# RX62N Group/RX63N Group

## Sample Program Using Embedded TCP/IP M3S-T4-Tiny

### Introduction

This document explains TCP/IP and driver that operated on RX62N and RX63N. This documents name is "Introduction Guide".

The TCP/IP protocol stack for embedded system is Tiny TCP/IP library “RX Family TCP/IP for Embedded system M3S-T4-Tiny” (which called The T4). T4 is provided as library format and user can develop own system with this library to use TCP/IP function.

The driver program for using T4 include cmt driver, system timer, ethernet driver, t4 driver and common driver.

We prepared sample programs for each CPU board included in the Renesas Starter Kit+. This sample program shows how to setup CPU board, PC settings, network connections to confirm correct sample program behavior.

Please refer to the following URL to know the latest information about T4.

https://www.renesas.com/mw/t4

echo server sample:R20AN0051

T4 is provided as Firmware Integration Technology (FIT) Module. Please refer to the URL to know FIT outline.

https://www.renesas.com/us/en/products/software-tools/software-os-middleware-driver/softwarepackage/fit.html

### Target Device

RX62N Group

RX63N Group

### Target Compilers

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

For details of the confirmed operation contents of each compiler, refer to "2 Operation Confirmation Conditions".

**Contents**

# 1. Overview

## 1.1 Overview

The sample program provides TCP/IP protocol, device driver (common driver, CMT driver, Ethernet driver, System timer and T4 driver) and sample for RX62N group, RX63N group.

## 2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

**Table 2-1 Operation Confirmation Conditions for RX62N**

| Item | Contents |
|---|---|
| MCU Used | R5F562N8BDBG (RX62N Group) |
| Operating frequencies | - Main clock: 12 MHz<br>- PLL: 12 MHz (main clock)<br>- System clock (ICLK): 96 MHz (Main clock multiply by 8)<br>- Peripheral module clock (PCLK): 48 MHz (Main clock multiply by 4)<br>- External bus clock (BCLK): 24 MHz (Main clock multiply by 2) |
| Operating voltage | 3.3V |
| Integrated development environment | Renesas Electronics e$^2$ studio Version 7.6.0<br>IAR Embedded Workbench for Renesas RX 4.12.1 |
| C compiler | Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00<br>Compiler option: The following option is added to the default settings of the integrated development environment.<br>-lang = c99 |
| | GCC for Renesas RX 4.8.4.201902<br>Compiler option: The following option is added to the default settings of the integrated development environment.<br>-std=gnu99 |
| | IAR C/C++ Compiler for Renesas RX version 4.12.1<br>Compiler option: The default settings of the integrated development environment. |
| iodefine.h version | Version 1.4 |
| Endian | little endian/big endian |
| Operating mode | Single-chip mode |
| Processer mode | Supervisor mode |
| Sample program version | Version 1.00 |
| Board used | Renesas Starter Kit+ for RX62N (R0K5562Nxxxxxxx) |

**Table 2-2 Operation Confirmation Conditions for RX63N**

| Item | Contents |
|---|---|
| MCU Used | R5F563NFDDFC (RX63N Group) |
| Operating frequencies | - Main clock: 12 MHz<br>- PLL: 192 MHz (main clock divided by 1 and multiplied by 16)<br>- System clock (ICLK): 96 MHz (PLL divided by 2)<br>- Peripheral module clock A (PCLKA): 96 MHz (PLL divided by 2)<br>- Peripheral module clock B (PCLKB): 48 MHz (PLL divided by 4)<br>- External bus clock (BCLK): 48 MHz (PLL divided by 4)<br>- FlashIF clock (FCLK): 48 MHz (PLL divided by 4)<br>- IEBUS clock (IECLK): 48 MHz (PLL divided by 4) |
| Operating voltage | 3.3V |
| Integrated development environment | Renesas Electronics e$^2$ studio Version 7.6.0<br>IAR Embedded Workbench for Renesas RX 4.12.1 |
| C compiler | Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00<br>Compiler option: The following option is added to the default settings of the integrated development environment.<br>-lang = c99 |
| | GCC for Renesas RX 4.8.4.201902<br>Compiler option: The following option is added to the default settings of the integrated development environment.<br>-std=gnu99 |
| | IAR C/C++ Compiler for Renesas RX version 4.12.1<br>Compiler option: The default settings of the integrated development environment. |
| iodefine.h version | Version 1.8A |
| Endian | Big endian/little endian |
| Operating mode | Single-chip mode |
| Processer mode | Supervisor mode |
| Sample program version | Version 1.00 |
| Board used | Renesas Starter Kit+ for RX63N (R0K50563Nxxxxxx) |

## 3. Related Documents

RX Family TCP/IP for Embedded system M3S-T4-Tiny Introduction Guide Firmware Integration Technology (R20AN0051)

# 4. Hardware Configuration

## 4.1 Pin setting example for using RSK+RX62N/RSK+RX63N

Table 4-1 shows pin setting example for using RSK+RX62N/RSK+RX63N. Figure 4-1 and Figure 4-2 show examples of connection to the PHY-LSI.

**Table 4-1 Pin setting example for using RSK+RX62N/RSK+RX63N**

| Case of Using MII Mode | Case of Using RMII Mode*3 | I/O Port |
|---|---|---|
| ET_TX_CLK | | PC4 |
| ET_RX_CLK | REF50CK | P76 |
| ET_TX_EN | RMII_TXD_EN | P80 |
| ET_ETXD3 | | PC6 |
| ET_ETXD2 | | PC5 |
| ET_ETXD1 | RMII_TXD1 | P82 |
| ET_ETXD0 | RMII_TXD0 | P81 |
| ET_TX_ER | | PC3 |
| ET_RX_DV | | PC2 |
| ET_ERXD3 | | PC0 |
| ET_ERXD2 | | PC1 |
| ET_ERXD1 | RMII_RXD1 | P74 |
| ET_ERXD0 | RMII_RXD0 | P75 |
| ET_RX_ER | RMII_RX_ER | P77 |
| ET_CRS | RMII_CRS_DV | P83 |
| ET_COL | | PC7 |
| ET_MDC | | P72 |
| ET_MDIO | | P71 |
| ET_LINKSTA | | P54 *1 |
| ET_EXOUT | | - *2 |
| ET_WOL | | - *2 |

Notes: 1. Setting is not required if the setting of #define ETHER_CFG_USE_LINKSTA is 0.

Notes: 2. Setting is not required because these pin are not used in Ethernet driver module.

Notes: 3. RMII mode can only be used with RSK + RX63N.

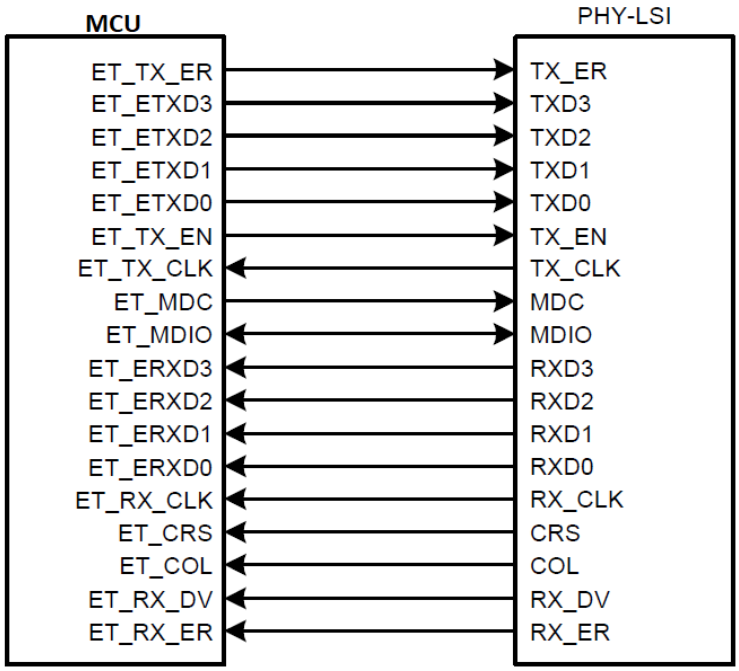



**Figure 4-1 Example of connection between RX62N, RX63N and PHY(MII)**

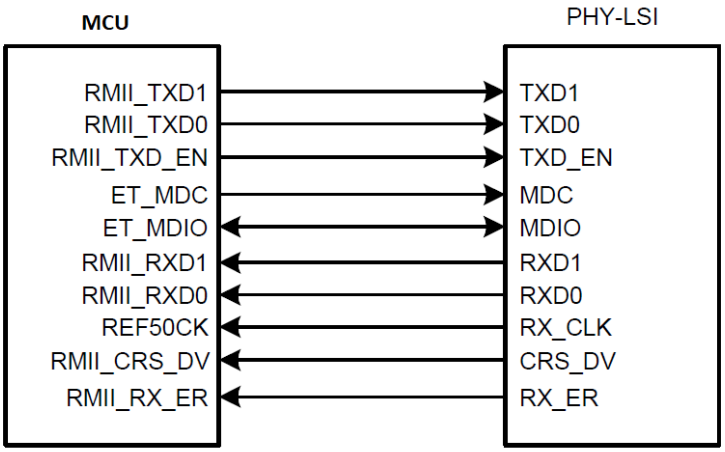



**Figure 4-2 Example of connection between RX62N, RX63N and PHY(RMII)**

# 5. Software

## 5.1 Sample Program Structure

This sample program includes following files.

**Table 5-1 Sample Program Structure**

| File/Directory name | detail |
|---|---|
| r01an5161xx0100-rx62nrx63n-t4-connectivity | |
| workspace | |
| r01an5161_rx62n_t4_connectivity_ccrx | CCRX sample project for RX62N |
| r01an5161_rx63n_t4_connectivity_ccrx | CCRX sample project for RX63N |
| r01an5161_rx62n_t4_connectivity_gcc | GCC sample project for RX62N |
| r01an5161_rx63n_t4_connectivity_gcc | GCC sample project for RX63N |
| r01an5161_rx62n_t4_connectivity_iar | IAR sample project for RX62N |
| r01an5161_rx63n_t4_connectivity_iar | IAR sample project for RX63N |
| r01an5161jj0100-rx62nrx63n-t4-connectivity.pdf | Introduction Guide(Japanese) |
| r01an5161ej0100-rx62nrx63n-t4-connectivity.pdf | Introduction Guide(English this document) |
| readme_j.txt | readme file(Japanese) |
| readme_e.txt | readme file(English) |

For example, the CCRX sample project for RX63N will contain the files listed in table Table 5-2 below.

**Table 5-2 Sample project for RX63N Structure**

| File/Directory name | detail |
|---|---|
| r01an5161_rx63n_t4_connectivity_ccrx | |
| doc | Document |
| r01an5161jj0100-rx62nrx63n-t4-connectivity.pdf | Introduction Guide(Japanese) |
| r01an5161ej0100-rx62nrx63n-t4-connectivity.pdf | Introduction Guide(English this document) |
| generate | MCU reset program |
| r01an5161_src | Sample program source |
| main.c | main function |
| echo_srv_sample.h | echo server sample program header file |
| r_pincfg | Pin setting |
| mcu_initial | Clock setting, non-existent port setting, stop peripheral functions that operate after reset |
| r_t4_rx | T4 module |
| tcp_blocking_sample | TCP Blocking Call |
| echo_srv.c | TCP blocking server echo |
| tcp_nonblocking_cancel_sample | TCP Non-blocking cancel Call |
| echo_srv.c | TCP non-blocking cancel server echo |
| tcp_nonblocking_sample | TCP Non-blocking Call |
| echo_srv.c | TCP non-blocking server echo |
| udp_blocking_sample | UDP Blocking Call |
| echo_srv.c | UDP blocking server echo |
| udp_nonblocking_sample | UDP Non-blocking Call |
| echo_srv.c | UDP non-blocking server echo |
| driver | Driver folder |
| cmt_driver | CMT driver module |
| common_driver | Common driver module |
| ether_driver | Ether driver module |
| systime_driver | System timer module |
| t4_driver | T4 driver module |
| r_driver_rx_config.h | The driver configuration file |
| HardwareDebug | Debug folder |
| Release | Release folder |
| .settings | Project settings folder |
| .cprojcet | Cproject file |
| .project | Project file |
| r01an5161_rx63n_t4_connectivity.x.launch | Launch file |
| r01an5161_rx63n_t4_connectivity.rcpc | RCPC file |

The folder to which the contents of r_t4_rx is extracted will contain the files listed in table Table 5-3 below.

**Table 5-3 Structure of the T4 FIT Modules**

| File/Directory name | detail |
|---|---|
| T4 FIT Module body (r_t4_rx) | |
| T4 Library (lib) | |
| cc-rx | Renesas CC-RX |
| T4_Library_ether_ccrx_rxv1_big.lib | T4 Library (RXV1 core, Big endian, for Ethernet) |
| T4_Library_ether_ccrx_rxv1_ether_little.lib | T4 Library (RXV1 core, Little endian, for Ethernet) |
| T4_Library_ether_ccrx_rxv1_ether_big_debug.lib | T4 Library includes debug information. (RXV1 core, Big endian, for Ethernet/for QE for TCP/IP) |
| T4_Library_ether_ccrx_rxv1_ether_little_debug.lib | T4 Library includes debug information. (RXV1 core, Little endian, for Ethernet/for QE for TCP/IP) |
| GCC | |
| libT4_Library_ether_gcc_rxv1_big.a | T4 Library (RXV1 core, Big endian, for Ethernet) |
| libT4_Library_ether_gcc_rxv1_little.a | T4 Library (RXV1 core, Little endian, for Ethernet) |
| IAR | |
| T4_Library_ether_iar_rxv1_big.a | T4 Library (RXV1 core, Big endian, for Ethernet) |
| T4_Library_ether_iar_rxv1_little.a | T4 Library (RXV1 core, Little endian, for Ethernet) |
| r_t4_itcpip.h | T4 Library header file |
| r_stdint.h | Standard data type header file |
| r_mw_version.h | Middleware version header file |
| T4 Document (doc) | |
| ja | |
| r20uw0031jj0111-t4tiny.pdf | User's Manual (Japanese) |
| r20uw0032jj0108-t4tiny.pdf | Ethernet Driver Interface Specification (Japanese) |
| r20an0051jj0209-rx-t4.pdf | T4 Introduction Guide (Japanese) |
| en | |
| r20uw0031ej0111-t4tiny.pdf | User's Manual (English) |
| r20uw0032ej0108-t4tiny.pdf | Ethernet Driver Interface Specification (English) |
| r20an0051ej0209-rx-t4.pdf | T4 Introduction Guide (English) |
| T4 Library make environment (make_lib) | |
| make_lib.zip | T4 Library make environment (includes source code) |
| T4 config reference (ref) | |
| config_tcpudp_reference.tpl | T4 Config file (template) |
| r_t4_rx_config_reference.h | T4 Config header(reference) |
| src | |
| config_tcpudp.c | T4 Config file |
| readme.txt | readme |

The folder to which the contents of driver is extracted will contain the files listed in table Table 5-4 below.

**Table 5-4 Structure of the Driver Modules**

| File/Directory name | detail |
|---|---|
| driver | |
| cmt_driver | |
| src | |
| cmt_driver.c | CMT driver source file |
| cmt_if.h | CMT driver interface file |
| common_driver | |
| src | |
| drv_locking.c | Locking source file |
| r_rx_compiler.h | File that unifies the function definition of each compiler |
| r_rx_inrtinsic_funtions.c | A file that defines built-in functions that are not implemented by the GCC / IAR compiler |
| r_rx_inrtinsic_funtions.h | File that unifies the definition of intrinsic functions of each compiler |
| common_driver.h | Common driver interface file |
| ether_driver | |
| src | |
| phy | |
| phy.c | Ethernet PHY device driver |
| phy.h | Ethernet PHY device driver interface file |
| targets | |
| rx63n | |
| ether_setting_rx63n.c | Interrupt and PHY management interface setting file |
| ether_driver.c | Ethernet driver source file |
| ether_private.h | Ethernet internal macro, structure and function define |
| ether_if.h | Ethernet driver interface |
| systime_driver | |
| src | |
| sys_time_driver.c | System timer source file |
| sys_time_private.h | System timer internal macro, structure and function define |
| sys_time_if.h | System timer interface |
| t4_driver | |
| src | |
| ether_callback.c | Callback function for Ethernet source file |
| t4_driver.c | t4 driver source file |
| timer.c | timer source file |
| timer.h | timer interface file |
| r_driver_rx_config.h | driver configuration file |

## 5.2 Sample source

Each project files including source file below.

- TCP blocking call (tcp_blocking directory)
- TCP Non-blocking cancel call (tcp_nonblocking_cancel directory)
- TCP Non-blocking call (tcp_nonblocking directory)
- UDP blocking call (udp_blocking directory)
- UDP Non-blocking call (udp_nonblocking directory)

The sample program has a common main function. The main function calls an echo_srv() function. Five above patterns are implementations of the echo back server and please use one pattern of project.

### 5.2.1 Flow of the TCP Echo Back Server Function (for Blocking Call)

Flow of the Echo Back Server Function

Cf. Figure 5.2 Flow of the TCP Echo Back Server Function (for Blocking Call).

### 5.2.2 Flow of the TCP Echo Back Server Function (for Non-blocking Call)

Flow of the Echo Back Server Function

Cf. Figure 5.3 Flow of the TCP Echo Back Server Function (for Non-Blocking Call).

Flow of the Callback Function

Cf. Figure 5.4 Flow of the TCP Callback Function (for Non-Blocking Call).

### 5.2.3 Flow of the UDP Echo Back Server Function (for Blocking Call)

Flow of the Echo Back Server Function

Cf. Figure 5.5 Flow of the UDP Echo Back Server Function (for Blocking Call).

### 5.2.4 Flow of the UDP Echo Back Server Function (for Non-blocking Call)

Flow of the Echo Back Server Function

Cf. Figure 5.6 Flow of the UDP Echo Back Server Function (for Non-Blocking Call).

Flow of the Callback Function

Cf. Figure 5.7 Flow of the UDP Callback Function (for Non-Blocking Call)

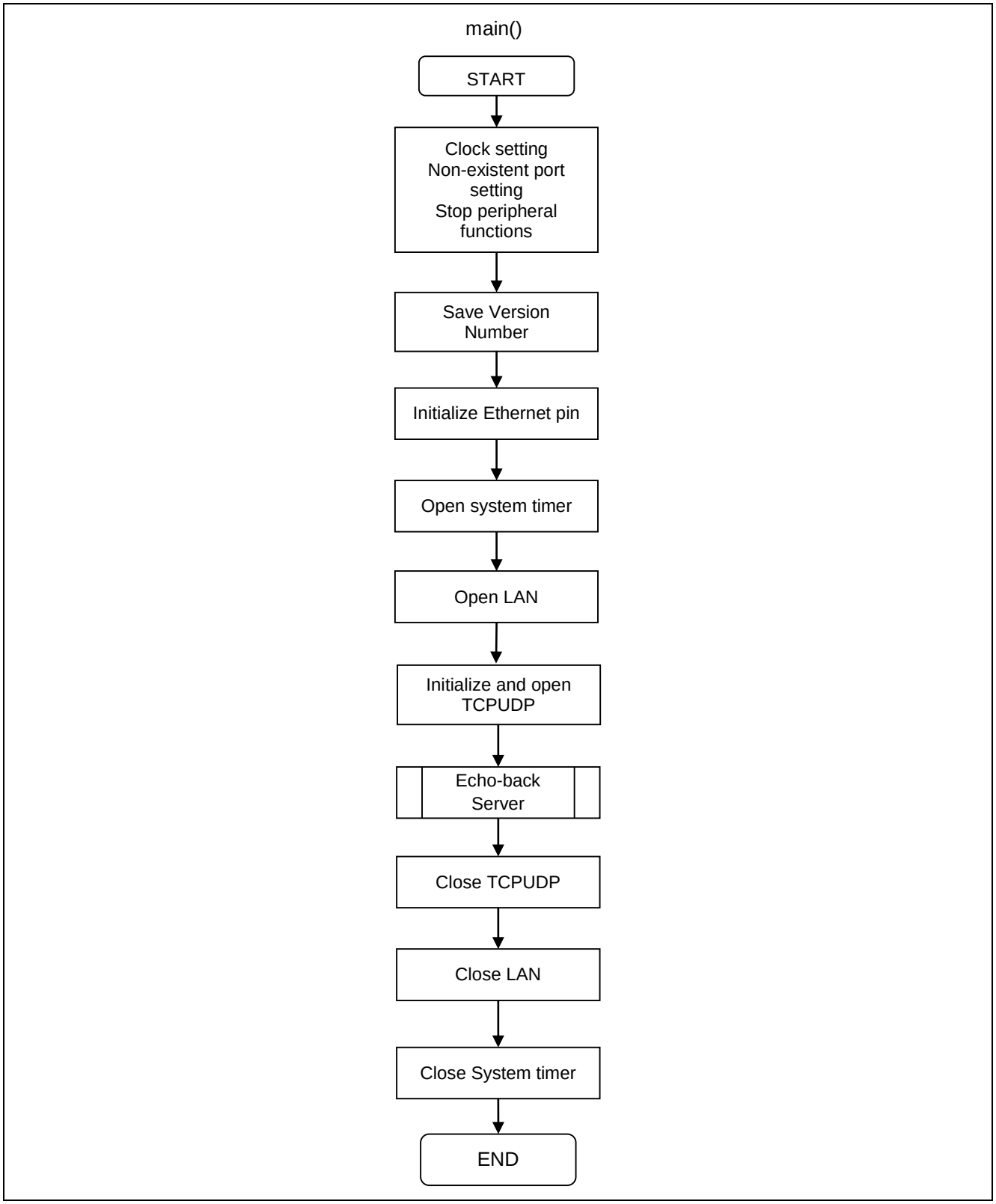



**Figure 5.1 Flow of the Main Function**

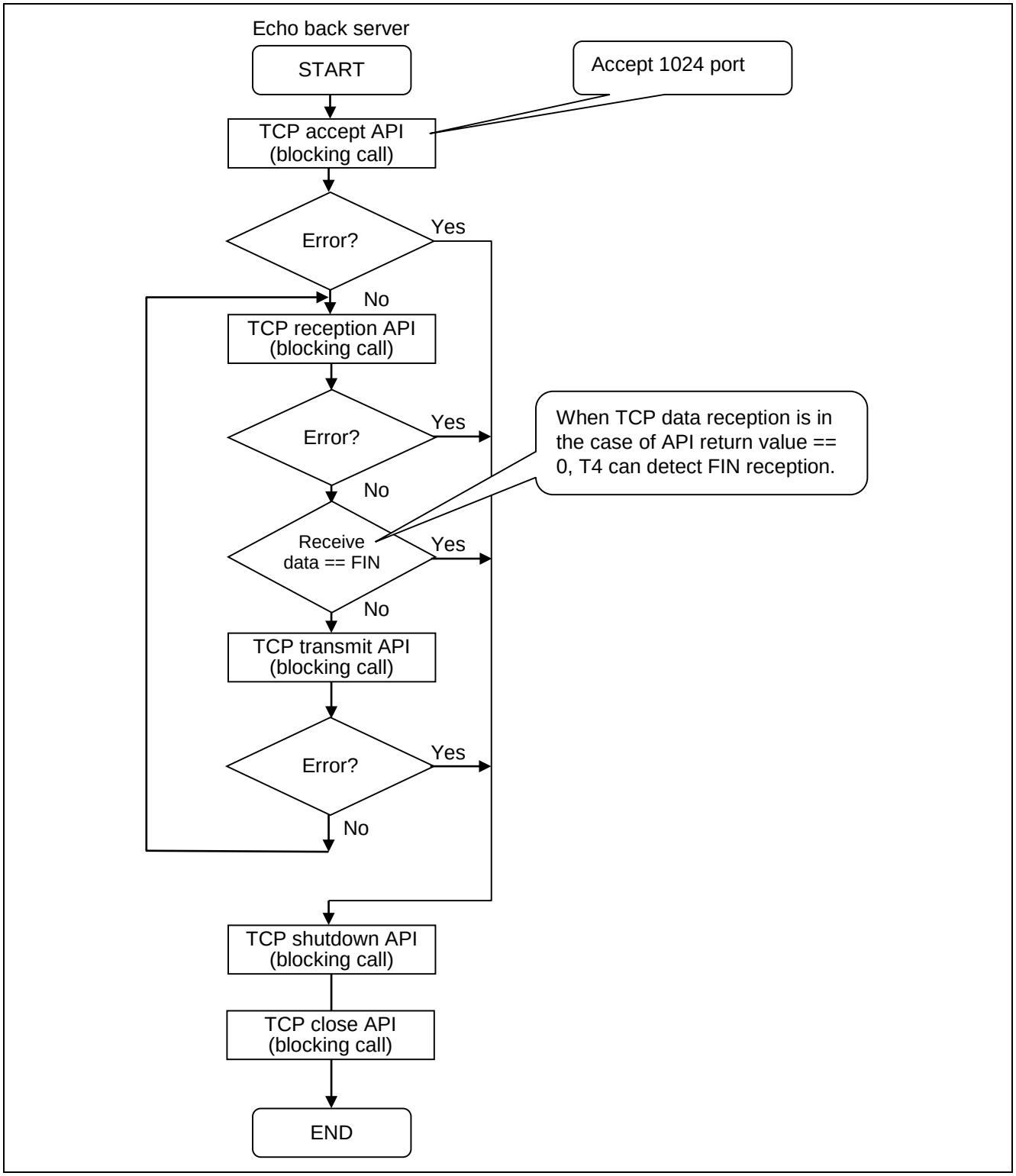



**Figure 5.2 Flow of the TCP Echo Back Server Function (for Blocking Call)**

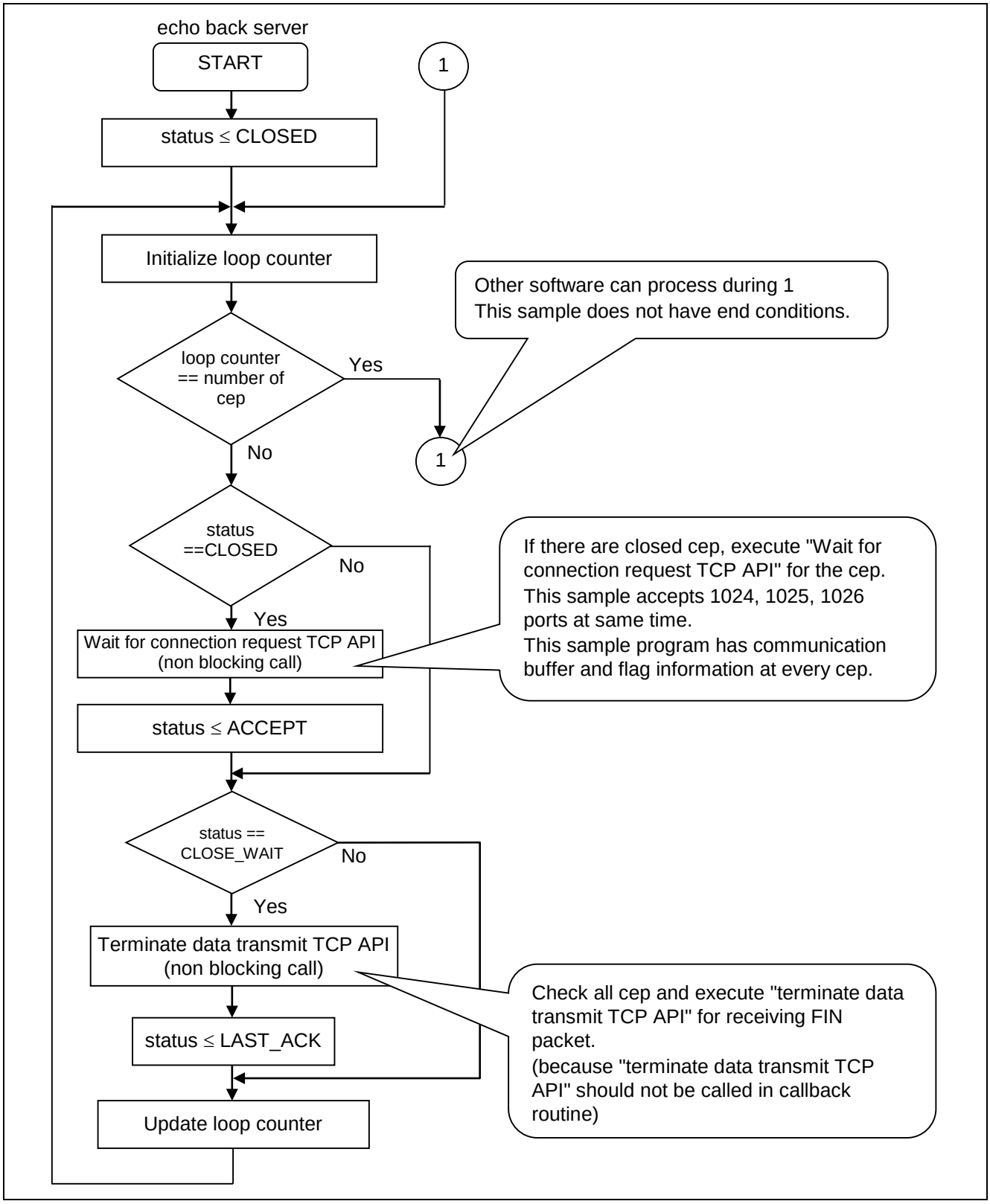



**Figure 5.3 Flow of the TCP Echo Back Server Function (for Non-Blocking Call)**

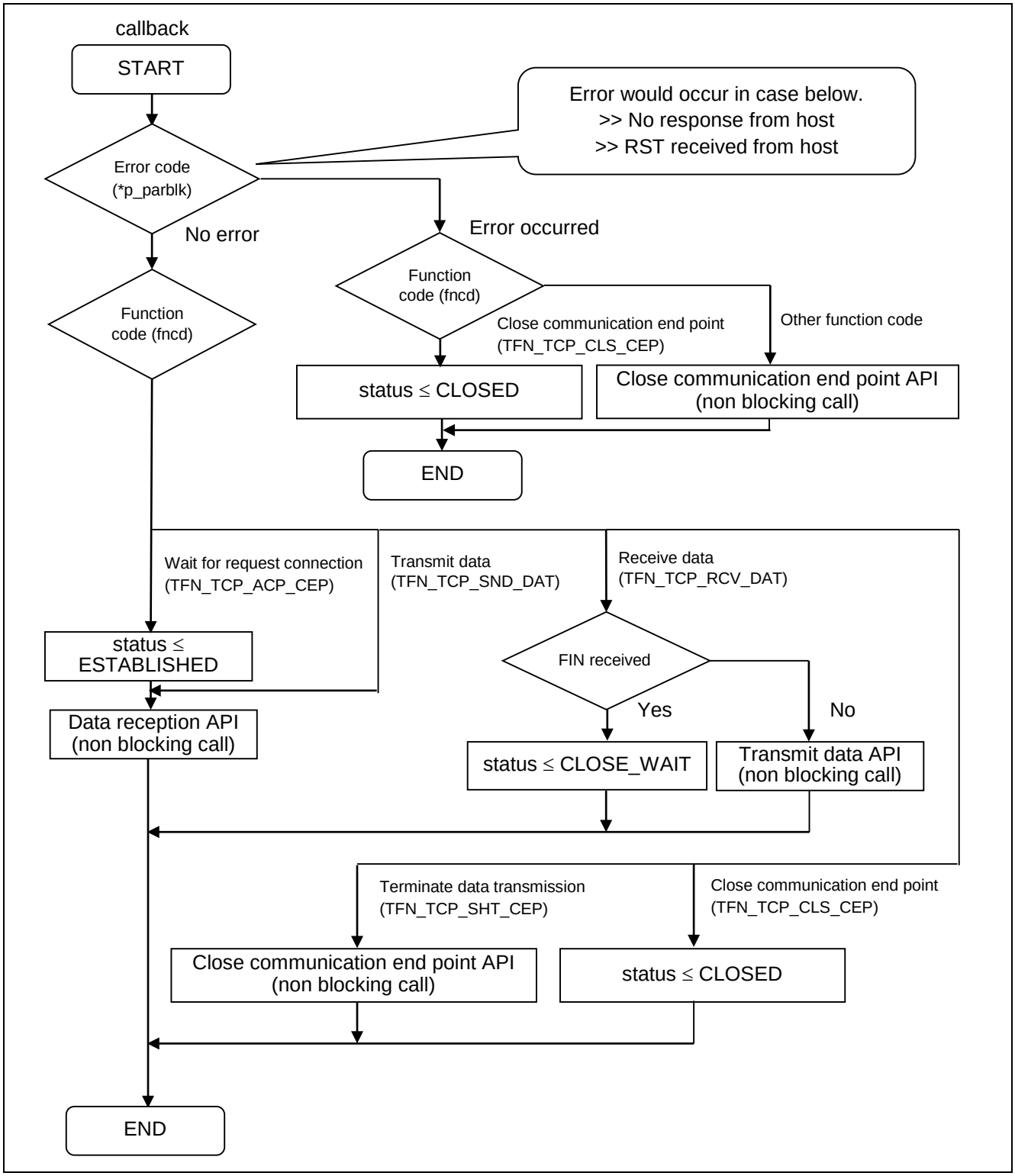



**Figure 5.4 Flow of the TCP Callback Function (for Non-Blocking Call)**

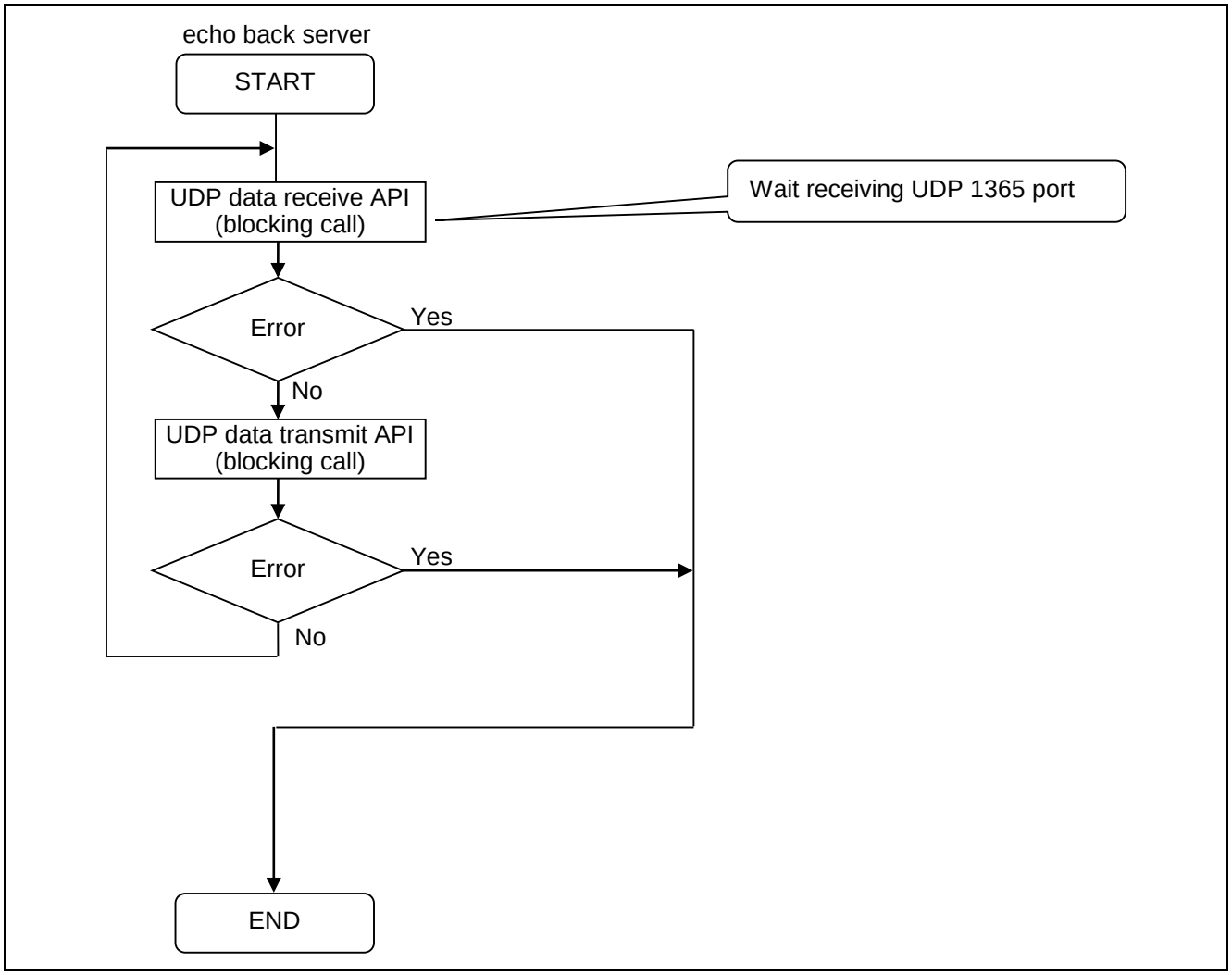



**Figure 5.5 Flow of the UDP Echo Back Server Function (for Blocking Call)**

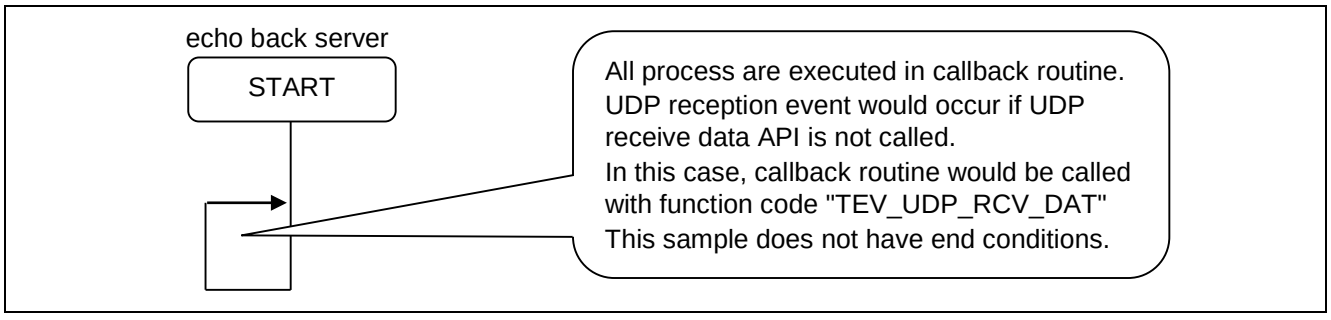



**Figure 5.6 Flow of the UDP Echo Back Server Function (for Non-Blocking Call)**

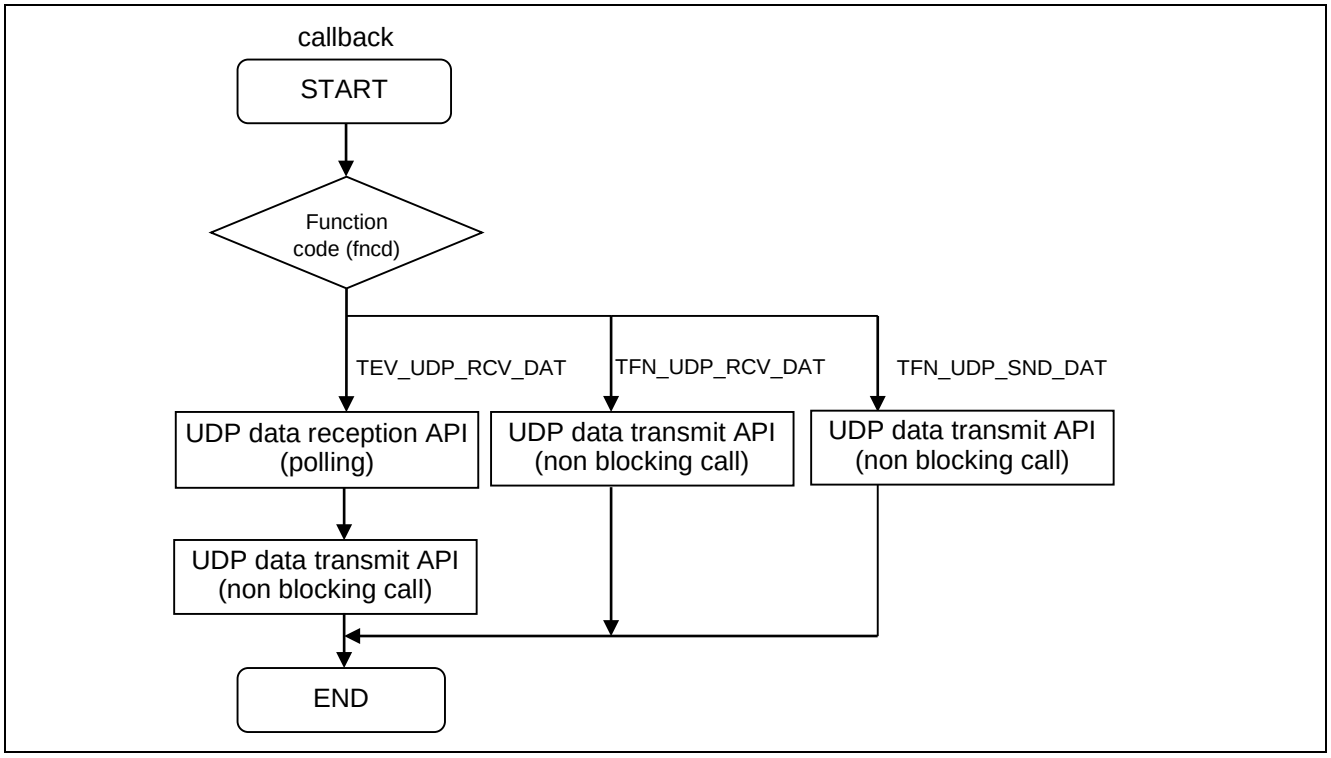



**Figure 5.7 Flow of the UDP Callback Function (for Non-Blocking Call)**

## 5.3 Environment for RX62N/RX63N sample program

Sample program projects for RX62N, RX63N are provided in the package, and the projects can import to e$^2$ studio and IAR Embedded Workbench.

### 5.3.1 Software Structure

Figure 5.8 shows the configuration diagram of each driver.

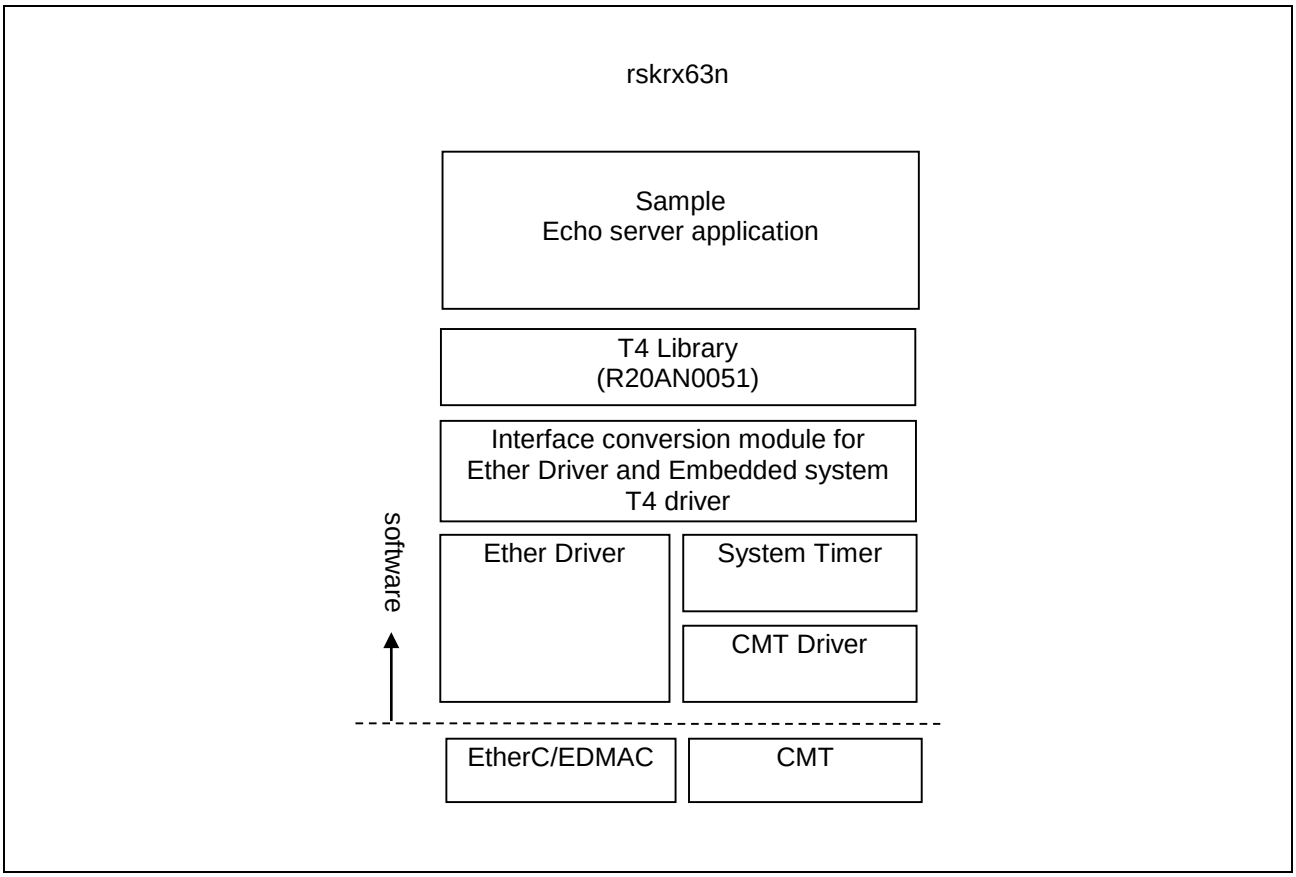



**Figure 5.8 Configuration diagram of each driver**

### 5.3.2 Method of converting e² studio to CS+ project

e² studio project can be converted to CS+ project to use *.rcpc file included in e² studio project. This section shows method of converting (in this example using sample program projects for RX63N).

- Start CS+ for CC, push the "GO" button in "e² studio / CubeSuite / …".
- Select "e² studio project file (*.rcpc) " and, open the *.rcpc file.
- "Project convert settings" window would open, and please select project in the project tree.
- Project settings on the right side of project tree, please select MCU "RX63N" -> "R5F563NFDxFC" and push the "OK" button. CS+ outputs the converted project.
- Please select "CC-RX" in the project tree.
- Please select "RXv2 architecture" in the "common option" tab -> "CPU" -> "command set architecture"
- echo_srv.c is registered in the each folders (project tree -> file -> src).
  -> TCP-blocking, TCP-non-blocking, UDP-blocking, UDP-nonblocking sample.
  Please remove echo_srv.c from the project tree excluding you need to work.
- Build the project
- Please set the debug tools fitting for your environment. User can confirm working sample program after this.

## 5.4 Confirm sample program

How to confirm Ethernet sample program

### 5.4.1 Environment

#### (1) Connecting the hardware

Setup Hardware connections

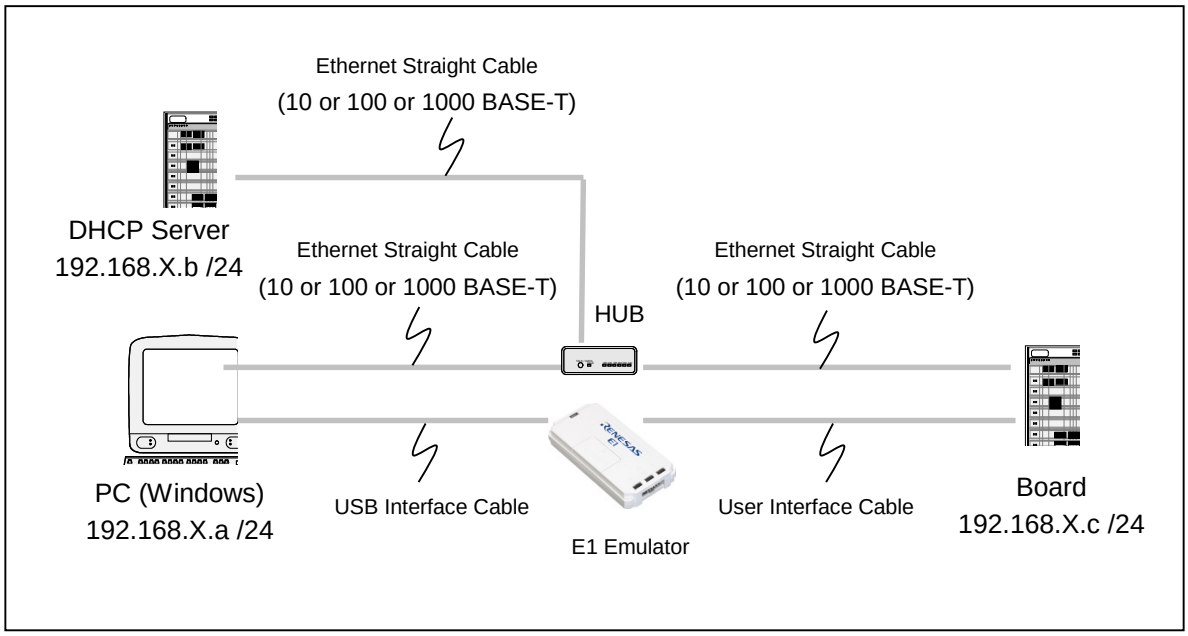



**Figure 5.9 Ethernet sample program environment**

We have confirmed using the Ethernet-switch product introduced in below.

- NETGEAR: GS108E

This Ethernet-switch has the function called “port mirroring function”, this function provides monitoring function for Ethernet. This Ethernet-switch can realize packet monitoring environment if user uses normally Ethernet-switch.

For example, please refer to the figure below, the board A transfers data to board B, normally Ethernet-switch filters packet and only outputs to the port connected to board B. If “port mirroring function” exists on Ethernet-switch, it copies data board B port and port mirroring port.

This function provides to monitor for peer-to-peer communication.

We recommend “Wireshark” for packet monitor. Please use “promiscuous mode” for peer-to-peer communication.

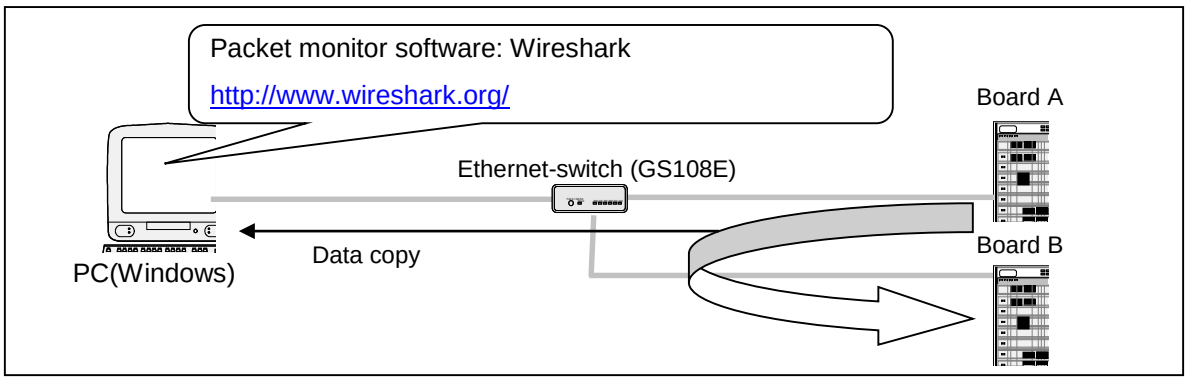



**Figure 5.10 Ethernet sample program confirmation environment**

#### (2) PC setting

Windows 7:

Control Panel -> Network -> Adaptor setting -> Local Network Connection

Network Tab -> Internet Protocol Version 4 (TCP/IPv4) -> Property

Please select the “Auto IP address configuration” in following dialog.

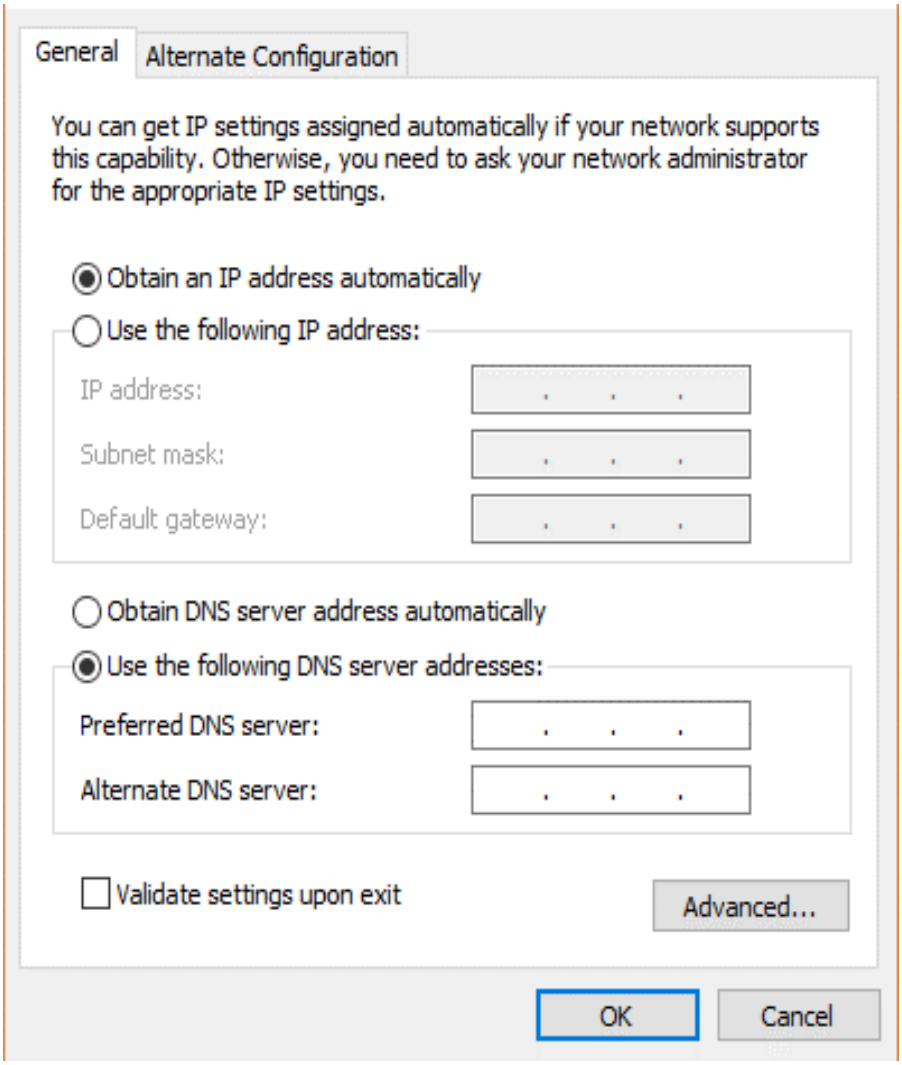



**Figure 5.11 Internet Protocol Version 4 (TCP/IPv4) Property**

After setting, please push OK button.

### (3) Start Sample program (sample folder)

- Start e² studio and open the sample program.
- From the [Project] menu, click the [Build Project].
- Connect E1 Emulator, and from the [Run] menu, click the [Debug].
- The program is run by clicking button in the [Debug] view , or pressing [F8] key.

### (4) Confirm IP Address for MCU

IP address will be allocated by DHCP server when sample program executes.

User can confirm the allocated IP address value using Renesas Debug Virtual Console on e² studio. In the GCC sample project, the IP address is not displayed on the Renesas debug virtual console. Check directly in the debug window.

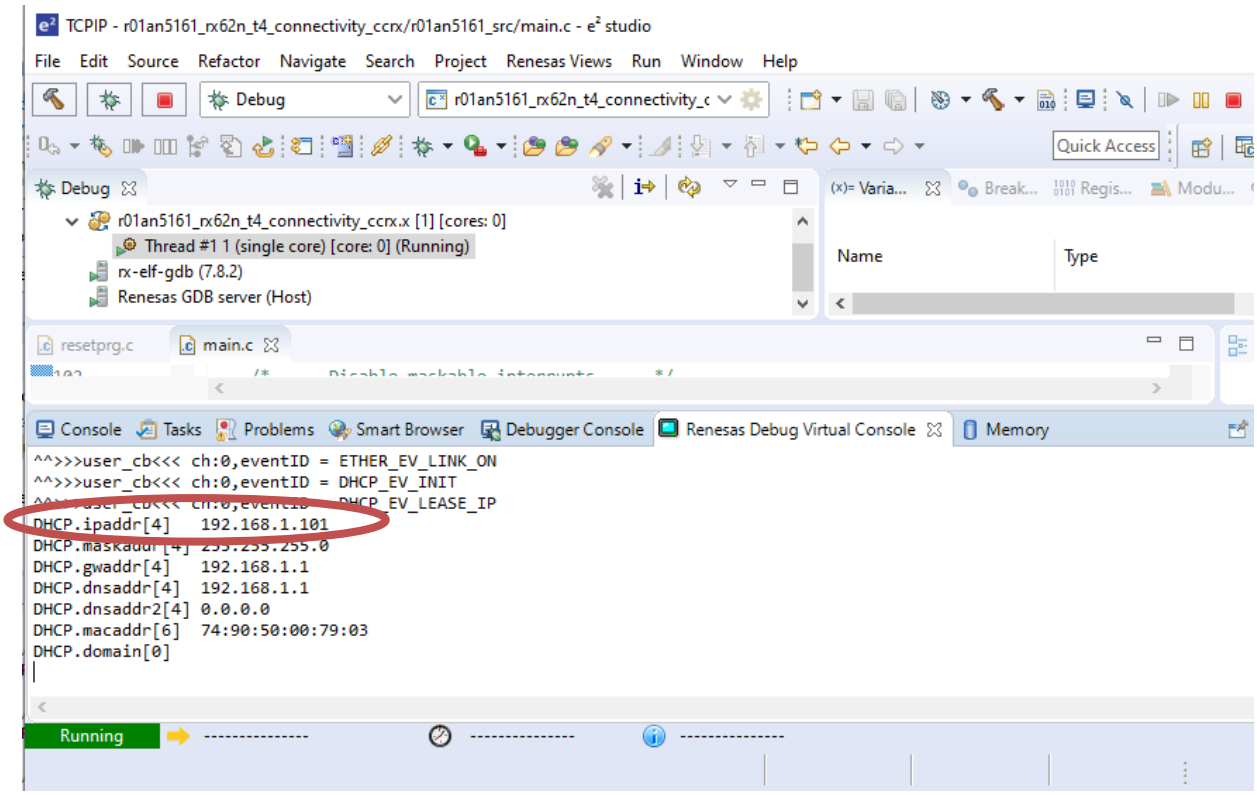



**Figure 5.12 Example about Display of IP address**

Figure Example about display of IP Address shows IP address is 192.168.1.101 is allocated to MCU.

Please execute ipconfig in command prompt in order to confirm PC (Windows) IP address if need.

### (5) Communication inspection

Execute ping command to MCU in command prompt.

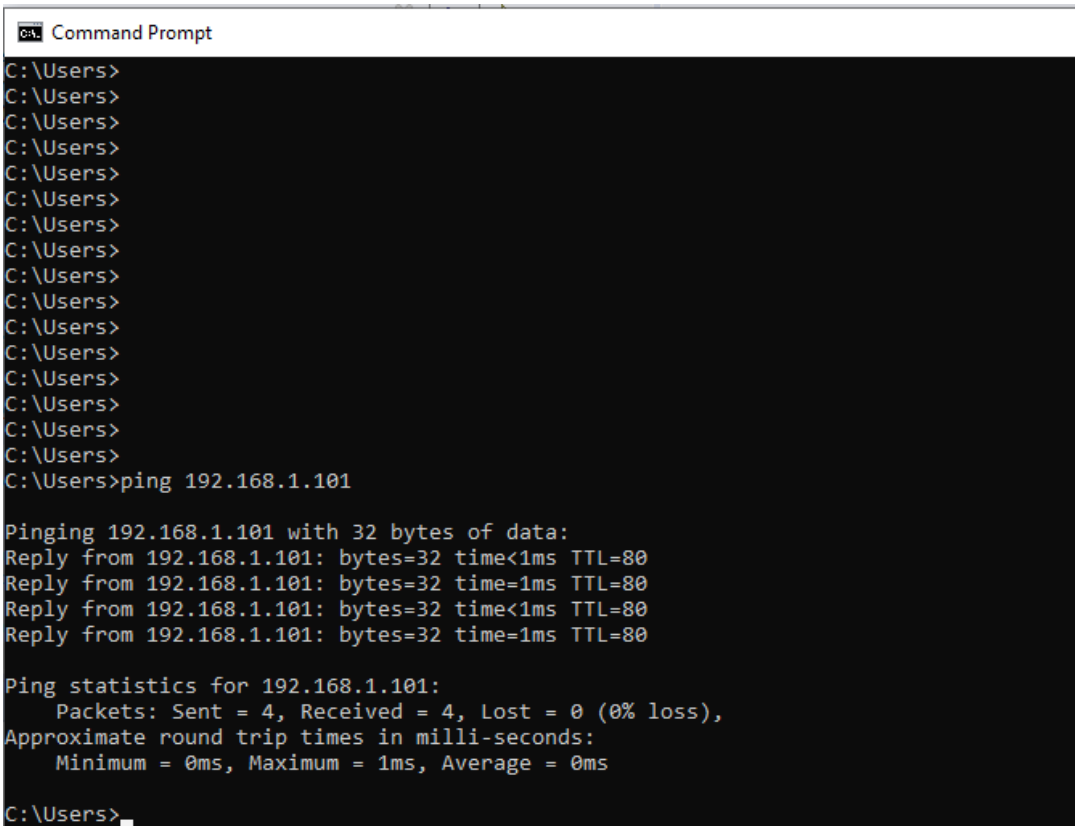



**Figure 5.13 Example about Execution of ping**

### 5.4.2 Confirm TCP connection

Execute telnet in command prompt

(1) **Enable telnet(Windows7 only)**

- Start -> Control Panel -> Program and Function

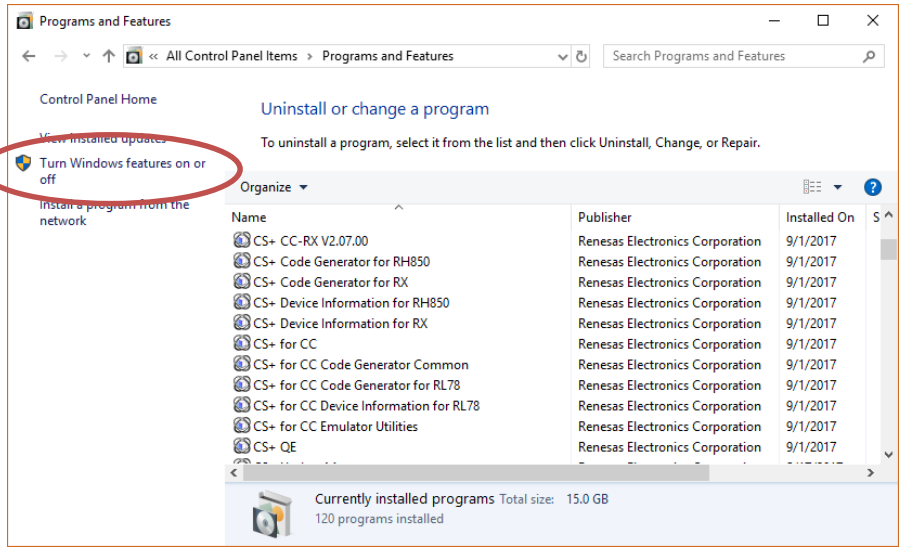



**Figure 5.14 Program and Function**

- Please check Telnet client

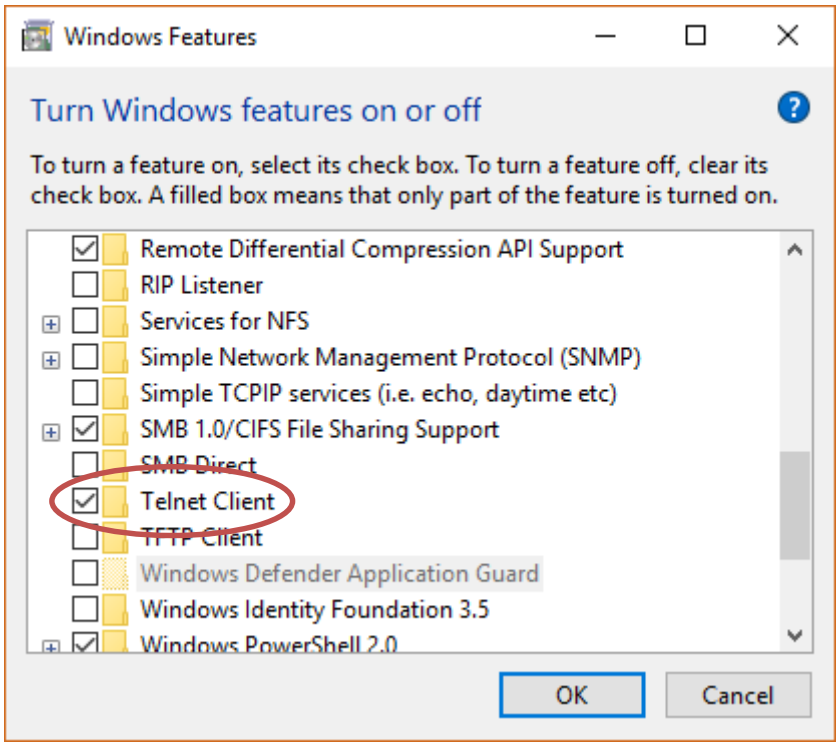



**Figure 5.15 Telnet Client**

### (2) Single LAN Port

Execute the command shown below at the MS-DOS prompt of your computer.

[TCP blocking call sample program]
telnet 192.168.X.c 1024

[TCP none-blocking call sample program (multiple communication end point can be used at same time)]
telnet 192.168.X.c 1024
telnet 192.168.X.c 1025

### (3) Several LAN Port

Execute one of the following depending on the execution environment of the sample program.
To establish connections, run the following command at the MS-DOS prompt on the PC.

[TCP blocking call sample program]
telnet 192.168.X.c 1024
telnet 192.168.X.d 1025

[TCP none-blocking call sample program(multiple communication end point can be used at same time)]
telnet 192.168.X.c 1024
telnet 192.168.X.c 1025

telnet 192.168.X.d 1026
telnet 192.168.X.d 1027

### (4) End of communication

Please input “telnet 192.168.X.c 1024” in command prompt. (192.168.X.c is the IP address allocated to MCU.)

Please input any keyboard input.

It is OK to confirm the data echo-back.

Please input Ctrl + “]” and next, input “quit[enter key]” makes disconnection.

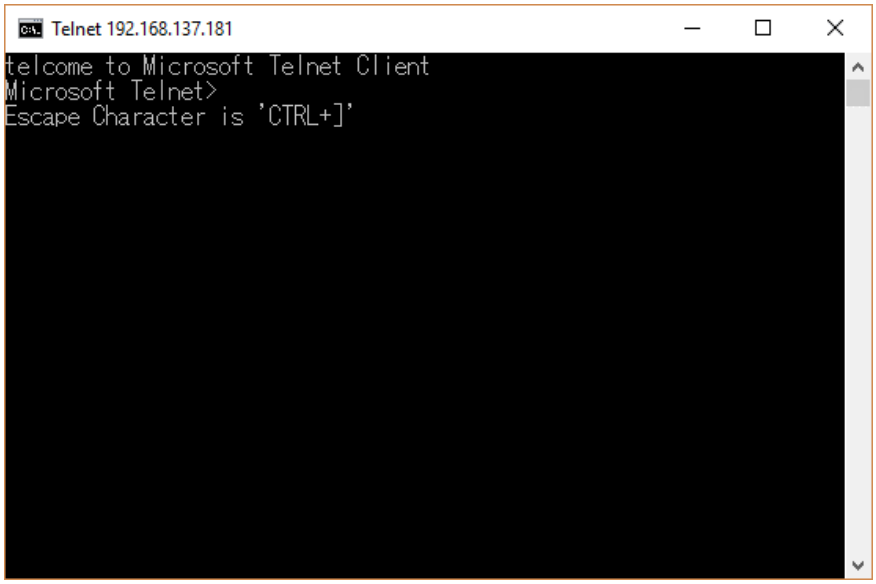



**Figure 5.16 Disconnect**

## 5.4.3 Confirm UDP connection

User uses PC free tool that can generate UDP packet.

### (1) Preparation of UDP software

Get free software that can send and receive UDP data at the following site.:
Socket Debugger Free https://www.udom.co.jp/sdg/
(This tool includes Japanese character)

### (2) Single LAN Port

User uses PC free tool that can generate UDP packet. Setting is below.

[UDP blocking call sample program]
Destination IP address: 192.168.X.c port number 1365

[UDP none-blocking call sample program(multiple communication end point can be used at same time)]
Destination IP address: 192.168.X.c port number 1365
Destination IP address: 192.168.X.c port number 1366

### (3) Several LAN Port

User uses PC free tool that can generate UDP packet. Setting is below.

[UDP blocking call sample program]
Destination IP address: 192.168.X.c port number 1365
Destination IP address: 192.168.X.d port number 1366

[UDP none-blocking call sample program(multiple communication end point can be used at same time)]
Destination IP address: 192.168.X.c port number 1367
Destination IP address: 192.168.X.c port number 1368

Destination IP address: 192.168.X.d port number 1369
Destination IP address: 192.168.X.d port number 1370

### (4) End of communication

Since UDP does not establish a connection, there is no communication termination command.

## 5.5 Configuration Overview

The configuration option settings of this module are located in r_t4_rx_config.h and r_driver_rx_config.h. The r_t4_rx_config.h's option names and setting values are in Section 2.7 of R20AN0051EJ0209. The r_driver_rx_config.h's option names and setting values are listed in the table below:

| Configuration options in r_driver_rx_config.h | |
|---|---|
| DRV_MCU_RX63_ALL<br>Note: Default value = 1 | Select whether the RX63N group is used.<br>If not use RX63N group, please comment out this macro. |
| DRV_MCU_RX62_ALL<br>Note: Default value = 1 | Select whether the RX62N group is used.<br>If not use RX62N group, please comment out this macro. |
| DRV_PCLKB_HZ<br>Note: Default value = 48000000 | Setting the frequency of PCLKB of RX63N<br>For the setting range, refer to RX63N Group UMH. |
| DRV_PCLK_HZ<br>Note: Default value = 48000000 | Setting the frequency of PCLK of RX62N<br>For the setting range, refer to RX62N Group UMH. |
| CMT_RX_CFG_IPR<br>Note: Default value = 5 | Interrupt priority level used for CMT interrupts |
| ETHER_CFG_MODE_SEL<br>Note: Default value = 0 | Sets the interface between ETHERC and the Ethernet PHY-LSI.<br>If set to 0, MII (Media Independent Interface) is selected.<br>If set to 1, RMII (Reduced Media Independent Interface) is selected. |
| ETHER_CFG_CH0_PHY_ADDRESS<br>Note: Default value = 31 | Specify the PHY-LSI address used by ETHERC channel 0.<br>Specify a value between 0 and 31. |
| ETHER_CFG_EMAC_RX_DESCRIPTORS<br>Note: Default value = 1 | Sets the number of receive descriptors.<br>This must be set to a value 1 or greater |
| ETHER_CFG_EMAC_TX_DESCRIPTORS<br>Note: Default value = 1 | Sets the number of transmit descriptors.<br>This must be set to a value 1 or greater |
| ETHER_CFG_BUFSIZE<br>Note: Default value = 1,536 | Specify the size of the transmit buffer or receive buffer.<br>The buffer is aligned with 32-byte boundaries, so specify a value that is a multiple of 32 bytes. |
| ETHER_CFG_EINT_INT_PRIORITY<br>Note: Default value = 2 | Sets the priority level of the EINT interrupt.<br>This must be set to a value in the range 1 to 15. |
| ETHER_CFG_CH0_PHY_ACCESS<br>Note: Default value = 0 | Specify the PHY access channel used by ETHERC channel 0.<br>When 0 is specified, ETHERC0 is used for PHY register access<br>When 1 is specified, ETHERC1 is used for PHY register access. |
| ETHER_CFG_PHY_MII_WAIT<br>Note: Default value = 8 | Specify the loop count of software loop used for read or write in PHY-LSI. Set the number of loops according to the PHY-LSI to be used.<br>Specify a value of 1 or greater. |
| ETHER_CFG_PHY_DELAY_RESET<br>Note: Default value = 0x00020000 | Specify the loop count used for timeout processing of PHY-LSI reset completion wait. Set the number of loops according to the PHY-LSI to be used. |

| | |
|---|---|
| ETHER_CFG_LINK_PRESENT<br>Note: Default value = 0 | Specify the polarity of the link signal output by the PHY-LSI.<br>When 0 is specified, link-up and link-down correspond respectively to the fall and rise of the LINKSTA signal.<br>When 1 is specified, link-up and link-down correspond respectively to the rise and fall of the LINKSTA signal. |
| ETHER_CFG_USE_LINKSTA<br>Note: Default value = 0 | Specify whether or not to use the PHY-LSI status register instead of the LINKSTA signal when a change in the link status is detected.<br>When 0 is specified, the PHY-LSI status register is used.<br>When 1 is specified, the LINKSTA signal is used. |
| ETHER_CFG_USE_PHY_KSZ8041NL<br>Note: Default value = 0 | Specify whether or not the KSZ8041NL PHY-LSI from Micrel is used.<br>When 0 is specified, the KSZ8041 is not used.<br>When 1 is specified, the KSZ8041 is used. |

## 5.6 Section setting

### 5.6.1 CCRX for Renesas RX section setting example

The following settings are required in the e$^2$ studio linker setting.

- Add the section.

**Adding the Section**

Add B_ETHERNET_BUFFERS_1, B_RX_DESC_1 and B_TX_DESC_1 section to the RAM.

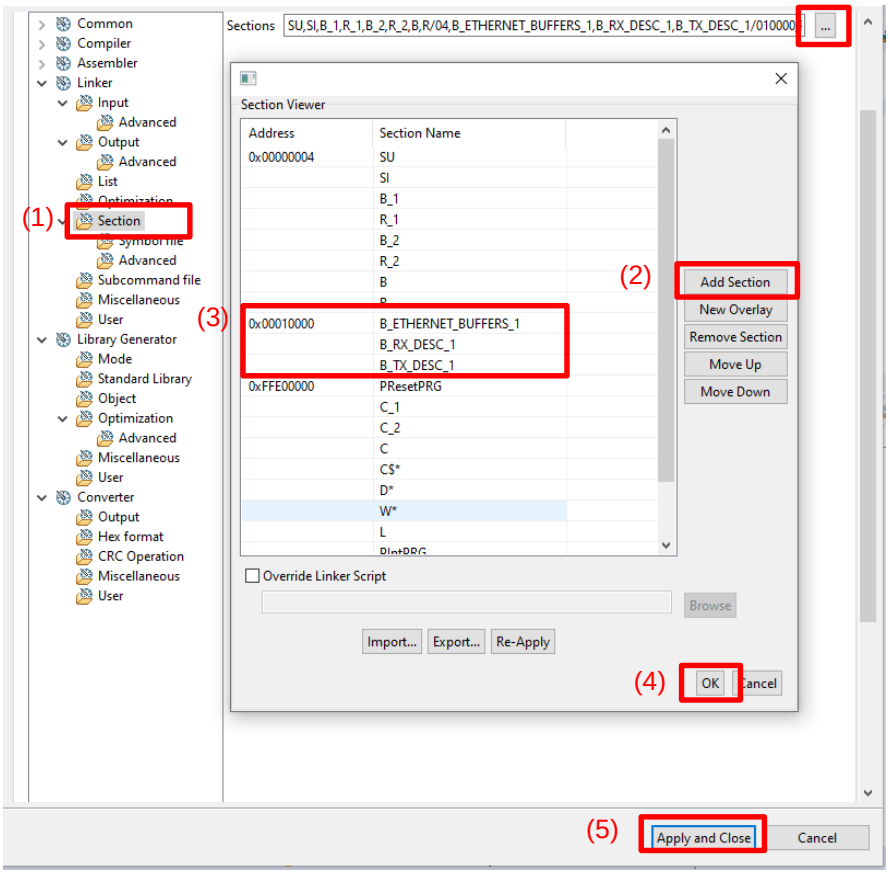
(1) Click the Section.

(2) Click the Add section button.

(3) Type Address and the Section Name.

(4) Click the OK button.

(5) Click the Apply and Close button.



### 5.6.2 GCC for Renesas RX section setting example

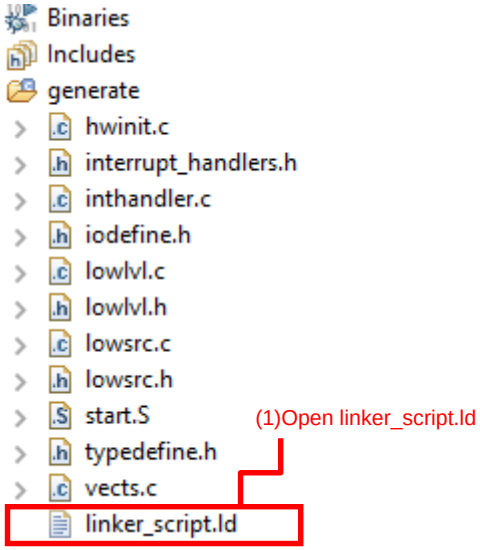
Edit the linker_script.ld file and add sections and symbols.

**Add Sections and Symbols**

Add the following code.

```
B_ETHERNET_BUFFERS_1 0x00010000 (NOLOAD) : AT(0x00010000)
{
        _B_ETHERNET_BUFFERS_1_start = .;
        *(B_ETHERNET_BUFFERS_1)
        _B_ETHERNET_BUFFERS_1_end = .;
} >RAM
B_RX_DESC_1 (NOLOAD) :
{
        _B_RX_DESC_1_start = .;
        *(B_RX_DESC_1)
        _B_RX_DESC_1_end = .;
} >RAM
B_TX_DESC_1 (NOLOAD) :
{
        _B_TX_DESC_1_start = .;
        *(B_TX_DESC_1)
        _B_TX_DESC_1_end = .;
} >RAM
```

(1) Open "linker_script.ld" from Project Explorer.



(2) Click linker_script.ld.

(3) Enter code.

### 5.6.3 IAR C/C++ Compiler for Renesas RX section setting example

Edit the icf file and add section settings.

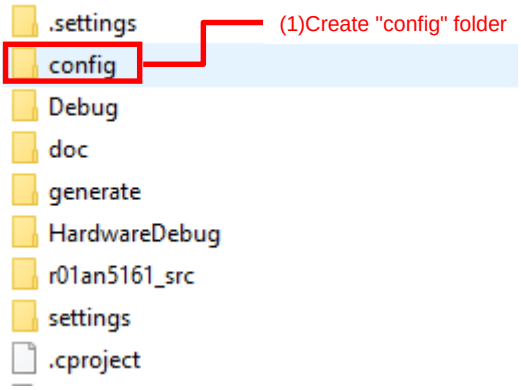
The icf file to be edited depends on the target device of the project, so please confirm and edit the upper 8 digits of the model name of the device to be used.

As an example, edit "lnkr5f563nf.icf" with RX63N (R5F563NFDxFC).

The following is an example of editing on the RX63N (R5F563NFDxFC).

Add section settings.

(1) Create "config" folder in project folder.

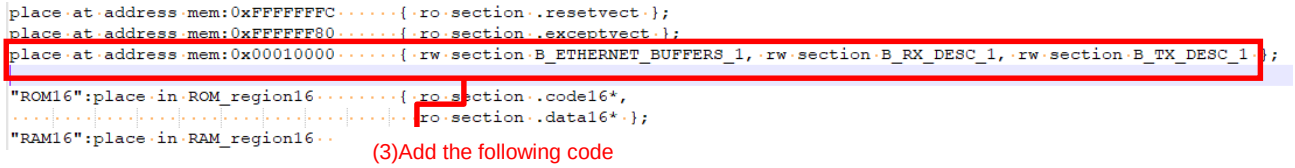



(2) Copy "lnkr5f563nf.icf" from "\rx\config" where IAR C/C++ Compiler for Renesas RX ("EWRX") is installed to the "config" folder in the project folder.

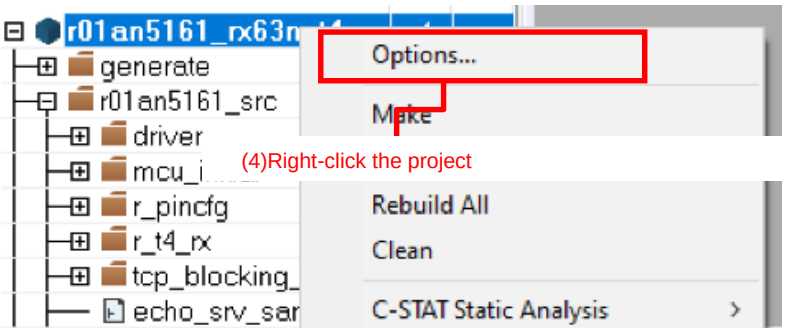
The installation default is "C: \Program Files (x86)\IAR Systems\Embedded Workbench 8.3".


(3) Open the copied "lnkr5f563nf.icf" file and add the following code.

```
place at address mem:0x00010000      { rw section B_ETHERNET_BUFFERS_1, rw section
B_RX_DESC_1, rw section B_TX_DESC_1 };
```



(4) Open the project from EWRX, right-click the project in the workspace and open options.

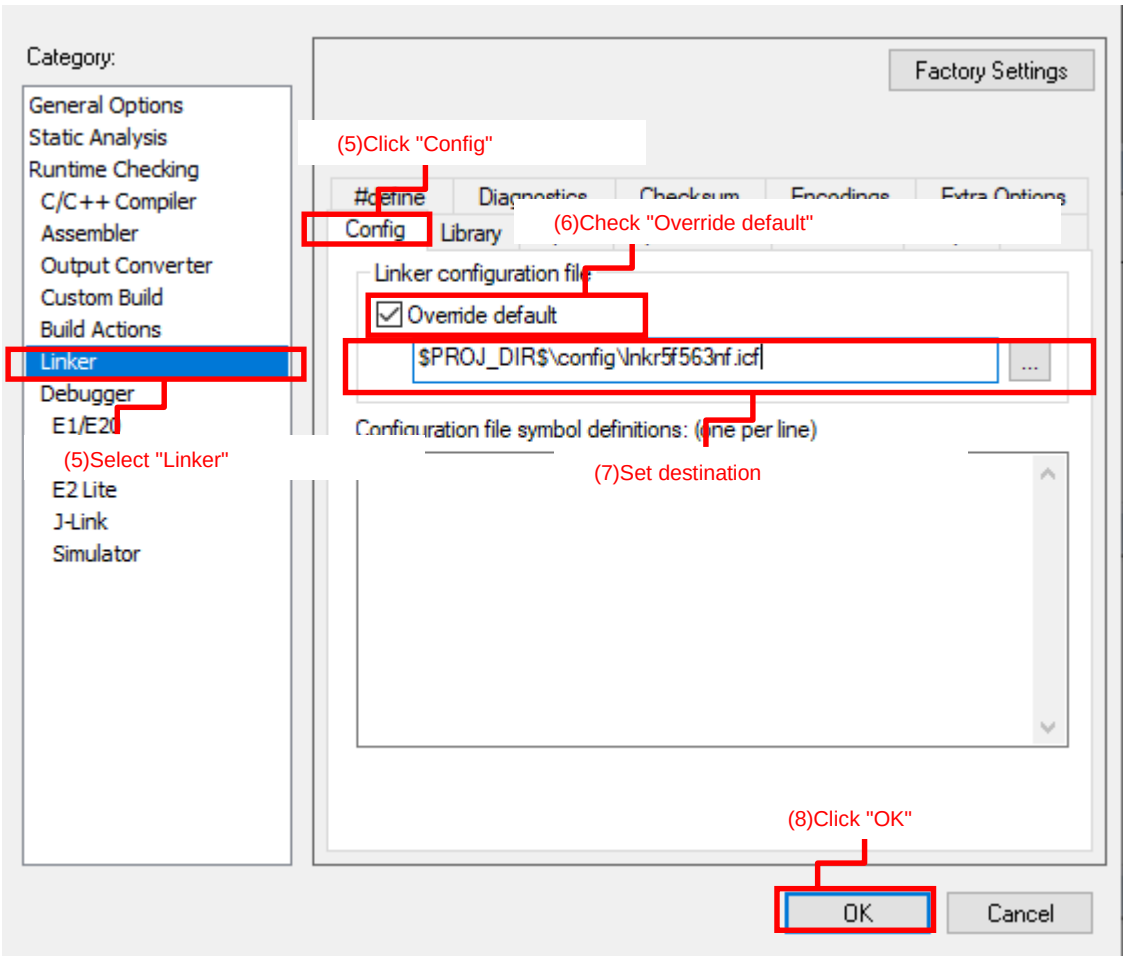



(5) Select "Category: Linker" and click "Config".

(6) Check "Override default".

(7) Set the file edited in step (3) as a reference destination.

(8) Click "OK".



## 5.7 Option-Setting Memory

Table 5-5 lists the Option-Setting Memory Configured in the Sample Code. When necessary, set a value suited to the user system.

**Table 5-5 Option-Setting Memory Configured in the Sample Code**

| Symbol | Address | Setting Value | Contents |
|---|---|---|---|
| OFS0 | FFFF FF8Fh to FFFF FF8Ch | FFFF FFFFh | The IWDT is stopped after a reset.<br>The WDT is stopped after a reset. |
| OFS1 | FFFF FF8Bh to FFFF FF88h | FFFF FFFFh | The voltage monitor 0 reset is disabled after a reset.<br>HOCO oscillation is disabled after a reset. |
| MDES | FFFF FF83h to FFFF FF80h | FFFF FFFFh | Little endian |

Notes:

1. Option-Setting memory only for RX63N.

## 6. Sample code

Sample code can be downloaded from the Renesas Electronics website.

## 7. Notes

### 7.1 T4 Library

(1) Specify the size of 15bit or less for the third argument "INT len" of tcp_rcv_dat() and tcp_snd_dat().
(2) Specify the size of 15bit or less for argument "TMO tmout" of tcp_acp_cep(), tcp_con_cep(), cp_cls_cep(), tcp_rcv_dat(), tcp_snd_dat(), udp_snd_dat() and udp_rcv_dat().
(3) This library can be used with Microcontroller Options fint_register=0 (Fast interrupt vectorregister [None]). The default for this option is fint_register=0.

### 7.2 Sample Program

(1) The sample program for little endian mode is only included.
(2) The MAC address of the sample program is stored in _myethaddr variable of config_tcpudp.c.
Change an initial value of the _myethaddr (MAC address) variable if necessary according to the system.

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Dec 22, 19 | - | First released. |

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

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

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

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

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

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

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

# Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

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

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

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

(Rev.4.0-1 November 2017)

## Corporate Headquarters

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

## Trademarks

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/.