

RZ/N1D, RZ/N1S, RZ/N1L Group

R01AN4412EJ0120

Rev.1.20

Quick Start Guide: Modbus Slave Software

Dec 28, 2018

Introduction

This document explains the procedure of Modbus protocol communication using Modbus stack for Renesas RZ/N1 platform.

Target Device

RZ/N1D, RZ/N1S, RZ/N1L

Contents

1. Overview	3
1.1 Purpose	3
1.2 Scope.....	3
1.3 Abbreviations/Definitions.....	4
1.4 References	4
2. Features	5
3. Project Setup	6
3.1 Requirements	6
3.2 Hardware	7
3.2.1 Serial port connection	7
3.2.2 Ethernet connection	9
3.3 Sample Application.....	10
3.4 Running the sample application.....	12
3.4.1 Standalone Variant – RZ/N1D and RZ/N1S	12
3.4.2 Standalone Variant – RZ/N1L.....	13
3.4.3 Core To Core variant – RZ/N1D (Communication Core)	15
3.4.4 Core To Core variant – RZ/N1D (Application Core).....	15
4. CODESYS setup	16
4.1 Installation and start-up	16
4.2 Create project	16
4.3 Device configuration.....	18
4.3.1 Configuration of Modbus Serial slave device	20
4.3.2 Configuration of Modbus Serial master device	20
4.3.3 Configuration of Modbus TCP master device.....	21
4.4 Connection to Software PLC.....	22
4.5 Device settings.....	26
4.5.1 Modbus Serial Slave device settings.....	27

4.5.2	Modbus Serial Master device settings	29
4.5.3	Modbus TCP Master device settings	31
5.	Test communication with the sample application.....	33
5.1	Execution procedure	33
5.1.1	Modbus Serial Master execution procedure	33
5.1.2	Modbus Serial Master Slave execution procedure.....	33
5.1.3	Modbus TCP Server execution procedure	34
5.1.4	Modbus TCP-Serial Gateway execution procedure.....	34
5.2	Method of operate the sample application	35
5.2.1	Starting the sample application.....	35
5.2.2	Terminal software setting	35
5.2.3	Method of select the serial communication setting	36
5.2.4	Method of select the execution function	37
5.2.5	Method of select Modbus gateway settings.....	38
5.3	CODESYS operation method	39
5.3.1	Method of set execution functions and I/O mapping	39
5.3.2	Operation start / end procedure of software PLC application	41
5.3.3	Method of setting the read function read value	42
5.3.4	Method of setting the write function read value	43
5.3.5	Execution result confirmation method	44
5.3.6	Method of check the write function write result	45
5.3.7	Method of check the read function result	46
6.	Defined value of sample application	47
7.	Limitations	49
8.	Website and Support.....	50

1. Overview

This document describes how to run Modbus on the RZ/N1 Series. It is possible either to run a standalone variant only using one core, the Cortex[®]-M3-core, or to use two separate cores in the case of the RZ/N1D communicating via Core To Core. Both cores feature the GOAL (Generic OS Abstraction Layer) which handles the communication of the cores and provides basic functionality e.g. timer handling.

The Modbus protocol runs on Cortex-M3-core in both the standalone and the Core To Core variant. Its task is the communication with other operators, therefore the alias name of the Cortex-M3-core is communication core (CC) in this document.

In the Core To Core variant the user application is executed on the Linux based Cortex-A7-core. This core is also named as application core (AC). In the standalone variant the user application is running on the communication core, either.

Modbus is an application layer messaging protocol developed by Modicon for PLC and is widely used in industrial automation and data collection systems. Modbus does not prescribe a physical layer such as a communication medium and communicates between devices of different types of buses or networks connected to the network.

There are the following two versions of Modbus protocol.

- Version for serial communication such as RS-232C and RS-485. (Modbus Serial)
- Version for networks supporting Ethernet and other Internet protocol suites. (Modbus TCP)

In addition, Modbus Serial has the following two transmission modes depending on the difference in the expression method of the value in the message frame.

- RTU mode (Modbus RTU)
- ASCII mode (Modbus ASCII)

Modbus stack for the Renesas RZ/N1 platform supports Modbus RTU, Modbus ASCII and Modbus TCP.

1.1 Purpose

This document explains the test procedure of Modbus protocol communication using Modbus stack for Renesas RZ / N1 platform. Evaluate using the software PLC CODESYS for the communication destination device.

1.2 Scope

This manual explains only the sample application for checking Modbus protocol communication and the environment setting procedure and operation method necessary for testing against CODESYS.

1.3 Abbreviations/Definitions

Table 1. Abbreviations/Definitions

Index	Abbreviations /Definitions	Description
1	PLC	Programmable Logic Controller
2	I/O	Input / Output
3	IP	Internet Protocol
4	TCP	Transmission Control Protocol
5	RTU	Remote Terminal Unit
6	ASCII	American Standard Code for Information Interchange
7	UART	Universal Asynchronous Receiver-Transmitter
8	USB	Universal Serial Bus
9	PWR	PoWeR
10	GND	GrounND
11	RTS	Ready To Send
12	LAN	Local Area Network
13	PC	Personal Computer

1.4 References

Technical information on Modbus is available from the Modbus Organization website, and information on RZ/N1 is available from the Renesas Electronics website.

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

Table 1.2 Technical Inputs

Index	Technical Inputs
1	Modbus_Application_Protocol_Vxxx.pdf
2	PI_MBUS_xxx.pdf
3	Modbus Serial Line Protocol and Implementation Guide Vxxx
4	Modbus_Messaging_Implementation_Guide_Vxxx.pdf
5	r01qs0020edxxxx-rzn1l-quick-start-guide.pdf
6	r01qs0021edxxxx-rzn1s-quick-start-guide.pdf
7	r01qs0022edxxxx-rzn1d-quick-start-guide.pdf
8	RZ_N1_EB_Board_Setup_Notes.Vxxx.pdf
9	RZN1-CODESYS-Quick-StartupGuide.pdf

2. Features

The Modbus stack supports Modbus RTU, Modbus ASCII and Modbus TCP and operates in the six operation modes shown in the table below.

Table 2.1 Modbus stack operation mode

Operation	Overview
Modbus RTU Master	Works as Master of Modbus RTU.
Modbus RTU Slave	Works as Slave of Modbus RTU.
Modbus ASCII Master	Works as Master of Modbus ASCII.
Modbus ASCII Slave	Works as Slave of Modbus ASCII.
Modbus TCP Server	Works as Server of Modbus TCP.
Modbus TCP-Serial Gateway	In addition to the server function of Modbus TCP, routes requests from Modbus TCP client device to Modbus Serial network.

Modbus stack supports nine public function codes in the table below.

Table 2.2 Modbus stack support function code

Function code	Detail
01	Read Coils
02	Read Discrete Inputs
03	Read Holding Registers
04	Read Input Register
05	Write Single Coil
06	Write Single Register
15	Write Multiple Coils
16	Write Multiple Registers
23	Read/Write Multiple Registers

Each function code is defined by communication protocol. For details and other Modbus protocol specifications, refer to the protocol specification.

3. Project Setup

For the general setting procedure of RZ/N1, follow the startup guide below.

- RZ/N1D-DB : r01qs0020edxxxx-rzn1l-quick-start-guide.pdf
- RZ/N1S-DB : r01qs0021edxxxx-rzn1s-quick-start-guide.pdf
- RZ/N1L-DB : r01qs0022edxxxx-rzn1d-quick-start-guide.pdf

3.1 Requirements

Table 3.1 Requirements

Item	Description
Board	Renesas Electronics • RZ/N1D-DB Development Board or • RZ/N1S-DB Development Board or • RZ/N1L-DB Development Board If expansion board is used for RZ/N1D-DB, RZ/N1S-DB, requires • RZ/N1-EB Expansion Board
IDE	IAR Systems Embedded Workbench® for ARM Version 8.22.1 or later
Emulator	IAR Systems I-jet
Software PLC	3S-Smart Software Solutions CODESYS Version 3.5 or later

3.2 Hardware

Take care of following the setup guidelines for the RZ/N1 Demo Board from the RZ/N1x quick start documentation from the home Solution Kit folder.

r01qs0020edxxxx-rzn1l-quick-start-guide.pdf for RZ/N1L

r01qs0021edxxxx-rzn1s-quick-start-guide.pdf for RZ/N1S

r01qs0022edxxxx-rzn1d-quick-start-guide.pdf for RZ/N1D

In case of testing Modbus Serial or Modbus TCP-Serial gateway, be sure to set the serial port properly.

The settings required for Modbus protocol communication are described below.

3.2.1 Serial port connection

In case of testing Modbus Serial, connect RZ/N1 and software PLC with RS-485 serial.

Input the UART output of RZ/N1 to the RS485 driver, connect the output from RS485 and the PC on which CODESYS runs, using conversion cable as needed. General evaluation connection of RZ/N1L and RZ/N1-EB is shown below.

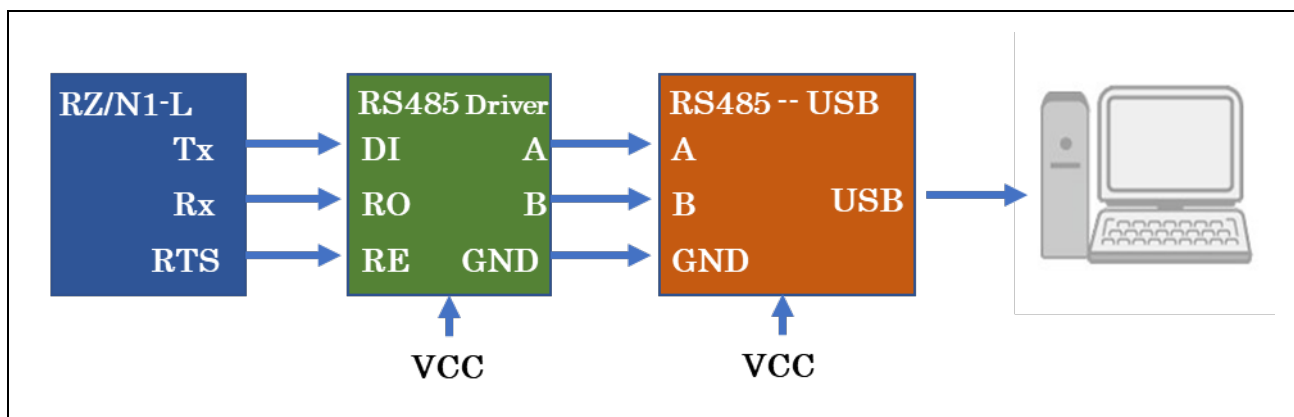


Figure 3.1: General evaluation connection of serial port of RZ/N1L UART interface Connection

Table 3.2 Switch setting of UART input / output of RZ/N1L

SW	Description	Setting
SW5-1	Switching of multiplexer circuit (RMII / MII, PMOD selection)	ON (PMOD)

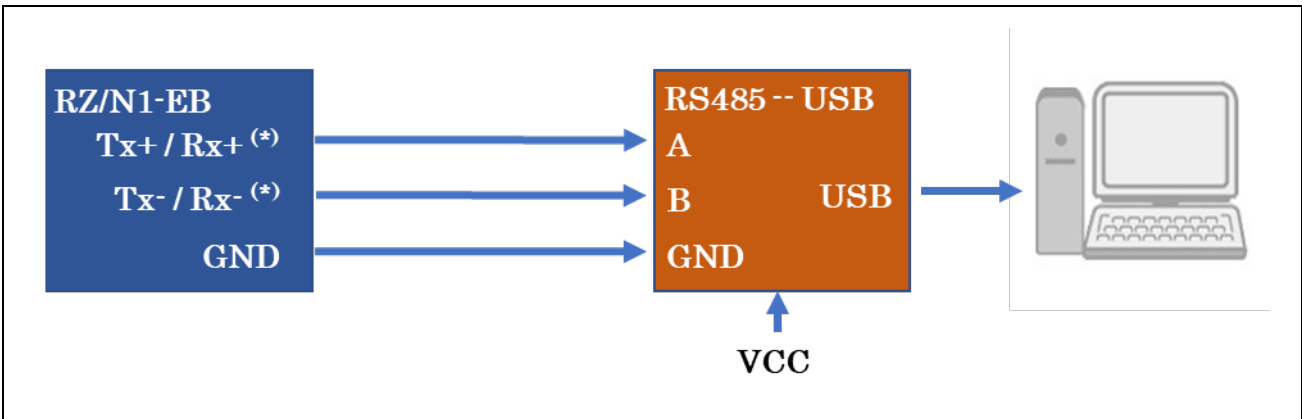


Figure 3.2: General evaluation connection of serial port of RZ/N1-EB RS485 interface Connection

(*)

1. RS485 interface (J8 connector) of RZ/N1-EB outputs Tx+, Tx-, Rx+, Rx-,
For half-duplex communication, short Tx+ and Rx+, Tx- and Rx-.

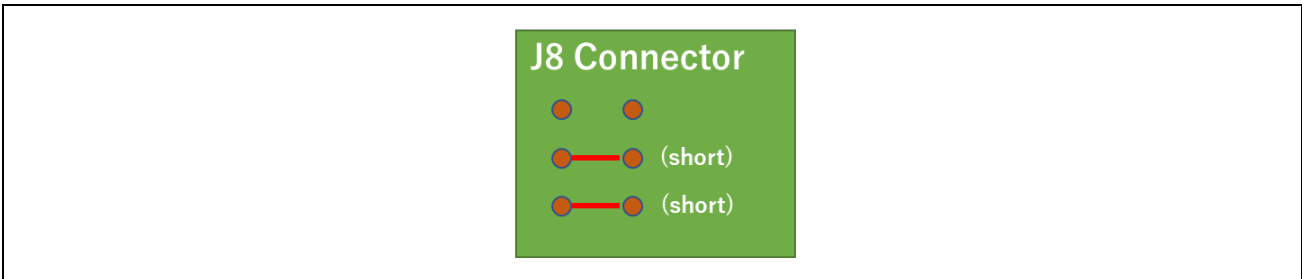


Figure 3.3: EB board modification (back view)

2. Set the S3-1 switch to ON.

Table 3.3 Half duplex communication setting of EB board

SW	Description	Setting
SW3-1	-	OFF
SW3-2	-	OFF
SW3-3	-	OFF
SW3-4	Half duplex communication	ON

Sample application requires terminal input / output by serial communication. Connect the following connector of RZ/N1 and the PC on which terminal software runs with the USB cable.

- RZ/N1D-DB : CN10
- RZ/N1S-DB : CN6
- RZ/N1L-DB : CN6

3.2.2 Ethernet connection

In case of testing Modbus TCP or Modbus TCP-Serial gateway, connect RZ/N 1 with Software PLC by Ethernet.

Connect the following connector of RZ/N1 and the PC running CODESYS with a LAN cable.

- RZ/N1D-DB : CN1 (ETH 4) or CN 4 (ETH 5)
- RZ/N1S-DB : CN1 (ETH 4) or CN 5 (ETH 5)
- RZ/N1L-DB : CN1 (ETH 4) or CN 5 (ETH 5)
- RZ/N1-EB : J23 (PHY2) or J24 (PHY3)

In case of using the port of RZ/N1-EB, the following jumper setting is required.

Table 3.4 Jumper setting of RZ/N1-EB Ethernet connection

Jumper	Description	Setting
CN15	MDC setting of PHY2 / PHY3 (selection of MDC1 / MDC2)	Connect 2-3 pins (MDC2 selected)
CN16	MDIO setting of PHY2 / PHY3 (selection of MDIO1 / MDIO2)	Connect 2-3 pins (MDIO2 selected)

In case of testing the Modbus TCP-Serial gateway, connect the RZ/N1 and the Modbus Serial slave device with the RS-485 serial in addition to the above settings.

Refer to "3.2.1 Serial port connection" for the connection method.

Sample application of Modbus TCP-Serial gateway requires terminal input / output by serial communication, connect the following connector of RZ/N1 and the PC on which terminal software runs with a USB cable.

- RZ/N1D-DB : CN10
- RZ/N1S-DB : CN6

3.3 Sample Application

In order to confirm Modbus protocol communication, several of sample applications are supported. And each stack is a different mode of operation. For each application, can check the communication of function codes 1 to 6, 15, 16, and 23 supported by the Modbus stack. The following examples can be found in the folder goal/appl/goal_mbs/.

- 00_rpc_cc_master – communication core (Core To Core variant only)
- 00_rpc_cc_slave – communication core (Core To Core variant only)
- 01_tcp_server – Modbus TCP Server application (Standalone variant, Core To Core variant for AC)
- 02_serial_master – Modbus Serial Master application (Standalone variant, Core To Core variant for AC)
- 03_serial_slave – Modbus Serial Slave application (Standalone variant, Core To Core variant for AC)
- 04_tcp_ser_gateway – Modbus TCP-Serial Gateway application (Standalone variant, Core To Core variant for AC)

Table 3.5 Modbus sample application operation mode

Sample application	Operation mode
Modbus Serial Master	Modbus RTU master / Modbus ASCII master (select transmission mode)
Modbus Serial Slave	Modbus RTU slave / Modbus ASCII slave (select transmission mode)
Modbus TCP Server	Modbus TCP Server.
Modbus TCP-Serial Gateway	Modbus TCP-Serial Gateway

Follow the instructions in the "Software Setup" chapter of the startup guide.

- Write U-Boot to QSPI flash (It is not necessary when using RZ/N1L-DB.)
- Building the source code of the Modbus sample application

To build the source code, use the corresponding workspace file in the table below. The build setting is "Debug-RAM".

Table 3.6 Modbus sample application workspace file for Standalone Variant

Sample application		Board	Workspace file
Sample	Source code		
Modbus Serial Master	goal\appl\goal_mbs\02_serial_master	RZ/N1L-DB	goal\projects\goal_mbs\02_serial_master\iar\renesas\rzn1l_demo_board\rzn1l_demo_board.eww
		RZ/N1D-DB	goal\projects\goal_mbs\02_serial_master\iar\renesas\rzn1d_demo_board\eb\rzn1d_demo_board_eb.eww
		RZ/N1S-DB	goal\projects\goal_mbs\02_serial_master\iar\renesas\rzn1s_demo_board\eb\rzn1s_demo_board_eb.eww
		+ RZ/N1-EB	
Modbus Serial Slave	goal\appl\goal_mbs\03_serial_slave	RZ/N1L-DB	goal\projects\goal_mbs\03_serial_slave\iar\renesas\rzn1l_demo_board\rzn1l_demo_board.eww
		RZ/N1D-DB	goal\projects\goal_mbs\03_serial_slave\iar\renesas\rzn1d_demo_board\eb\rzn1d_demo_board_eb.eww
		RZ/N1S-DB	goal\projects\goal_mbs\03_serial_slave\iar\renesas\rzn1s_demo_board\eb\rzn1s_demo_board_eb.eww
		+ RZ/N1-EB	
Modbus TCP Server	goal\appl\goal_mbs\01_tcp_server	RZ/N1D-DB	goal\projects\goal_mbs\01_tcp_server\iar\renesas\rzn1d_demo_board\rzn1d_demo_board.eww
		RZ/N1S-DB	goal\projects\goal_mbs\01_tcp_server\iar\renesas\rzn1s_demo_board\rzn1s_demo_board.eww
		RZ/N1L-DB	goal\projects\goal_mbs\01_tcp_server\iar\renesas\rzn1l_demo_board\rzn1l_demo_board.eww

Modbus TCP- Serial Gateway	goal\appl\goal_mbs\ 04_tcp_ser_gateway	RZ/N1D-DB	goal\projects\goal_mbs\01_tcp_server\iar\renesas\ rzn1d_demo_board_eb\rzn1d_demo_board_eb.eww
		+ RZ/N1-EB	
		RZ/N1S-DB	goal\projects\goal_mbs\01_tcp_server\iar\renesas\ rzn1s_demo_board_eb\rzn1s_demo_board_eb.eww
		+ RZ/N1-EB	
		RZ/N1L-DB	goal\projects\goal_mbs\04_tcp_ser_gateway\iar\renesas\ rzn1l_demo_board\rzn1l_demo_board.eww
		+ RZ/N1-EB	
		RZ/N1D-DB	goal\projects\goal_mbs\04_tcp_ser_gateway\iar\renesas\ rzn1d_demo_board_eb\rzn1d_demo_board_eb.eww
		+ RZ/N1-EB	
		RZ/N1S-DB	goal\projects\goal_mbs\04_tcp_ser_gateway\iar\renesas\ rzn1s_demo_board_eb\rzn1s_demo_board_eb.eww
		+ RZ/N1-EB	

For Core To Core variant, use the following workspace file.

Table 3.7 Modbus sample application workspace file for Core To Core Variant

Sample application		Board	Workspace file
Sample	Source code		
Modbus Serial Master	goal\appl\goal_mbs\ 00_rpc_cc_master	RZ/N1D-DB + RZ/N1-EB	goal\projects\goal_mbs_rpc_cc\00_goal_rpc_demo_m aster\iar\renesas\rzn1d_demo_board_eb\rzn1d_demo_ board_eb.eww
Modbus Serial Slave	goal\appl\goal_mbs\ 00_rpc_cc_slave	RZ/N1D-DB + RZ/N1-EB	goal\projects\goal_mbs_rpc_cc\00_goal_rpc_demo_sl ave\iar\renesas\rzn1d_demo_board_eb\rzn1d_demo_b oard_eb.eww
Modbus TCP Server	goal\appl\goal_mbs\ 00_rpc_cc_slave	RZ/N1D-DB + RZ/N1-EB	goal\projects\goal_mbs_rpc_cc\00_goal_rpc_demo_sl ave\iar\renesas\rzn1d_demo_board\rzn1d_demo_boar d.eww
Modbus TCP- Serial Gateway	goal\appl\goal_mbs\ 00_rpc_cc_slave	RZ/N1S-DB + RZ/N1-EB	goal\projects\goal_mbs_rpc_cc\00_goal_rpc_demo_sl ave\iar\renesas\rzn1d_demo_board_eb\rzn1d_demo_b oard_eb.eww

3.4 Running the sample application

The RZ/N1D and RZ/N1S use the U-Boot bootloader for initial setup of the hardware and loading of the Cortex-M3 firmware. Additionally the RZ/N1D U-Boot bootloader is used for booting the Linux Kernel. The RZ/N1L is working without any bootloader. This chapter describes how to install the management software on the flash of the board. If no bootloader was yet installed on the RZ/N1D and RZ/N1S please refer to the Linux documentation - Quick Start Guide for U-Boot and Linux - *RZ/N1x-Quick-Start-Guide.pdf*.

There are many similarities between the derivatives of the RZ/N1 series but some minor difference, too. Therefore here is a more detailed explanation how to run a sample application on each.

All standalone projects and the CC project of the Core To Core variant contain different workspaces for each board variant. The project workspaces ending on *_eb contain the configuration for the CPU Board together with the extension board (4 switch ports). The other project workspaces contain the configuration for working with the CPU Board only.

3.4.1 Standalone Variant – RZ/N1D and RZ/N1S

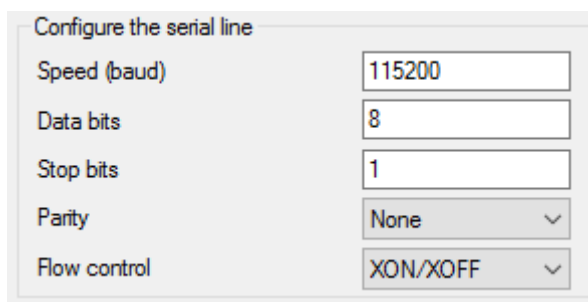
It is possible to load the code via debugger into RAM, which is a very fast approach to test the user application, or to flash the Cortex-M3 core. In both cases and must be built using IAR Embedded Workbench.

For the target workspace file, refer to “3.3 Sample Application”

3.4.1.1 Loading application into RAM via IAR Embedded Workbench

To compile a project, follow these steps:

1. Start the IAR Embedded Workbench
2. Open a project via Open/Workspace
3. Go to the workspace folder and open it:
4. In case the CPU board is used together with the extension board, please ensure to select the correct IAR-project.
Compile the project via “Project/Compile” or “Project/Rebuild all”.
5. Power up the device
6. Open a serial terminal e.g. PuTTY and connect it to the serial interface where the UART is connected to (also see section 3.2 for selecting the correct device). The following settings must be configured for the connection:



Configure the serial line	
Speed (baud)	115200
Data bits	8
Stop bits	1
Parity	None
Flow control	XON/XOFF

Figure 3.4: Serial Terminal settings RZ/N1D and RZ/N1S

7. Press any key on your keyboard to interrupt the bootloader.
8. Ensure to configure the U-boot boot command to release the Cortex-M3 core after reset. This is done by the command:


```
setenv bootcmd "mw 0x04000004 1 && rzn1_start_cm3 && loop 0 1"
```

 followed by


```
saveenv
```

 and reset the board.
9. Connect the debugger to the system via the "Download and Debug" button of the IAR Embedded Workbench.
10. After the Debug view opened, click on the "Go" button.

3.4.1.2 Loading application into flash via dfu-util

The board uses the u-boot bootloader for initial setup of the hardware and loading of the Cortex-M3 core firmware. This chapter describes how to install the compiled management software on the flash of the board. If no bootloader was yet installed on the board please refer to the Linux documentation – Quick Start Guide for U-Boot and Linux - *RZ/N1x-Quick-Start-Guide.pdf*.

The following steps describe the installation of management software:

1. Connect a Linux PC to the board according to section 3.2.
2. Power up the board.
3. Hit any key to stop the autoboot of the u-boot
4. Type "*dfu*" in the serial terminal of the board and hit enter.
5. On a Linux terminal start the command

```
sudo dfu-util -a'sf_cm3' -D FIRMWARE.bin
```

Replace FIRMWARE.bin with the file name of the software to install. The binary is placed at the subfolder Debug-RAM\Exe of the IAR project folder.

6. When the download process is complete, press Ctrl+C on u-boot.
7. If the autoboot command was already configured, go to step 10.
8. Set the autoboot command in the u-boot:


```
setenv bootcmd "sfprobe && sfread 0x4000000 d0000 80000 && rzn1_start_cm3 && loop 0 1"
```
9. Save the command to the flash: *saveenv*
10. Reset the device

3.4.2 Standalone Variant – RZ/N1L

The RZ/N1L does not use any bootloaders. If any application is stored in flash, it will be started automatically. Both, loading into RAM and flash can be done using IAR Embedded Workbench .

For the target workspace file, refer to "3.3 Sample Application"

1. Start the IAR Embedded Workbench
2. Open a project via Open/Workspace
3. Go to the workspace folder and open it:
4. Compile the project via "Project/Compile" or "Project/Rebuild all".
5. Power up the device
6. Open a serial terminal e.g. PuTTY and connect it to the serial interface where the UART is connected to (also see section 3.2 for selecting the correct device). The following settings must be configured for the connection:

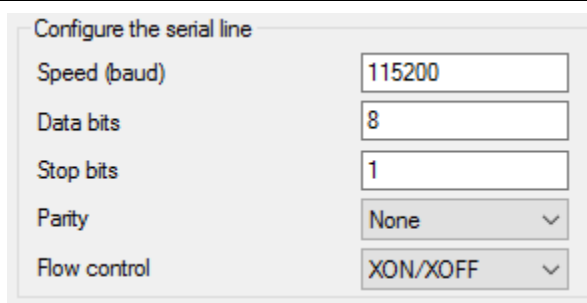


Figure 3.5: Serial Terminal settings RZ/N1L

7. Choose either the Debug-RAM or the Debug-ROM configuration. First it is used for debugging via IAR Embedded Workbench, second is loading the application into the flash.

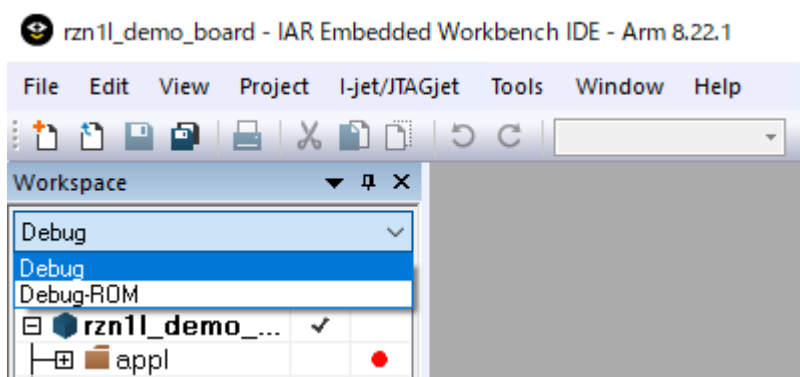


Figure 3.6: IAR Embedded Workbench Configurations RAM and ROM for RZ/N1L

8. If you have chosen Debug-ROM click on “Download and Debug”. Change reset mode to “Hardware” and press “Stop Debugging”. This is not necessary if you choose the Debug-RAM configuration.

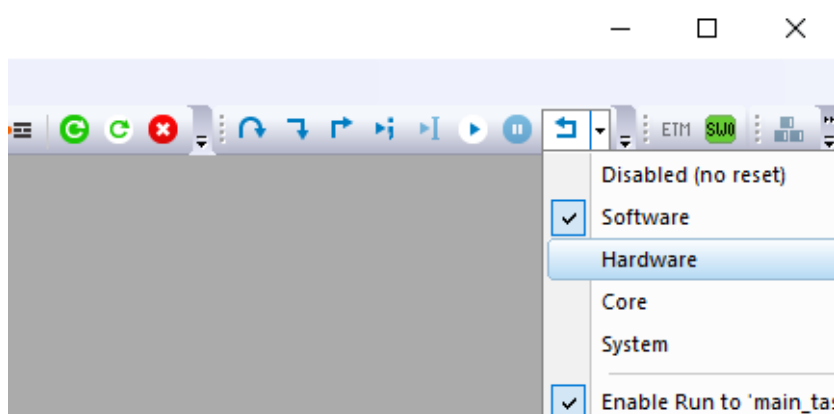


Figure 3.7: Changing Reset mode of RZ/N1L in Debug-ROM configuration

9. Connect the debugger to the system via the “Download and Debug” button of the IAR Embedded Workbench.
 10. After the Debug view opened, click on the “Go” button.

3.4.3 Core To Core variant – RZ/N1D (Communication Core)

The binary file for the Cortex-M3 core is located in the board type related IAR Embedded Workbench folder. For the target workspace file, refer to “3.3 Sample Application”

Load the binary file to the flash according to the following steps.

1. Connect a Linux PC to the board according to section 3.2
2. Power up the board.
3. Hit any key to stop the autoboot of the U-Boot
4. Type *dfu* in the serial terminal of the board and hit enter.
5. On a Linux terminal start the command

```
sudo dfu-util -a "sf_cm3" -D FIRMWARE
```

Replace FIRMWARE.bin with the file name of the software to install. The binary is placed at the subfolder Debug-RAM\Exe of the IAR project folder.

6. When the download process is complete, press Ctrl+C on u-boot.
7. If the autoboot command was already configured, go to step 10.
8. Set the autoboot command in the u-boot:

```
setenv bootcmd "sf probe && sf read 0x4000000 d0000 80000 && sf read 0x80fe0000 b0000 20000 && sf read  
0x80008000 1d0000 f00000 && rzn1_start_cm3 && sleep 4 && bootm 0x80008000 - 0x80fe0000"
```

9. Save the command to the flash: *saveenv*
10. Reset the device

3.4.4 Core To Core variant – RZ/N1D (Application Core)

The user application runs on the Linux system of the Cortex-A7. Its binary must be created by GCC and downloaded to the RZ/N1 board manually.

3.5.4.1 Building and downloading the user application

The following steps describe, how to build a binary and download it to the RZ/N1 board.

1. Navigate with the terminal of a Linux PC to the project of the application core at
goal/projects/goal_mbs_rpc_ac/01_tcp_server/gcc.
goal/projects/goal_mbs_rpc_ac/02_serial_master/gcc.
goal/projects/goal_mbs_rpc_ac/03_serial_slave/gcc.
goal/projects/goal_mbs_rpc_ac/04_tcp_ser_gateway/gcc.
2. Start the build process by executing the Makefile by typing
make
3. Select as target platform “rzn_a7_demo_board”.
4. Power up the board and wait till Linux booted successfully.
5. Copy the binary file *build/rzn_a7_demo_board/goal_rzn_a7_demo_board.bin* to the RZ/N1 board by e.g. secure copy (scp).
6. Start the binary file on the target by typing the commands
./goal_rzn_a7_demo_board.bin

The GOAL setups the connection to the communication core via Core To Core and starts the user application. The initialization is done when the log message “GOAL initialized” is printed at the terminal, if logging is activated.

4. CODESYS setup

4.1 Installation and start-up

Install CODESYS on PC and start it. Follow the startup guide for the procedure. CODESYS software and startup guide can be obtained from the following site

- Software : https://store.codesys.com/codesys.html?__store=en

4.2 Create project

Create a new project.

(1) Select the menu [File] → [New Project].

(2) In the New Project dialog, select [Projects] in [Categories], [Standard project] in [Templates], enter an arbitrary project name in [Name], save destination path in [Location] and confirm with [OK] to confirm.

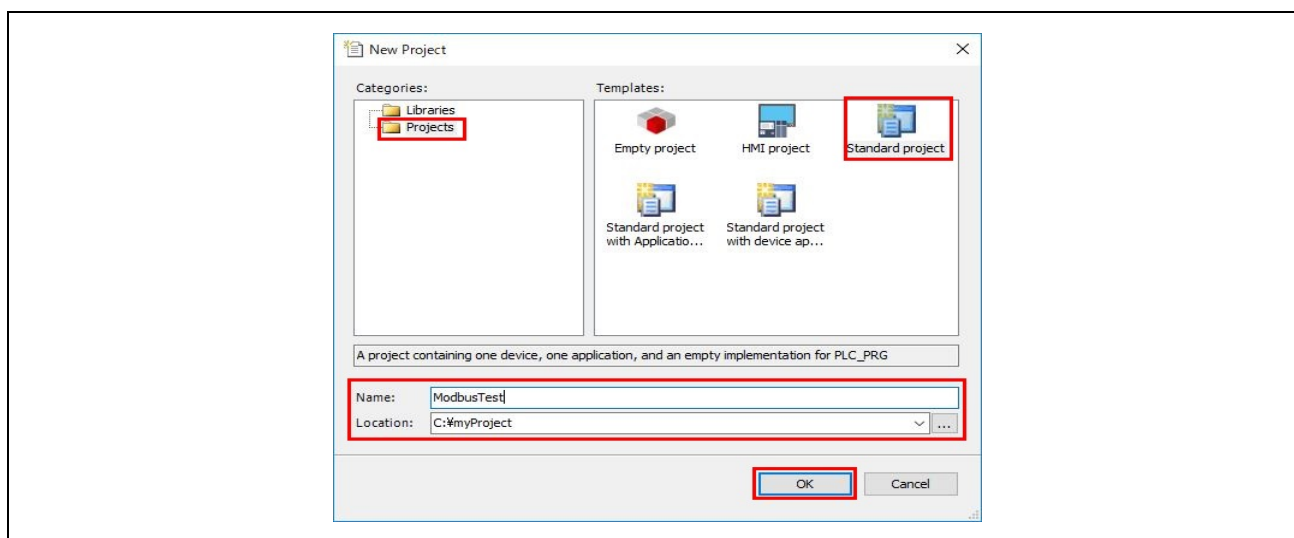


Figure 4.1: New Project dialog

(3) In the Standard Project dialog, select [CODESYS Control Win V3] for [Device] and [Structured Text (ST)] for [PLC_PRG in] and confirm with [OK]

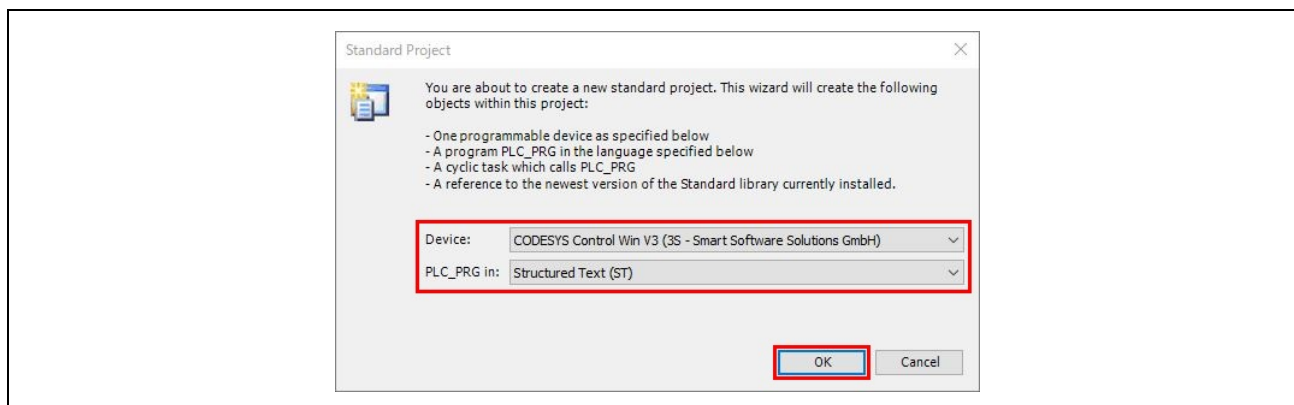


Figure 4.2: Standard Project dialog

New project is created, and the device tree is displayed in the device window.

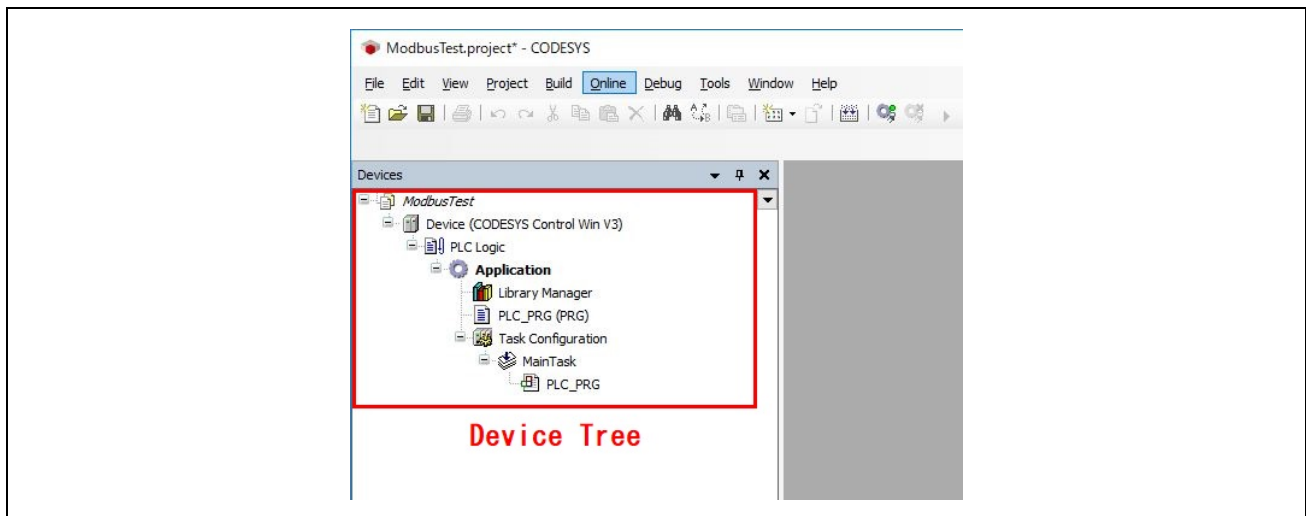


Figure 4.3: Device tree in the device window

4.3 Device configuration

Add a device object to the device tree of the device window according to the application to be used. Device object represents a hardware module.

(1) Select the "Device" object (right click) → [Add Device] in the device tree.

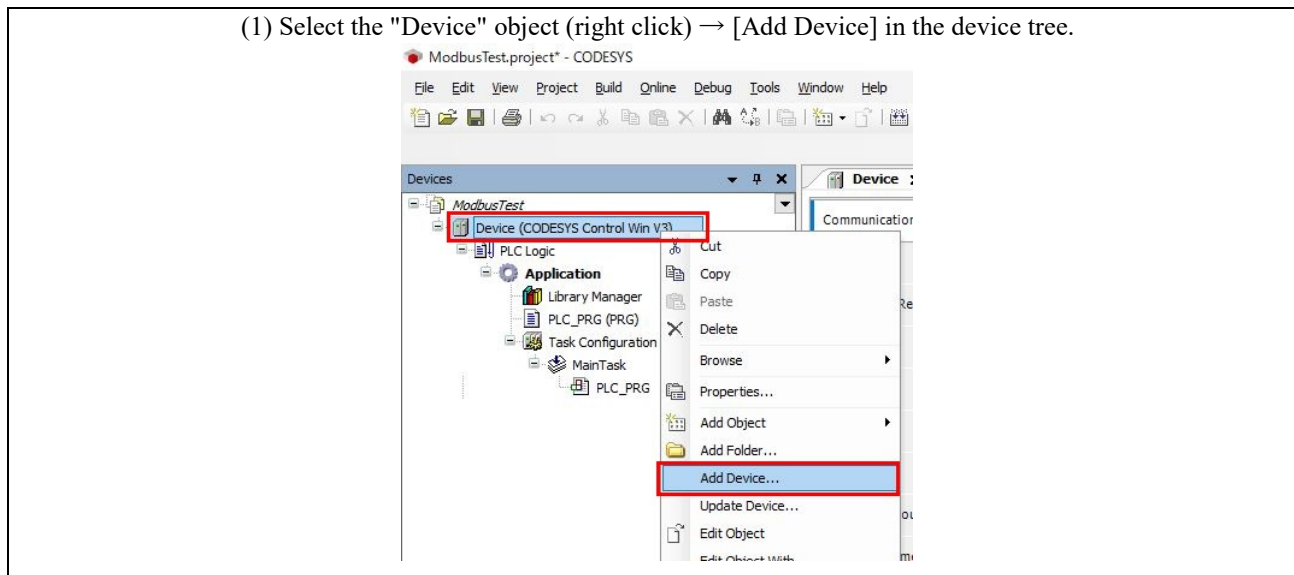


Figure 4.4: Add Device

(2) Select the device to add in the Add Device dialogue box and confirm with [Add Device].

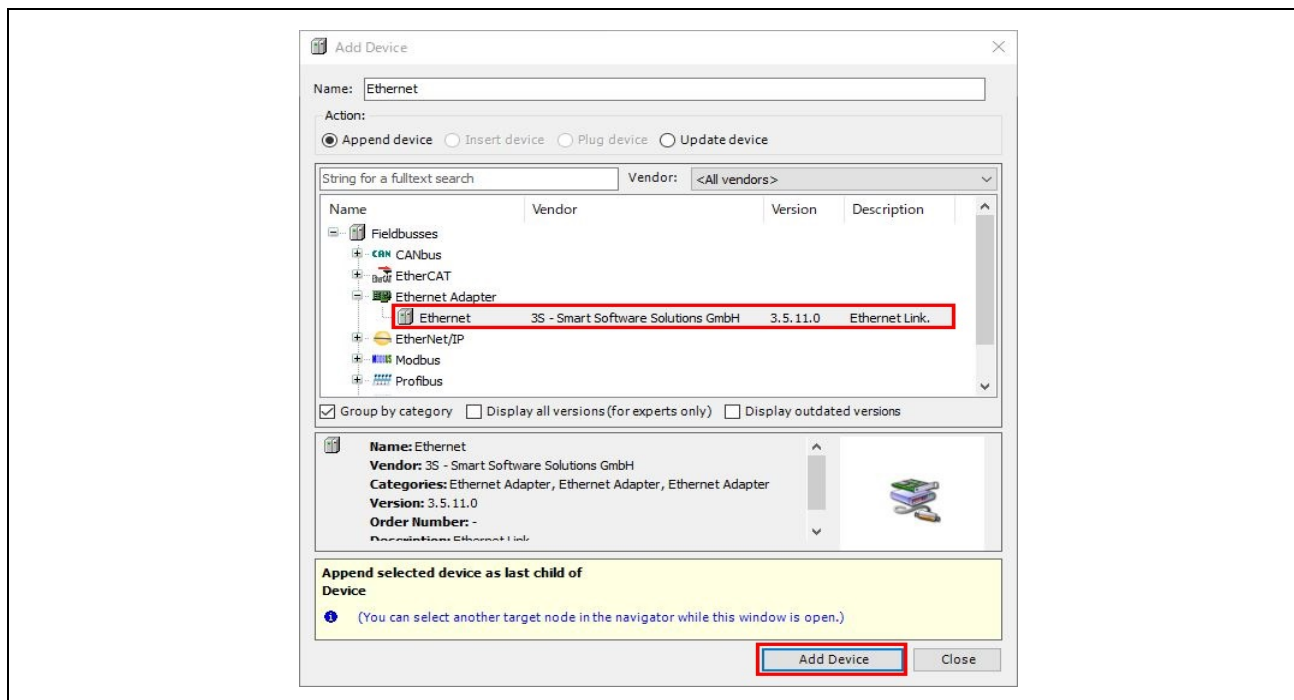


Figure 4.5: Add Device dialog

(3) The device object added in (2) is displayed in the device tree of the device window. When clicking and selecting this device, the display of the Add Device dialog will switch to the contents of the selected device, so please add child device object in the same procedure as (2). Close all dialogs after closing the dialog.

After adding all, close the dialog with [Close].

In case of close the Add Device dialog in the middle, right-click the target device object in the device window and select [Add Device] to open the dialog again.

4.3.1 Configuration of Modbus Serial slave device

In case of using the sample application of Modbus Serial master, configure the Modbus Serial slave device of the communication destination as follows.

Device		...	[1]
- Modbus COM	([Fieldbuses] → [Modbus] → [Modbus Serial Port])	...	[2]
- Modbus Serial Device	([Fieldbuses] → [Modbus] → [Modbus Serial Device])	...	[3]

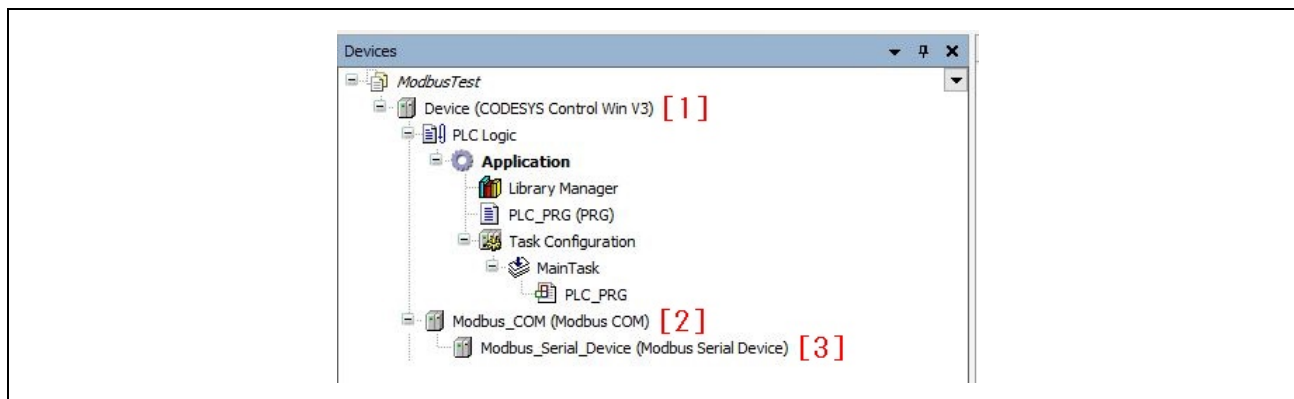


Figure 4.6: Device tree of Modbus Serial slave device

4.3.2 Configuration of Modbus Serial master device

In case of using the sample application of Modbus Serial slave, configure the Modbus Serial master device of the communication destination as follows.

Device		...	[1]
- Modbus COM	([Fieldbuses] → [Modbus] → [Modbus Serial Port])	...	[2]
- Modbus Master	([Fieldbuses] → [Modbus] → [Modbus Serial Master])	...	[3]
- Modbus Slave	([Fieldbuses] → [Modbus] → [Modbus Serial Slave])	...	[4]

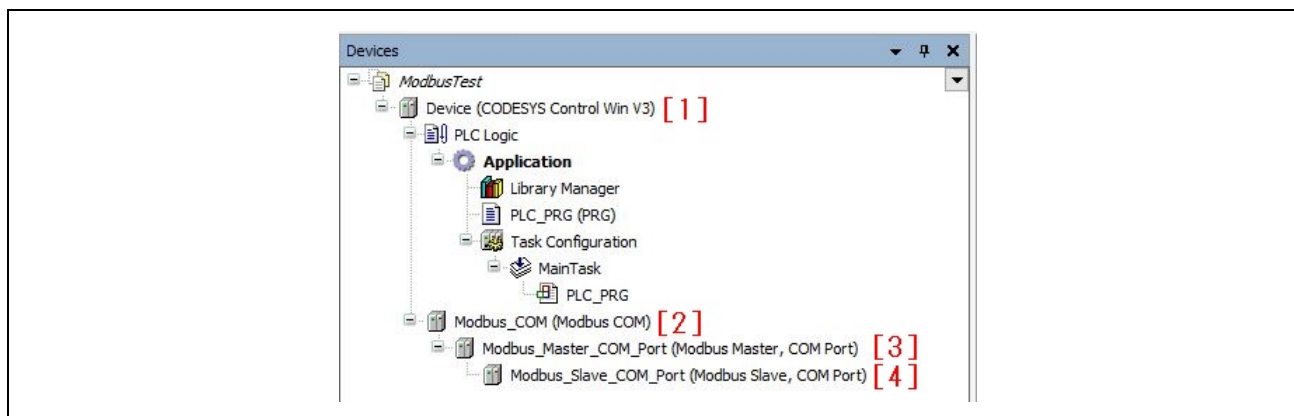


Figure 4.7: Device tree of Modbus Serial master device

4.3.3 Configuration of Modbus TCP master device

In case of using the sample application of Modbus TCP slave or Modbus TCP-Serial gateway, configure the Modbus TCP master device of the communication destination as follows.

- Modbus TCP slave

Device		...	[1]
- Ethernet	([Fieldbuses] → [Ethernet Adapter])	...	[2]
- Modbus TCP Master	([Fieldbuses] → [Modbus] → [Modbus TCP Master])	...	[3]
- Modbus TCP Slave	([Fieldbuses] → [Modbus] → [Modbus TCP Slave])	...	[4]

- Modbus TCP-Serial gateway

Device		...	[1]
- Ethernet	([Fieldbuses] → [Ethernet Adapter])	...	[2]
- Modbus TCP Master	([Fieldbuses] → [Modbus] → [Modbus TCP Master])	...	[3]
- Modbus TCP Slave	([Fieldbuses] → [Modbus] → [Modbus TCP Slave])	...	[4]
- Modbus Slave	([Fieldbuses] → [Modbus] → [Modbus Serial Slave])	...	[5]

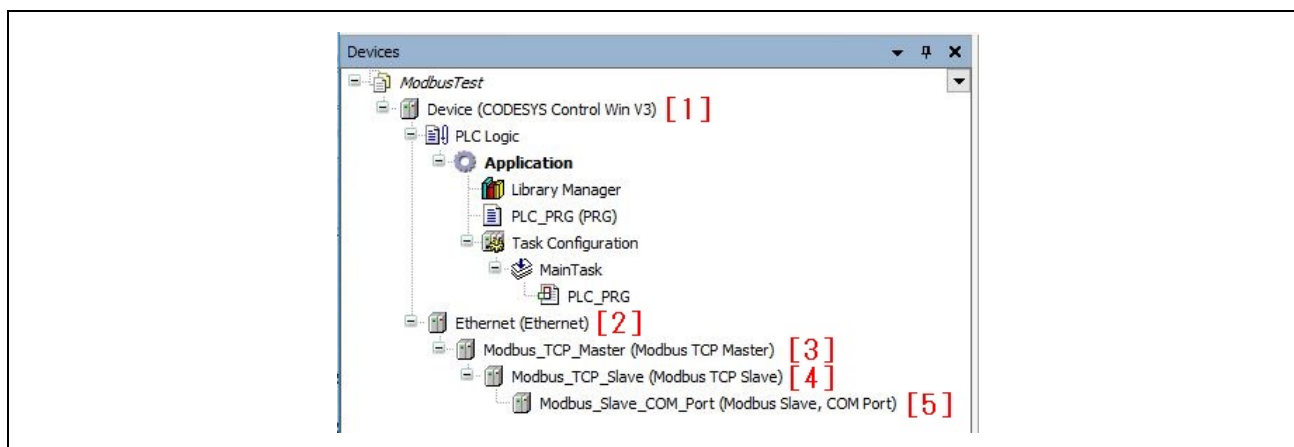


Figure 4.8: Device tree of Modbus TCP master device

4.4 Connection to Software PLC

Connect to the software PLC via the gateway server.

- (1) Select [CODESYS Gateway SysTray] (right click) → [Start Gateway] in the system tray to start the gateway server.
Confirm that the icon displays changes from " (stop) " to " (running)".
(If already started it do not need to run it.)

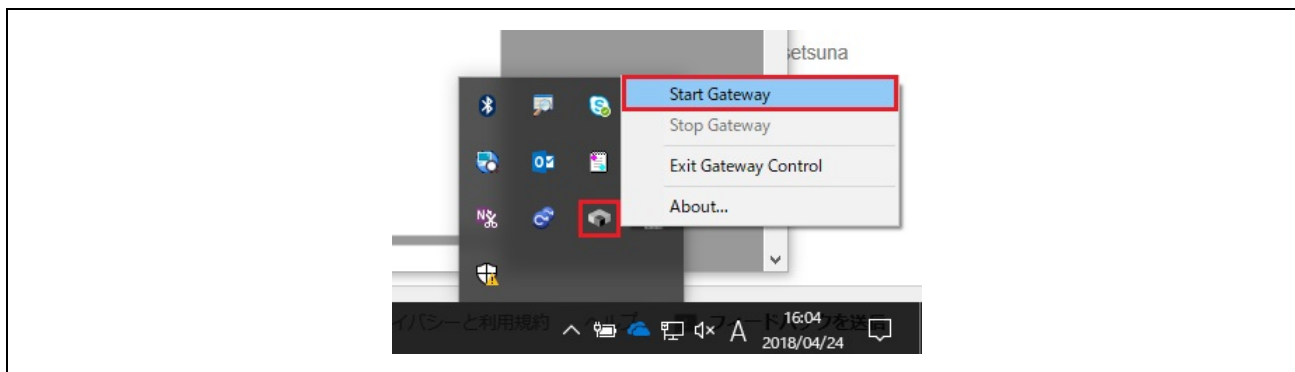


Figure 4.9: Starting the CODESYS gateway server

- (2) Select [CODESYS Control Win SysTray] (right click) → [Start PLC] in the system tray to activate the software PLC. Confirm that the icon displays changes from " (stop)" to " (running)".
(If already started it do not need to run it.)

In case of message about security appears, take necessary measures as necessary and click [OK].

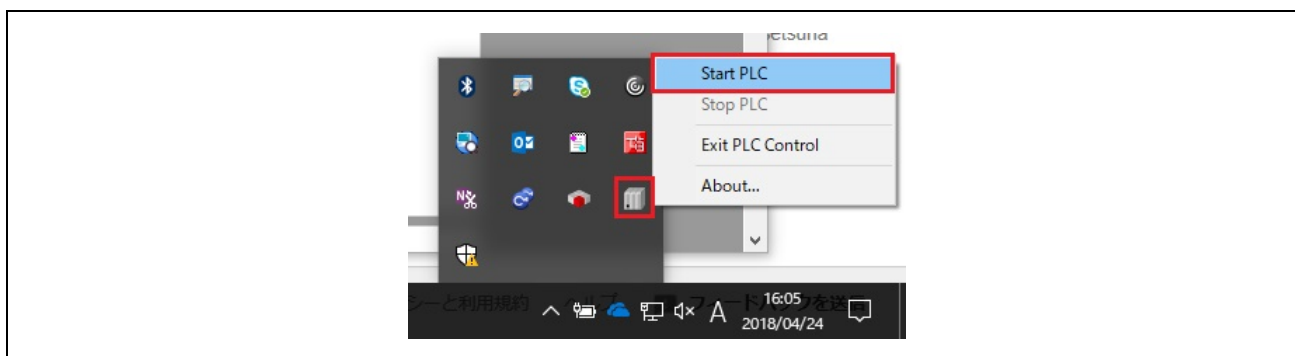


Figure 4.10: Starting the CODESYS software PLC

(3) Define the gateway on CODESYS. Double-click the "Device" object in the device window to open the device editor and select the "Communication Settings" tab. Click [Gateway] → [Add new gateway] at the top of the Communication Settings dialog.

(If the gateway is already displayed in the dialogue, proceed to (6)).

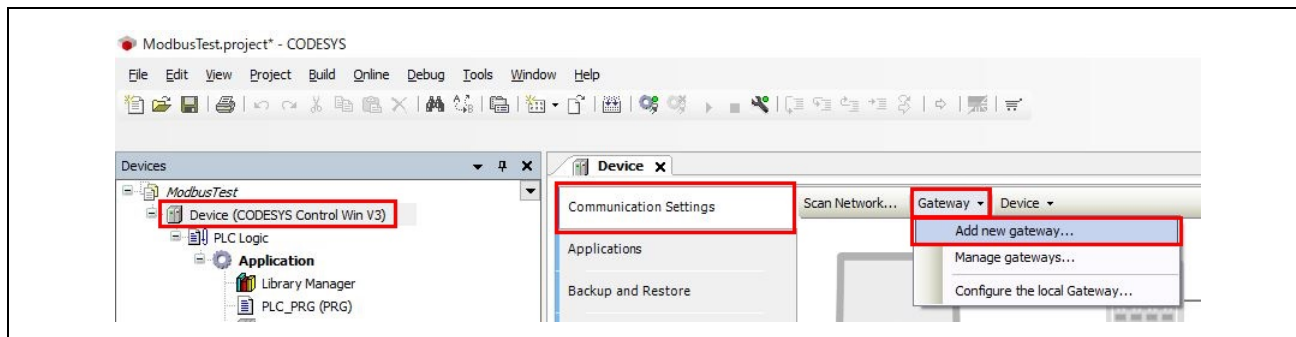


Figure 4.11: Add a new gateway

(4) In the Gateway dialog, select [TCP / IP] for [Driver], enter [localhost] for [IP-Address] and confirm with [OK].

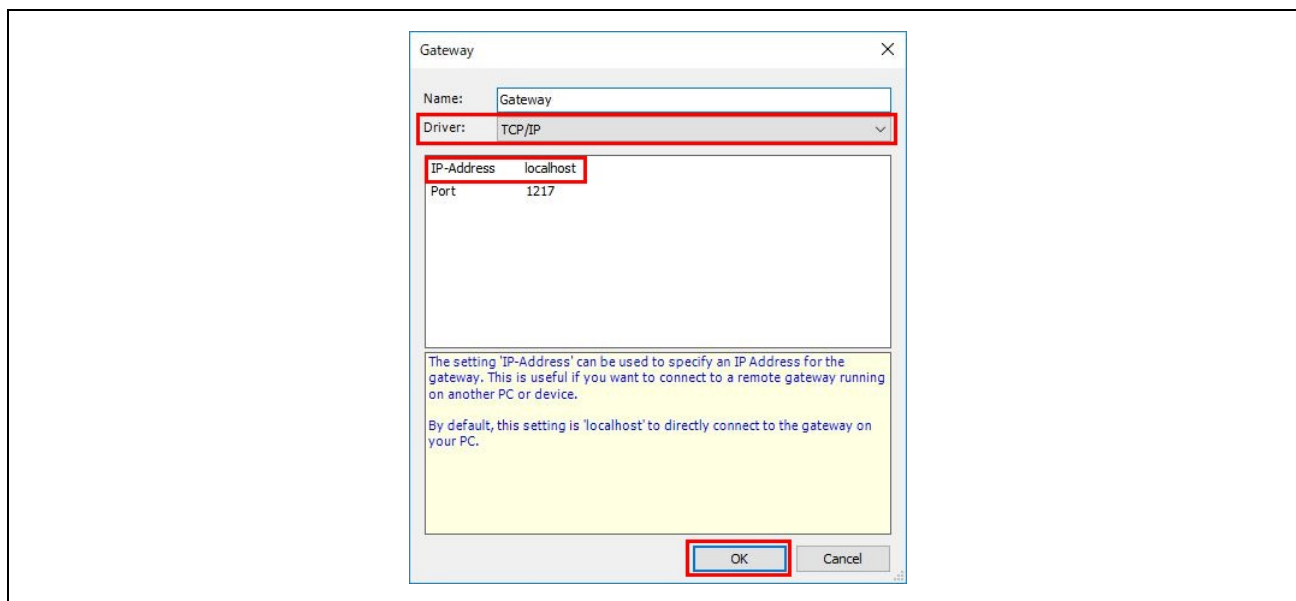


Figure 4.12: Gateway dialog

- (5) Gateway entry appears in the Communication Settings dialog. Check that the bottom right circle is green (normal execution state).

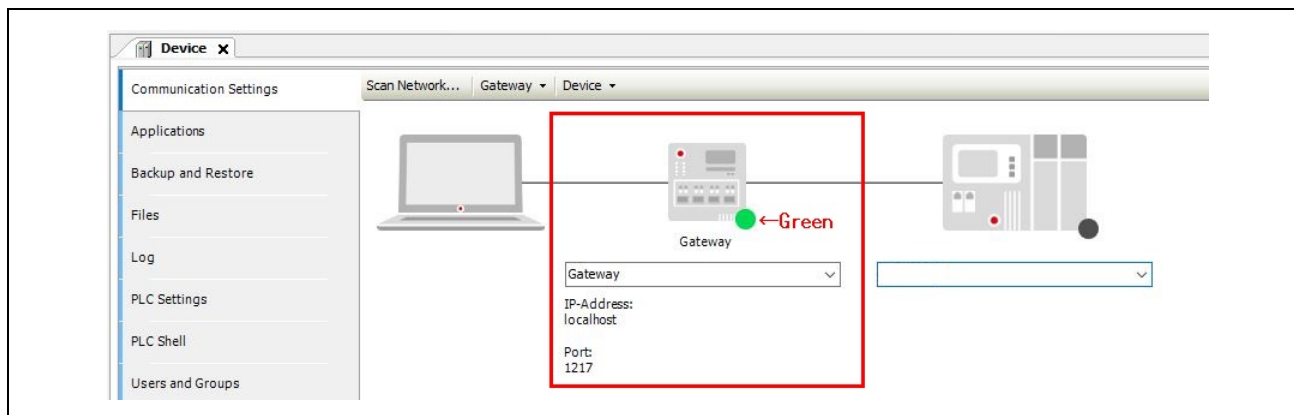


Figure 4.13: Gateway display of Communication Settings dialog

- (6) Scan the network and search for available PLCs. Click [Scan Network] at the top of the Communication Settings dialog.

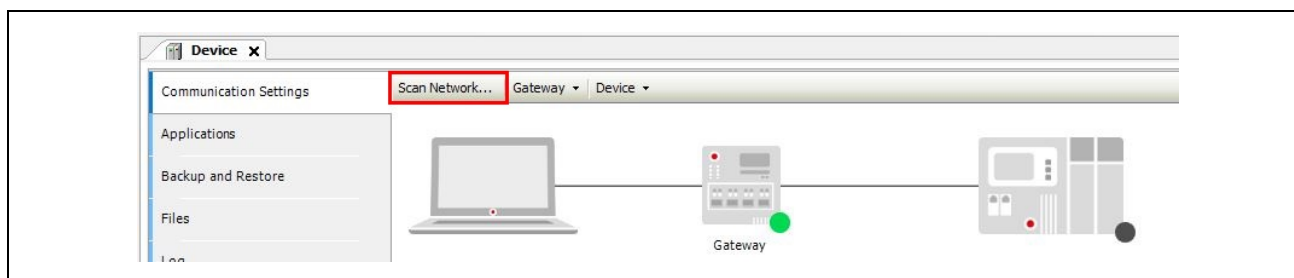


Figure 4.14: Network scan start

- (7) Click [Scan network] in the Select Device dialog. Since the software PLC started by (2) is displayed, double click it to activate it. The device name of the software PLC is the computer name of the PC. If it is not displayed, check (1), (2) again.

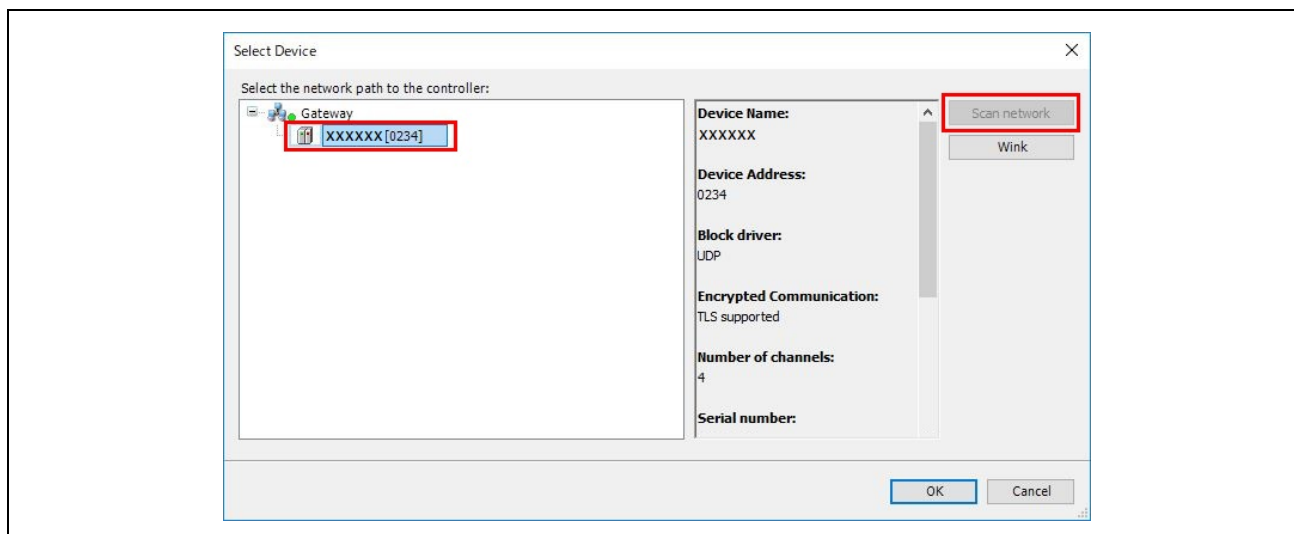


Figure 4.15: Select Device dialog

- (8) The software PLC is displayed in the Communication Settings dialog. Check that the bottom right circle is green (normal execution state).

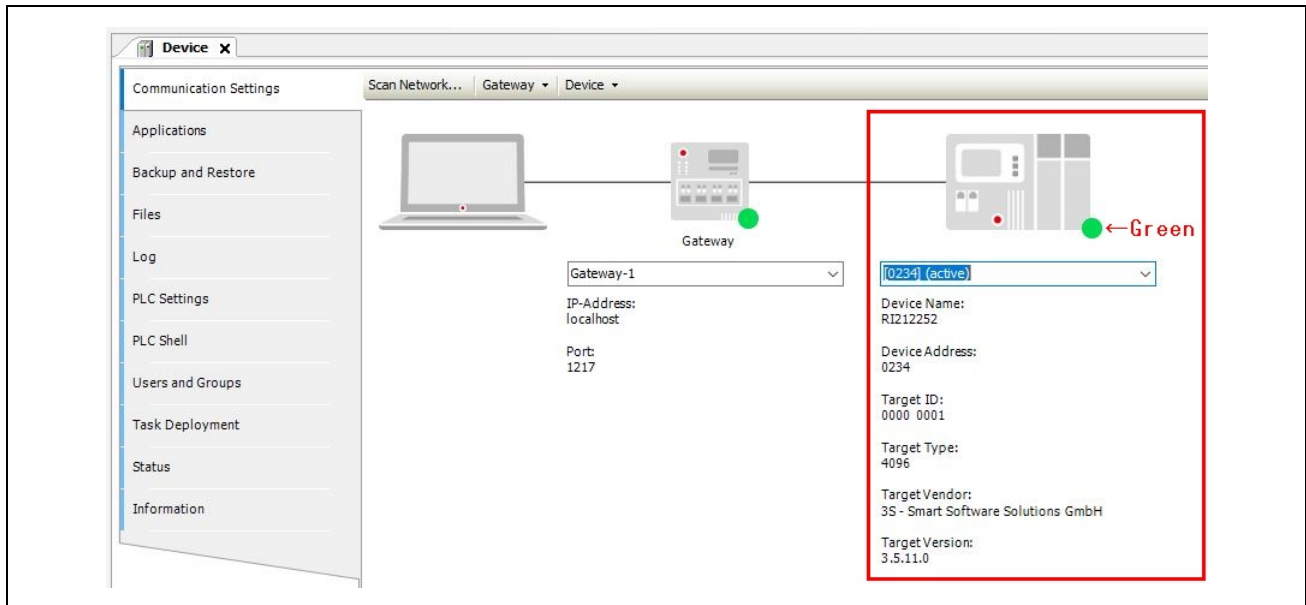


Figure 4.16: Software PLC display of Communication Settings dialog

4.5 Device settings

Configure each device object in the device tree configured in "4.3 Device Configuration" setting in the following procedure.

- (1) Double-click the target device object and open the device editor in the main window.
- (2) Select the tab on the left side of the device editor and open the dialog with setting target item.
- (3) Set the value to the displayed setting item.

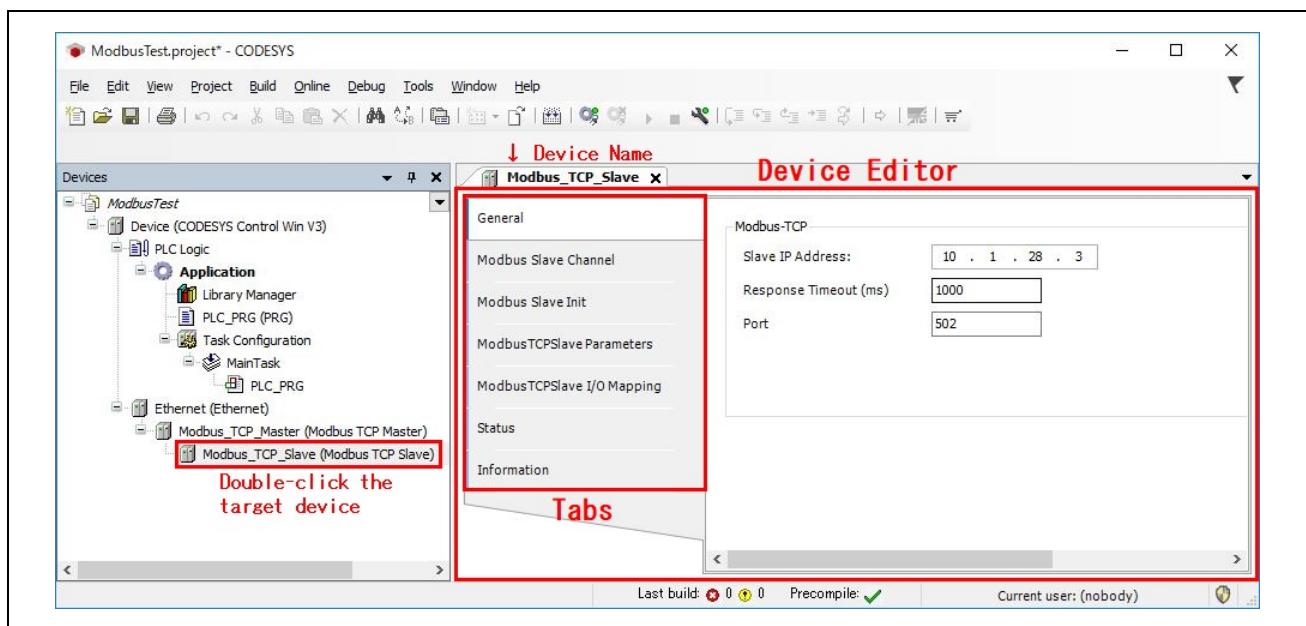


Figure 4.17: Device Editor

4.5.1 Modbus Serial Slave device settings

Settings of the Modbus Serial slave device are shown in the table below.

Table 4.1 Setting of Modbus Serial slave device

Device object	Tab	Dialog setting items		Setting value
[2] Modbus COM	General	COM Port		PC connection COM port
		Baud Rate		Baud rate to execute
		Parity		NONE
		Data Bits		8 (RTU) 7 (ASCII)
		Stop Bits		1
[3] Modbus Serial Device	General	Unit ID		1
		Holding Registers		16
		Input Registers		16
		Start	Coils	2000
		Addresses	Discrete Inputs	2000
			Holding Register	2000
			Input Register	2000
	Modbus Serial Device	Always update variables		Enabled 2
	I/O Mapping			(Always in bus cycle task)

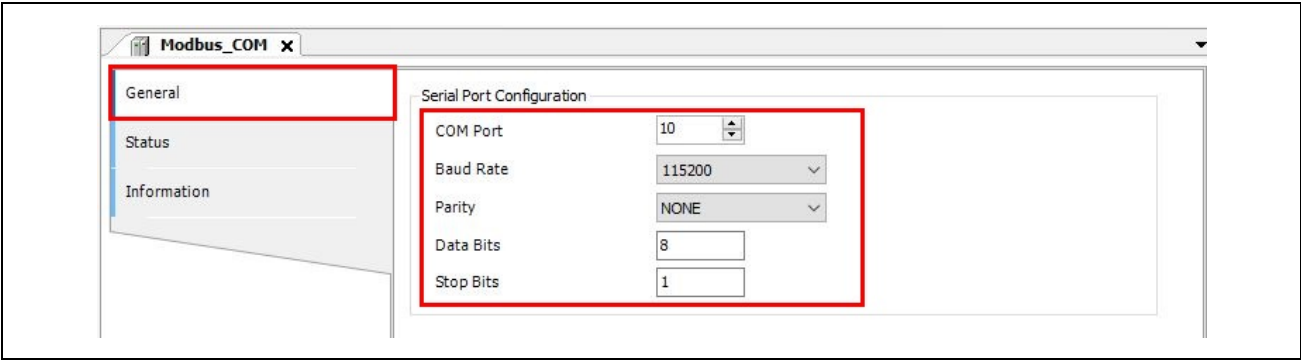


Figure 4.18: Modbus Serial slave device setting - Modbus COM object

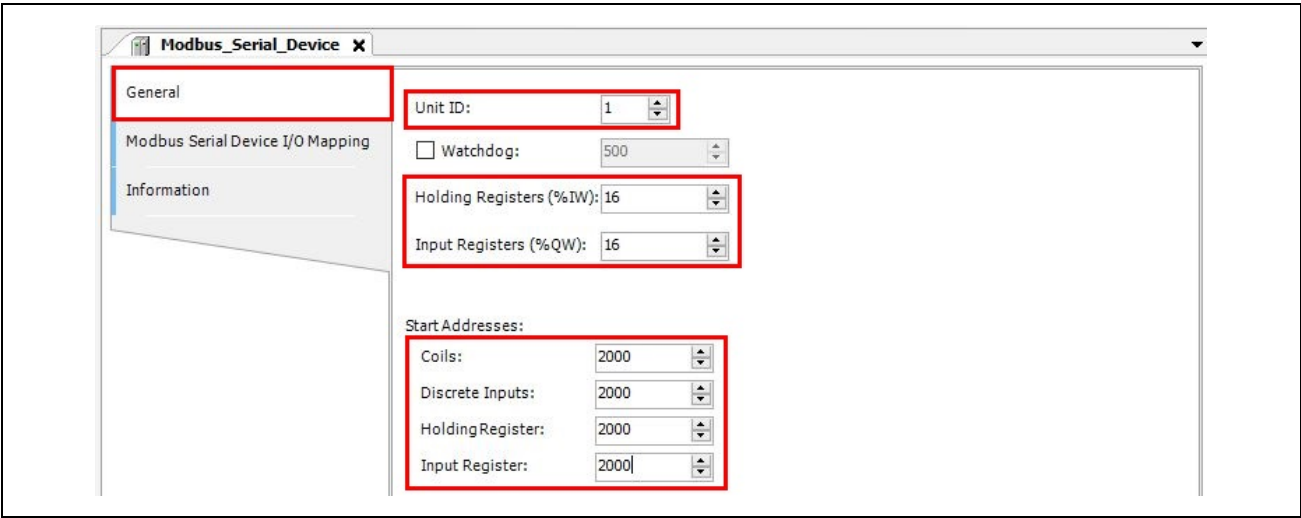


Figure 4.19: Modbus Serial slave device setting - Modbus Serial Device object (1)

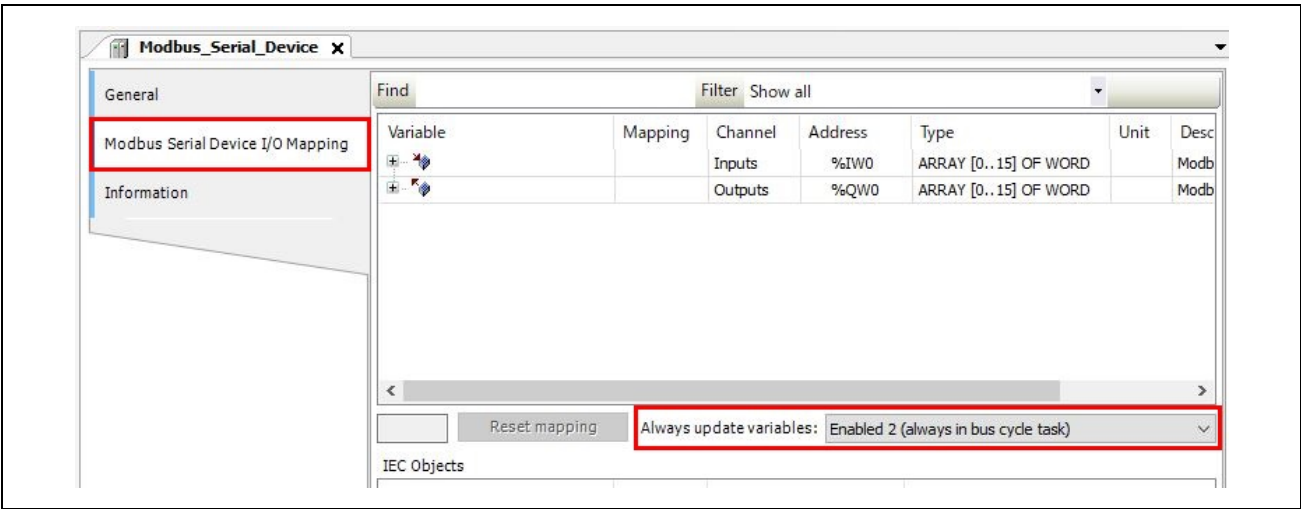


Figure 4.20: Modbus Serial slave device setting - Modbus Serial Device object (2)

4.5.2 Modbus Serial Master device settings

Settings of the Modbus Serial master device are shown in the table below.

Table 4.2 Setting of Modbus Serial master device

Device object	Tab	Dialog setting items	Setting value
[2] Modbus COM	General	COM Port	PC connection COM port
		Baud Rate	Baud rate to execute
		Parity	NONE
		Data Bits	8 (RTU) 7 (ASCII)
		Stop Bits	1
[3] Modbus Master	General	Transmission Mode	Transmission mode to be executed
		auto-restart communication	When reconnecting at error occurrence
[4] Modbus Slave	General	Slave Address	1

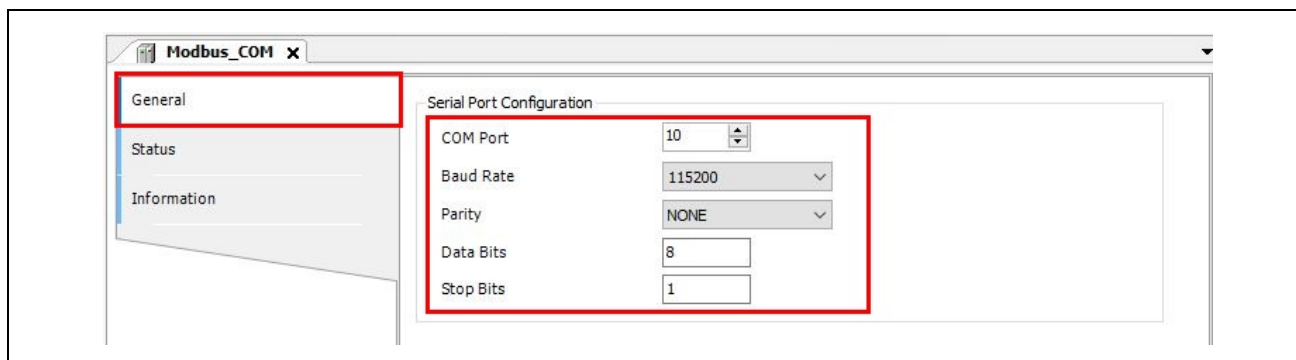


Figure 4.21: Modbus Serial master device setting - Modbus COM object

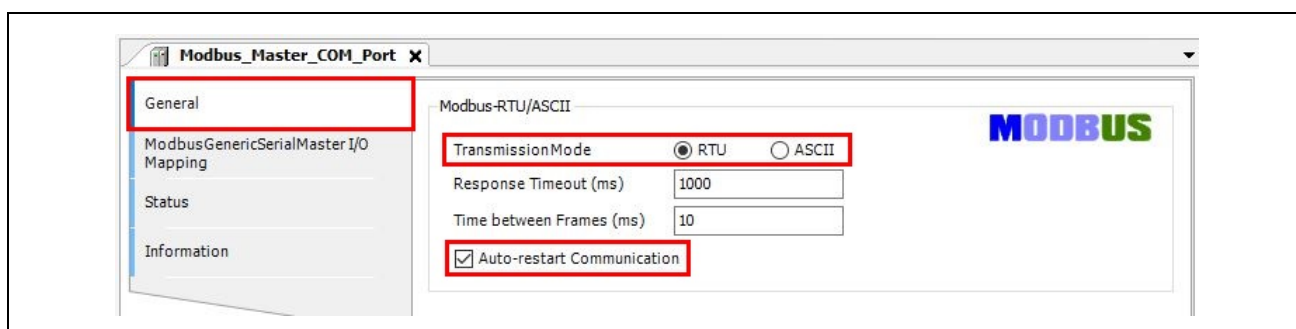


Figure 4.22: Modbus Serial master device configuration - Modbus Master object

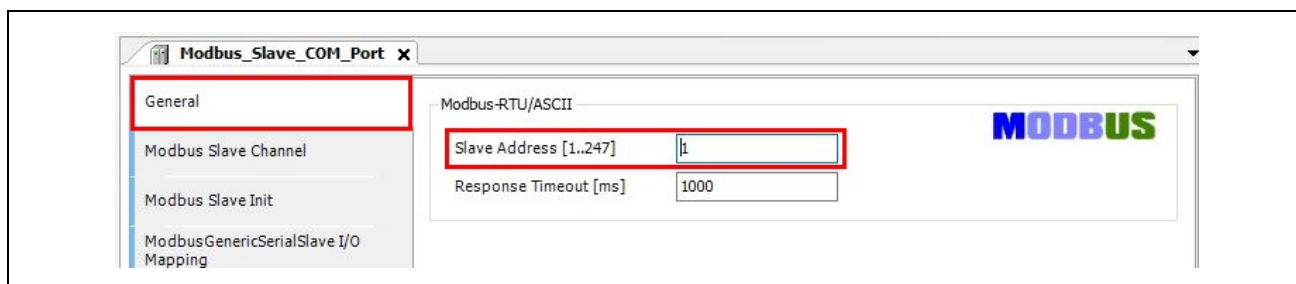


Figure 4.23: Modbus Serial master device setting - Modbus Slave object

4.5.3 Modbus TCP Master device settings

Since the "Ethernet" object setting reflects the Ethernet setting of the PC, first set up the PC.

Table 4.3 PC's Ethernet settings

Item	Setting value
IP address	10.1.28.X (X is other than 1,3)
Subnet mask	255.255.255.0
Default gateway	No setting

Settings of the Modbus TCP master device are shown in the table below.

Table 4.3 CODESYS Modbus TCP master setting

Device object	Tab	Dialog setting items	Setting value
[2] Ethernet	General	Interface	Select interface connected to RZ/N1 board
		IP Address	Same as PC setting
		Subnet Mask	
		Default Gateway	
[3] Modbus TCP Master	General	Auto-reconnect	When reconnecting at error occurrence
[4] Modbus TCP Slave	General	Slave IP Address	10.1.28.3
		Port	502 or 1024
[5] Modbus Slave	General	Slave Address	Address of the connected slave device

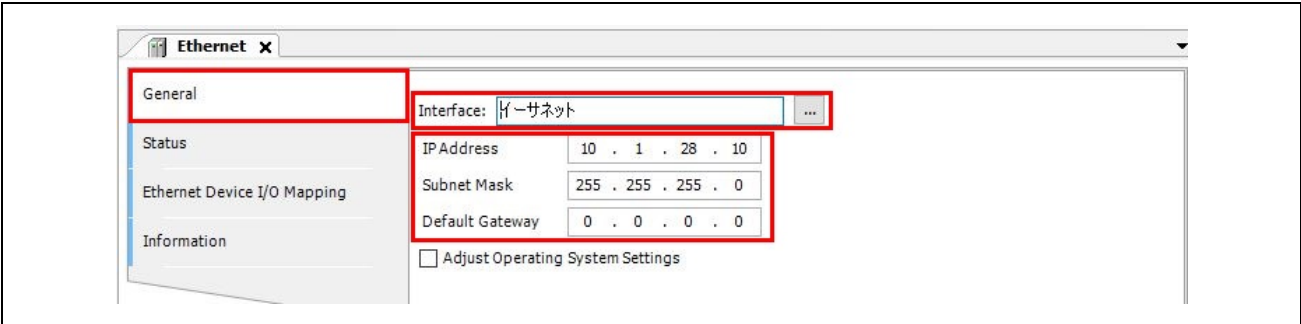


Figure 4.24: Modbus TCP master device configuration - Ethernet object

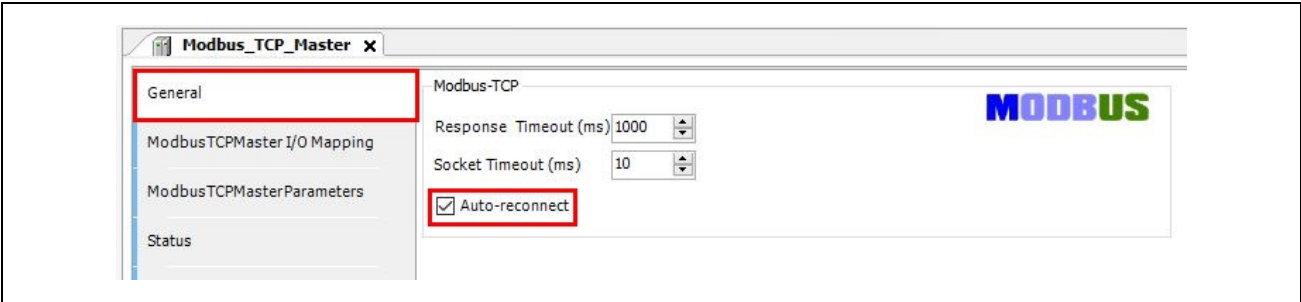


Figure 4.25: Modbus TCP master device configuration - Modbus TCP Master object

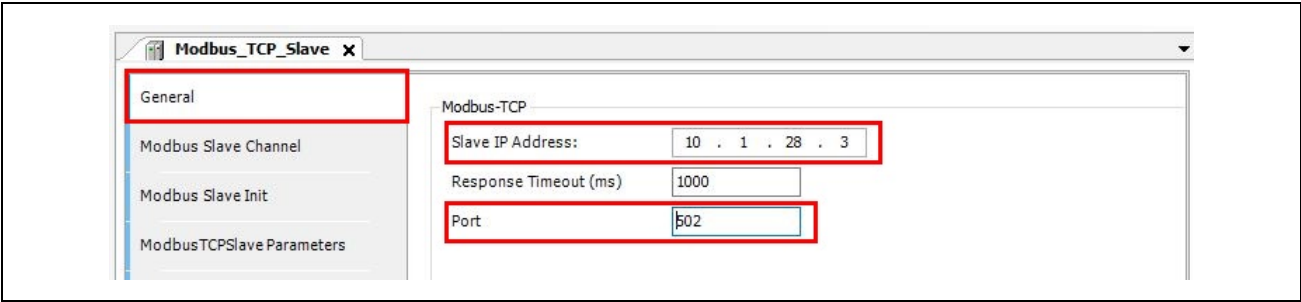


Figure 4.26: Modbus TCP master device configuration - Modbus TCP Slave object

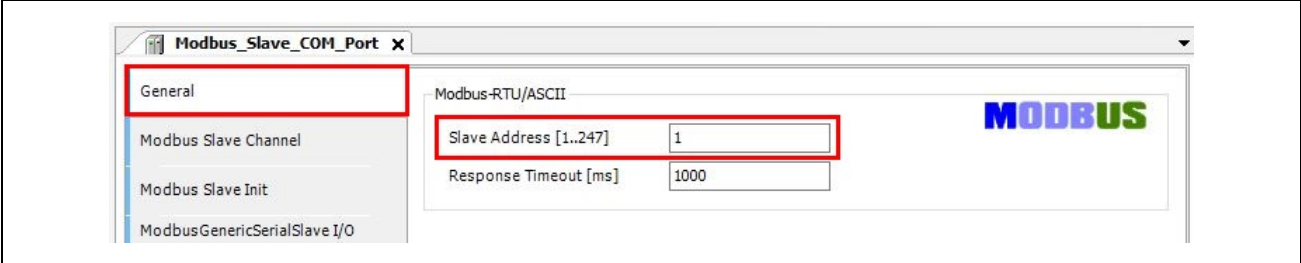


Figure 4.27: Modbus TCP master device configuration - Modbus Slave object

5. Test communication with the sample application

5.1 Execution procedure

This is a procedure to communicate using the sample application. For details on how to operate each procedure, refer to "5.2 Sample application operation method" and "5.3 CODESYS operation method"

5.1.1 Modbus Serial Master execution procedure

Procedure for executing the Modbus Serial master sample application is shown in the table below, Communication destination is a software PLC that operates as a Modbus Serial slave device.

Table 5.1 Sample application execution sequence of Modbus Serial master

Master operation	Slave operation	Operation contents
(1)	Start operation of software PLC application	Refer to 5.3.2.
(2) Start-up		Refer to 5.2.1
(3) Start terminal software		Refer to 5.2.2
(4) Selection of serial communication setting		Refer to 5.2.3
(5)	Read value setting of Read Function	Execute as necessary Refer to 5.3.3
(6) Select execution function		Refer to 5.2.4
(7)	Write function check result check	Execute as necessary Refer to 5.3.6
(5) - (7) can be executed repeatedly.		

5.1.2 Modbus Serial Master Slave execution procedure

Procedure for executing the Modbus Serial slave sample application is shown in the table below, Communication destination is a software PLC that operates as a Modbus Serial slave device

Table 5.2 Sample application execution sequence of Modbus Serial slave

Master operation	Slave operation	Operation contents
(1)	Start-up	Refer to 5.2.1
(2)	Start terminal software	Refer to 5.2.2
(3)	Selection of serial communication setting	Refer to 5.2.3
(4) Select execution function		Refer to 5.3.1
(5) Start operation of software PLC application		Refer to 5.3.2
(6) Write function check result check		Execute as necessary Refer to 5.3.4
(7) Confirm execution result		Refer to 5.3.5
(8) Read function read result check		Execute as necessary Refer to 5.3.7
(6), (8) can be executed repeatedly.		
(9) Stop operation of software PLC application		Refer to 5.3.2
(4) - (9) can be executed repeatedly.		

5.1.3 Modbus TCP Server execution procedure

Procedure for executing the Modbus TCP Server sample application is shown in the table below, Communication destination is a software PLC that operates as a Modbus TCP Master device.

Table 5.3 Sample application execution sequence of Modbus TCP slave

Master operation	Slave operation	Operation contents
(1)	Start-up	Refer to 5.2.1
(2) Select execution function		Refer to 5.3.1
(3) Start operation of software PLC application		Refer to 5.3.2
(4) Write function check result check		Execute as necessary Refer to 5.3.4
(5) Confirm execution result		Refer to 5.3.5
(6) Read function read result check		Execute as necessary Refer to 5.3.7
(4)~(6) can be executed repeatedly		
(7) Stop operation of software PLC application		Refer to 5.3.2
(2)~(7) can be executed repeatedly		

5.1.4 Modbus TCP-Serial Gateway execution procedure

Procedure for executing the Modbus TCP Serial Gateway sample application is shown in the table below, Communication destination is a software PLC that operates as a Modbus TCP Master device.

Table 5.4 Sample application execution sequence of Modbus TCP Serial Gateway

Master operation	Slave operation	Operation contents
(1)	Start-up	Refer to 5.2.1
(2)	Start terminal software	Refer to 5.2.2
(3)	Selection of serial communication setting	Refer to 5.2.5
(4) Select execution function for Modbus TCP Slave		Execute as necessary Refer to 5.3.1
(5) Select execution function for Modbus Slave		Execute as necessary Refer to 5.3.1
(6) Start operation of software PLC application		Refer to 5.3.2
(7) Write function check result check		Execute as necessary Refer to 5.3.4
(8) Confirm execution result		Refer to 5.3.5
(9) Read function read result check		Execute as necessary Refer to 5.3.7
(8)~(10) can be executed repeatedly.		
(11) Stop operation of software PLC application		Refer to 5.3.2
(5)~(11) can be executed repeatedly.		

5.2 Method of operate the sample application

This section explains how to operate the application when testing the communication by the sample application.

5.2.1 Starting the sample application

Run the source code that was built with "3.3 Sample application" with the debugger.

(1) Turn on the power of the RZ/N1 board.

(2) Download the program using the EWARM menu [Project] → [Download and Debug], or the Download and Debug button shown below.

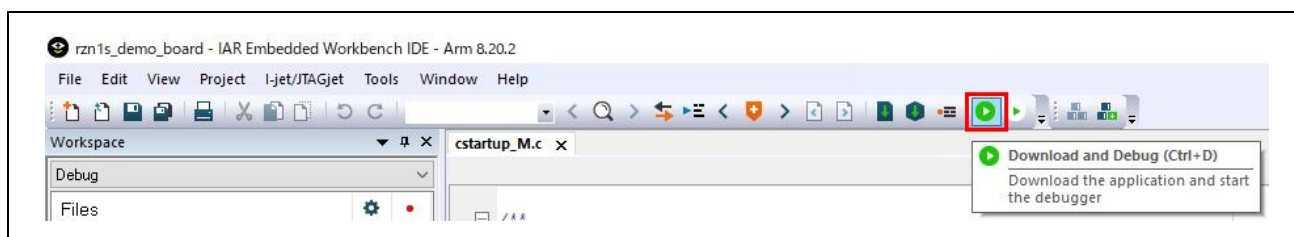


Figure 5.1: EWARM download button

(3) Execute the program using the menu [Debug] → [Go], or the Go button in the figure below.

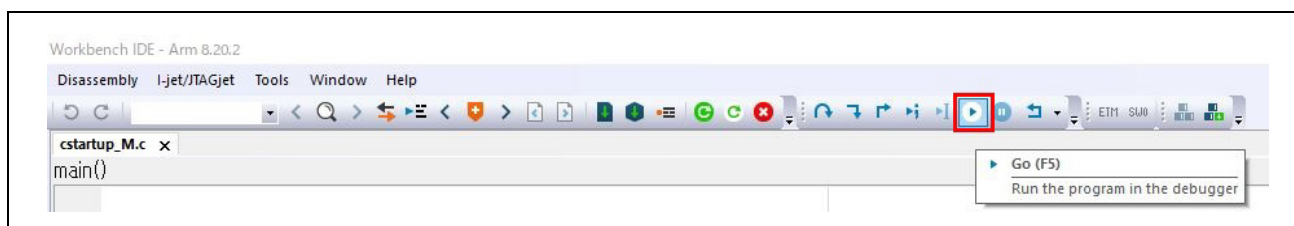


Figure 5.2: EWARM execution button

5.2.2 Terminal software setting

Some sample applications require terminal input by serial communication, and terminal software is required. Set the following table for terminal software.

Table 5.5 Terminal software serial port setting

Item	Setting
Connection COM port	The third port of the four COM ports
Baud rate	115200
Data bit	8
Parity	None
Stop bit	1

5.2.3 Method of select the serial communication setting

If the sample application is a Modbus Serial master or slave, the following menu will be displayed on the terminal software after starting, and serial communication setting will be selected. The setting cannot be changed on the way. Enter the menu number of the setting you want to test from the keyboard. Be sure to select so that the baud rate matches the CODESYS transmission mode set in "4.5 Device Settings".

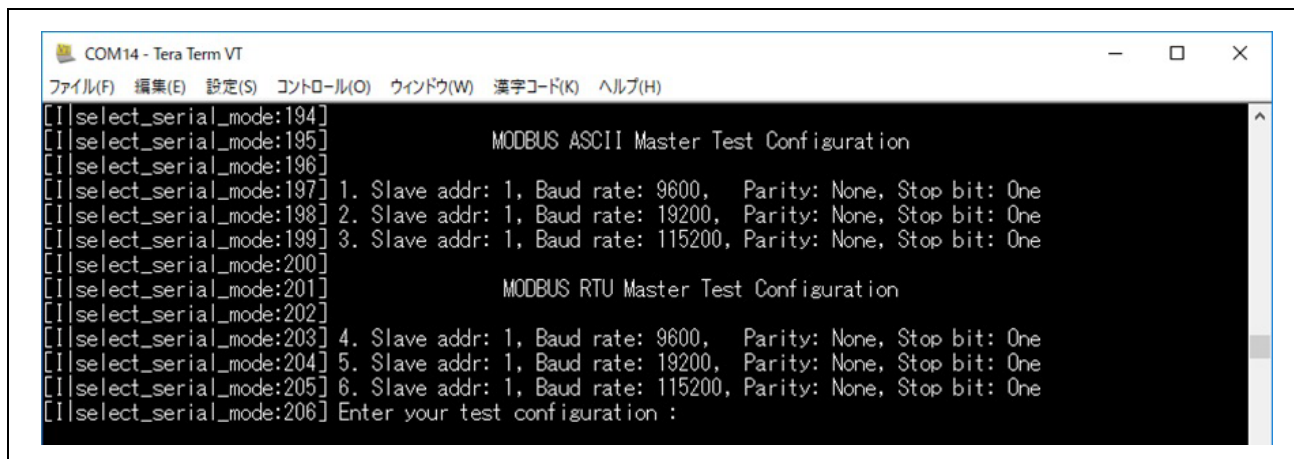


Figure 5.3: Select serial port setting menu screen

Table 5.6 Menu number and contents of serial communication setting

Menu No	Transmission mode	Slave D	Baud rate	Parity	Stop bit
1	ASCII	1	9600	None	1bit
2			19200		
3			115200		
4	RTU		9600		
5			19200		
6			115200		

5.2.4 Method of select the execution function

If the sample application is Modbus Serial master, after selecting the serial port setting, the following menu is displayed on the terminal software and the execution function is selected. Enter the menu number of the function want to test from the keyboard. Send a request message once with one input. The execution result is displayed after input. Success if "SUCCESS" is displayed, failure when "FAILURE" is displayed.

After the execution result is displayed, the menu is displayed again, and it becomes selectable. Select "0" to end the test.

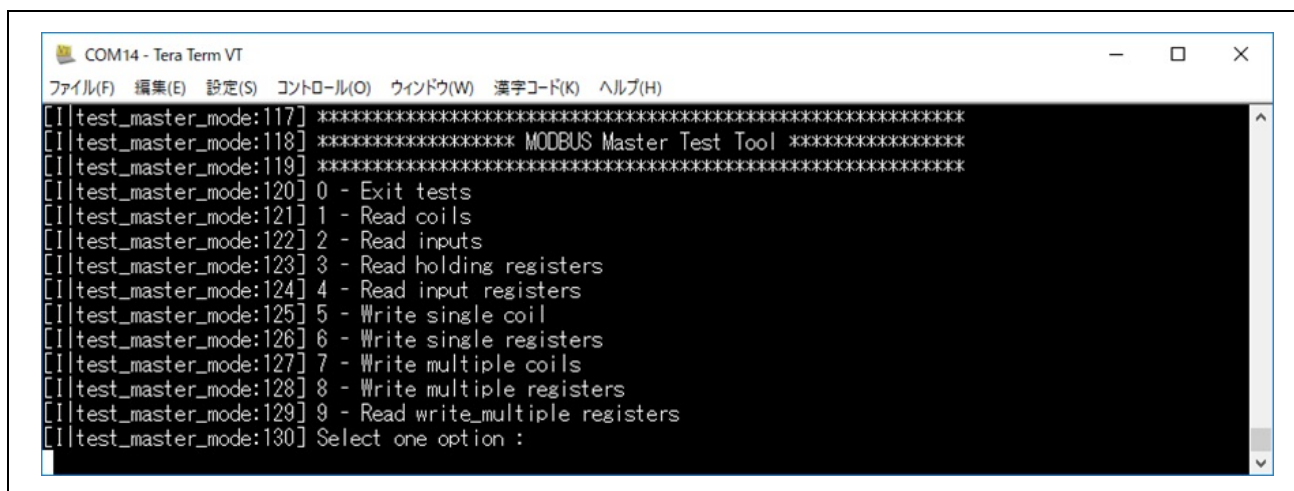


Figure 5.4: Execution function selection menu screen

The menu number and the contents of the request message issued by the sample application are listed in the table below.

Table 5.7 Menu number and execution function contents

Menu No		Function cord	Address	Data	Write value
0		(Test end)			
1	1	Read Coils	2000	1 Coil	-
2	2	Read Discrete Inputs	2000	16 Pins	-
3	3	Read Holding Registers	2000	16 Registers	-
4	4	Read Input Register	2000	16 Registers	-
5	5	Write Single Coil	2000	1 Coil	0xFF00
6	6	Write Single Register	2000	1 Register	0x55AA
7	15	Write Multiple Coils	2000	16 Coils	All 0
8	16	Write Multiple Registers	2000	16 Registers	All 0
9	23	Read/Write Multiple Registers	(Read)	2000	8 Registers
			(Write)	2000	16 Registers
					All 0

5.2.5 Method of select Modbus gateway settings

In case of the sample application is a Modbus TCP-Serial gateway, the following menu is displayed on the terminal software after selection, and the setting of Modbus gateway setting is selected. The setting cannot be changed on the way. Enter the number of the setting you want to test from the keyboard. Be sure to select the transmission mode so that it matches the slave RZ/N1 device. "C. DISABLE_FUNC_CODE" is not supported by Core To Core variant

```

COM10 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) 漢字コード(K) ヘルプ(H)
[I|start_tcp_gateway_mode:71] *****
[I|start_tcp_gateway_mode:72] ***** MODBUS Gateway Test Configuration *****
[I|start_tcp_gateway_mode:73] *****
[I|start_tcp_gateway_mode:74] 0. EXIT: Exit Test
[I|start_tcp_gateway_mode:75] 1. ENABLE_GATEWAY_TCP: Enable gateway as TCP Server
[I|start_tcp_gateway_mode:76] 2. ENABLE_GATEWAY_RTU: Enable gateway as TCP Server with RTU Serial device
[I|start_tcp_gateway_mode:77] 3. ENABLE_GATEWAY_ASCII: Enable gateway as TCP Server with ASCII Serial device
[I|start_tcp_gateway_mode:78] 4. DISABLE_GATEWAY_TCP: Disable gateway as TCP Server
[I|start_tcp_gateway_mode:79] 5. DISABLE_GATEWAY_RTU: Disable gateway as TCP Server with RTU Serial device
[I|start_tcp_gateway_mode:80] 6. DISABLE_GATEWAY_ASCII: Disable gateway as TCP Server with ASCII Serial device
[I|start_tcp_gateway_mode:81] 7. ENABLE_SECURITY_ACCEPT_IP_LIST_CONN_EST: Enable Security mode with accept IP list(PC IP beigh
[I|start_tcp_gateway_mode:82] 8. ENABLE_SECURITY_ACCEPT_IP_LIST_CONN_REJECT: Enable Security mode with accept IP list(PC IP noh
[I|start_tcp_gateway_mode:83] 9. ENABLE_SECURITY_REJECT_IP_LIST_CONN_REJECT: Enable Security mode with reject IP list(PC IP beh
[I|start_tcp_gateway_mode:84] a. ENABLE_SECURITY_REJECT_IP_LIST_CONN_EST: Enable Security mode with reject IP list(PC IP not beh
[I|start_tcp_gateway_mode:85] b. DISABLE_SECURITY: Disable Security mode
[I|start_tcp_gateway_mode:86] c. DISABLE_FUNC_CODE: Function code is not supported
[I|start_tcp_gateway_mode:88]
Enter your test configuration :

```

Figure 5.5: Modbus gateway setting selection menu screen

Table 5.6 Menu number and execution function contents

Menu No	Security settings	Function of Modbus TCP Server	Transmission mode	Baud rate	Parity	Stop bit
0			(Test end)			
1	None	Enable	RTU	115200	None	1bit
2			RTU			
3			ASCII			
4		Disabled	RTU			
5			RTU			
6			ASCII			
7	Set the ACCEPT IP	Enable	RTU			
8	Address					
9	Set the REJECT IP					
a	Address					
b	None					
	None	Enable				
c		Function codes 3 and 5 are not supported	RTU			

In case of select "7", "8" or "9" or "a" for the menu number, enter the IP address to be permitted or rejected after selection, enter "Enter" at the end of input. The input format is "X.X.X.X (Enter)". Enter three IP addresses. If the number of IP addresses you want to allow or deny is less than 3, please enter the same address.

5.3 CODESYS operation method

This section explains how to operate CODESYS when testing communication using the sample application.

5.3.1 Method of set execution functions and I/O mapping

In case of the software PLC operates as a master device, set the execution function in the slave device object in the device tree. The setting procedure is described below.

This setting is done with logged out from the software PLC. If you are logged in, please refer to "5.3.2 Procedure to start / stop operation of software PLC application" before logging out.

- (1) Double-click the slave device object ("Modbus Slave" object or "Modbus TCP Slave" object) in the device window to open the device editor.
- (2) Select the [Modbus Slave Channel] tab and click [Add Channel]

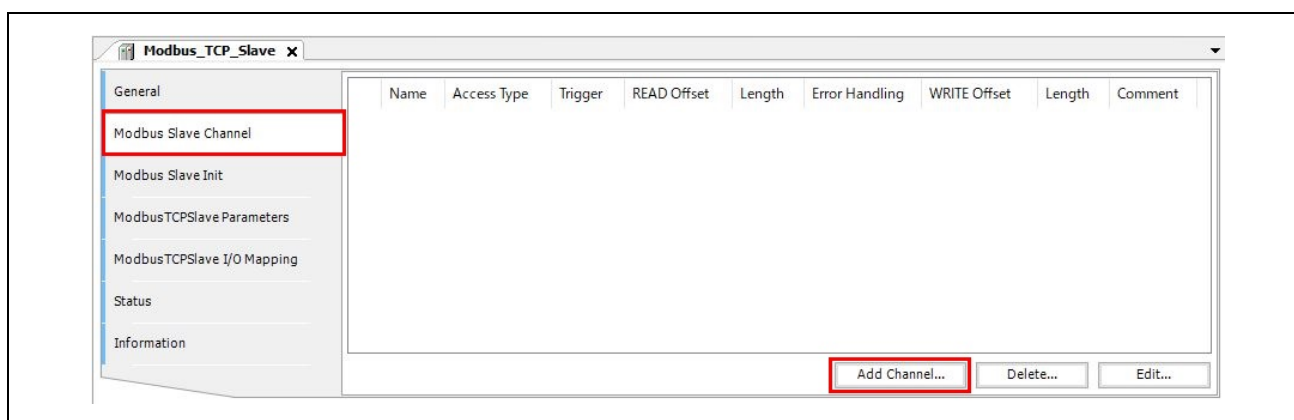


Figure 5.6: Add channel

- (3) In the Modbus Channel dialog, select the function to be executed for [Access Type] of [Channel], select [READ Register] for Read function, [Offset] and [Length], and click [OK].

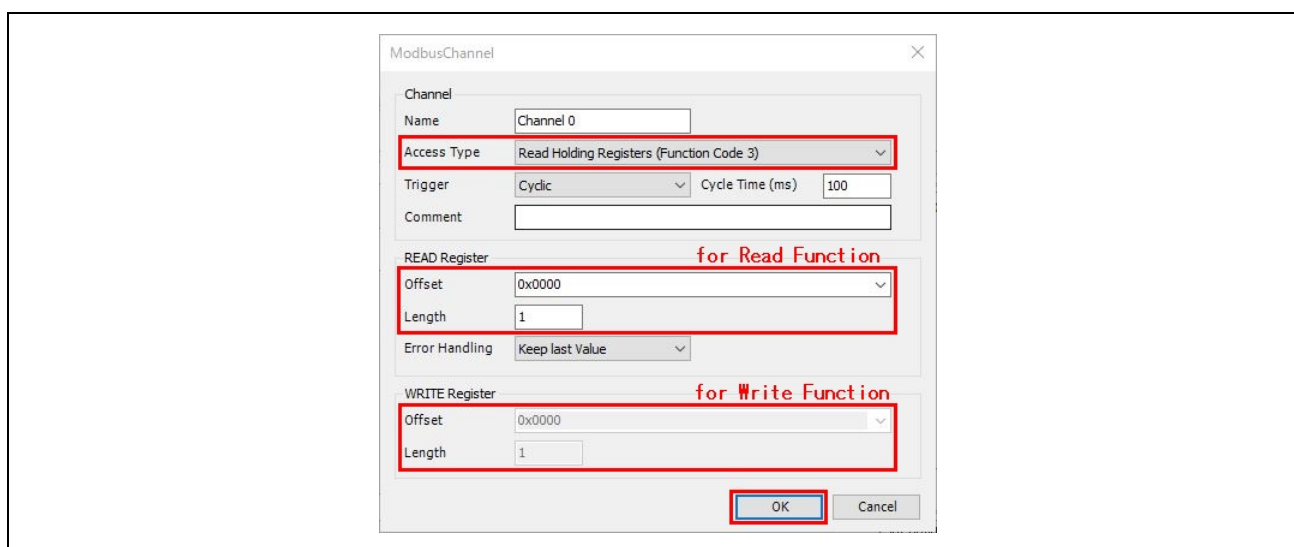


Figure 5.7: Modbus Channel dialog

(4) To execute more than one function, add the channel in the same procedure as (2), (3).

(5) The set channels are automatically I / O mapped. The I / O mapping can be confirmed on the tab below.

- "Modbus Slave" object : [Modbus Generic Serial Slave I / O Mapping] tab
- "Modbus TCP Slave" object : [Modbus TCP Slave I / O Mapping] tab

Explain display of I/O mapping.

- [Channel] column
This is the name of the channel set in (2) - (4), and it is displayed in the [Name] column of the [Modbus Slave Channel] tab.
- [Address] column
% <I / O type> <type> <BYTE position>. <BIT position>.
 - I / O type : "I" for Input (Read function), "Q" for Output (Write function)
 - Data type : "W" for WORD type, "B" for BYTE type, "X" for BIT type
 - BYTE position : mapped BYTE position (mapped through channel by I / O)
 - BIT position : mapped BIT position (mapped in the channel)
- [Type] column
Data types such as WORD / BYTE / BIT / BOOL and ARRAY [0..N] OF WORD / BYTE are displayed.

(6) Set [Always update variables] on the I / O mapping tab to [Enabled 2]. With this setting, the value of the I/O variable will be updated automatically.

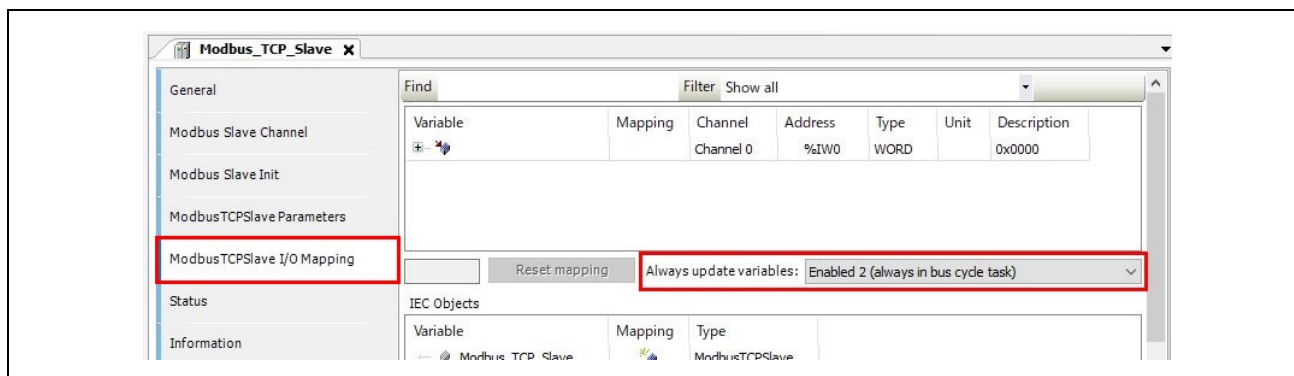


Figure 5.8: I/O variable update setting

5.3.2 Operation start / end procedure of software PLC application

Procedure to start and terminate the operation of the software PLC application is described below. The start and end operations are performed after logging in to the software PLC.

- (1) Log in to the software PLC using the menu [Online] → [Login] or the [Login] button in the figure below. At this time, a message about downloading the application may be displayed, but click [Yes].

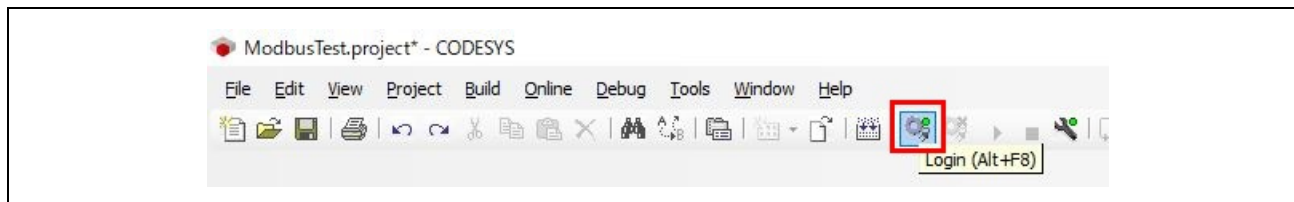


Figure 5.9: Login button to Software PLC

- (2) Start application of Software PLC with menu [Debug] → [Start], or [Start] button shown below. (It is not necessary if already started driving.)

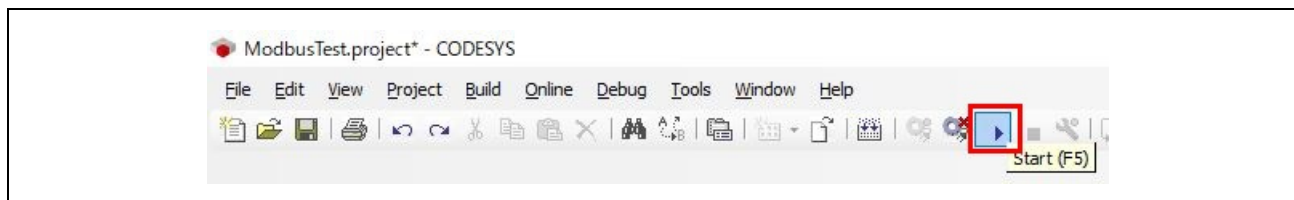


Figure 5.10: Application operation start button of Software PLC

- (3) To stop the operation of the application of the software PLC, click the menu [Debug] → [Stop], or the [Stop] button in the figure below.

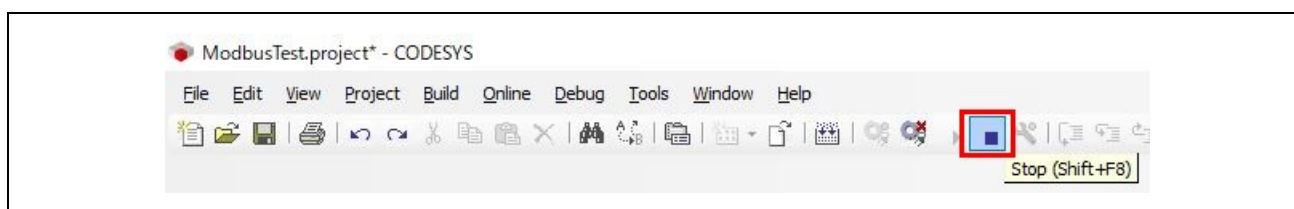


Figure 5.11: Application operation stop button of Software PLC

- (4) To log out from the PLC, click the menu [Online] → [Logout] or the [Logout] button in the figure below. It is also possible to log out while running the application without executing (3).

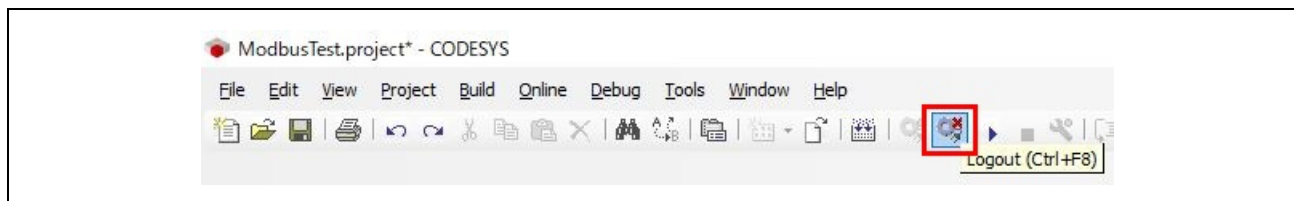


Figure 5.12: Logout button to Software PLC

5.3.3 Method of setting the read function read value

In case of the software PLC operates as a slave device, can change the read value of the Read function (code 4) for Input Registers. The setting procedure is described below. This setting is operated with software PLC logged in.

(1) Double-click the "Modbus Serial" object in the device window to open the device editor.

(2) Select the [Modbus Serial Device I/O Mapping] tab and enter the I/O corresponding to the address you want to change from the I/O indicated as "Outputs" in the [Channel] column of the I/O mapping list

Displayed I/O is the number of registers from the start address set on the [General] tab. For details on displaying I/O mapping, refer to "5.3.1 Execution function setting method and I/O mapping".

(3) Double-click the [Prepared Value] column of the corresponding line and enter the write value. For N-ary notation, "N # (N numerical value)" is assumed. In the case of Bit data, can switch "TRUE" / "FALSE" / "No setting" by clicking.

Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit	Description
		Inputs	%IW0	ARRAY [0..15] ...				Modbus Holding Registers
		Outputs	%QW0	ARRAY [0..15] ...				Modbus Input Registers
		Outputs[0]	%QW0	WORD	0			RegisterAddress:2000 (Start address)
		Outputs[1]	%QW1	WORD	0			
		Outputs[2]	%QW2	WORD	0			
		Outputs[3]	%QW3	WORD	0			
		Outputs[4]	%QW4	WORD	0			
		Outputs[5]	%QW5	WORD	0			
		Outputs[6]	%QW6	WORD	0			
		Outputs[7]	%QW7	WORD	0			
		Outputs[8]	%QW8	WORD	0			
		Outputs[9]	%QW9	WORD	0			
		Outputs[10]	%QW10	WORD	0			
		Outputs[11]	%QW11	WORD	0			
		Outputs[12]	%QW12	WORD	0			
		Outputs[13]	%QW13	WORD	0			Change the value to "0xFFFF"
		Outputs[14]	%QW14	WORD	0	16#FFFF		RegisterAddress:2014
		Outputs[15]	%QW15	WORD	0			

Figure 5.13: Read value setting example of Read function

(4) Values are written to the Software PLC in the menu [Debug] → [Write Values]. The [Prepared Value] column disappears and the value in the [Current Value] column is updated.

5.3.4 Method of setting the write function read value

In case of the software PLC operates as a slave device, can change the read value of the Read function (code 4) for Input Registers. The setting procedure is described below. it while logging in to the software PLC.

- (1) Double-click the slave device object ("Modbus Slave" object or "Modbus TCP Slave" object) in the device window to open the device editor.
- (2) Select the I/O mapping tab ([Modbus Generic Serial Slave I/O Mapping] or [Modbus TCP Slave I/O Mapping]) and specify the I/O want to change from the I/O mapped with [5.3.1 Execution function setting method and I/O mapping].
- (3) Double-click the [Prepared Value] column of the corresponding line and enter the write value. For N-ary notation, "N #" (N numerical value)" is assumed. In the case of Bit data, can switch "TRUE" / "FALSE" / "No setting" by clicking.

Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit	Description
		Channel 0	%QW0	WORD	0			0x0000
		Bit 0	%QX0.0	BOOL	FALSE	TRUE		
		Bit 1	%QX0.1	BOOL	FALSE	Change the value of Bit0 to "TRUE"		
		Bit 2	%QX0.2	BOOL	FALSE			
		Bit 3	%QX0.3	BOOL	FALSE			
		Bit 4	%QX0.4	BOOL	FALSE			
		Bit 5	%QX0.5	BOOL	FALSE			
		Bit 6	%QX0.6	BOOL	FALSE			
		Bit 7	%QX0.7	BOOL	FALSE			
		Bit 8	%QX1.0	BOOL	FALSE			
		Bit 9	%QX1.1	BOOL	FALSE			
		Bit 10	%QX1.2	BOOL	FALSE			
		Bit 11	%QX1.3	BOOL	FALSE			
		Bit 12	%QX1.4	BOOL	FALSE			
		Bit 13	%QX1.5	BOOL	FALSE			
		Bit 14	%QX1.6	BOOL	FALSE			
		Bit 15	%QX1.7	BOOL	FALSE	Change the value to "0xFFFF"		
		Channel 0	%QW1	WORD	0	16#FFFF		0x0001

Figure 5.14: Write value setting example of Write function

- (4) Values are written to the Software PLC using the menu [Debug] → [Write values]. The Prepared Value column disappears and the value in the [Current Value] column is updated.

【 Caution 】

The slave sample application returns a response when receiving a write function request message from the master but does not write to Coil or Register.

5.3.5 Execution result confirmation method

In case of the software PLC operates as a master device, the presence or absence of error response of the slave RZ/N1 and the occurrence of communication error can be known by the variable value of the device object. The confirmation procedure is described below. Please make sure the auto reconnection setting of the master device object is disabled so that the value at the time of error occurrence is not updated.

(1) Select the watch window to display from the menu [View] → [Watch] and open it.

(2) Enter the following expression in the [Expression] field of the watch window.

- Error occurrence : IoConfig_Globals.< Slave device object name>.xError
- Error code (content) : IoConfig_Globals.< Slave device object name>.byModbusError

(3) Check the value in the [Value] column. The meanings of the values are shown in the table below.



Expression	Application	Type	Value	Prepare
IoConfig_Globals.Modbus_TCP_Slave.xError	Device.Application	BOOL	TRUE	
IoConfig_Globals.Modbus_TCP_Slave.byModbusError	Device.Application	MB_ERRORCODES	ILLEGAL_DATA_ADD...	

Figure 5.15: Watch example screen of response error code

Table 5.9 Error occurrence status of slave device object Contents of variable

Value (BOOL type)		State
FALSE	(0)	No error
TRUE	(1)	Error

Table 5.10 Content of error code variable of slave device object

Value (ENUM MB_ErrorCodesType)		State	
RESPONSE_SUCCESS	(0x00)	Normal	
ILLEGAL_FUNCTION	(0x01)	Error response	
ILLEGAL_DATA_ADDRESS	(0x02)	(Conforms to exception codes of Modbus specification)	
ILLEGAL_DATA_VALUE	(0x03)		
SLAVE_DEVICE_FAILURE	(0x04)		
ACKNOWLEDGE	(0x05)		
SLAVE_DEVICE_BUSY	(0x06)		
MEMORY_PARITY_ERROR	(0x08)		
GATEWAY_PATH_UNAVAILABLE	(0x0A)		
GATEWAY_DEVICE_FAILED_TO_RESPOND	(0x0B)		
RESPONSE_TIMEOUT	(0xA1)	Communication	No response within time
RESPONSE_CRC_FAIL	(0xA2)	error	Checksum is invalid
RESPONSE_WRONG_SLAVE	(0xA3)		Response from illegal slave
RESPONSE_WRONG_FUNCTIONCODE	(0xA4)		Invalid function code
TCP_COMMUNICATION_ERROR	(0xA5)		Unable to send due to TCP port error
RESPONSE_INVALID_DATA	(0xA6)		Data corruption
RESPONSE_INVALID_PROTOCOL	(0xA7)		Protocol incorrect
RESPONSE_INVALID_HEADER	(0xA8)		Invalid header
UNDEFINED	(0xFF)	Initial state etc.	Undefined state

5.3.6 Method of check the write function write result

In case of software PLC operates as a slave device, you can check the register value after writing by the Write function (codes 6, 16, 23) to Holding Registers in the following way.

- (1) Double-click the "Modbus Serial" object in the device window to open the device editor.
- (2) Select the [Modbus Serial Device I/O Mapping] tab and enter the I/O corresponding to the write target address from the I/O indicated as "Input" in the [Channel] column of the I/O mapping.
The displayed I/O is the number of registers from the start address set on the [General] tab. For details on displaying I/O mapping, refer to [5.3.1 Execution function setting method and I/O mapping].
- (3) Register values are automatically displayed in the [Current Value] column. Check the value of the corresponding line.

Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit	Description
		Inputs	%IW0	ARRAY [0..15]...				Modbus Holding Registers
		Inputs[0]	%IW0	WORD	0			
		Inputs[1]	%IW1	WORD	0			
		Inputs[2]	%IW2	WORD	0			
		Inputs[3]	%IW3	WORD	0			
		Inputs[4]	%IW4	WORD	0			
		Inputs[5]	%IW5	WORD	0			
		Inputs[6]	%IW6	WORD	0			
		Inputs[7]	%IW7	WORD	0			
		Inputs[8]	%IW8	WORD	0			
		Inputs[9]	%IW9	WORD	0			
		Inputs[10]	%IW10	WORD	0			
		Inputs[11]	%IW11	WORD	0			
		Inputs[12]	%IW12	WORD	0			
		Inputs[13]	%IW13	WORD	0			
		Inputs[14]	%IW14	WORD	0			
		Inputs[15]	%IW15	WORD	0			
		Outputs	%QW0	ARRAY [0..15]...				Modbus Input Registers

Figure 5.16: Example of checking the read value of the Read function

5.3.7 Method of check the read function result

In case of the software PLC operates as the master device, you can check the read value returned by the Read function response message in the following way.

- (1) Double-click the slave device object ("Modbus Slave" object or "Modbus TCP Slave" object) in the device window to open the device editor.
- (2) Select the I/O mapping tab ([Modbus Generic Serial Slave I/O Mapping] or [Modbus TCP Slave I/O Mapping]) and specify the I/O want to change from the I/O mapped with [5.3.1 Execution function setting method and I/O mapping].
- (3) The read value is automatically displayed in the [Current Value] column. Check the value of the corresponding line.

Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit	Description
		Channel 0	%IW0	WORD	1			0x0000
		Bit 0	%IX0.0	BOOL	TRUE			
		Bit 1	%IX0.1	BOOL	FALSE			
		Bit 2	%IX0.2	BOOL	FALSE			
		Bit 3	%IX0.3	BOOL	FALSE			
		Bit 4	%IX0.4	BOOL	FALSE			
		Bit 5	%IX0.5	BOOL	FALSE			
		Bit 6	%IX0.6	BOOL	FALSE			
		Bit 7	%IX0.7	BOOL	FALSE			
		Bit 8	%IX1.0	BOOL	FALSE			
		Bit 9	%IX1.1	BOOL	FALSE			
		Bit 10	%IX1.2	BOOL	FALSE			
		Bit 11	%IX1.3	BOOL	FALSE			
		Bit 12	%IX1.4	BOOL	FALSE			
		Bit 13	%IX1.5	BOOL	FALSE			
		Bit 14	%IX1.6	BOOL	FALSE			
		Bit 15	%IX1.7	BOOL	FALSE			

Figure 5.17: Example of checking the read value of the Read function

【 Caution 】

Slave sample application does not read Coil or Register when receiving a read function request message from the master. Instead, it responds by reading the data in the table below.

Table 5.11 Read data of sample application

Function code	Read data
1 Read Coils	- Modbus TCP server, Modbus Serial slave - Regardless of the address, a value incremented from 1 in 1-byte order
2 Read Discrete Inputs	- Modbus TCP-Serial gateway - Regardless of the address, the value decremented in order from 0xFF for each byte
3 Read Holding Registers	- Modbus TCP server, Modbus Serial slave - Regardless of the address, the value incremented sequentially from 1 in each word
4 Read Input Register	- Modbus TCP-Serial gateway - Regardless of the address, the value decremented sequentially from 0x00FF for each word
23 Read/Write Multiple Registers	- Modbus TCP-Serial gateway - Regardless of the address, the value decremented sequentially from 0x00FF for each word

6. Defined value of sample application

Setting of the sample application is determined by the following definition.

■ Definition of RZ/N1 board

In the preprocessor setting of the EWARM project, define the board type.

<Setting method>

(1) Select EWARM menu [Project] → [Options]

(2) Category of Option dialog box [C / C ++ Compiler] → [Preprocessor] tab → Set to [Defined symbols]

Table 6.1 Define symbol of board definition

Setting		Define symbol
Renesas RZ/N1 Platform		RENESAS_CFG_PLAT_RZN=1
CPU Board	RZ/N1D-DB	RZN1D=1
	RZ/N1S-DB	RZN1S=1
	RZ/N1L-DB	RZN1L=1
Expansion board (RZ/N1-EB)		RENESAS_CFG_BOARD_RZN1EB=1

■ Modbus stack definition

In case of operating the Modbus stack, define the symbols in the table below in the preprocessor settings of the EWARM project. And, the Modbus stack uses the TCP/IP stack and Ethernet communication, the relationship definition is also required.

< Setting method >

(1) Select EWARM menu [Project] → [Options]

(2) Category of Option dialog box [C / C ++ Compiler] → [Pre-processor] tab → Set to [Defined symbols]

Table 6.2 Define symbol of Modbus stack definition

Setting	Define symbol
Modbus Stack	GOAL_CONFIG_MODBUS_STACK=1
TCP/IP Stack	GOAL_CONFIG_TCPIP_STACK=1
	GOAL_CONFIG_TCPIP_TCP=1
	GOAL_FEAT_LWIP_INIT=1
Ethernet communication	GOAL_CONFIG_ETHERNET=1
	GOAL_CONFIG_PHY_DETECTION=1
	GOAL_CONFIG_PHY_MC_KSZ8041=1
	GOAL_CONFIG_PHY_MARVELL_88E1512=1 (using expansion board)
RS486 communication	GOAL_CONFIG_MB_UART_HALF_DUPLEX=1

■ Modbus's Ethernet definition

In the sample application of Modbus TCP Server or Modbus TCP-Serial gateway, can change the Ethernet settings according to the macro definition value in the source code.

<Definition file>

- Modbus TCP Server :

goal\appl\goal_mbs\01_tcp_server\ goal_mbs_tcp_server_demo.c

- Modbus TCP-Serial Gateway :

goal\appl\goal_mbs\04_tcp_ser_gateway\ goal_mbs_tcp_ser_gw_demo.c

Table 6.3 Macro definition of Modbus Ethernet configuration

Macro	Definition contents	Setting value	Setting Example
MBS_TCP_IP_ADDR	IP address	GOAL_NET_IPV4	<Case of 10.1.28.1>
MBS_TCP_SUBNET_MASK	sub-net mask	Use function	GOAL_NET_IPV4(10, 1, 28, 1)
MBS_TCP_GATEWAY	default gateway		
MBS_ADDITIONAL_TCP_PORTS	Number of additional ports (Added to 502)	number	<Case of Port 1024> 1024

7. Limitations

- In case of operating as a Modbus TCP server or a Modbus TCP-Serial gateway, the Modbus stack supports connection with 8 or less client units.
- Sample application of the Modbus TCP server uses an error response in response to a read/write request to a specific address.
 - In case of Modbus TCP server and Modbus TCP-Serial gateway
 - If the start address is 0x5001 to 0x5FFF, respond with SLAVE DEVICE FAILURE (Exception code = 0x04).
 - If the start address is 0x8000, responds with ILLEGAL DATA ADDRESS (Exception code = 0x02).
 - If the end address is 0xFFFF or more, responds with ILLEGAL DATA ADDRESS (Exception code = 0x02).
 - In case of Modbus serial slave
 - If the start address is 0x5001 to 0x5FFF, respond with SLAVE DEVICE FAILURE (Exception code = 0x04).
 - If the start address is 0xF000 or more, responds with ILLEGAL DATA ADDRESS (Exception code = 0x02).

Read/Write Multiple Registers with function code 23, if either one of Read or Write matches, an error response will result.

In address setting with CODESYS, the start address is "[Offset]" and the end address is "[Offset] + [Length]-1" in step (3) of "5.3.1 Execution function setting method and I/O mapping" .

8. Website and Support

Renesas Electronics Website

<http://www.renesas.com/>

Inquiries

<http://www.renesas.com/contact/>

All trademarks and registered trademarks are the property of their respective owners.

Revision History

Rev.	Date	Description	
		Page	Summary
0.90	Jun 15, 2018	-	First edition issued
1.00	Oct 31, 2018	7	Add serial port connection description
1.10	Nov 30, 2018	7-11	Support added for RZ / N1-EB evaluation board
		45	Added support for multi-client connection of Modbus TCP server
1.20	Dec 28, 2018	3	Added explanation of two cores
		10	Added sample application description
			Added Table of workspace files for Core To Core variant
		12-15	Added execution procedure of sample application
		38	Changed menu description

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

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

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

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

3. Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

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

- 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.
- IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers Inc
- MODBUS® is a registered trademark of the MODBUS-IDA Organization.
- CODESYS is a registered trademark of 3S-Smart Software Solutions GmbH, Germany.
- Additionally all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)



SALES OFFICES

Renesas Electronics Corporation

<http://www.renesas.com>

Refer to "<http://www.renesas.com/>" for the latest and detailed information.

Renesas Electronics America Inc.

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

Renesas Electronics Canada Limited

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

Renesas Electronics Europe Limited

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH

Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.

Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.

Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited

Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.

13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.

80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.

Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.

No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.

17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338