000-4-1 | 「雪」ダ ● ● ● ● ● ● ● ● ● ● ● | · · · 🖻 | | Q | 18 B C/ | ++ 🗘 Debu |
|--|------------------------------------|---|---------------------|------------------|-----------------|-----------|------------|
| ebug × 🗉 🐘 🖬 🖷 🗆 | i resetprg.c × | | 🕪 Varia 👒 Break 🛤 | Modul 눱 Proje 🐒 | Expre 🗣 Event 🗵 | 2 Perip | IOR " |
| update_firmware_rx231 HardwareDebug [Renesas GDB Hardware Debugging] | | void PowerON_Reset_PC(void) | | | × H % E | B PC: 0/- | OA: 0/2 |
| B update_firmware_nx231.x [1] [cores: 0] | 142 143 | <pre>/* Stack pointers are setup prior to calling this function - see comments above */</pre> | Туре | Address | Data | Count 1 | imer Chann |
| P Thread #1 1 (single core) [core: 0] (Suspended : Signal : SIGTRAP:Trace/breakp | 143 | /* stack pointers are setup prior to calling this function - see comments above */ | Trace Start | | | | |
| PowerON_Reset_PC() at resetprg.c:141 0xffffcc61 | 145 | /* Initialise the MCU processor word and Exception Table Register */ | Trace Stop | | | | |
| J nx-elf-gdb -nx-force-v2 (7.8.2) | 146 | <pre>#ifRENESAS_VERSION_ >= 0x01010000</pre> | Trace Reco | | | | |
| Renesas GDB server (Host) | 147 ffffcc6f | <pre>set_intb((void *)_sectop("C\$VECT"));</pre> | Event Brea | | | | |
| | 148 ffffcc78 | | C Timer Start | | | | |
| | 149 | <pre>>#else set_intb((unsigned long)_sectop("C\$VECT"));</pre> | Timer Stop | | | | |
| | 151 | <pre>set_into((unsigned long)_sectop(CSVEC()); set_extb((unsigned long)_sectop("EXCEPTVECT"));</pre> | | | | | |
| | 152 | fendif | | | | | |
| | 153 | | | | | | |
| | 154 | | | | | | |
| | 155 | /* Initialize FPSW for floating-point operations */ | | | | | |
| | 156 157 | <pre>#ifdefROZ #define FPU ROUND 0x00000001 /* Let FPSW RMbits=01 (round to zero) */</pre> | | | | | |
| | 158 | #delse PP_MUMAD 02000000001 /* Let PSA MODITS-001 (round to zero) */ | | | | | |
| | 159 | #define FPU ROULD 0x000000000 /* Let FPSW RMbits=00 (round to nearest) */ | | | | | |
| | 160 | Bendif | | | | | |
| | 161 | e #ifdefDOFF | | | | | |
| | 162 | #define FPU_DENOM 0x00008100 /* Let FPSW DNbit=1 (denorma) as zero) */ | | | | | |
| | 163 164 | * Melse #define FPU DENOM 0x000000000 /* Let FPSW DMbit=0 (denormal as is) */ | | | | | |
| | 165 | <pre>#define FPU_DENUM 0X000000000 /* Let HSW DNDIT=0 (genormal as is) */ #endif</pre> | | | | | |
| | 166 | PERMAT | | | | | |
| | 167 ffffcc85 | set fpsw(FPSW init FPU_ROUND FPU_DENOM); | | | | | |
| | 168 | And a second s | | | | | |
| | 169 | /* Switch to high-speed operation */ | | | | | |
| | 170 ffffcc88 171 ffffcc8b | operating_frequency_set(); | | | | | |
| | 171 +++++cc8b | usb_lpc_clock_source_select(); | | | | | |
| | 173 | /* If the warm start Pre C runtime callback is enabled, then call it. */ | | | | | |
| | 174 | #if BSP CFG USER WARM START CALLBACK PRE INITC ENABLED == 1 | | | | | |
| | 175 | BSP_CFG_USER_WARM_START_PRE_C_FUNCTION(); | | | | | |
| | 176 | Pendif | | | | | |
| | 177 | | | | | | |
| | 178 179 ffffcc8e | /* Initialize C runtime environment */ _INITSCT(): | | | | | |
| | 199 ********** | _inition(); | | | | | |
| | 181 | /* If the warm start Post C runtime callback is enabled, then call it. */ | < | | | | |
| | 182 | ■ #1 # RSP CEG USER WARM START CALLBACK POST INITC ENABLED == 1 | | | | · | |
| | | <u>۲</u> | Project Saved Templ | | | | |
| | Console × III F | egisters 💱 Problems 🧠 Smart Browser 🕷 Debugger Console 🔅 Debug Shell 🔋 Memory | | 8.3 | (法)派司部 | | • 🖻 • 🕾 |
| | update_firmware_rx | 31 HardwareDebug [Renesas GDB Hardware Debugging] | | | | | |
| | USB Bus Power | 5.0918 V | | | | | |
| | Finished target | connection | | | | | |
| | GDB: 55760 | | | | | | |
| | Target connect: | | | | | | |
| | Target connect: Starting downlo | | | | | | |
| | Finished downlo | | | | | | |
| | | du memory', has failed with error report, Firmware API CFFWInterface::FFWMCUCmd_DUMP failed with error: A timeout | error has occurred | in emulator firm | ware processin | c. | |
| | | | | | | e. | |
| | Hardware breaks | oint set at address 0xffffe060 | | | | | |
| | | | | | | | |
| | | | | | | | |

7. Click the Resume icon on the toolbar to execute the program. The program breaks at the beginning of the main function. * The following screenshots illustrate an example configuration for the RX231 group.

| File Edit Source Refactor Navigate Search Project | Renesas Views Run Window Help | |
|--|------------------------------------|--|
| 🐔 🐞 🔳 🕸 Debug 🛛 🗸 🖸 update_firmw | rare_rx231 Har 🗸 🔅 📑 🕶 🔚 🐚 🖷 🖛 🗞 | • 🗟 📲 🖉 🧶 🧶 • 🖉 🗣 🔍 • 🖉 • 🖉 • 🖉 • 🖉 • |
| > Debug × | 🖻 🦌 i 🏘 🕴 🗖 🚺 resetprg.c 🛛 | 7 |
| C update_firmware_rx231 HardwareDebug [Renesas GE W update_firmware_rx231x [1] [core: 0] M Thread #1 1 (single core) [core: 0] (Suspended P OwerON. Reset_PC() at resetprg.c:141 0xffff M rx-elf-gdb -rx-force-v2 (7.8.2) Renesas GDB server (Host) | : Signal : SIGTRAP:Trace/breakp | <pre>{ /* Stack pointers a setup prior to calling this fu Click /* Initialise the MCU processor word and Exception T #ifRENESAS_VERSION_ >= 0x01010000 set_intb((void *)_sectop("C\$VECT")); </pre> |



8. Click the Resume icon on the toolbar again after the break in step 7. * The following screenshots illustrate an example configuration for the RX231 group.



9. Confirm the following message is output in the terminal software.

RX130 group:

```
RX130 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute program
>
```

RX140 group:

```
RX140 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Execute program
>
```

RX231 group:

```
RX231 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute program
>
```

RX261 group:

```
RX261 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute program
>
```


5. Application Overview

5.1 Configuration of the Firmware Update Program

This section explains the configuration of the firmware update program, which is a sample program for this application note. This program is stored in the default area of the start-up program protection. Constants data (such as initial values of variables and string literals) are stored in the constants data area in ROM. There are two constants data areas: constants data area 1 and constants data area 2, and constants data are stored in either one of them.

A constants data area consists of three areas: version information storage area, constants data storage area, and write complete information storage area.

Figure 5.1 shows Memory Map of the Firmware Update Program, Table 5.1 shows Configuration of the Constants Data Area.



Figure 5.1 Memory Map of the Firmware Update Program

| Table 5.1 | Configuration of the Constants Data Area |
|-----------|--|
|-----------|--|

| Area Name | Description |
|---|--|
| Version information storage area | Area for storing the version information of the firmware update program. |
| Constants data storage area | Area for storing constants data |
| Write complete information storage area | Area into which the version information is programmed when the firmware update program is updated. |



5.2 Operation Overview

This section explains the sample programs; firmware update program and firmware program in this application note.

The firmware update program is stored in the default area of the start-up program protection and the constants data area 1. The firmware update program receives the firmware program (.mot file) through the serial communication using the XMODEM/SUM protocol and program the firmware into the code flash memory. Then the default area and the alternate area of the star-up program protection are switched temporarily to program the code flash memory except the default area and the constants data area. Since this can protect the firmware update program, if programming the firmware failed, for example, due to temporary blackout, the firmware firmware still can be updated by restarting the firmware update program.

The firmware program is programmed in the area other than the default area of the start-up program protection area and constants data area using the firmware update program. The firmware program outputs the message to the host PC using the serial communication. When the firmware receives the command from the host PC, it switches between the default area and the alternate area, and then executes a software reset. This causes the firmware update program to restart.

Table 5.2 lists the Functional Comparison of Sample Programs.

| Function | Firmware Update Program | Firmware Program |
|---|-------------------------|------------------|
| Erasing/programming the code flash memory | Supported | Not supported |
| Switching the start-up program protection area and software reset | Supported | Supported |

Table 5.2 Functional Comparison of Sample Programs

5.2.1 Programming the Firmware Update Program

Follow the procedure in 4.3 Debugging the Project to program the firmware update program in the default area of the start-up program protection area and the constants data area. Alternatively, start in boot mode and use the Renesas Flash Programmer to program the firmware update program in the default area of the start-up program protection area and the constants data area.

For details on using the Renesas Flash Programmer, refer to the user's manual for the Renesas Flash Programmer.



5.2.2 Programming the Firmware Program

The following procedure describes the flow of operation to program the firmware program using the firmware update program.

1. The firmware update program is launched in single-chip mode. The firmware update program starts up the SCI and outputs the menu displayed in the terminal software on the host PC.



Figure 5.2 Launching the Firmware Update Program (for RX231 Group)



2. The firmware program update command is sent from the terminal software to program the firmware program. The firmware update program places the flash memory rewrite processing into the on-chip RAM so that programming the code flash memory is performed from the on-chip RAM.



Figure 5.3 Sending the Firmware Program Update Command (for RX231 Group)



3. The firmware update program branches to flash memory rewrite processing in the on-chip RAM and erases the alternate area and the firmware program area. After erasing the code flash memory, it returns to the firmware update program in the default area.



Figure 5.4 Erasing the Code Flash Memory (for RX231 Group)



4. The firmware program is transmitted using the terminal software. The firmware update program analyzes the received data and stores the data for programming the code flash memory into the write buffer of the on-chip RAM.



Figure 5.5 Transmitting the Firmware Program (for RX231 Group)



5. When the write buffer in the on-chip RAM becomes full with the data, the firmware update program branches to flash memory rewrite processing in the on-chip RAM. Flash memory rewrite processing temporarily switches between the default area and the alternate area of the start-up program protection according to the setting of the flash initial setting register (FISR). Then it programs the data in the write buffer to the code flash memory. After the data has been programmed, the start-up program protection area are switched back to the original.



Figure 5.6 Programming the Firmware Program (for RX231 Group)



- 6. Steps 4 and 5 are repeated until all firmware program data are programmed.
- 7. After the firmware program is programmed, the firmware update program branches to firmware start-up processing. The firmware update program switches between the default area and the alternate area of the start-up program protection permanently according to the extra area setting, and then executes a software reset. The firmware program is launched.



Figure 5.7 Software Reset and Launching the Firmware Program (for RX231 Group)



5.2.3 Updating the Firmware Program

This section describes the flow of operation to update the firmware program using the firmware update program.

1. The default area and the alternate area of the start-up program protection are switched permanently according to the extra area setting by the firmware program which has not yet been updated. Then the device is reset.



Figure 5.8 Operation of the Firmware Program Before Being Updated (for RX231 Group)



2. The firmware update program is launched. The firmware update program starts up the SCI and outputs the menu displayed in the terminal software on the host PC.



Figure 5.9 Launching the Firmware Update Program (for RX231 Group)



3. The firmware program update command is sent from the terminal software to update the firmware program. The firmware program update program places the flash memory rewrite processing into the on-chip RAM so that programming the code flash memory is performed from the on-chip RAM.



Figure 5.10 Sending the Firmware Program Update Command (for RX231 Group)



4. The firmware update program branches to flash memory rewrite processing in the on-chip RAM and erases the alternate area and the firmware program area. After erasing the code flash memory, it returns to the firmware update program in the default area.



Figure 5.11 Erasing the Code Flash Memory (for RX231 Group)



5. The new firmware program is transmitted using the terminal software. The firmware update program analyzes the received data and stores the data for programming the code flash memory into the write buffer of the on-chip RAM.



Figure 5.12 Transmitting the Firmware Program (for RX231 Group)



RX100/RX200 Series Updating Firmware Using Start-Up Program Protection and Serial Communication

6. When the write buffer in the on-chip RAM becomes full with the data, the firmware update program branches to flash memory rewrite processing in the on-chip RAM. Flash memory rewrite processing temporarily switches between the default area and the alternate area of the start-up program protection according to the setting of the flash initial setting register (FISR). Then the data stored in the write buffer is programmed into the code flash memory. After the programming has been completed, the start-up program protection areas are switched back to the original.



Figure 5.13 Programming the Firmware Program (for RX231 Group)



- 7. Steps 5 and 6 are repeated until all new firmware program data are programmed.
- 8. After the new firmware program is programmed, the firmware update program branches to firmware startup processing. The firmware update program switches between the default area and the alternate area of the start-up program protection permanently according to the extra area setting, and then executes a software reset. The updated firmware program is launched.



Figure 5.14 Software Reset and Launching the Updated Firmware Program (for RX231 Group)



5.2.4 Updating the Firmware Update Program

This section describes the flow of operation to update the firmware update program using the firmware update program. Note that the firmware program on the code flash memory is erased when updating the firmware update program. Thus the firmware program must be programmed after updating the firmware update program.

1. After the default area and the alternate area of the startup program protection have been switched permanently according to the extra area setting by the firmware program, the device is reset.



Figure 5.15 Firmware Program Operation (for RX231 Group)



2. The firmware update program is launched. The firmware update program starts up the SCI and outputs the menu displayed in the terminal software on the host PC.



Figure 5.16 Launching the Firmware Update Program (for RX231 Group)



3. The firmware update program update command is sent from the terminal software to update the firmware update program. The firmware update program places the flash memory rewrite processing in the on-chip RAM so that programming the code flash memory is performed from the on-chip RAM.



Figure 5.17 Sending the Firmware update program Update Command (for RX231 Group)



4. The firmware update program branches to flash memory rewrite processing in the on-chip RAM and erases the alternate area and the constants data area 2. After erasing the code flash memory, it returns to the firmware update program in the default area.



Figure 5.18 Erasing the Code Flash Memory (for RX231 Group)



5. The new firmware update program is transmitted by the terminal software. The firmware update program before being updated analyzes the received data and stores the data for programming the code flash memory into the write buffer of the on-chip RAM.



Figure 5.19 Transmitting the Firmware Update Program (for RX231 Group)



RX100/RX200 Series Updating Firmware Using Start-Up Program Protection and Serial Communication

6. When the write buffer becomes full with the data in the on-chip RAM, the firmware update program branches to flash memory rewrite processing in the on-chip RAM. Flash memory rewrite processing temporarily switches between the default area and the alternate area of the start-up program protection according to the setting of the flash initial setting register (FISR). Then the data stored in the write buffer is programmed into the code flash memory. After the programming has been completed, the start-up program protection areas are switched back to the original.



Figure 5.20 Programming the Firmware Update Program (for RX231 Group)



- 7. Steps 5 and 6 are repeated until all data of the new firmware update program are programmed.
- 8. After the new firmware update program is programmed, the version information of the new firmware update program is retrieved from the version information storage area 2 in the constants data area 2, and then programmed into the write complete information storage area 2.



Figure 5.21 Programming write complete information storage area (for RX231 group)



9. After the version information is programmed into the write complete information storage area, the firmware update program branches to firmware start-up processing. The firmware update program before being updated switches between the default area and the alternate area of the start-up program protection permanently according to the extra area setting, and then executes a software reset. The updated firmware update program is launched.



Figure 5.22 Software Reset and Launching the Updated Firmware Update Program (for RX231 Group)



5.2.5 Firmware Program Operation

This section describes flow of operation to confirm the firmware update program operation using the firmware program.

1. The firmware program is programmed as described in 5.2.2 Programming the Firmware.



Figure 5.23 Programming and Launching the Firmware Program with the Firmware Update Program (for RX231 Group)



2. The firmware program is launched. The firmware program starts up the SCI and outputs the message to the terminal software on the host PC.



Figure 5.24 Outputting the Message (for RX231 Group)



3. The command is sent from the terminal software. The firmware program places processing for operation for the flash memory in the on-chip RAM so that switching the start-up program protection area and software reset are performed in the on-chip RAM.



Figure 5.25 Receiving the Command (for RX231 Group)



4. The firmware program switches between the default area and the alternate area of the start-up program protection permanently according to the extra area setting, and then executes a software reset. The firmware update program is launched.



Figure 5.26 Software Reset and Launching the Firmware Update Program (for RX231 Group)



5.2.6 Recovery when Programming Failed

When programming the code flash memory using the firmware update program and if the programming failed, for example, due to temporary blackout, the programming can be performed again by resetting the device. The following explains how this works:

The firmware update program temporarily switches between the default area and the alternate area of the start-up program protection by the setting of the flash initial setting register (FISR) before erasing and programming the code flash memory. Then it programs the code flash memory except the default area. The setting of the flash initial setting register (FISR) is initialized by resetting the device. The default area and the alternate area after the register initialization are determined according to the start-up area setting in the extra area. When programming the code flash memory by the firmware update program, the start-up area setting for the extra area is set to launch from the default area. Therefore resetting the device causes the firmware update program in the default area to be launched and enables programming the code flash memory again.



5.2.7 How to Create a Firmware Update Program for Updating

When creating a firmware update program to be used when updating the firmware update program, change the items shown below.

(1) Changing the section settings

In the section settings, change the start address of each section. The firmware update program for updating needs to store constants data in the constants data area that is not used by the running firmware update program. Therefore, if the program before the update is using constants data area 2, set the start address as shown in Table 5.3. If the program before the update is using constants data area 1, set the start address as shown in Table 5.4.

Table 5.3 Start Address of Each Section When Constants Data is Stored in Constants Data Area 1

| | Section | | | |
|-------------|------------|------------|----------------|--|
| Device | FW_UP_VER | C_1 | FW_UP_COMPLETE | |
| RX130 group | 0xFFFF6800 | 0xFFFF6808 | 0xFFFF73F0 | |
| RX140 group | 0xFFFF6000 | 0xFFFF6008 | 0xFFF6FF0 | |
| RX231 group | | | | |
| RX261 group | | | | |

| Table 5.4 | Start Address of Each | h Section When Constants | s Data is Stored in Constants | Data Area 2 |
|-----------|-----------------------|--------------------------|-------------------------------|-------------|
|-----------|-----------------------|--------------------------|-------------------------------|-------------|

| | Section | | | |
|-------------|------------|------------|----------------|--|
| Device | FW_UP_VER | C_1 | FW_UP_COMPLETE | |
| RX130 group | 0xFFFF7400 | 0xFFFF7408 | 0xFFFF7FF0 | |
| RX140 group | 0xFFFF7000 | 0xFFFF7008 | 0xFFFF7FF0 | |
| RX231 group | | | | |
| RX261 group | | | | |

(2) Changing the version information

Change the version information of the firmware update program. To change the version information, change the value of the constant FW_UP_PROGRAM_VERSION in main.c. Version information should be 4-digit hexadecimal number from 0 to 9.

The following shows version information setting examples:

• For Ver3.05:

| fine FW UP PROGRAM VERSION (0x0305 |
|------------------------------------|
|------------------------------------|

• For Ver10.20

#define FW UP PROGRAM VERSION (0x1020u)



5.3 Process Flowchart and Screen Output: Firmware Update Program

The firmware update program uses the serial communication to output the message to the terminal software on the host PC and branches to an appropriate processing according to the command input from the terminal software.

5.3.1 Main Processing

Main processing of the firmware update program initializes the SCI FIT module and the Flash FIT module, and uses the SCI to display the menu in the terminal software on the host PC. Then main processing waits for a key input from the terminal software and branches to an appropriate processing according to the key input.

(1) **Process flowchart**

Figure 5.27 shows the Flowchart of Main Processing.



Figure 5.27 Flowchart of Main Processing



(2) Screen output in the terminal software

After the firmware update program is launched, the following message is output.

RX130 group:

```
RX130 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute firmware
>
```

Figure 5.28 Screen Output by Main Processing (RX130 Group)

RX140 group:

```
RX140 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute firmware
>
```

Figure 5.29 Screen Output by Main Processing (RX140 Group)

RX231 group:

```
RX231 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute firmware
>
```

Figure 5.30 Screen Output by Main Processing (RX231 Group)

RX261 group:

```
RX261 firmware update using Start-Up Program Protection menu ver1.00
1...Update firmware program
2...Update firmware update program
3...Execute firmware
>
```

Figure 5.31 Screen Output by Main Processing (RX261 Group)

When '1' is entered, the firmware program is updated. The update mode is set to firmware program update mode. After the update mode is set, the process branches to the firmware update processing.

When '2' is entered, the firmware update program is updated. The update mode is set to either firmware update program update mode 2. After the update mode is set, the process branches to the firmware update processing.

When '3' is entered, the firmware start-up processing is executed.

When a line feed is entered, the menu is displayed again.



(3) Update Mode

When the firmware update program is used to update a program, the update mode shown in Table 5.5 is set to branch into the firmware update processing.

| Update Mode | Description |
|--------------------------------|--|
| Firmware program update mode | This mode updates the firmware program. |
| | Programming or erasing is performed only for the firmware program |
| | area and alternate area of the start-up program protection. |
| Firmware update program update | This mode updates the firmware update program. |
| mode 1 | Programming or erasing is performed only for the constants data area |
| | 1 and alternate area of the start-up program protection. |
| Firmware update program update | This mode updates the firmware update program. |
| mode 2 | Programming or erasing is performed only for the constants data area |
| | 2 and alternate area of the start-up program protection. |

Table 5.5 Overview of the Update Mode

5.3.2 Firmware Update Processing

When '1' or '2' is entered in main processing, firmware update processing is executed. The update program is received through the serial communication using the XMODEM/SUM protocol and then programmed into the code flash memory.

(1) **Process flowchart**

Figure 5.32 shows the Flowchart of Firmware Update Processing.





Figure 5.32 Flowchart of Firmware Update Processing



(2) Screen output in the terminal software

1. Confirmation message for updating

Firmware update processing outputs the following confirmation message for updating.

```
Erase flash memory and write firmware (Y/N)?
Figure 5.33 Screen Output to Confirm Updating
```

2. Downloading the file

When 'Y' or 'y' is entered, the processing erases the code flash memory, outputs the message as shown Erasing has been done. Start Xmodem download...

Figure 5.34, and waits for the firmware being received. Transmit the .mot file with the XMODEM/SUM protocol from the terminal software. For transmitting a file with the XMODEM/SUM protocol from the terminal software, refer to the user's manual for the terminal software.

Erasing has been done. Start Xmodem download...

Figure 5.34 Screen Output for Downloading the File

3. Completion of the firmware update

When the firmware has been updated, the following message is output.

```
Updating firmware has been done.
>
```

Figure 5.35 Screen Output upon Completion of the Firmware Update

4. Error output

If an error occurs during firmware update, any of the following message is output according to the error.

Figure 5.36 Screen Output upon Error Occurrence

5. Canceling the update

If a key other than 'Y' or 'y' is entered in response to the confirmation message for updating in 1. above, the following message is output and the update is canceled.

Command canceled.

>

Figure 5.37 Screen Output when Updating is Canceled



5.3.3 Firmware Start-up Processing

When '3' is entered in main processing, firmware start-up processing is executed. The firmware is launched by executing a software reset after switching the start-up program protection area.

If the firmware program or firmware update program is updated, firmware start-up processing is executed without displaying the menu.

(1) **Process flowchart**

Figure 5.38 shows the Flowchart of Program Start-up Processing.



Figure 5.38 Flowchart of Program Start-up Processing



(2) Screen output in the terminal software

1. Confirmation message for launching the program.

The firmware start-up processing outputs the following confirmation message for launching the firmware.

Execute firmware (Y/N)?

Figure 5.39 Screen Output for Launching the Firmware

2. Confirmation message for launching the firmware update program

The firmware start-up processing after the firmware update program is updated outputs the message as shown in Figure 5.40 to confirm the new firmware update program version.

Execute new firmware update program Ver[new-firmware-update-program-version](Y/N)?

Figure 5.40 Screen Output for Launching the New Firmware Update Program

3. Launching the firmware

If 'Y' or 'y' is entered in response to the message above, the processing outputs the following message, switches the start-up program protection area, and executes a software reset.

Switch Start-Up area and do software reset.

Figure 5.41 Screen Output for Switching the Start-up Program Protection Area and Software Reset

4. Error output

If an error occurs during launching the firmware, any of the following message is output according to the error.

Figure 5.42 Screen Output upon Error Occurrence

5. Canceling the firmware start-up

If a key other than 'Y' or 'y' is entered in response to the confirmation message for launching the firmware in 1. above, the following message is output and the update is canceled.

Command canceled. >

Figure 5.43 Screen Output when the Firmware Start-up is Canceled


5.4 **Process Flowchart and Screen Output: Firmware Program**

The firmware program uses the serial communication to output the message to the terminal software on the host. If a line feed is entered in the terminal software, the firmware switches the start-up program protection area, executes the software reset, and launch the firmware update program.

(1) **Process flowchart**

Figure 5.44 shows the Flowchart of the Firmware Program.



Figure 5.44 Flowchart of the Firmware Program



(2) Screen output in the terminal software

1. Firmware program start-up

The firmware program outputs the following message to inform that the firmware program is starting.

```
This program is the sample firmware.
Push Enter key to execute firmware update.
```

Figure 5.45 Screen Output when the Firmware Program is Starting-up

When a line feed is entered, the firmware switches the start-up program protection area, executes a software reset, and launch the firmware update program.

2. Error output

If an error occurs during launching the firmware update program, any of the following message is output according to the error.

Figure 5.46 Screen Output upon Error Occurrence



5.5 Detailed Information of the Firmware Update Program

5.5.1 File Composition

Table 5.6 lists the Files Used in the Firmware Update Program and Table 5.7 lists the Standard Include Files Used in the Firmware Update Program. Files generated by the FIT module and files generated by the integrated development environment are not included in these tables.

| File Name | Outline | |
|----------------------|---|--|
| main.c | Main source file | |
| main.h | Main interface file | |
| r_xmodem.c | XMODEM source file | |
| r_xmodem_if.h | XMODEM interface file | |
| r_fw_up_rx.c | Firmware update source file | |
| r_fw_up_rx_if.h | Firmware update interface file | |
| r_fw_up_rx_private.h | Firmware update header file | |
| r_fw_up_buf.c | Source file to process the firmware data buffer | |
| r_fw_up_buf.h | Header file to process the firmware data buffer | |

Table 5.6 Files Used in the Firmware Update Program

Table 5.7 Standard Include Files Used in the Firmware Update Program

| File Name | Outline |
|-----------|---|
| stdbool.h | Defines macros regarding the Boolean type and the Boolean value. |
| stdint.h | Defines macros by declaring the integer type of the specified width. |
| stdlib.h | Library for standard C programming processing such as storage area management |
| string.h | Library for processing such as string comparison and copy. |



5.5.2 Constants

Table 5.8 to Table 5.13 list constants used in the firmware update program.

| Constant | Setting Value | Description |
|----------------------------|-------------------------|-----------------------------------|
| FW_UP_PROGRAM_VERSION | (0x0100u) | Version information |
| DUMMY_DATA | (0xAA55AA55AA55AA55) | Dummy data to be stored |
| | | before the write complete |
| | | information storage area |
| MASK_NUM | (0x0Fu) | Mask for getting lower 4 bits |
| ASCII_CODE_NUM | (0x30u) | Character code of '0' |
| ASCII_CODE_POINT | (0x2Eu) | Character code of '.' |
| DELAY_NUM | (1u) | Delay time to be passed as |
| | | an argument to the |
| | | R_BSP_SoftwareDelay |
| | (| function |
| TIMEOUT_NUM | (10000u) | Count value for determining |
| | | the 10-second time out |
| | (4.) | period |
| RECV_BYTE_SIZE | (1u) | 1 byte size for receiving |
| SEND_BYTE_SIZE | (1u) | 1 byte size for transmitting |
| COMMAND_UPDATE_FIRM | ('1') | Character code for the input |
| | | command (for firmware |
| | (10) | program update command) |
| COMMAND_UPDATE_FIRM_UPDATE | ('2') | Character code for the input |
| | | command (for firmware |
| | | update program update command) |
| COMMAND_EXEC_PROGRAM | ('3') | Character code for the input |
| | (3) | command (for firmware start- |
| | | up command) |
| COMMAND_YES_UPPER | ('Y') | Character code for the input |
| | (1) | command ("Y") |
| COMMAND_YES_LOWER | ('y') | Character code for the input |
| | ()) | command ("y") |
| COMMAND CR | ('\r') | Character code for the input |
| | | command (line feed) |
| STRING_MAX_SIZE | RX130 group: | Maximum size for an output |
| | (SCI_CFG_CH1_TX_BUFSIZ) | • |
| | RX140 group: | - |
| | (SCI_CFG_CH1_TX_BUFSIZ) | |
| | RX231 group: | |
| | (SCI_CFG_CH5_TX_BUFSIZ) | |
| | RX261 group: | |
| | (SCI_CFG_CH6_TX_BUFSIZ) | |



| Constant | Setting Value | Description |
|------------------|---|--|
| XM_SOH | (0x01u) | XMODEM control code (SOH) |
| XM_EOT | (0x04u) | XMODEM control code (EOT) |
| XM_ACK | (0x06u) | XMODEM control code (ACK) |
| XM_NAK | (0x15u) | XMODEM control code (NAK) |
| XM_CAN | (0x18u) | XMODEM control code (CAN) |
| XM_HEADER_SIZE | (1+1+1) | Header size of the XMODEM data block (the number of bytes) |
| XM_DATA_SIZE | (128u) | Data size of the XMODEM data block (the number of bytes) |
| XM_SUM_SIZE | (1u) | Checksum size of the XMODEM data block (the number of bytes) |
| XM_BLOCK_SIZE | (XM_HEADER_SIZE + XM_DATA_SIZE + XM_SUM_SIZE) | XMODEM data block size (the number of bytes) |
| XM_RETRY_COUNT | (10u) | The number of retries upon XMODEM data transfer timeout |
| UINT8T_0 | (0u) | 0 in uint8_t |
| UINT8T_1 | (1u) | 1 in uint8_t |
| COMPLEMENT_CHECK | (0xFFu) | The numerical value to confirm the complement of the block number |

 Table 5.9 Constants Used in the Firmware Update Program (r_xmodem.c)



| Constant | Setting Value | Description |
|------------------------------|--|---|
| FW_UP_BINARY_BUF_SIZE | (256u) | Buffer size for data to be programmed in the code flash memory |
| FW_UP_BINARY_BUF_NUM | (2u) | The number of buffers for data to be programmed in the code flash memory |
| FW_UP_BUF_NUM | (60u) | The number of arrays to store analyzed Motorola S record data |
| FW_UP_FIRM_ST_ADDRESS | RX130 group: (FLASH_CF_BLOCK_127) RX140 group: (FLASH_CF_BLOCK_127) RX231 group: (FLASH_CF_BLOCK_255) RX261 group: (FLASH_CF_BLOCK_255) | Start address of the area to program the firmware |
| FW_UP_FIRM_EN_ADDRESS | RX130 group: (FLASH_CF_BLOCK_37 - 1) RX140 group: (FLASH_CF_BLOCK_19 - 1) RX231 group: (FLASH_CF_BLOCK_19 - 1) RX261 group: (FLASH_CF_BLOCK_19 - 1) | End address of the area to program the firmware |
| FW_UP_CONST_1_ST_ADDRESS | RX130 group: (FLASH_CF_BLOCK_37) RX140 group: (FLASH_CF_BLOCK_19) RX231 group: (FLASH_CF_BLOCK_19) RX261 group: (FLASH_CF_BLOCK_19) | Start address of the constants data area 1 |
| FW_UP_CONST_1_FIN_ST_ADDRESS | RX130 group: (FLASH_CF_BLOCK_34 - 256) RX140 group: (FLASH_CF_BLOCK_17 - 256) RX231 group: (FLASH_CF_BLOCK_17 - 256) RX261 group: (FLASH_CF_BLOCK_17 - 256) | Start address to be used for final programming processing in constants data area 1 |

Table 5.10 Constants Used in the Firmware Update Program (r_fw_up_rx_private.h) (1/3)



| Constant | Setting Value | Description |
|------------------------------|---------------------------|------------------------------|
| FW_UP_COMPLETE_1_ST_ADDRESS | RX130 group: | Start address of the write |
| | (FLASH_CF_BLOCK_34 - 8) | complete information |
| | RX140 group: | storage area 1 |
| | (FLASH_CF_BLOCK_17 - 8) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_17 - 8) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_17 - 8) | |
| FW_UP_CONST_1_EN_ADDRESS | RX130 group: | End address of the |
| | (FLASH_CF_BLOCK_34 - 1) | constants data area 1 |
| | RX140 group: | |
| | (FLASH_CF_BLOCK_17 - 1) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_17 - 1) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_17 - 1) | |
| FW_UP_CONST_2_ST_ADDRESS | RX130 group: | Start address of the |
| | (FLASH_CF_BLOCK_34) | constants data area 2 |
| | RX140 group: | |
| | (FLASH_CF_BLOCK_17) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_17) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_17) | |
| FW_UP_CONST_2_FIN_ST_ADDRESS | RX130 group: | Start address to be used for |
| | (FLASH_CF_BLOCK_31 - 256) | final programming |
| | RX140 group: | processing in constants |
| | (FLASH_CF_BLOCK_15 - 256) | data area 2 |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_15 - 256) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_15 - 256) | |
| FW_UP_COMPLETE_2_ST_ADDRESS | RX130 group: | Start address of the write |
| | (FLASH_CF_BLOCK_31 - 8) | complete information |
| | RX140 group: | storage area 2 |
| | (FLASH_CF_BLOCK_15 - 8) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_15 - 8) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_15 - 8) | |

 Table 5.11
 Constants Used in the Firmware Update Program (r_fw_up_rx_private.h) (2/3)



| Constant | Setting Value | Description |
|--------------------------|-------------------------|-----------------------------------|
| FW_UP_CONST_2_EN_ADDRESS | RX130 group: | End address of the constants |
| | (FLASH_CF_BLOCK_31 - 1) | data area 2 |
| | RX140 group: | |
| | (FLASH_CF_BLOCK_15 - 1) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_15 - 1) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_15 - 1) | |
| FW_UP_STUP_ST_ADDRESS | RX130 group: | Start address of the default area |
| | (FLASH_CF_BLOCK_15) | of the start-up program |
| | RX140 group: | protection |
| | (FLASH_CF_BLOCK_7) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_7) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_7) | |
| FW_UP_STUP_EN_ADDRESS | (FLASH_CF_BLOCK_END) | End address of the default area |
| FW_OF_STOF_EN_ADDRESS | (FLASH_CF_BLOCK_END) | of the start-up program |
| | | protection |
| FW_UP_FIRM_BLOCK_NUM | RX130 group: (90u) | The number of blocks for the |
| | RX140 group: (108u) | area to program the firmware |
| | RX231 group: (236u) | |
| | RX261 group: (236u) | |
| FW_UP_CONST_BLOCK_NUM | RX130 group: (3u) | The number of blocks for the |
| FW_UF_CONST_BLUCK_NUM | | constants data area |
| | RX140 group: (2u) | constants data area |
| | RX231 group: (2u) | |
| | RX261 group: (2u) | |
| FW_UP_STUP_BLOCK_NUM | RX130 group: (16u) | The number of blocks for the |
| | RX140 group: (8u) | default area of the start-up |
| | RX231 group: (8u) | program protection |
| | RX261 group: (8u) | |
| FW_UP_FIRM_RESETVECT | RX130 group: | Address of the reset vector of |
| | (FLASH_CF_BLOCK_15 – 4) | the firmware program |
| | RX140 group: | |
| | (FLASH_CF_BLOCK_7-4) | |
| | RX231 group: | |
| | (FLASH_CF_BLOCK_7 – 4) | |
| | RX261 group: | |
| | (FLASH_CF_BLOCK_7 – 4) | |
| FW UP BLANK VALUE | (0xFFFFFFFF) | Read value when the code flash |
| | (| memory is blank. |
| PRCR_KEY | (0xA500u) | Key code of PRCR register |
| SET_PRC1 | (0x0002u) | Value for setting the PRC1 bit of |
| | · / | 5 |
| | | the PRCR register |
| MCU_RESET | (0xA501u) | A value that is set in the SWRR |

Table 5.12 Constants Used in the Firmware Update Program (r_fw_up_rx_private.h) (3/3)

| Constant | Setting Value | Description |
|------------------------|---------------|--|
| MOT_S_CHECK_SUM_FIELD | (0x02u) | The number of characters for the checksum field in the Motorola S-record format |
| ADDRESS_LENGTH_S1 | (0x04u) | The number of characters for the address field in the Motorola S-record format (S1 type) |
| ADDRESS_LENGTH_S2 | (0x06u) | The number of characters for the address field in the Motorola S-record format (S2 type) |
| ADDRESS_LENGTH_S3 | (0x08u) | The number of characters for the address field in the Motorola S-record format (S3 type) |
| BUF_LOCK | (1u) | The specified buffer of Motorola S-record format is locked. |
| BUF_UNLOCK | (0u) | The specified buffer of Motorola S-record format is open. |
| MOT_RECORD_S0 | (0u) | Record type in Motorola S-record format (S0 type) |
| MOT_RECORD_S1 | (1u) | Record type in Motorola S-record format (S1 type) |
| MOT_RECORD_S2 | (2u) | Record type in Motorola S-record format (S2 type) |
| MOT_RECORD_S3 | (3u) | Record type in Motorola S-record format (S3 type) |
| MOT_RECORD_S7 | (7u) | Record type in Motorola S-record format (S7 type) |
| MOT_RECORD_S8 | (8u) | Record type in Motorola S-record format (S8 type) |
| MOT_RECORD_S9 | (9u) | Record type in Motorola S-record format (S9 type) |
| MASK_LOWER_BYTE | (0x000000FFu) | Mask for getting lower 1 byte. |
| ASCII_CODE_0 | (0x30u) | Character code of '0'. |
| ASCII_CODE_9 | (0x39u) | Character code of '9'. |
| ASCII_CODE_UPPER_A | (0x41u) | Character code of 'A'. |
| ASCII_CODE_UPPER_F | (0x46u) | Character code of 'F'. |
| ASCII_CODE_LOWER_A | (0x61u) | Character code of 'a'. |
| ASCII_CODE_LOWER_F | (0x66u) | Character code of 'f'. |
| CONVERT_HEX_NUM | (0x0Fu) | Value for converting character code from 0 to 9 to hexadecimal. |
| CONVERT_HEX_UPPER_CHAR | (0x37u) | Value for converting character code from A to F to hexadecimal. |
| CONVERT_HEX_LOWER_CHAR | (0x57u) | Value for converting character code from a to f to hexadecimal. |

| Table 5.13 | Constants Used in the Firmware Update Program (r_fw_up_buf.h) |
|------------|---|
| | oonotanto oooa in tho i inninaro opaato i rogram (i_in_up_bann) |



5.5.3 **Type Definitions**

Figure 5.47 to Figure 5.50 show type definitions used in the firmware update program.

```
typedef enum e xmodem proc stage
{
     XMODEM PROC END = 0,
     XMODEM PROCESSING,
     XMODEM SOH_RECEIVED
} e xmodem proc stage t;
typedef struct st xmodem states
{
    uint8_t retry_counter;
uint8_t expected_block_number;
uint8_t recv_buf_index;
uint8_t can_counter;
uint8_t * p_recv_buf;
e xmodem_proc_starse t retry_counter;
     e_xmodem_proc_stage_t proc_stage;
    xm_recv_func_t recv_func;
xm_send_func_t send_func;
xm_exec_func_t exec_func;
} st_xmodem_states_t;
```

Figure 5.47 Type Definitions Used in the Firmware Update Program (r_xmodem.c)

```
typedef enum e xmodem err
{
   XMODEM SUCCESS,
   XMODEM SEND ERR,
   XMODEM RECV ERR,
   XMODEM TIMEOUT,
   XMODEM PROC BLOCK ERR,
   XMODEM RECV CAN,
   XMODEM DATA ERR
} e xmodem err t;
typedef e xmodem err t (*xm recv func t)(uint8 t* p arg);
typedef e xmodem err t (*xm send func t)(uint8 t arg);
typedef e_xmodem_err_t (*xm_exec_func_t)(const uint8_t* p_buf, uint16_t size);
```

Figure 5.48 Type Definitions Used in the Firmware Update Program (r xmodem if.h)



```
typedef enum e update mode t
{
   UPDATE FW,
   UPDATE FW UP 1,
   UPDATE FW UP 2,
} update mode t;
typedef enum e fw up return t
{
   FW UP SUCCESS,
   FW UP ERR OPENED,
   FW UP ERR NOT OPEN,
   FW UP ERR NULL PTR,
   FW UP ERR INVALID RECORD,
   FW UP ERR BUF FULL,
   FW UP ERR BUF EMPTY,
   FW UP ERR INITIALIZE,
   FW UP ERR ERASE,
   FW UP ERR WRITE,
   FW UP ERR VERIFY,
   FW UP ERR SWITCH AREA,
   FW UP ERR INVALID ADDRESS,
   FW UP ERR INVALID RESETVECT,
   FW UP ERR INTERNAL
} fw up return t;
typedef struct st fw up fl data t
{
   uint32 t src addr;
   uint32 t dst addr;
   uint32 t len;
   uint16 t count;
} fw_up_fl_data_t;
```

Figure 5.49 Type Definitions Used in the Firmware Update Program (r_fw_up_rx_if.h)



```
typedef enum fw up mot s cnt t
{
    STATE MOT S RECORD MARK = 0,
   STATE MOT S RECORD TYPE,
   STATE_MOT_S_LENGTH_1,
   STATE MOT S LENGTH 2,
   STATE MOT S ADDRESS,
   STATE MOT S DATA,
    STATE MOT S CHKSUM 1,
    STATE MOT S CHKSUM 2
} fw up mot s cnt t;
typedef struct MotSBufS
   uint8_t addr_length;
uint8_t data_length;
uint8_t * p_address;
uint8_t * p_data;
uint8_t type;
uint8_t act;
{
    struct MotSBufS * p_next;
} fw up mot s buf t;
typedef struct WriteDataS
{
   uint32_t addr;
uint32_t len;
uint8_t data[FW_UP_BINARY_BUF_SIZE];
   struct WriteDataS * p next;
    struct WriteDataS * p prev;
} fw up write data t;
```

```
Figure 5.50 Type Definitions Used in the Firmware Update Program (r_fw_up_buf.h)
```



5.5.4 Variables

Table 5.14 to Table 5.17 list static variables and Table 5.18 to Table 5.19 lists const variables.

Table 5.14 static Variables Used in the Firmware Update Program (main.c)

| Туре | Variable | Description | Function |
|-------------------------|---------------------|-------------------------------|--------------------------------|
| static uint8_t | s_update_mode | Update mode | main |
| | | | block_proc_xm |
| | | | update_firmware |
| | | | exec_firmware |
| | | | set_write_complete_information |
| static uint8_t | s_update_complete_f | Update complete flag | main |
| | lag | | update_firmware |
| | | | exec_firmware |
| static sci_hdl | s_sci_handle | SCI module control | main |
| | | handle | send_byte_xm |
| | | | recv_byte_xm |
| | | | update_firmware |
| | | | exec_firmware |
| | | | send_string_sci |
| static volatile bool | s_sci_send_end_flag | SCI transmit complete | sci_callback |
| | | determination flag | send_string_sci |
| static volatile int32_t | s_timeout_count | Timeout determination counter | recv_byte_xm |

Table 5.15 static Variable Used in the Firmware Update Program (r_xmodem.c)

| Туре | Variable | Description | Function |
|----------------|---------------------------|----------------------------|-------------|
| static uint8_t | s_recv_buf[XM_BLOCK_SIZE] | XMODEM receive data buffer | exec_xmodem |

Table 5.16 static Variable Used in the Firmwaer Update Program (r_fw_up_rx.c)

| Туре | Variable | Description | Function |
|-------------|-------------|--|--|
| static bool | s_is_opened | Firmware update initialization complete flag | fw_up_open fw_up_close fw_up_put_data fw_up_get_data erase_firmware write_firmware switch_start_up_and_reset |



| Туре | Variable | Description | Function | |
|------------------------------|---------------------------------------|---|---|--|
| static fw_up_mot_s_buf_t | *sp_app_put_mot_s_buf | Pointer to the Motorola S-record data buffer currently used for Motorola S format analysis processing | fw_up_buf_init fw_up_put_mot_s | |
| static fw_up_mot_s_buf_t | *sp_app_get_mot_s_buf | Pointer to the Motorola S-record data buffer currently used for processing to create data to be programmed into the code flash memory | fw_up_buf_init fw_up_get_binary | |
| static fw_up_mot_s_buf_t | s_mot_s_buf[FW_UP_ BUF_NUM] | Buffer to store the contents of the Motorola S-record data | fw_up_buf_init fw_up_memory_init | |
| static fw_up_write_data_t | *sp_app_write_buf | Pointer to the current data buffer for programming the code flash memory | fw_up_buf_init fw_up_get_binary | |
| static fw_up_write_data_t | s_write_buf[FW_UP_ BINARY_BUF_NUM] | Buffer to store the data for programming the code flash memory | fw_up_buf_init | |
| static fw_up_mot_s_cnt_t | s_mot_s_data_state | Analysis state of the Motorola S-record data | fw_up_buf_init fw_up_put_mot_s | |
| static uint32_t | s_write_current_address | Current address to program in the code flash memory | fw_up_buf_init fw_up_get_binary | |
| static bool | s_detect_terminal_flag | Detection flag for the endpoint of the record | fw_up_buf_init fw_up_put_mot_s fw_up_get_binary | |

| Table 5.17 | static Variables Used in the Firmware Update Program | (r_fw_up_buf.c) |
|------------|--|-----------------|



| Туре | Variable | Description | Function |
|----------------------|-----------------------------|--|--|
| const uint32_t | g_program_version | Firmware update program version | show_menu_start_up |
| const uint64_t | g_dummy_data | | — |
| static const uint8_t | s_string_menu0[] | RX130group: "RX130 firmware update using Start-Up Program Protection menu ver" RX140group: "RX140 firmware update using Start-Up Program Protection menu ver" RX231 group: "RX231 firmware update using Start-Up Program Protection menu ver" RX261 group: "RX261 firmware update using Start-Up Program Protection menu ver" | show_menu_start_up |
| static const uint8_t | s_string_menu1[] | "1…Update firmware program∖r∖n" | show_menu_start_up |
| static const uint8_t | s_string_menu2[] | "2Update firmware update program\r\n" | show_menu_start_up |
| static const uint8_t | s_string_menu3[] | "3Execute firmware\r\n" | show_menu_start_up |
| static const uint8_t | s_string_input[] | "> " | show_menu_start_up update_firmware exec_firmware |
| static const uint8_t | s_string_crlf[] | "\r\n" | main update_firmware exec_firmware |
| static const uint8_t | s_string_update[] | "Erase flash memory and write firmware (Y/N)?" | update_firmware |
| static const uint8_t | s_string_erase_success[] | "Erasing has been done.\r\n" | update_firmware |
| static const uint8_t | s_string_download [] | "Start Xmodem download…\r\n" | update_firmware |
| static const uint8_t | s_string_finish_xmodem[] | "Updating firmware has been done.\r\n" | update_firmware |
| static const uint8_t | s_string_exec_firm[] | "Execute firmware (Y/N)?" | exec_firmware |
| static const uint8_t | s_string_reset[] | "Switch Start-Up area and do software reset.\r\n" | exec_firmware |
| static const_uin8_t | s_string_exec_firm_update[] | "Execute new firmware update program Ver" | exec_firmware |
| static const uint8_t | s_string_y_n[] | " (Y/N)?" | exec_firmware |
| static const uint8_t | s_string_cancel[] | "Command canceled.\r\n" | update_firmware exec_firmware |

 Table 5.18 const Variables Used in the Firmware Update Program (main.c) (1/2)



Updating Firmware Using Start-Up Program Protection and Serial Communication

| Туре | Variable | Description | Function | |
|----------------------|----------------------------|--|-----------------|--|
| static const uint8_t | s_string_flash_err[] | "Flash module error.\r\n" | main | |
| static const uint8_t | s_string_erase_err[] | "Erasing error.\r\n" | update_firmware | |
| static const uint8_t | s_string_set_info_err[] | "Set write complete information error.\r\n" | update_firmware | |
| static const uint8_t | s_string_send_err[] | "Send error.\r\n" | update_firmware | |
| static const uint8_t | s_string_recv_err[] | "Receive error.\r\n" | update_firmware | |
| static const uint8_t | s_string_timeout[] | "Timeout.\r\n" | update_firmware | |
| static const uint8_t | s_string_block_err[] | "Block processing error.\r\n" | update_firmware | |
| static const uint8_t | s_string_data_err[] | "Data error.\r\n" | update_firmware | |
| static const uint8_t | s_string_fin_update_err[] | "Finalize update error.\r\n" | update_firmware | |
| | | | exec_firmware | |
| static const uint8_t | s_string_init_update_err[] | "Initialize update error.\r\n" | update_firmware | |
| | | | exec_firmware | |
| static const uint8_t | s_string_resetvect_err[] | "Reset vector of the firmware is invalid.\r\n" | exec_firmware | |
| static const uint8_t | s_string_switch_err[] | "Switching Start-Up area error.\r\n" | exec_firmware | |
| | | | | |

Table 5.19 const Variables Used in the Firmware Update Program (main.c) (2/2)



5.5.5 Functions

Table 5.20 lists the Functions Used in the Firmware Update Program, Table 5.21 lists the FIT Module Functions Used in the Firmware Update Program, Table 5.22 to Table 5.25 lists the e² studio Smart Configurator Generated Function Used in the Firmware Update Program.



| Function | Description | Defined File |
|--------------------------------|--|---------------|
| main | Main processing | main.c |
| show_menu_start_up | Displaying menu | main.c |
| sci_callback | Callback function for the SCI FIT module to check completion of an SCI transmission | main.c |
| send_byte_xm | Callback function for XMODEM protocol to transmit 1-byte data | main.c |
| recv_byte_xm | Callback function for XMODEM protocol to receive 1-byte data | main.c |
| block_proc_xm | Callback function for XMODEM protocol for data processing of 1-data block | main.c |
| update_firmware | Firmware update processing | main.c |
| exec_firmware | Firmware start-up processing | main.c |
| send_string_sci | Transmitting strings | main.c |
| set_write_complete_information | Programming the version information to the write complete information storage area | main.c |
| show_version | Display version | main.c |
| exec_xmodem | XMODEM protocol processing | r_xmodem.c |
| xmodem_recv_soh | Receiving the header of XMODEM protocol data block | r_xmodem.c |
| xmodem_check_eot | Checking the header of XMODEM protocol data block | r_xmodem.c |
| xmodem_recv_block | Receiving 1-data block of XMODEM protocol | r_xmodem.c |
| xmodem_analyze_block | Analyzing XMODEM protocol data block | r_xmodem.c |
| xmodem_proc_data | Processing data for 1 data block of XMODEM protocol | r_xmodem.c |
| xmodem_send_response | Response for XMODEM protocol | r_xmodem.c |
| fw_up_open_flash | Flash FIT module initialization | r_fw_up_rx.c |
| fw_up_open | Firmware update initialization | r_fw_up_rx.c |
| fw_up_close | Completing firmware update | r_fw_up_rx.c |
| copy_update_ramprog | Copying RAM program | r_fw_up_rx.c |
| analyze_and_write_data | Analyzing receive data and programming code flash memory | r_fw_up_rx.c |
| fw_up_put_data | Analyzing receive data | r_fw_up_rx.c |
| fw_up_get_data | Obtaining programming data for the code flash memory | r_fw_up_rx.c |
| erase_firmware | Erasing the code flash memory | r_fw_up_rx.c |
| write_firmware | Programming the code flash memory | r_fw_up_rx.c |
| switch_start_up_and_reset | Switching the start-up program protection area and software reset | r_fw_up_rx.c |
| fw_up_buf_init | Initializing buffer for firmware update | r_fw_up_buf.c |
| fw_up_memory_init | Initializing pointer to the buffer | r_fw_up_buf.c |
| fw_up_put_mot_s | Analyzing Motorola S-record data | r_fw_up_buf.c |
| fw_up_get_binary | Obtaining programming data for the code flash memory | r_fw_up_buf.c |
| fw_up_ascii_to_hexbyte | Converting ASCII to binary | r_fw_up_buf.c |

Table 5.20 Functions Used in the Firmware Update Program



| Table 3.21 TTT Module Functions Osed in the Finnware opdate Frogram | | | | |
|---|------------------|--|--------------------------------|--|
| Function | FIT Module | Application | Function (Used in) | |
| R_FLASH_Open | Flash FIT module | Initializing the Flash FIT module | fw_up_open_flash | |
| R_FLASH_Erase | Flash FIT module | Erasing the code flash memory | erase_firmware | |
| R_FLASH_Write | Flash FIT module | Programming the code | write_firmware | |
| | | flash memory | set_write_complete_information | |
| R_FLASH_Control | Flash FIT module | Switching the start-up | erase_firmware | |
| | | program protection area | write_firmware | |
| | | | switch_start_up_and_reset | |
| R_SCI_Open | SCI FIT module | Starting up the SCI | main | |
| R_SCI_Control | SCI FIT module | Enabling the transmit end interrupt | main | |
| R_SCI_Send | SCI FIT module | Transmitting the SCI data | send_byte_xm | |
| | | - | send_string_sci | |
| R_SCI_Receive | SCI FIT module | Receiving the SCI data | main | |
| | | | recv_byte_xm | |
| | | | update_firmware exec_firmware | |

Table 5.21 FIT Module Functions Used in the Firmware Update Program

Table 5.22 e² studio Smart Configurator Generated Function Used in the Firmware Update Program (RX130 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI1 | SCI FIT module | Pin setting for the SCI | main |

Table 5.23 e² studio Smart Configurator Generated Function Used in the Firmware Update Program (RX140 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI1 | SCI FIT module | Pin setting for the SCI | main |

Table 5.24 e² studio Smart Configurator Generated Function Used in the Firmware Update Program
(RX231 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI5 | SCI FIT module | Pin setting for the SCI | main |

Table 5.25 e² studio Smart Configurator Generated Function Used in the Firmware Update Program
(RX261 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI6 | SCI FIT module | Pin setting for the SCI | main |



5.6 Detailed Information of the Firmware

5.6.1 File Composition

Table 5.26 lists the Files Used in the Firmware and Table 5.27 lists the Standard Include Files Used in the Firmware. Files generated by the FIT module and files generated by the integrated development environment are not included in these tables.

| File Name | Outline | |
|------------------------------------|---|--|
| main.c | Main source file | |
| main.h ^{*1} | Main interface file | |
| r_fw_up_rx.c ^{*1} | Firmware update source file | |
| r_fw_up_rx_if.h ^{*1} | Firmware update interface file | |
| r_fw_up_rx_private.h ^{*1} | Firmware update header file | |
| r_fw_up_buf.c ^{*1} | Source file to process the firmware data buffer | |
| r_fw_up_buf.h ^{*1} | Header file to process the firmware data buffer | |

Table 5.26Files Used in the Firmware

Note: 1. This is the same file used in the firmware update program.

Table 5.27 Standard Include Files Used in the Firmware

| File Name | Outline | |
|-----------|--|--|
| stdbool.h | Defines macros regarding the Boolean type and the Boolean value. | |
| stdint.h | Defines macros by declaring the integer type of the specified width. | |
| stdlib.h | Library for standard C programming processing such as storage | |
| | area management | |
| string.h | Library for processing such as string comparison and copy. | |

5.6.2 Constants

Table 5.28 lists Constants Used in the Firmware (main.c). For constants defined in the same file as the firmware update program, refer to 5.5.2 Constants.

| Constant | Setting Value | Description |
|-----------------|--|---|
| RECV_BYTE_SIZE | (1) | 1 byte size for receiving |
| COMMAND_CR | ('\r') | Character code for the input command: line feed |
| STRING_MAX_SIZE | RX130 group: (SCI_CFG_CH1_TX_BUFSIZ) RX140 group: (SCI_CFG_CH1_TX_BUFSIZ) RX231 group: (SCI_CFG_CH5_TX_BUFSIZ) RX261 group: (SCI_CFG_CH6_TX_BUFSIZ) | Maximum size for an output string |

Table 5.28 Constants Used in the Firmware (main.c)

5.6.3 Type Definitions

For type definitions, refer to 5.5.3 Type Definitions.



5.6.4 Variables

Table 5.29 lists static Variables Used in the Firmware (main.c) and Table 5.30 lists const Variables Used in the Firmware (main.c). For variables defined in the same file as the firmware update program, refer to 5.5.4 Variables.

| Table 5.29 static Variables Used in the Firmware (main.c) | | | |
|---|---------------------|-------------------------------------|-----------------|
| Туре | Variable | Description | Function |
| static sci_hdl | s_sci_handle | SCI module control handle | main |
| | | | send_string_sci |
| static volatile bool | s_sci_send_end_flag | SCI transmit complete determination | sci_callback |
| | | flag | send_string_sci |

Table 5.20 static Variables Llead in the Firmware (main c)

| Table 5.50 Const variables Osed in the Firmware (main.c) | | | |
|--|----------------------------|---|--------------------|
| Туре | Variable | Description | Function |
| static const uint8_t | s_string_menu0[] | "This program is the sample firmware.\r\n" | show_menu_start_up |
| static const uint8_t | s_string_menu1[] | "Push Enter key to execute firmware update.\r\n" | show_menu_start_up |
| static const uint8_t | s_string_input[] | "> " | show_menu_start_up |
| static const uint8_t | s_string_crlf[] | "\r\n" | main |
| static const uint8_t | s_string_reset[] | "Switch Start-Up area and do software reset.\r\n" | main |
| static const uint8_t | s_string_flash_err[] | "Flash module error.\r\n" | main |
| static const uint8_t | s_string_switch_err[] | "Switching Start-Up area error.\r\n" | main |
| static const uint8_t | s_string_init_update_err[] | "Initialize update error.\r\n" | main |
| static const uint8_t | s_string_fin_update_err[] | "Finalize update error.\r\n" | main |
| static const uint8_t | s_string_resetvect_err[] | "Reset vector of the firmware update is invalid.\r\n" | main |

Table 5.30 const Variables Used in the Firmware (main.c)



5.6.5 Functions

Table 5.31 lists the Functions Used in the Firmware, Table 5.32 lists the FIT Module Functions Used in the Firmware, Table 5.33 to Table 5.36 lists the e^2 studio Smart Configurator Generated Function Used in the Firmware. These tables do not include functions that are defined in the same file as the firmware update program and are not used by the firmware.

| Function | Description | Defined File |
|---------------------------|--|---------------|
| main | Main processing | main.c |
| show_menu_start_up | Displaying menu | main.c |
| sci_callback | Callback function for the SCI FIT module to check completion of an SCI transmission | main.c |
| send_string_sci | Transmitting strings | main.c |
| fw_up_open_flash | Flash FIT module initialization | r_fw_up_rx.c |
| fw_up_open | Firmware update initialization | r_fw_up_rx.c |
| fw_up_close | Completing firmware update r_fw_up | |
| copy_update_ramprog | Copying RAM program | r_fw_up_rx.c |
| switch_start_up_and_reset | Switching the start-up program protection area and software reset | r_fw_up_rx.c |
| fw_up_buf_init | Initializing buffer for firmware update | r_fw_up_buf.c |
| fw_up_memory_init | Initializing pointer to the buffer r_fw_up_t | |

Table 5.31 Functions Used in the Firmware

| Table 5.32 FIT Module Functions Used in the Firmwa |
|--|
|--|

| Function | FIT Module | Application | Function (Used in) |
|-----------------|------------------|--|-------------------------------|
| R_FLASH_Open | Flash FIT module | Initializing the Flash FIT module | fw_up_open_flash |
| R_FLASH_Control | Flash FIT module | Switching the start-up program protection area | switch_start_up_and _reset |
| R_SCI_Open | SCI FIT module | Starting up the SCI | main |
| R_SCI_Control | SCI FIT module | Enabling the transmit end interrupt | main |
| R_SCI_Send | SCI FIT module | Transmitting the SCI data | send_string_sci |
| R_SCI_Receive | SCI FIT module | Receiving the SCI data | main |



| Table 5.33 e ² studio Smart Configurator Generated Function Used in the Firmware (RX130 Group) | | | |
|---|----------------|-------------------------|--------------------|
| Function | FIT Module | Application | Function (Used in) |
| R_SCI_PinSet_SCI1 | SCI FIT module | Pin setting for the SCI | main |
| | | | |

Table 5.34 e² studio Smart Configurator Generated Function Used in the Firmware (RX140 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI1 | SCI FIT module | Pin setting for the SCI | main |

Table 5.35 e² studio Smart Configurator Generated Function Used in the Firmware (RX231 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI5 | SCI FIT module | Pin setting for the SCI | main |

Table 5.36 e² studio Smart Configurator Generated Function Used in the Firmware (RX261 Group)

| Function | FIT Module | Application | Function (Used in) |
|-------------------|----------------|-------------------------|--------------------|
| R_SCI_PinSet_SCI6 | SCI FIT module | Pin setting for the SCI | main |



6. Import a Project

The sample programs are distributed in e² studio project format. This section shows how to import a project into CS+. After importing the sample project, make sure to confirm build and debugger setting.

6.1 Importing a Project into CS+

To use sample programs in CS+, follow the steps below to import them into CS+. In projects managed by CS +, do not use space codes, multibyte characters, and symbols such as "\$", "#", "%" in folder names or paths to them.

(Note that depending on the version of CS+ you are using, the interface may appear somewhat different from the screenshots below.)



Figure 6.1 Importing a Project into CS+



7. Reference Documents

- RX110 Group User's Manual: Hardware (R01UH0421)
- RX111 Group User's Manual: Hardware (R01UH0365)
- RX113 Group User's Manual: Hardware (R01UH0448)
- RX130 Group User's Manual: Hardware (R01UH0560)
- RX140 Group User's Manual: Hardware (R01UH0905)
- RX230 Group, RX231 Group User's Manual: Hardware (R01UH0496)
- RX260 Group, RX261 Group User's Manual: Hardware (R01UH1045) (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.)
- C compiler manual
- RX Family C/C++ compiler Package (The latest version can be downloaded from the Renesas Electronics website.)



Revision History

| 1 | | Descript | Description | | |
|------|-----------|----------|---|--|--|
| Rev. | Date | Page | Summary | | |
| 1.00 | Sep.25.17 | | First edition issued | | |
| 1.10 | Sep.26.18 | | RX100 series added | | |
| 1.20 | Mar.24.22 | All | Added the following to the target devices. | | |
| | | | RX140 group | | |
| | | | Changed the specifications of the sample programs for the following reasons: | | |
| | | | To reduce the size of the firmware update program. | | |
| | | | Removed the CMT module | | |
| | | | • Separated the constants part from the program part. | | |
| | | | To improve convenience when updating a program.Changed firmware start-up procedure after programming is complete. | | |
| | | | Revised Table 1.2. | | |
| | | | Modified the setting items in Table 1.5. | | |
| | | | Modified URLs in "2.2 Compiler Package" and "2.3 Renesas Flash Programmer". | | |
| | | | Updated "3. Setting Up the Project" and "4. Operation Confirmation". | | |
| | | | Added "5.1 Configuration of the Firmware Update Program". | | |
| | | | Changed the figures used in "5.2 Operation Overview". | | |
| | | | Added "5.2.7 How to Create a Firmware Update Program for Updating". | | |
| | | | Changed the figures used in "5.3 Process Flowchart and Screen Output: Firmware Update Program" and "5.4 Process Flowchart and Screen Output: Firmware Program". | | |
| | | | Updated "5.5 Detailed Information of the Firmware Update Program" and "5.6 Detailed Information of the Firmware". | | |
| | | | Revised "6. Import a Project". | | |
| | | | Added "7. Reference Documents". | | |
| 1.30 | Sep.05.24 | All | Added the following to the target devices. | | |
| | | | RX261 group | | |



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 is supplied until the 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

- 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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
- 5. 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.
- 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.

- 7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
- 8. 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.
- 9. 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, 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.
- 10. 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.
- 11. 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.
- 12. 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.
- This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
 Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.
- (Note1) "Renease 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.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

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

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/.