

RX72M Group

Communications Board Modbus Startup Manual

Introduction

This application note is a quick start guide for Modbus communication with the RX72M communication board for industrial network evaluation.

Target Device

RX72M



Contents

| 1. | Overview | 4 |
|-----------|---|------|
| 1.1 | Feature | 4 |
| 1.2 | Operating Environment | 5 |
| 1.3 | Reference document | 6 |
| 2. | Hardware | 7 |
| 2.1 | Setting up the Board | 7 |
| 2.2 | Selecting the Power Source | 8 |
| 2.3 | Debug environment | 8 |
| 2 | Installing the e ² studie | 0 |
| Э. | | 9 |
| 3.1 | Installing the CC-RX Compiler V3.06.00 | 9 |
| 3.2 | Registering the Tool Chain | 9 |
| 4. | Sample application | .11 |
| 4.1 | Overview | . 11 |
| 4.1.1 | Modbus protocol stack (Sample program) | . 11 |
| 4.1.2 | 2 Modbus application (Sample program) | . 11 |
| 4.2 | Block diagram | . 11 |
| 5. | Test communication by sample application | .12 |
| 5.1 | Hardware connection | . 12 |
| 5.1.1 | Modbus TCP Server Stack Mode | . 12 |
| 5.1.2 | 2 Modbus RTU/ASCII stack mode | . 13 |
| 5.1.3 | Modbus TCP Serial gateway stack mode | . 14 |
| 6 | Project Setup | 15 |
| 0. 6 1 | Modbus TCP Server Setup Procedure | 15 |
| 6.2 | Modbus RTI //ASCII Setup Procedure | 20 |
| 6.3 | Modbus TCP Serial Cateway Setup Procedure | . 20 |
| 0.5 | Noubus for Senar Galeway Setup Frocedure | . 20 |
| 7. | Setup a master tool & Demonstration | .26 |
| 7.1 | Modbus TCP Server | . 26 |
| 7.2 | Modbus RTU/ASCII Slave | . 27 |
| 7.3 | Modbus RTU/ASCII Master | . 28 |
| 7.4 | Modbus TCP Serial Gateway | . 29 |
| 8. | Appendix | .30 |
| 8.1 | Appendix A. DHCP mode | . 30 |
| 8.2 | Appendix B. Setting the baud rate | . 30 |
| 8.3 | Appendix C. Setting Slave ID | . 30 |
| 8.4 | Appendix D. Multi-client configuration | . 31 |



| 9. | Limitations | 32 |
|-----|---------------|----|
| Rev | ision History | 33 |



1. Overview

This is a document of Modbus protocol stack that runs on RX72M, and gives an outline of functions, application programming interface (API), and application sample when developing and implementing an application that uses the protocol stack.

This package supports Ethernet-based Modbus TCP, RS-485 serial communication-based Modbus RTU, and Modbus ASCII protocols.

1.1 Feature

The Modbus protocol is a communication protocol developed by Modicon Inc. (Schneider Electric SA.) for programmable logic controllers (PLCs), the specifications of which are publicly available. Refer to the protocol specification (PI-MBUS-300 Rev. J).

The Modbus protocol stack for RX72M allows easy development of the following applications: The stack mode is specified by the initialization API at application execution time.

- Modbus RTU slave
- Modbus RTU master
- Modbus ASCII slave
- Modbus ASCII master
- Modbus TCP server
- Modbus TCP gateway

The Modbus protocol stack for RX72M supports the following nine function codes:

- 1 (0x01)-Read coils
- 2 (0x02)-Read discrete input
- 3 (0x03) Read holding registers
- 4 (0x04) Read input registers
- 5 (0x05)-Write single coil
- 6 (0x06)-Write single register
- 15 (0x0F)-Write multiple coils
- 16 (0x10) Write multiple registers
- 23 (0x17) Read / Write multiple registers

For more information about Modbus, please refer to the following site.

http://www.modbus.org

[Modicon Modbus Protocol Reference Guide Rev.J] (PI_MBUS_300.pdf)

[Modbus Application Protocol Specification V1.1b3] (Modbus_Application_Protocol_V1_1b3.pdf)

Note) The version number may be different due to the update. Please refer to the latest manual.



1.2 Operating Environment

The sample program covered in this manual run in the environment below.

Table 1.1 Operating Environment

| Item | Description |
|------------------------------------|---|
| Board | RX72M communications board |
| | TS-TCS07298 from Tessera Technology |
| CPU | RX CPU (RXv3) |
| Operating frequency | CPU clock (CPUCLK): 240 MHz |
| Operating voltage | 3.3 V |
| Operating modes | Single chip mode |
| | Boot mode (SCI interface) |
| | Boot mode (USB interface) |
| | Boot mode (FINE interface) |
| Device requirements | R5F572MNDDBD |
| | Code flash memory |
| | Capacity: 2/4 Mbytes |
| | ROM cache: 8 Kbytes |
| | Data flash memory |
| | Capacity: 32 Kbytes |
| | RAM/extended RAM |
| | Capacity: 512 Kbytes/512 Kbytes |
| Communications protocol | Modbus |
| Integrated development environment | e2Studio 2024-10 or later |
| Tool chain | C/C ++ compiler package V3.06.00 or later for RX family |
| Emulator (ICE) | Renesas E2 Lite |
| Evaluation tool | ModbusDemoApplication.exe: Modbus evaluation test program |



1.3 Reference document

Technical information on Modbus is available from the Modbus Organization site, and information on the RX72M communication board is available from the Renesas Electronics site.

- Modbus Organization's site : <u>http://www.modbus.org</u>
- Renesas Electronics website : <u>http://www.renesas.com</u>

Table 1.2 Modbus related documents

| Item | Description |
|------|---|
| 1 | Modbus_Application_Protocol_V1_1b3.pdf |
| 2 | PI_MBUS_300.pdf |
| 3 | Modbus_over_serial_line_V1_02.pdf |
| 4 | Modbus_Messaging_Implementation_Guide_V1_0b.pdf |

Table 1.3 RX72M Communication board related documents

| ltem | Description |
|------|---|
| 1 | RX72M Group User's Manual: Hardware (R01UH0804EJ) |
| 2 | RX72M Communication Board Schematic (rx72m-com) |

Table 1.4 Emulator related documents

Item Description

| 1 | E1 / E20 Emulator, E2 Emulator Lite User's Manual Supplement (RX User System Design) (R20UT0399EJ) |
|---|--|
| 2 | RX Family E1 / E20 Emulator User's Manual (R20UT0398EJ) |

2. Hardware

For detailed information on the board, refer to the RX72M Group Communications Board Hardware Manual.



Figure 2.1 Configuration of the RX72M Communications Board

2.1 Setting up the Board

Before supplying power to the board, set up jumpers and connect the cables. In addition, make settings for the JTAG configuration mode. This mode is normally used with a short circuit between jumper pins 2 and 3. However, if the hot plug-in function is to be used, change the combination to jumper pins 1 and 2. For the detailed locations of the related parts, refer to the *RX72M Communications Board Hardware Manual*.



Figure 2.2 Setting up Jumper Pins



2.2 Selecting the Power Source

Power to the RX72M can be supplied from a 5-V DC power source or through the USB port. Use whichever is suitable for the configuration of your operating environment.

2.3 Debug environment

Source code debugging is performed by connecting a CPU board to a PC via the E2 Emulator Lite. The connection between the CPU board, the E2 emulator Lite and the host PC is shown in Figure 2.3.



Figure 2.3 Connection between CPU board, E2 emulator Lite and host PC

The document about the CPU board is shown below.

- Reference document
 - RX72M Communication Board Schematic (rx72m-com)
- CPU board schematic
 - RX72M Group User's Manual: Hardware (R01UH0804EJ)
 Hardware specifications (pin layout, memory map, specifications of peripheral functions, electrical characteristics, timing) and operation description

The document about the Renesas development tool (E2 Emulator Lite) is shown below.

- Reference document
 - E1 / E20 Emulator, E2 Emulator Lite User's Manual Supplement (RX User System Design) (R20UT0399EJ)
 - RX Family E1 / E20 Emulator User's Manual (R20UT0398EJ)

3. Installing the e² studio

Download RX72M compatible e2studio (2024-10 or later) from the following website. https://www.renesas.com/e2studio_download

3.1 Installing the CC-RX Compiler V3.06.00

The compiler selection screen appears while installing e2studio. By selecting [Renesas CCRX v3.0 6.00] and selecting [Next], CC-RX V3.0 6.00 compiler compatible with RX72M will be installed together.

| 🛃 Renesas CC-RX Compilers | | * ^ |
|--|--|-----|
| □ Renesas CCRX v3.00.00 v3.00.00 Renesas C/C++ Compiler Package ダウンロード・サイズ: 22.5 MB Requires: • Renesas Tool License Manage ☑ Renesas CCRX v3.06.00 v3.01.00 Renesas C/C++ Compiler Package ダウンロード・サイズ: 21.4 MB Requires: • Renesas Tool License Manage | for RX Family v3.00.00 er - 2.2.1 for RX Family v3.01.00 er - 2.2.1 | |
| 🙀 GCC for Renesas RX (Registration | Required) | * |
| GCC for Renesas RX 4.8.4.201803 GCC for Renesas RX 4.8.4.201803 ダウンロード・サイズ: 72.8 MB | 4.8.4.201803 | |
| GCC for Renesas RX 4.8.4.201801 | 4.8.4.201801 | ~ |

Figure 3.1 e2studio-Compiler selection

To start e2studio, please run "e2studio.exe" located in the installed folder below.

e2_studio_rx72m\eclipse

3.2 Registering the Tool Chain

Register the CC-RX compiler v3.06.00 so that it can be used with the e² studio for RX72M.

- (1) Start the e^2 studio for RX72M.
- (2) Select [File] \rightarrow [New] \rightarrow [C/C++Project].



Figure 3.2 e2studio-Project selection



(3) In the [Templates for New C/C++ Project] dialog box, select [Renesas RX] \rightarrow [Renesas CC-RX C/C++ Executable Project] \rightarrow [Next].

| 💽 New (| C/C++ Project | 0 | \times |
|------------------------|-----------------|--|----------|
| Templat | tes for New C/C | ++ Project | |
| All CMake Make | Cebur | Makefile Project Create a Makefile project using CDT's new Core Build System. Over existing content or a new project with op | ^ |
| Renesas F Renesas F | RX | Renesas CC-RX C/C++ Executable Project A C/C++ Project for Renesas RX using the Renesas CC-RX toolchain. | |
| | RX | Renesas CC-RX C/C++ Library Project A C/C++ Library Project for Renesas RX using the Renesas CC-RX toolchain. | |
| | Renesas | Renesas Debug Only Project Renesas Debug Only Project | |

Figure 3.3 e2studio-Project selection

- (4) In the [New Renesas CC-RX C/C++ Executable Project] dialog box, enter a desired project name and select [Next].
- (5) In the [Select toolchain, device & debug settings] dialog box, select [Toolchain Management] under [Toolchain Settings].
- (6) In the [Renesas Toolchain Management] dialog box, select [Add] → [Browse...] to refer to the installation folder "C:¥Renesas¥RX¥3_0_6". The registration was successful if "v3.06.00 has been added under "Renesas CCRX".

| Scan for installed toolchains c | on startup |
|-------------------------------------|---|
| Toolchain Type ✔ ✔ Renesas CC-RX | Installation Path |
| ✓ v3.06.00 | C:¥Program Files (x86)¥Renesas¥RX¥3_6_0¥ |
| ✓ v3.03.00 | C:¥Program Files (x86)¥Renesas¥RX¥3_3_0¥ |
| ✓ v3.01.00 | C:¥Program Files (x86)¥Renesas Electronics¥CS+¥CC¥CC-RX¥V3.01.00¥ |
| ✓ v2.08.01 | C:¥Program Files (x86)¥Renesas¥RX¥2_8_1¥ |
| 🗸 🗹 GCC for Renesas RX | |
| 8.3.0.202405 | C:¥ProgramData¥GCC for Renesas RX 8.3.0.202405-GNURX-ELF¥rx-elf¥rx-elf¥ |
| 8.3.0.202311 | C:¥ProgramData¥GCC for Renesas RX 8.3.0.202311-GNURX-ELF¥rx-elf¥rx-elf¥ |
| ✓ 8.3.0.202305 | C:¥ProgramData¥GCC for Renesas RX 8.3.0.202305-GNURX-ELF¥rx-elf¥rx-elf¥ |
| KPIT GNURX-ELF Toolcha | in |





4. Sample application

4.1 Overview

This sample application program can be divided three blocks broadly.

- 1. Real time OS and TCP/IP stack.
- 2. Modbus protocol stack sample program, which uses FRTOS+TCP.
- 3. Modbus application sample program, which uses Modbus protocol stack sample program.

4.1.1 Modbus protocol stack (Sample program)

This sample application includes a sample program of Modbus protocol stack, which provides TCP or serial communication based on Modbus protocol. This sample program uses RTOS and TCP/IP stack (for Modbus TCP protocol).

4.1.2 Modbus application (Sample program)

This sample application includes Modbus application sample program, which use FRTOS+TCP and Modbus protocols stack sample program.

Please refer to this chapter and chapter 5. Test communication by sample application in detail.

4.2 Block diagram

A block diagram of this sample application is shown in Figure 4.1.



Figure 4.1

Feature block diagram of sample application

5. Test communication by sample application

5.1 Hardware connection

The Modbus protocol stack has different hardware connection methods depending on the stack mode.

5.1.1 Modbus TCP Server Stack Mode



Figure 5.1

Hardware connection example in Modbus TCP server stack mode

In this sample program, the IP address is automatically acquired from the DHCP server. When the sample program starts, the log message shown in Figure 5.2 is output to the terminal software.

| COM8 - Tera Term VT | |
|--|--|
| ïle Edit Setup Control Window Help | |
| * Modbus-TCP,Gateway Sample Program with uC3,uNet3 Start ** ip Address : 192.168.1.200 ip Mask : 255.255.255.0 ip Gateway : 0.0.0.0 net3_socket net3_bind : port (502) net3_listen isten_fd = 1 | |

Figure 5.2 Modbus TCP sample program startup log output message



5.1.2 Modbus RTU/ASCII stack mode

Use RS-485 communication to connect the RX72M communication board to the PC. The RS-485 transceiver module on the RX72M communication board is connected to the serial I / F channel 10 of the MCU, and the J6 pin is connected to RS-485.

Table 5.1 RS-485 I/F connection pin for Modbus RTU/ASCII (channel 6)

| MCU port Channel 10 | RS-485 transceiver connection pin (MCU side) | RS-485 transceiver connection pin (J6 side) | |
|---------------------|---|--|--|
| P86(RXD10) | R | A(1) | |
| PC7(TXD10) | D | B(2) | |
| P80(RTS10) | DE | GND(3) | |
| Board_3V3 | VCC | - | |
| GND | GND | - | |
| P86(RXD10) | R | A(1) | |

Figure 5.4 is an example of connection when setting up Modbus RTU / ASCII communication with RX72M communication board.



Figure 5.3 Hardware connection example in Modbus RTU/ASCII stack mode



5.1.3 Modbus TCP Serial gateway stack mode

Change the LED display on the Modbus RTU / ASCII slave device via the gateway device using the application that runs on Windows PC and the Modbus command.

Two evaluation boards are required to check the operation. It becomes evaluation board A (for gateway device) and evaluation board B (RTU / ASCII slave device)

On the evaluation board B (RTU / ASCII slave device) side, please download the RTU_SLAVE executable file of Modbus RTU / ASCII stack referring to the mounting procedure of Modbus RTU / ASCII stack.

| Table 5.1 | Gateway | / device | default settings |
|-----------|---------|----------|------------------|
| | | | - |

| Operation mode | MODBUS_RTU_MASTER_MODE |
|----------------|------------------------|
| Baud rate | 115200bps |
| Parity | None |
| Stop bit | 1 |



Figure 5.4 Hardware connection example in Modbus TCP serial gateway stack mode

[Limitations]

The gateway function uses the function of Modbus RTU / ASCII master mode.

Therefore, in Modbus TCP serial gateway mode, Modbus communication is only possible with function codes supported in Modbus RTU / ASCII master mode.



6. Project Setup

Describes the procedure to execute communication in the sample application. Complete the hardware connection according to the protocol stack mode to be operated by the sample application referring to 5.1 Hardware Connection in advance.

6.1 Modbus TCP Server Setup Procedure

This chapter describes the procedure for building Modbus TCP.

Import the sample project. After the program is started, by selecting [File] → [Import] → [Existing Projects into Workspace]. Check the "select root directory" and click "Browse..." button.
 →Select "rx72m_modbus_eva" → Check "modbus_rx72m" → [Finish].

| Import | | | | × |
|---|------------------------------------|---|------------|-----|
| Import Projects Select a directory to sear | rch for existing Eclipse projects. | | | 7 |
| Select root directory: | C:¥rx72m_modbus_eva | ~ | Browse | |
| ○ Select archive file: | | ~ | Browse | |
| Projects: | | | | |
| modbus_rx72m (C | :¥rx72m_modbus_eva) | | Select Al | I |
| | | | Deselect A | All |

2. Open "cofiguration.xml".

| Project Explorer X |
|--|
| ✓ p→ modbus_rx72m (in rx72m_modbus_eva) [TCP_SERVER_FRE] |
| > 🔊 Includes |
| > 😂 r_modbus_rx |
| > 😂 src |
| 🐲 modbus_rx72m.scfg |
| modbus_rx72m ASCII_MASTER_FREERTOS.launch |
| modbus_rx72m ASCII_SLAVE_FREERTOS.launch |
| modbus_rx72m RTU_MASTER_FREERTOS.launch |
| modbus_rx72m RTU_SLAVE_FREERTOS.launch |
| modbus_rx72m TCP_GATEWAY_FREERTOS.launch |
| modbus_rx72m TCP_SERVER_FREERTOS.launch |
| > ⑦ Developer Assistance |



 Select the "Components" tab. Confirm that all components are enabled.



4. Generate the code with the "Generate Code" button.

| modbus_rx72m.scf Software comp configuration | g × | Generate Code Ge | enerate Report |
|--|--|------------------|----------------|
| Components in a Components in a Components in a Components in a Config Co | Leven and the second se | Configure | ٤ |

| 0. | eeleet [Balla eelingulaterie] | | [.002 | | | | |
|----|-------------------------------|---|----------|---------------|-----------------|----|---|
| | New Go Into | > | guration | Generate Code | Generate Report | 21 | , |
| | OU IIILO | | | | | | |

5. Select [Build Configurations] \rightarrow [Set Active] \rightarrow [TCP_SERVER_FREERTOS]

| | | Go Into | > | gu | ation | Generate Code | Generate Report | |
|------|----|-------------------------------|---------------|-----|-------------|---------------|--------------------|---------|
| | | Open in New Window Show In | Alt+Shift+W > | - | Configure | e | ١ | |
| I | D | Сору | Ctrl+C | h | | | | |
| au | ß | Paste | Ctrl+V | H. | | | | |
| n | × | Delete | Delete | н. | | | | |
| ur | | Source | > | н. | | | | |
| cł | | Move | | н. | | | | |
| 31. | | Rename | F2 | | | | | |
| nc 👔 | 1 | Import | | | | | | |
| E | 1 | Export | | н. | | | | |
| | | Build Project | | | | | | |
| | | Clean Project | | н. | | | | |
| 4 | 2 | Refresh | F5 | н. | | | | |
| | | Close Project | | н. | | | | |
| | | Close Unrelated Project | | | | | | |
| | | Build Targets | > | | | | | |
| | | Index | > | | | | | |
| | | Build Configurations | > | | Set Active | > | 1 ASCII_MASTER_FF | REERTOS |
| | | Source | > | | Manage | | 2 ASCII_SLAVE_FREE | ERTOS |
| (| D | Run As | > | | Build All | | 3 RTU_MASTER_FRE | ERTOS |
| - | 14 | Debug As | > | | Clean All | | 4 KTU_SLAVE_FREE | |
| 41 | | Team | > | | Build Selec | cted | 6 TCP_GATEWAY_FR | ERIOS |
| | | Compare With | > | por | Pins Pins | interrupts | UTCP_SERVER_FREE | 11103 |

6. Execute the build.

| Project Explorer × modbus_rx72m (in rx72m_modble) | ъ us | eva) ITCP SERVEI | modbus_rx/2m.sctg × |
|--|---------|---|---------------------------------|
| > 🔊 Includes > 🥝 r_modbus_rx | | Go Into | |
| > 😂 src > 🏷 trash | | Open in New Win Show In | ndow Alt+Shift+W > |
| modbus_rx72m.scfg modbus_rx72m ASCII_MASTE modbus_rx72m ASCII_SLAVE_ modbus_rx72m RTU_MASTEF modbus_rx72m RTU_SLAVE_F modbus_rx72m TCP_GATEWA | | Copy Paste Delete Source Move Rename | Ctrl+C Ctrl+V Delete > |
| modbus_rx72m TCP_SERVER_ Developer Assistance | | Import Export | |
| l | | Build Project | |

7. After connecting the board and J-Link, start debugging in the following procedure.



8. Select the drop-down menu next to the bug icon and selecting "Debugger Configurations "



 $[Renesas DBG Hardware Debugging] \rightarrow [modbus_rx72m_TCP_SERVER_FREERTOS] item, then press [Debug].$

| 9 🔐 🕼 🗎 🗮 🖪 🏹 🗸 | | 2 |
|--|--|---|
| type filter text | Name: moddus_tx/2mTCP_SERVER_FREERIOS | |
| C/C++ Application | Project: | ^ |
| CAC++ Remote Application Script | modbus_rx72m Browse | |
| GDB Hardware Debugging | C/C++ Application: | |
| C GDB Simulator Debugging (RH850) | TCP_SERVER_FREERTOS/modbus_rx72m.x | |
| | Variables Search Project Browse Build (if required) before launching | |
| modbus_rx72m RTU_MASTER_FREERTOS | Build Configuration: Use Active | |
| modbus_rx72m RTU_SLAVE_FREERTOS modbus_rx72m TCP_GATEWAY_FREERTOS modbus_rx72m TCP_SERVER_FREERTOS | C Enable auto build O Disable auto build | ~ |
| 💽 Renesas Simulator Debugging (RX, RL78) | Baunt And | |

Following dialog will appear, so switch to the debug screen.

| 🖸 Con | firm Perspective Switch | × |
|-------|---|---|
| ? | This kind of launch is configured to open the Debug perspective when it suspends. This Debug perspective supports application debugging by providing views for displayin the debug stack, variables and breakpoints. Switch to this perspective? | g |
| Rem | nember my decision Switch No | |



9. Click the "Resume" button. The program will run.

| File Edit Source Re | factor Navigate Search | n Project Renesas | Views Run | Renesas | AI V |
|---|--|----------------------|--------------|-----------|------|
| 🗐 😸 🕶 🐔 📲 🔇 | / 🏷 i 🗙 🕪 💷 🖀 🖞 | 8. 👁 🖹 🖬 🗞 | 🎋 🕶 💁 | - 00 | ・核 |
| 🎋 Debug 🗡 | | | 🖻 💥 i> | 000 | |
| ✓ C [™] modbus_rx72m | TCP_SERVER_FREERTOS [| Renesas GDB Hardwa | are Debuggin | ig] | |
| modbus_rx72 modbus_rx72 modbus_rx72 | 2m.x [1] [cores: 0] 1 (single core) [core: 0] (| Suspended : Signal : | SIGTRAP:Trac | e/breakpo | oint |



6.2 Modbus RTU/ASCII Setup Procedure

This chapter describes the procedure for building Modbus RTU/ASCII. Since the program import has been completed in steps 1, 2, 3, and 4 of Chapter 6.1, there is no need to import it separately.

This chapter describes the explanation starting from switching the Serial mode.

1. Select [Build Configurations] → [Set Active] → [xxxx_yyyy_FREERTOS] Select "xxxx_yyyy" if necessary.

ASCII_MASTER_FREERTOS ASCII_SLAVE_FREERTOS RTU_MASTER_FREERTOS RTU_SLAVE_FREERTOS

| | Go Into Open in New Window | | e auto variables in | to calling this function - see of this function but such variables |
|----------|---|---|--|--|
| | Show In Copy Paste Delete Source Move Rename | Alt+Shift+W> Ctrl+C Ctrl+V Delete > | available after you ctions have not been uctors of C++ object C) STACK_ENABLE == 1 USED(ustack_area); | change the stack from the I stack cleared and the data sections ha s have not been executed until th |
| 22 | Import Export Build Project Clean Project Refresh Close Project Close Unrelated Project | F5 | <pre>X) defined(GN the Interrupt Table B(R_BSP_SECTOP_INTVE CEPTION_TABLE the Exception Table the Exception Table</pre> | UC) Register */ CTTBL); Register */ |
| | Build Targets Index | > | B(K_BSP_SECTOP_EXCEP | IVECTIBE); |
| | Build Configurations | > | Set Active > | 1 ASCII_MASTER_FREERTOS |
| | Source | > | Manage | 2 ASCII_SLAVE_FREERIOS |
| ○ | Run As Debug As Team | > | Build All Clean All Build Selected | 4 RTU_SLAVE_FREERTOS 5 TCP_GATEWAY_FREERTOS 6 TCP_SCATEWAY_FREERTOS |

2. Execute the build.

| New Comparing the sector of the | ✓ | dbus eva) ITCP SERVER | |
|---|--|---|---------------------------------|
| > Src Open in New Window > Stash Show In Alt+Shift+W> Image: modbus_rx72m.scfg Copy Ctrl+C Image: modbus_rx72m ASCII_SLAVE Paste Ctrl+V Image: modbus_rx72m RTU_MASTEE Delete Delete Image: modbus_rx72m RTU_SLAVE, F Move Nove Image: modbus_rx72m RTU_SLAVE, F Move F2 | > 🔊 Includes > 🤒 r_modbus_rx | New Go Into | > |
| Image: modbus_nx72m.scfg Image: Copy Ctrl + C Image: modbus_nx72m ASCII_SLAVE Paste Ctrl + V Image: modbus_nx72m ASCII_SLAVE Delete Delete Image: modbus_nx72m RTU_MASTEF Source > Image: modbus_nx72m RTU_SLAVE_F Move Rename F2 Image: modbus_nx72m TCP_SERVER Image: Move Image: Move Image: Move Image: modbus_nx72m TCP_SERVER Image: Move Image: Move Image: Move Image: Move Image: Move Image: Move Image: Move Image: Move Image: Move Image: Mo | > 🐸 src > 🥟 trash | Open in New Window Show In | Alt+Shift+W> |
| modbus_rx72m TCP_SERVER Mport O Developer Assistance Export | modbus_rx72m.scfg modbus_rx72m ASCII_MAST modbus_rx72m ASCII_SLAVE modbus_rx72m RTU_MASTE modbus_rx72m RTU_SLAVE modbus_rx72m RTU_SLAVE modbus_rx72m TCP_GATEW | Copy Paste Source Move Rename | Ctrl+C Ctrl+V Delete > |
| | modbus_rx72m TCP_SERVEF Developer Assistance | import Export | |



- 3. After connecting the board and J-Link, start debugging in the following procedure.
- 4. Select the drop-down menu next to the bug icon and selecting "Debugger Configurations "



 $[Renesas DBG Hardware Debugging] \rightarrow [modbus_rx72m_xxxx_yyyy_FREERTOS] item, then press [Debug].$

ASCII_MASTER_FREERTOS ASCII_SLAVE_FREERTOS RTU_MASTER_FREERTOS RTU_SLAVE_FREERTOS

| ype filter text | 📄 Main 🎋 Debugger 🕨 Startup 🧤 Source 🔲 Common | |
|--|---|---|
| C/C++ Application C/C++ Remote Application | Project: | ^ |
| EASE Script | modbus_rx72m Browse | |
| C GDB Hardware Debugging | C/C++ Application: | |
| GDB Simulator Debugging (RH850) GLaunch Group CRenesas GDB Hardware Debugging modbus_rx72m ASCII_MASTER_FREERTOS modbus_rx72m ASCII_SLAVE_FREERTOS | TCP_SERVER_FREERTOS/modbus_rx72m.x | |
| | Variables Search Project Browse Build (if required) before launching | |
| modbus_rx72m RTU_MASTER_FREERTOS | Build Configuration: Use Active | / |
| modbus_rx72m RTU_SLAVE_FREERTOS | O Enable auto build O Disable auto build | |
| modbus rx72m TCP SERVER EREERTOS | Use workspace settings Configure Workspace Settings | ~ |



Following dialog will appear, so switch to the debug screen.

| 🖸 Con | firm Perspective Switch X |
|-------|---|
| 2 | 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? |
| Rem | Switch No |

5. Click the "Resume" button. The program will run.



6.3 Modbus TCP Serial Gateway Setup Procedure

This chapter describes the procedure for building Modbus TCP Serial Gateway. Since the program import has been completed in steps 1, 2, 3, and 4 of Chapter 6.1, there is no need to import it separately.

1. Select [Build Configurations] → [Set Active] → [TCP_GATEWAY_FREERTOS] Select "xxxx_yyyy" if necessary.

| | New Go Into | > | <pre>{ /* Stack pointers are setup prior to calling th </pre> |
|----------|---|---------------------------------|--|
| | Open in New Window Show In | Alt+Shift+W > | /* You can use auto variables in this function * will be unavailable after you change the state |
| | Copy Paste Delete Source Move | Ctrl+C Ctrl+V Delete > | <pre>/* The bss sections have not been cleared and t * and constructors of C++ objects have not bee #if defined(GNUC) #if BSP_CFG_USER_STACK_ENABLE == 1 INTERNAL_NOT_USED(ustack_area); #endif</pre> |
| 220 | Rename | F2 | <pre>INTERNAL_NOT_USED(istack_area); #endif</pre> |
| 4 | Export | | <pre>#if defined(CCRX) defined(GNUC)</pre> |
| 8 | Build Project Clean Project Refresh Close Project Close Unrelated Project | FS | <pre>/* Initialize the Interrupt Table Register */ R_BSP_SET_INTB(R_BSP_SECTOP_INTVECTTBL); #ifdef BSP_MCU_EXCEPTION_TABLE /* Initialize the Exception Table Register */ R BSP_SET_EXTB(R_BSP_SECTOP_EXCEPTVECTTBL):</pre> |
| | Build Targets Index | > | #endif |
| | Build Configurations | > | Set Active > ✓ 1 ASCII_MASTER_FREERTOS |
| | Source | > | Manage 2 ASCII_SLAVE_FREERTOS |
| 0 | Run As Debug As Team | > | Build All 3 RTU_MASTER_FREERTOS Clean All 4 RTU_SLAVE_FREERTOS Build Selected 5 TCP_GATEWAY_FREERTOS |
| | - | | 0 ICP_SERVER_FREERIOS |

2. Execute the build.

| Project Explorer X | eva) ITCP SERVEI | |
|--|---|---|
| > []] Includes []] Includes []] | Go Into Open in New Window Show In Alt+Shift+W> | |
| modbus_rx72m.scfg modbus_rx72m ASCII_MASTE modbus_rx72m ASCII_SLAVE modbus_rx72m RTU_MASTEF modbus_rx72m RTU_SLAVE_F modbus_rx72m RTU_SLAVE_F modbus_rx72m TCP_GATEWA | Copy Ctrl+C Paste Ctrl+V Delete Delete Source > Move F2 | - |
| modbus_rx72m TCP_SERVER O Developer Assistance | Import Export | |
| | Build Project | ļ |



- 3. After connecting the board and J-Link, start debugging in the following procedure.
- 4. Select the drop-down menu next to the bug icon and selecting "Debugger Configurations "



 $[Renesas DBG Hardware Debugging] \rightarrow [modbus_rx72m_TCP_GATEWAY_FREERTOS] item, then press [Debug].$

| 19 🖻 🗫 🖺 🗶 🖻 🏹 🗸 | Name: modbus_rx72m TCP_GATEWAY_FREERTOS | | |
|---|---|----------|--|
| type filter text | Main 🕸 Debugger 🕨 Startup 🔲 Common 🧤 Source | e | |
| C/C++ Application | Project: | ^ | |
| I EASE Script | modbus_rx72m | Browse | |
| GDB Hardware Debugging GDB Simulator Debugging (RH850) Launch Group Renesas GDB Hardware Debugging modbus_rx72m ASCII_MASTER_FREERTOS modbus_rx72m ASCII_SLAVE_FREERTOS modbus_rx72m RTU_MASTER_FREERTOS modbus_rx72m RTU_SLAVE_FREERTOS | C/C++ Application: | | |
| | TCP_GATEWAY_FREERTOS/modbus_rx72m.x | | |
| | Variables Search Project | Browse | |
| | Puild (if required) before launching | Diotisen | |
| | Build (in required) before faunching | | |
| | Build Configuration: Use Active | ~ | |
| | C Enable auto build Disable auto build | t | |
| modbus_rx72m TCP_GATEWAY_FREERTOS | Use workspace settings <u>Configure Workspace</u> | Settings | |
| Renesas Simulator Debugging (RX, RL78) | | • | |
| | Revert | Apply | |
| Filter matched 14 of 16 items | Revert | Apply | |



Following dialog will appear, so switch to the debug screen.

| 💽 Con | firm 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? |
| Rem | Switch No |

5. Click the "Resume" button. The program will run.



7. Setup a master tool & Demonstration

7.1 Modbus TCP Server

- 1. Connect "TCP Server" as described in section 5.1.1.
- 2. Open "ModbusDemoApplication.exe" which is included in this package.
- 3. Set the "Remote Modbus Server" IP Address (e.g., "192.168.1.100") and Port (e.g., "502").
- 4. Press "Connect", the coils will be updated periodically and LEDs 1-4 of the connected TCP server will be updated and flashing.

| 🖳 Modbus Demo | – 🗆 X |
|--------------------------|-------------------------------|
| File(F) Help | |
| Connection TCP server | Connect |
| Serial setting | L/O Coils 04 |
| 115200bps | Discrete Inputs 00 |
| NONE Parity | Slave ID updated periodically |
| Remote Modbus Server | Connection Timeout 5000 |
| 192.168.1.100 502 | |
| WRITE MULTIPLE COIL | |

Figure 7.1: ModbusDemoApplication tcp server setting

7.2 Modbus RTU/ASCII Slave

- 1. Connect "RTU/ASCII slave" as described in section 5.1.2.
- 2. Open "ModbusDemoApplication.exe" which is included in this package.

Set the Serial settings
 Connection : Serial Slave
 COMxx : Adapt to each environment
 Baud rate : Match the sample program settings
 Mode : RTU or ASCII
 Parity : No
 Stop bit : 1
 Slave ID : 1

4. Press "Connect", the coils will be updated periodically and LEDs 1-4 of the connected RTU/ASCII slaves will be updated and flashing.

| 🖳 Modbus Demo | – 🗆 X |
|--|--|
| File(F) Help | |
| Connection Serial Slave | Connect |
| Serial setting COM9 115200bps RTU NONE Parity 1 stop bit V | I/O 04 Discrete Inputs 00 Slave ID 1 Updated periodically Connection Timeout |
| Remote Modbus Server Port IP Adress Port 192.168.1.100 502 Info: Serial Slave thread ends. | |

Figure 7.2: ModbusDemoApplication RTU/ASCII slave setting

7.3 Modbus RTU/ASCII Master

- 1. Connect "RTU/ASCII slave" as described in section 5.1.2.
- 2. Open "ModbusDemoApplication.exe" which is included in this package.
- 3. Set the Serial settings

Connection : Serial Master COMxx : Adapt to each environment Baud rate : Match the sample program settings Mode : RTU or ASCII Parity : No Stop bit : 1 Slave ID : 1

4. Press "Connect" and the coil will wait for input. As you enter values, LEDs 1 to 4 on the connected Master will update and light up.

Ex), Input 00 : LED off Input 01 : LED1 on Input FF : LED1-4 on

| 🛃 Modbus Demo File(F) Help | – 🗆 X |
|---|---|
| Connection Serial Master ~ | Connect |
| Serial setting COM9 ~ 115200bps ~ RTU ~ NONE Parity ~ | I/O Coils FF Discrete Inputs 01 Slave ID 1 Input data |
| 1 stop bit ~ Remote Modbus Server | Connection Timeout 5000 |
| Info: Serial Master thread ends. | |

Figure 7.3: ModbusDemoApplication RTU/ASCII master setting

7.4 Modbus TCP Serial Gateway

- 1. Connect another "RTU slave" as described in section 5.1.3.
- 2. Open "ModbusDemoApplication.exe" which is included in this package.
- 3. Set the "Remote Modbus Server" IP Address (e.g., "192.168.1.100") and Port (e.g., "502").
- 4. Press "Connect", the coils will be updated periodically and LEDs 1-4 of the connected RTU slaves will be updated and flashing.

| Connection TCP Gateway Serial setting III5200bps RTU NONE Parity I stop bit Padress Port 192.168.1.100 502 Connect Connect Connect U/O Coils 20 Discrete Inputs 00 1 Connect Discrete Inputs 00 Slave ID Updated periodically | Modbus Demo | - 🗆 X |
|---|---|--|
| Serial setting 115200bps RTU NONE Parity I stop bit Slave ID 1 updated periodically Connection Timeout 5000 | Connection TCP Gateway | Connect |
| 1 stop bit Connection Timeout Remote Modbus Server 5000 IP Adress Port 192.168.1.100 502 | Serial setting | I/O Coils 20 Discrete Inputs 00 Slave ID 1 updated periodically |
| | 1 stop bit Remote Modbus Server IP Adress 192.168.1.100 | Connection Timeout 5000 |

Figure 7.4: ModbusDemoApplication TCP gateway setting



8. Appendix

8.1 Appendix A. DHCP mode

1. When operating with DHCP, enable the setting for "ipconfigUSE_DHCP" to "1". "rx72m_modbus_eva\src\frtos_config\FreeRTOSIPConfig.h"

2. Build and debug.



8.2 Appendix B. Setting the baud rate

1. In the release environment, the baud rate is set to 115200bps. To change the baud rate, change the following variable.

"rx72m_modbus_eva\src\modbus_init.c"

The values that can be set are 115200, 76800, 38400, 31250, 19200 or 9600.

2. Build and debug.



8.3 Appendix C. Setting Slave ID

- 1. When changing the Slave ID, please change the following settings. (renge : 1 to 247) "**rx72m_modbus_eva\src\modbus_init.c**"
- 2. Build and debug.

| <pre>#ifdef MODBUS_ASCII</pre> | /* ASCII Slave mode */↔ | |
|--------------------------------|--------------------------|------------------|
| #0100 | MODDUS_ASUII_SLAVE_MODE, | |
| #6126 | | |
| #endife | MODDUS_KIU_SEAVE_MODE, | |
| #end i e | 1). | /* Slave ID */+ |
| #endif⇔ | <u> </u> | /# 01000 ID #/ (|
| #endifk⊣ | | |
| #endif↩ #endif↩ #endifŀ | 1); | /* Slave ID */← |

8.4 Appendix D. Multi-client configuration

This project supports multiple clients.

The initial state is enabled, and clients that can receive will be able to register their IP addresses. Please add your IP address in the section below.

In the initial state, the only valid IP address is "192.168.1.101"

Example of adding "192.168.1.102" and "192.168.1.103".

"rx72m_modbus_eva\src\modbus_init.c"

```
uint32_t modbus_init(void);
```

```
@brief Initialize MODBUS protocol stack
@param none
@retval error code
*******
              */
uint32_t modbus_init(void)
{
  uint32_t ercd;
#if defined( MODBUS_RTU ) || defined( MODBUS_ASCII ) || defined( MODBUS_GATEWAY )
  serial stack init info t st init info;
  serial_gpio_cfg_t
                     st_gpio_cfg;
#endif
  slave_map_init_t st_slave_map;
#if defined( MODBUS_TCP ) || defined( MODBUS_GATEWAY )
  /* Enable IP table */
  Modbus_tcp_init_ip_table (ENABLE, ACCEPT);
  /* register IP address */
 ercd = Modbus_tcp_add_ip_addr ("192.168.1.101");
  ercd = Modbus_tcp_add_ip_addr ("192.168.1.102");
 ercd = Modbus_tcp_add_ip_addr ("192.168.1.103");
  if (ercd != ERR_OK)
  {
   return ercd;
  }
```

Note), The number of clients is limited to 8.



9. Limitations

None



Revision History

| | | Description | |
|------|----------------|-------------|--|
| Rev. | Date | Page | Summary |
| 1.00 | July. 31, 2019 | - | First edition issued |
| 1.01 | Nov. 01, 2019 | - | -Replaced due to lack of content in the configuration diagram |
| 1.02 | Feb. 07, 2020 | - | -Update Ether driver and startup file (prst.src) |
| | | | -Corrected the supply clock to PHY from 50MHz to 25Mhz |
| | | | -Corrected erroneous description of evaluation tool operation method |
| 1.03 | Nov 20, 2020 | - | -Update all documents by updating sample application. |
| | | | -Add explanations for Modbus application programs |
| | | | -Change the switch configuration and its explanation. |
| | | | -Remove modbus_init function from API |
| | | | -Remove modbus_init function from API |
| 2.00 | Dec. 25, 2024 | - | -Changed from uNET3 to FREERTOS+TCP |
| | | | -2 ports enabled |
| | | | |



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 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.

•Arm® and Cortex® are registered trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. •Ethernet is a registered trademark of Fuji Xerox Co. Ltd.

Ethernet is a registered trademark of Fuji Xerox Co. Ltd.

Modbus is a registered trademark of Schneider Electric, licensed to the Modbus Organization, Inc.
 IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers Inc

•Additionally all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners.

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.
- 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.
- 6. 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
 - "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) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
- (Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.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/.