

RX Family

TCP/IP for Embedded system M3S-T4-Tiny Introduction Guide Firmware Integration Technology

Introduction

This document explains TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.09 (hereafter referred to as "T4") that depends on MCUs. This documents name is "Introduction Guide".

T4 is the TCP/IP protocol stack for embedded system. T4 is provided as library format and user can develop own system with this library to use TCP/IP function. The peripherals of the MCU used for communication are Ethernet. The peripherals are internal Ethernet controller or external bus. The external bus connects to external Ethernet controller chip. We recommend RX64M, RX71M, or RX65N (has internal Ethernet controller) for Ethernet system, in case user selects RX family.

We prepared sample programs for each CPU board included in [the Renesas Starter Kit+](#), [the Gadget Renesas RX65N board \(GR-ROSE\)](#), the 3rd Party board. 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>

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

<https://www.renesas.com/software-tool/fit>

[Notice about confirm working on MCUs]

This application is only FIT Module about TCP/IP functions. The sample program that can be confirmed working on MCUs is not included. The sample program using T4 FIT module will be uploaded to the URL the below.

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

This figure shows 4 cases of T4 software stack. The software stack depends on the following conditions:

- ITRON TCP/IP APIs Style or Socket APIs style
- With TSIP (Trusted Secure IP) ^(Note) or Without TSIP

Note: In case of using MCU equipped TSIP

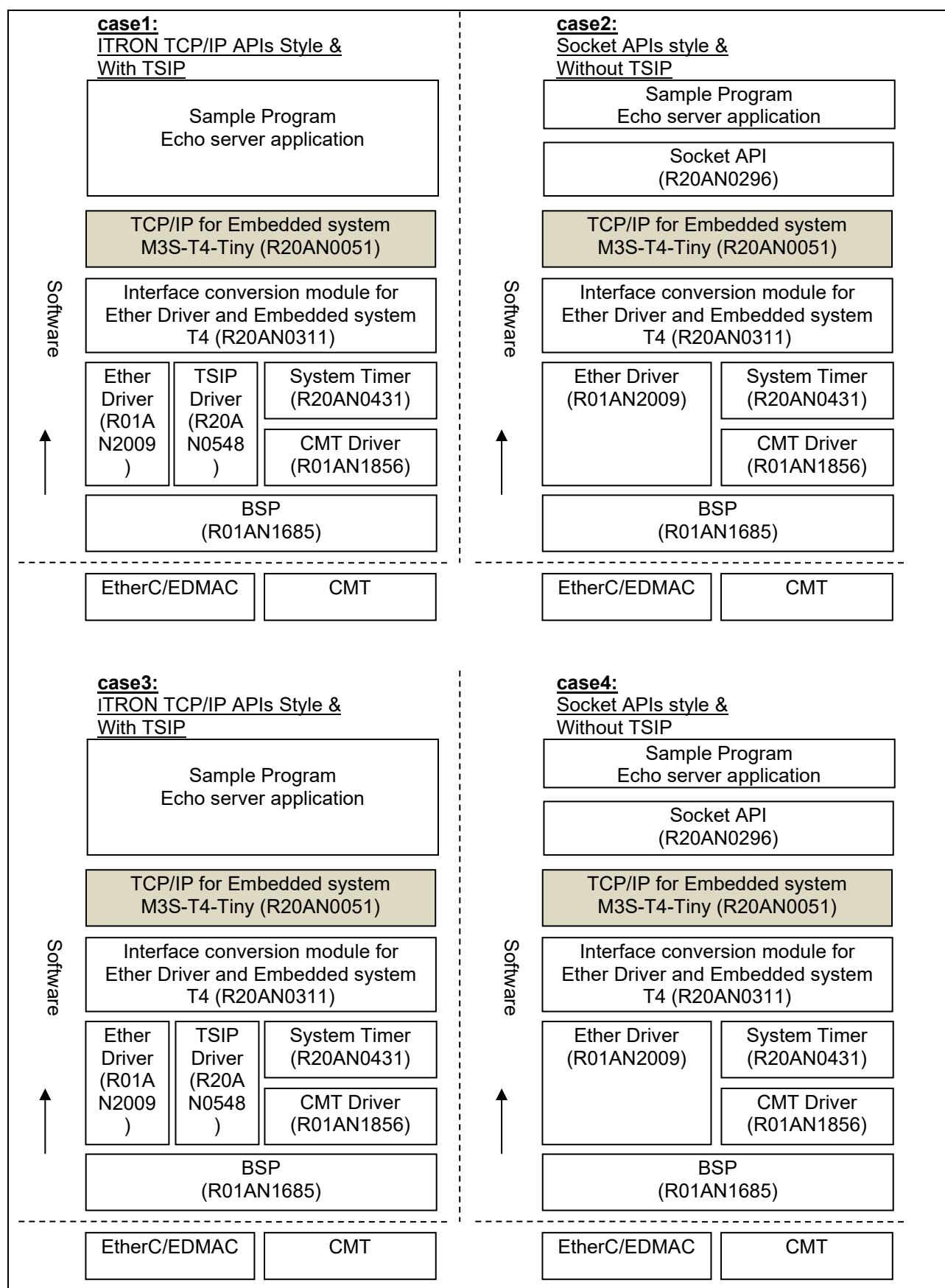


Figure 1.1 T4 Software Stack

[Notice about spec of T4]

T4 is assumed for easy application implementation. T4 does not have the function that "Socket interface" like Linux TCP/IP, next generation IP technology like IPSec and IPv6, router function like ICMP error notifying and routing protocol.

T4 may not implement the latest vulnerability countermeasures. Therefore, T4 is suitable for communication aimed at local connections rather than the Internet connections.

Target Device

RX family

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 "7.1 Confirmed Operation Environment".

Note that use of GCC for Renesas RX has limitation. Refer to "1.6 Limitations".

Related Documents

- Firmware Integration Technology User's Manual (R01AN1833)
- RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685)
- RX Family Adding Firmware Integration Technology Modules to Projects (R01AN1723)
- RX Family Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)
- RX Smart Configurator User's Guide: e² studio (R20AN0451)
- RX Smart Configurator User's Guide: CS+ (R20AN0470)
- RX Smart Configurator User's Guide: IAREW (R20AN0535)
- RX Family Interface conversion module for Ethernet Driver and Embedded system M3S-T4-Tiny Firmware Integration Technology (R20AN0311)
- RX Family Ethernet Module Using Firmware Integration Technology (R01AN2009)
- RX Family System Timer Module Firmware Integration Technology (R20AN0431)
- RX Family CMT Module Using Firmware Integration Technology (R01AN1856)
- TSIP (Trusted Secure IP) Module Firmware Integration Technology (Binary version) (R20AN0548)
- TCP/IP for Embedded system M3S-T4-Tiny User's Manual (R20UW0031)
- TCP/IP for Embedded system M3S-T4-Tiny: Ethernet Driver Interface Specification (R20UW0032)

Contents

1. Overview	6
1.1 T4 FIT Module	6
1.2 Overview of the T4 FIT Module	6
1.3 API Overview	6
1.4 File Structure	7
1.5 Processing Example	8
1.6 Limitations	9
2. API Information	10
3. T4 Library Information	16
3.1 Compile option	16
3.1.1 CC-RX	16
3.1.2 GCC	16
3.1.3 IAR	16
3.2 Version information	17
3.2.1 CC-RX	17
3.2.2 GCC	17
3.2.3 IAR	17
4. QE for TCP/IP	18
5. Sample Program	19
5.1 Flow of the TCP Echo Back Server Function (for Blocking Call)	19
5.1.1 Flow of the Echo Back Server Function	19
5.2 Flow of the TCP Echo Back Server Function (for Non-blocking Call)	19
5.2.1 Flow of the Echo Back Server Function	19
5.2.2 Flow of the Callback Function	19
5.3 Flow of the UDP Echo Back Server Function (for Blocking Call)	19
5.3.1 Flow of the Echo Back Server Function	19
5.4 Flow of the UDP Echo Back Server Function (for Non-blocking Call)	19
5.4.1 Flow of the Echo Back Server Function	19
5.4.2 Flow of the Callback Function	19
5.5 Component Setting	26
5.6 Environment for RX65N sample program	28
5.6.1 Using FIT Modules	28
5.6.2 Software Structure	29
5.6.3 Scheme of link the Library file	30
5.6.4 Method of converting e ² studio to CS+ project	30
5.7 Confirm sample program	31
5.7.1 Environment	31

(1)	Connecting the hardware	31
(2)	PC setting	32
(3)	Start Sample program (sample folder)	33
(4)	Confirm IP Address for MCU	33
(5)	Communication inspection	34
5.7.2	Confirm TCP connection	35
(1)	Enable telnet(Windows10 only)	35
(2)	Single LAN Port	36
(3)	Several LAN Port	36
(4)	End of communication	36
5.7.3	Confirm UDP connection	37
(1)	Preparation of UDP software	37
(2)	Single LAN Port	37
(3)	Several LAN Port	37
(4)	End of communication	37
6.	Notes	38
6.1	T4 Library	38
6.2	Smart Configurator	38
6.3	Sample Program	38
7.	Appendices	39
7.1	Confirmed Operation Environment	39
7.2	Software update information	40
	Revision History	43

1. Overview

1.1 T4 FIT Module

The T4 FIT module can be used by being implemented in a project as an API. See section 2.12 Adding the FIT Module to Your Project for details on methods to implement this FIT module into a project.

1.2 Overview of the T4 FIT Module

T4 FIT module is the TCP / IP communication protocol software library for the RX family.

TCP / IP communication can be realized by any device equipped with RX microcomputer.

Software Name: Embedded TCP/IP M3S-T4-Tiny for RX Family V.2.10
It outlines the file structure.

1.3 API Overview

Table 1.1 lists the API functions included in this module.

Table 1.1 API Functions

Function	Description
tcp_acp_cep	Wait for Connection Request (Passive Open)
tcp_con_cep	Wait for Connection Request (Active Open)
tcp_rcv_dat	Receive Data
tcp_snd_dat	Send Data
tcp_sht_cep	Terminate Data Transmission
tcp_cls_cep	Close Communication End Point
tcp_can_cep	Cancel Pending Processing
udp_rcv_dat	Receive Data
udp_snd_dat	Send Data
udp_can_cep	Cancel Pending Processing
tcpudp_get_ramsize	Calculation of work memory size used by T4 Library
tcpudp_open	Initialize the T4 Library
_process_tcpip	Process TCP/IP
tcpudp_close	Close the T4 Library
tcpudp_reset	Reset the T4 Library
igmp_join_group	Join to Multicast group
igmp_leave_group	Release from Multicast group

Library specification can be seen in user's manual (R20UW0031).

User's manual explains how to use this library, and APIs. And Ethernet driver interface specification (R20UW0032) explain how to make the user defined functions called from library.

1.4 File Structure

This application note includes following files.

Table 1.2 File Structure 1

File/Directory name		detail
r20an0051xx0210-rx-t4-connectivity.zip		
	r20an0051jj0210-rx-t4.pdf	Introduction Guide (Japanese)
	r20an0051ej0210-rx-t4.pdf	Introduction Guide (English this document)
FITDemos		
	rskrx65n_tcp_blocking	[e ² studio] (cc-rx) (rskrx65n) TCP Blocking Call
	rskrx65n_tcp_nonblocking_cancel	[e ² studio] (cc-rx) (rskrx65n) TCP Non-blocking cancel Call
	rskrx65n_tcp_nonblocking	[e ² studio] (cc-rx) (rskrx65n) TCP Non-blocking Call
	rskrx65n_udp_blocking	[e ² studio] (cc-rx) (rskrx65n) UDP Blocking Call
	rskrx65n_udp_nonblocking	[e ² studio] (cc-rx) (rskrx65n) UDP Non-blocking Call
FITModules		
	r_t4_rx_v2.10.xml	xml file
	r_t4_rx_v2.10.zip	fit modules
	r_t4_rx_v2.10_extend.mdf	mdf file

The folder to which the contents of r_t4_rx_v.2.10.zip is extracted will contain the files listed in table Table 1.3 Structure of the T4 FIT Modules below.

Table 1.3 Structure of the T4 FIT Modules

File/Directory name	Description
T4 FIT Module (r_t4_rx_v.2.10.zip)	
T4 config (r_config)	
r_t4_rx_config.h	T4 Config header
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)
r20uw0032jj0109-t4tiny.pdf	Ethernet Driver Interface Specification (Japanese)
r20an0051jj0210-rx-t4.pdf	Introduction Guide (Japanese)
en	
r20uw0031ej0111-t4tiny.pdf	User's Manual (English)
r20uw0032ej0109-t4tiny.pdf	Ethernet Driver Interface Specification (English)
r20an0051ej0210-rx-t4.pdf	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

1.5 Processing Example

Figure 5.1 Flow of the Main Function.

1.6 Limitations

About GCC for Renesas RX, T4 only supports GCC for Renesas RX 4.8.4.201803 or earlier. GCC for Renesas RX after this version cannot be used with T4 because GCC added the "in_addr_t" type and this type is also defined in T4, this makes build error of duplicate type of definition.

2. API Information

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

2.1 Hardware Requirements

The MCU used must support the following functions:

- Internal Ethernet controller or external bus that is connected to external Ethernet controller chip

2.2 Software Requirements

This driver is dependent upon the following FIT module:

- r_t4_driver_rx

2.3 Supported Toolchain

This driver has been confirmed to work with the toolchain listed in 7.1, Confirmed Operation Environment.

Note that use of GCC for Renesas RX has limitation. Refer to “1.6 Limitations”.

2.4 Interrupt Vector

None.

2.5 Header Files

All API calls and their supporting interface definitions are located in r_t4_itcpip.h.

2.6 Integer Types

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

2.7 Configuration Overview

The configuration option settings of this module are located in `r_t4_rx_config.h`. The option names and setting values are listed in the table below:

Configuration options in <code>r_t4_rx_config.h</code>	
T4_CFG_SYSTEM_CHANNEL_NUMBER Note: Default value = 1	Set the number of LAN ports. This option is set to " <code>_t4_channel_num</code> ".
T4_CFG_SYSTEM_DHCP Note: Default value = 1	Select whether to use DHCP. 1=Use, 0=No use This option is set to " <code>_t4_dhcp_enable</code> ".
T4_CFG_FIXED_IP_ADDRESS_CHx (x: 0-(T4_CFG_SYSTEM_CHANNEL_NUMBER-1))	(Setting is unnecessary when T4_CFG_SYSTEM_DHCP is 1) Set the IP address of each LAN socket. Separate bytes with a comma (,). Example: 192,168,0,3 This option is set to " <code>tcpudp_env[].ipaddr</code> ".
T4_CFG_FIXED_SABNET_MASK_CHx (x: 0-(T4_CFG_SYSTEM_CHANNEL_NUMBER-1))	(Setting is unnecessary when T4_CFG_SYSTEM_DHCP is 1) Set the IP address of each LAN socket. Separate bytes with a comma (,). Example: 255,255,255,0 This option is set to " <code>tcpudp_env[].maskaddr</code> ".
T4_CFG_FIXED_GATEWAY_ADDRESS_CHx (x: 0-(T4_CFG_SYSTEM_CHANNEL_NUMBER-1))	(Setting is unnecessary when T4_CFG_SYSTEM_DHCP is 1) Set the Gateway address of each LAN socket. Separate bytes with a comma (,). Example: 0,0,0,0 This option is set to " <code>tcpudp_env[].gwaddr</code> ".
T4_CFG_ETHER_CHx_MAC_ADDRESS (x: 0-(T4_CFG_SYSTEM_CHANNEL_NUMBER-1))	Set the MAC address of each LAN socket. Separate bytes with a comma (,). Example: 0x74,0x90,0x50,0x10,0xFE,0x77 This option is set to " <code>_myethaddr</code> ".
T4_CFG_SYSTEM_CALLBACK_FUNCTION_USE	Sets whether or not to register the system callback function. 1=Regist, 0=No regist This option is set to " <code>g_fp_user</code> ".
T4_CFG_SYSTEM_CALLBACK_FUNCTION_NAME_TMP	(Setting is unnecessary when T4_CFG_SYSTEM_CALLBACK_FUNCTION_USE is 0) Register the function name of the system callback. This option is set to " <code>g_fp_user</code> ".
T4_CFG_TCP_REPIDx_PORT_NUMBER (x: 1-4)	Set the port number of TCP reception port. This option is set to " <code>tcp_crep[].myaddr.portno</code> ".
T4_CFG_TCP_CEPIDx_CHANNEL (x: 1-6)	Set the LAN socket number that you want to link to each TCP endpoint. The range is 0 to (T4_CFG_SYSTEM_CHANNEL_NUMBER - 1). This option is set to " <code>tcp_ccep[].lan_port_number</code> ".
T4_CFG_TCP_CEPIDx_RECEIVE_WINDOW_SIZE (x: 1-6)	Sets the receive window size of each TCP endpoint. This option is set to " <code>tcp_ccep[].rbufsz</code> ".
T4_CFG_TCP_CEPIDx_CALLBACK_FUNCTION_USE (x: 1-6)	Sets whether to use the callback routine for each TCP endpoint. Set 0 for endpoints to be executed by blocking calls. Set 1 for endpoints to be executed with non-blocking calls. This option is set to " <code>tcp_ccep[].callback</code> ".

T4_CFG_TCP_CEPIDx_CALLBACK_FUNCTION_NAME_TMP (x: 1-6)	(Setting is unnecessary when T4_CFG_TCP_CEPIDx_CALLBACK_FUNCTION_USE is 0) Register the function name of the TCP callback. This option is set to "tcp_ccep[].callback".
T4_CFG_TCP_CEPIDx_KEEPAIVE_ENABLE (x: 1-6)	Sets whether to use the KEEPAIVE function for each TCP endpoint. 0 = invalid, 1 = enabled This option is set to "tcp_ccep[].keepalive_enable".
T4_CFG_TCP_MSS Note: Default value = 1460 (byte)	Set the MSS value of TCP communication. This option is set to "_tcp_mss".
T4_CFG_TCP_2MSL_TIME Note: Default value = 60 (second)	Set the 2MSL value of TCP communication. This option is set to "_tcp_2msl".
T4_CFG_TCP_MAX_TIMEOUT_PERIOD Note: Default value = 600 (second)	Set retransmission timeout time for TCP communication. This option is set to "_tcp_rt_tmo_rst".
T4_CFG_TCP_DIVIDE_SENDING_PACKET Note: Default value = 1	Selects divide sending packet function of TCP communication. 0 = invalid, 1 = enabled This option is set to "_tcp_dack".
T4_CFG_TCP_KEEPAIVE_START Note: Default value = 7200 (second)	Set Keepalive transmission start time of TCP communication. The setting range is 1 to 86400. This option is set to "_tcp_keepalive_start".
T4_CFG_TCP_KEEPAIVE_INTERVAL Note: Default value = 10 (second)	Set retransmission interval time of Keepalive of TCP communication. The setting range is 1 to 86400. This option is set to "_tcp_keepalive_interval".
T4_CFG_TCP_KEEPAIVE_COUNT Note: Default value = 10	Sets the maximum number of keepalive packet transmissions. The setting range is 0 to 0xffffffff. This option is set to "_tcp_keepalive_count".
T4_CFG_UDP_CEPIDx_CHANNEL (x: 1-6)	Set the LAN socket number that you want to link to each UDP endpoint. The range is 0 to (T4_CFG_SYSTEM_CHANNEL_NUMBER - 1). This option is set to "udp_ccep[].lan_port_number".
T4_CFG_UDP_CEPIDx_PORT_NUMBER (x: 1-6)	Set the port number of UDP reception port. This option is set to "udp_ccep[].myaddr.portno".
T4_CFG_UDP_CEPIDx_CALLBACK_FUNCTION_USE (x: 1-6)	Sets whether to use the callback routine for each UDP endpoint. Set 0 for endpoints to be executed by blocking calls. Set 1 for endpoints to be executed with non-blocking calls. This option is set to "udp_ccep[].callback".
T4_CFG_UDP_CEPIDx_CALLBACK_FUNCTION_NAME_TMP (x: 1-6)	(Setting is unnecessary when T4_CFG_UDP_CEPIDx_CALLBACK_FUNCTION_USE is 0) Register the function name of the UDP callback. This option is set to "udp_ccep[].callback".
T4_CFG_UDP_MULTICAST_TTL Note: Default value = 1	Sets TTL of IP datagram to be sent to multicast. This option is set to "_multi_ttl".
T4_CFG_UDP_BEHAVIOR_OF_RECEIVED_ZERO_CHECKSUM Note: Default value = 0	Select the behavior when UDP zero checksum is received. 0: Receive as a normal packet 1: Discard as abnormal packet This option is set to "_udp_enable_zerochecksum".
T4_CFG_IP_ARP_CACHE_TABLE_COUNT Note: Default value = 3	Set the number of cache entries for ARP. This option is set to "_ip_tblcnt".

2.8 Code Size

The sizes of ROM, RAM and maximum stack usage associated with this module are listed below.

The ROM (code and constants) and RAM (global data) sizes are determined by the build-time configuration options described in 2.7 Configuration Overview.

The values in the table below are confirmed under the following conditions.

Module Revision: r_t4_rx Rev.2.10

Compiler Version: Renesas Electronics C/C++ Compiler Package for RX Family V3.02.00

(The option of "lang = c99" is added to the default settings of the integrated development environment.)

GCC for Renesas RX 4.8.4.201803

(The option of "-std=gnu99" is added to the default settings of the integrated development environment.)

IAR C/C++ Compiler for Renesas RX version 4.14.1

(The default settings of the integrated development environment.)

ROM, RAM and Stack Code Sizes				
Device	Category	Memory Used		
		Renesas Compiler	GCC	IAR Compiler
RX65N (without TSIP)	ROM	25,435 bytes	23,345 bytes	24,827 bytes
	RAM	227 bytes	227 bytes	213 bytes
	STACK	436 bytes	-	764 bytes

Use the "CallWalker" to check your system stack size. Because the stack size is changed in case "Changed compile option", "Changed sample driver code", and whether the MCU equips TSIP feature, etc.

2.9 Parameters

Please refer to the user's manual for the parameter structure used by the API function of this module. The structure is located in r_t4_itcpip.h as are the prototype declarations of API functions.

2.10 Return Values

Please refer to the user's manual for the return values of API functions, this enumeration is located in r_t4_itcpip.h as are the prototype declarations of API functions.

2.11 Callback Function

Please refer to the user's manual for callback function,

2.12 Adding the FIT Module to Your Project

This module must be added to each project in which it is used. Renesas recommends the method using the Smart Configurator described in (1) or (3) or (5) below. However, the Smart Configurator only supports some RX devices. Please use the methods of (2) or (4) for RX devices that are not supported by the Smart Configurator.

- (1) Adding the FIT module to your project using the Smart Configurator in e² studio
By using the Smart Configurator in e² studio, the FIT module is automatically added to your project. Refer to “Renesas e² studio Smart Configurator User Guide (R20AN0451)” for details.
- (2) Adding the FIT module to your project using the FIT Configurator in e² studio
By using the FIT Configurator in e² studio, the FIT module is automatically added to your project. Refer to “Adding Firmware Integration Technology Modules to Projects (R01AN1723)” for details.
- (3) Adding the FIT module to your project using the Smart Configurator in CS+
By using the Smart Configurator Standalone version in CS+, the FIT module is automatically added to your project. Refer to “Renesas e² studio Smart Configurator User Guide (R20AN0451)” for details.
- (4) Adding the FIT module to your project in CS+
In CS+, please manually add the FIT module to your project. Refer to “Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)” for details.

2.13 "for", "while" and "do while" statements

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

The following shows example of description.

while statement example :

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

for statement example :

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

do while statement example :

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

3. T4 Library Information

This product corresponds to RX family.

3.1 Compile option

Library file is built with default compile option.

3.1.1 CC-RX

- compile option (little endian)
-isa=rxv1 -nofpu -lang=c99 -output=obj -obj_path=DefaultBuild -nologo
- compile option (The case of Little endian with debug information)
-isa=rxv1 -nofpu -lang=c99 -output=obj -obj_path=Debug -debug -optimize=0 -nologo
- compile option (big endian)
-isa=rxv1 -nofpu -endian=big -lang=c99 -output=obj -obj_path=DefaultBuild -nologo
- compile option (The case of Big endian with debug information)
-isa=rxv1 -nofpu -endian=big -lang=c99 -output=obj -obj_path=Debug -debug -optimize=0 -nologo

3.1.2 GCC

- compile option (little endian)
-Os -ffunction-sections -fdata-sections -Wstack-usage=100 -misa=v1 -mlittle-endian-data
-std=gnu99
- compile option (big endian)
-Os -ffunction-sections -fdata-sections -Wstack-usage=100 -misa=v1 -mbig-endian-data
-std=gnu99

3.1.3 IAR

- compile option (little endian)
--suppress_core_attribute -e -Oh --no_cross_call
--dlib_config "C:\Program Files (x86)\IAR Systems\Embedded Workbench 8.2\rx\LIB\dlrxflln.h"
--double=32 --data_model=f --endian l -D NDEBUG --core rxv1 --int=32 --fpu=none --align_func=1
- compile option (big endian)
--suppress_core_attribute -e -Oh --no_cross_call
--dlib_config "C:\Program Files (x86)\IAR Systems\Embedded Workbench 8.2\rx\LIB\dlrxflln.h"
--double=32 --data_model=f --endian b -D NDEBUG --core rxv1 --int=32 --fpu=none --align_func=1

3.2 Version information

T4 has version information as string data in `R_t4_version` variable library member. `R_t4_version` variable is defined in the `r_t4_itcpip.h`. T4 Library information is as below.

```
extern const mw_version_t R_t4_version;
```

3.2.1 CC-RX

- RXV1 core(little endian) Library file (For the Ethernet):

```
compiler = 0x03020000  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for RXV1 LITTLE endian.(Mar 10  
2021, 14:44:15) "
```

- RXV1 core (little endian) with debug information Library file (For the Ethernet):

```
compiler = 0x03020000  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for RXV1 LITTLE endian.(Mar 10  
2021, 14:46:54) "
```

- RXV1 core(big endian) Library file (For the Ethernet):

```
compiler = 0x03020000  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for RXV1 BIG endian.(Mar 10  
2021, 14:44:29) "
```

- RXV1 core (big endian) with debug information Library file (For the Ethernet):

```
compiler = 0x03020000  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for RXV1 BIG endian.(Mar 10  
2021, 14:47:06) "
```

3.2.2 GCC

- RXV1 core(little endian) Library file (For the Ethernet):

```
compiler = 0x00040804  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for GCC RX LITTLE endian.(Mar 10  
2021, 16:33:31) "
```

- RXV1 core(big endian) Library file (For the Ethernet):

```
compiler = 0x00040804  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for GCC RX BIG endian.(Mar 10  
2021, 16:33:43) "
```

3.2.3 IAR

- RXV1 core(little endian) Library file (For the Ethernet):

```
compiler = 0x0000019E  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for IAR RXV1 LITTLE endian.(Mar  
10 2021, 16:41:44) "
```

- RXV1 core(big endian) Library file (For the Ethernet):

```
compiler = 0x0000019E  
library = "M3S-T4-Tiny(Ethernet) version 2.09 for IAR RXV1 BIG endian.(Mar 10  
2021, 16:42:00) "
```

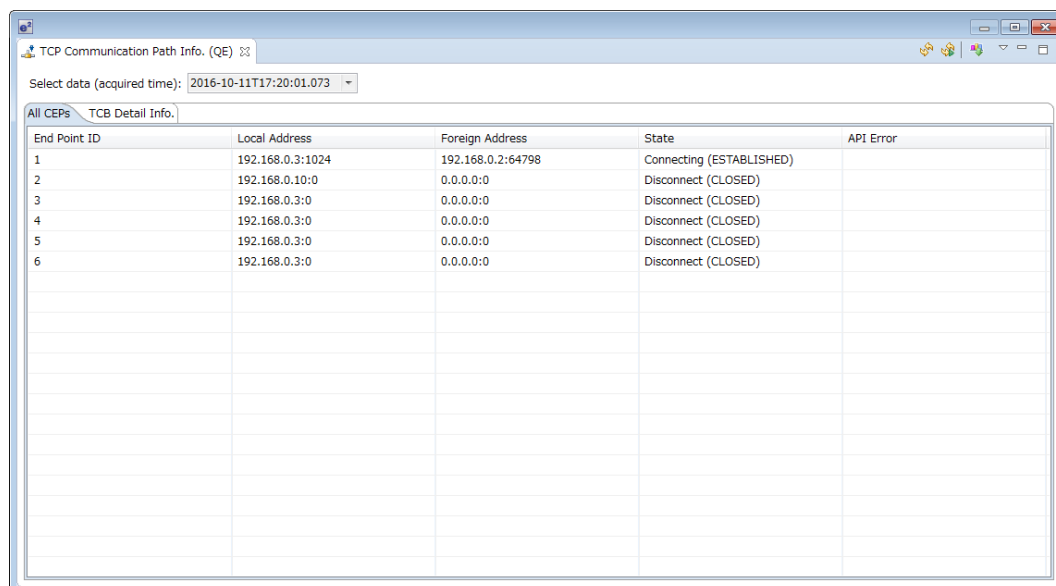
4. QE for TCP/IP

QE for TCP/IP provides the T4 debug information as e2 studio plug-in. This tool can display following information at realtime. This means debug efficiency will be improved.

- Which state is in TCP state machine?
- Which API is executed?
- Which error is occurred? etc.

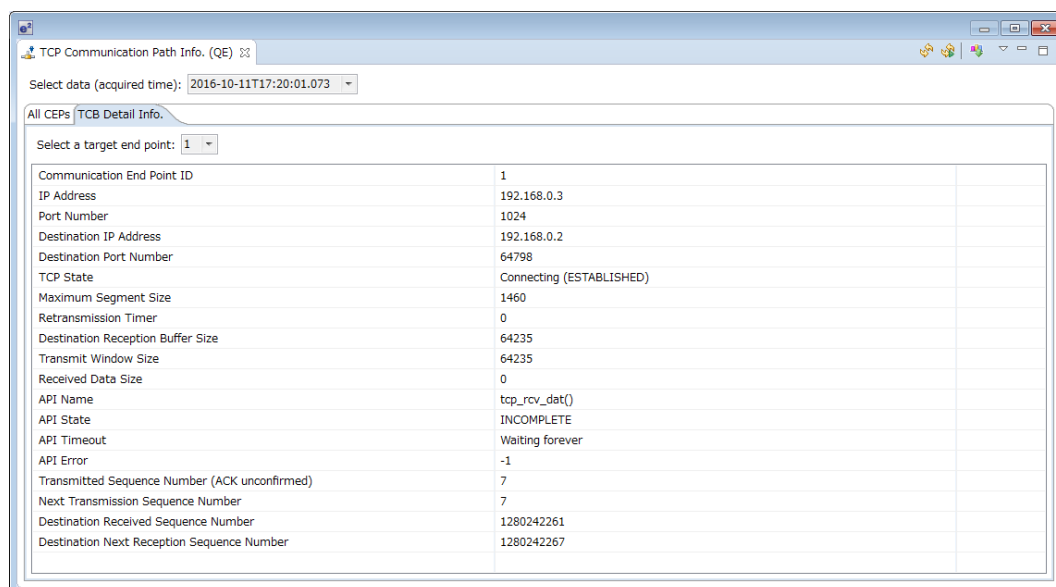
QE for TCP/IP can be downloaded from following URL. It is possible to use to plug-in this tool to e2 studio.

<https://www.renesas.com/us/en/products/software-tools/tools/solution-toolkit/qe-qe-for-tcp-ip.html>



End Point ID	Local Address	Foreign Address	State	API Error
1	192.168.0.3:1024	192.168.0.2:64798	Connecting (ESTABLISHED)	
2	192.168.0.10:0	0.0.0.0:0	Disconnect (CLOSED)	
3	192.168.0.3:0	0.0.0.0:0	Disconnect (CLOSED)	
4	192.168.0.3:0	0.0.0.0:0	Disconnect (CLOSED)	
5	192.168.0.3:0	0.0.0.0:0	Disconnect (CLOSED)	
6	192.168.0.3:0	0.0.0.0:0	Disconnect (CLOSED)	

Figure 4.1 Example of All TCP Endpoint List



Communication End Point ID	1
IP Address	192.168.0.3
Port Number	1024
Destination IP Address	192.168.0.2
Destination Port Number	64798
TCP State	Connecting (ESTABLISHED)
Maximum Segment Size	1460
Retransmission Timer	0
Destination Reception Buffer Size	64235
Transmit Window Size	64235
Received Data Size	0
API Name	tcp_rcv_dat()
API State	INCOMPLETE
API Timeout	Waiting forever
API Error	-1
Transmitted Sequence Number (ACK unconfirmed)	7
Next Transmission Sequence Number	7
Destination Received Sequence Number	1280242261
Destination Next Reception Sequence Number	1280242267

Figure 4.2 Example of Detail Information about a TCP Endpoint

5. Sample Program

T4 sample program prepares five 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.

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

5.1.1 Flow of the Echo Back Server Function

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

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

5.2.1 Flow of the Echo Back Server Function

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

5.2.2 Flow of the Callback Function

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

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

5.3.1 Flow of the Echo Back Server Function

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

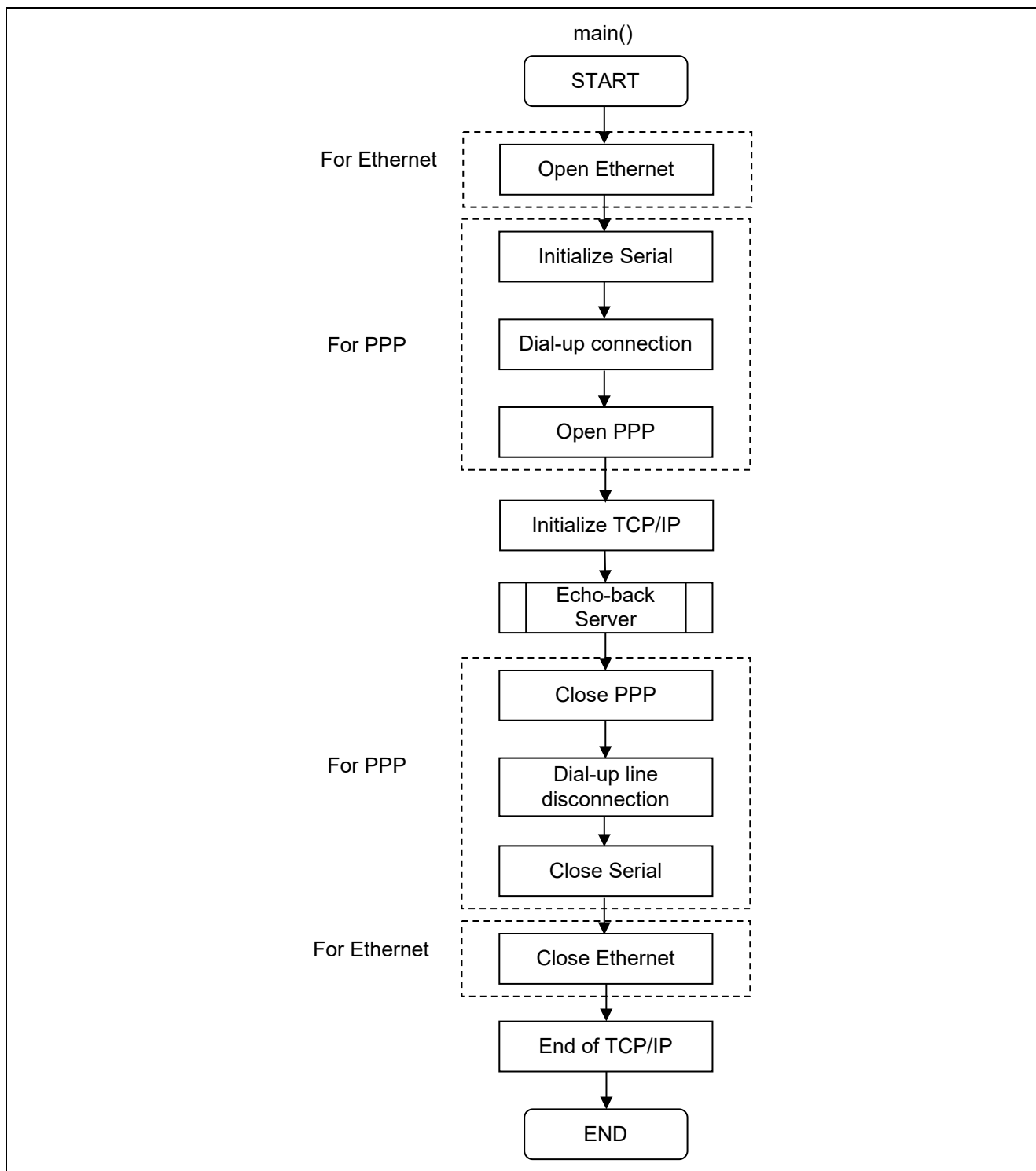
5.4 Flow of the UDP Echo Back Server Function (for Non-blocking Call)

5.4.1 Flow of the Echo Back Server Function

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

5.4.2 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**

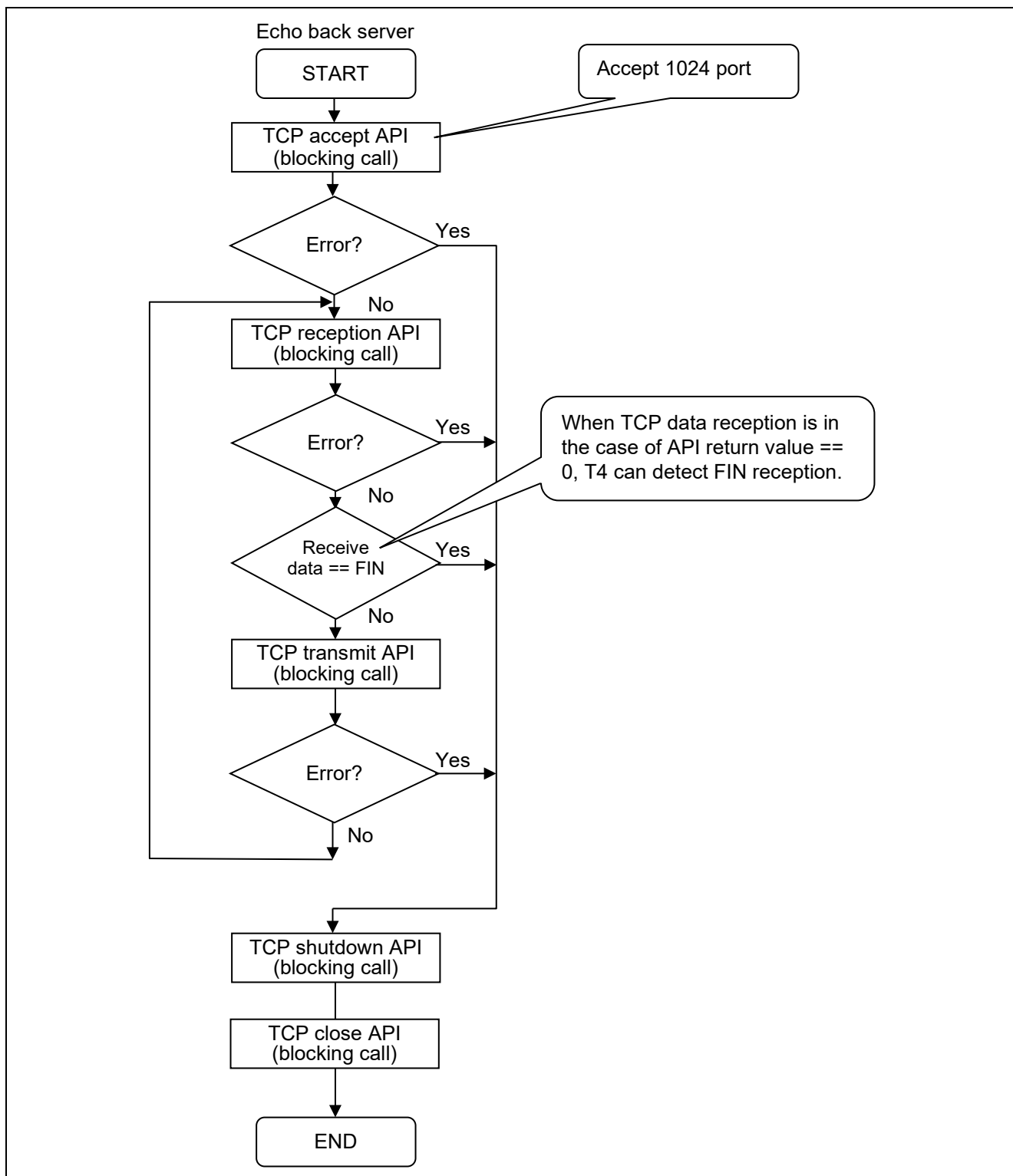


Figure 5.2 Flow of the TCP Echo Back Server Function (for Blocking Call)

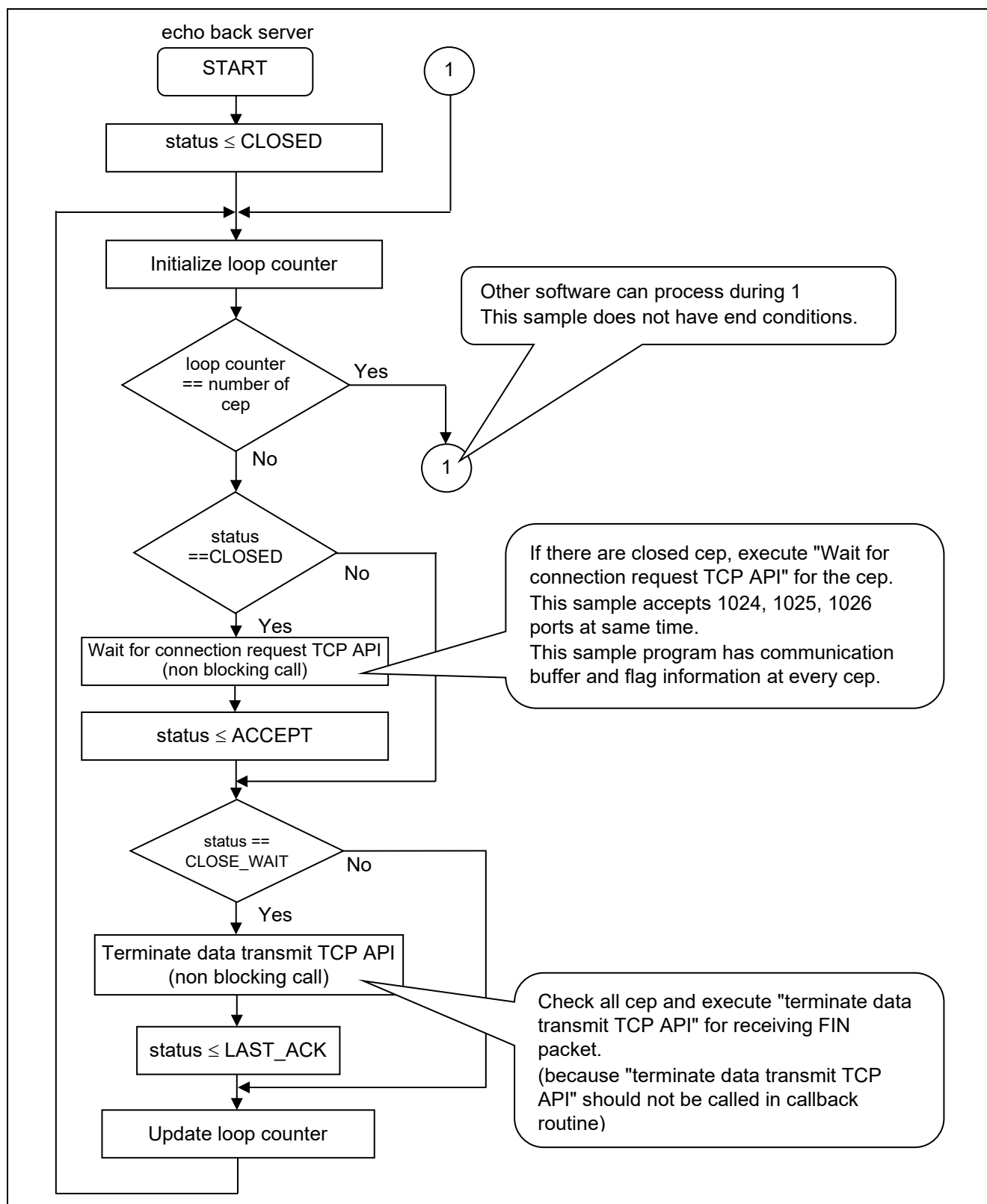


Figure 5.3 Flow of the TCP Echo Back Server Function (for Non-Blocking Call)

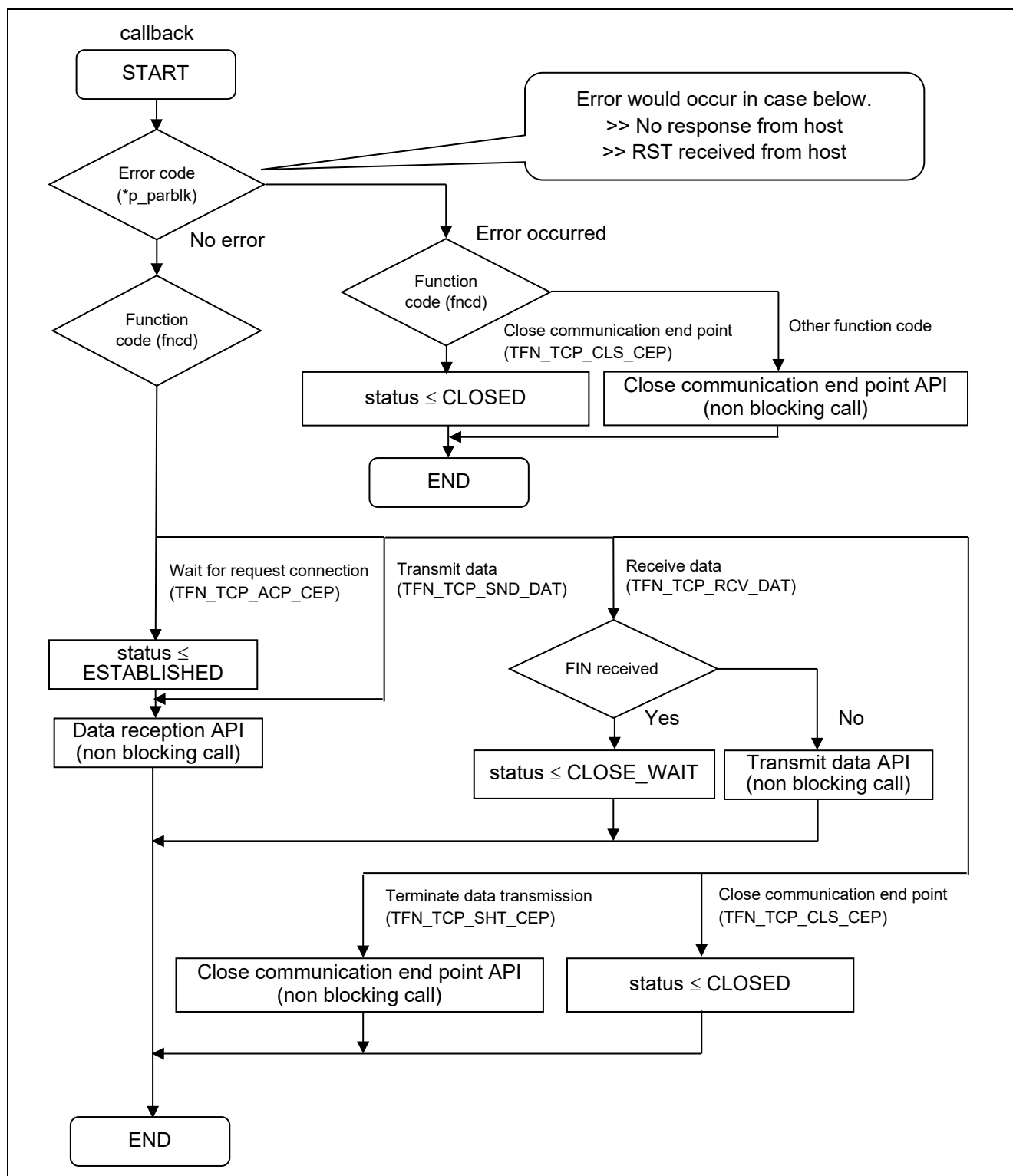


Figure 5.4 Flow of the TCP Callback Function (for Non-Blocking Call)

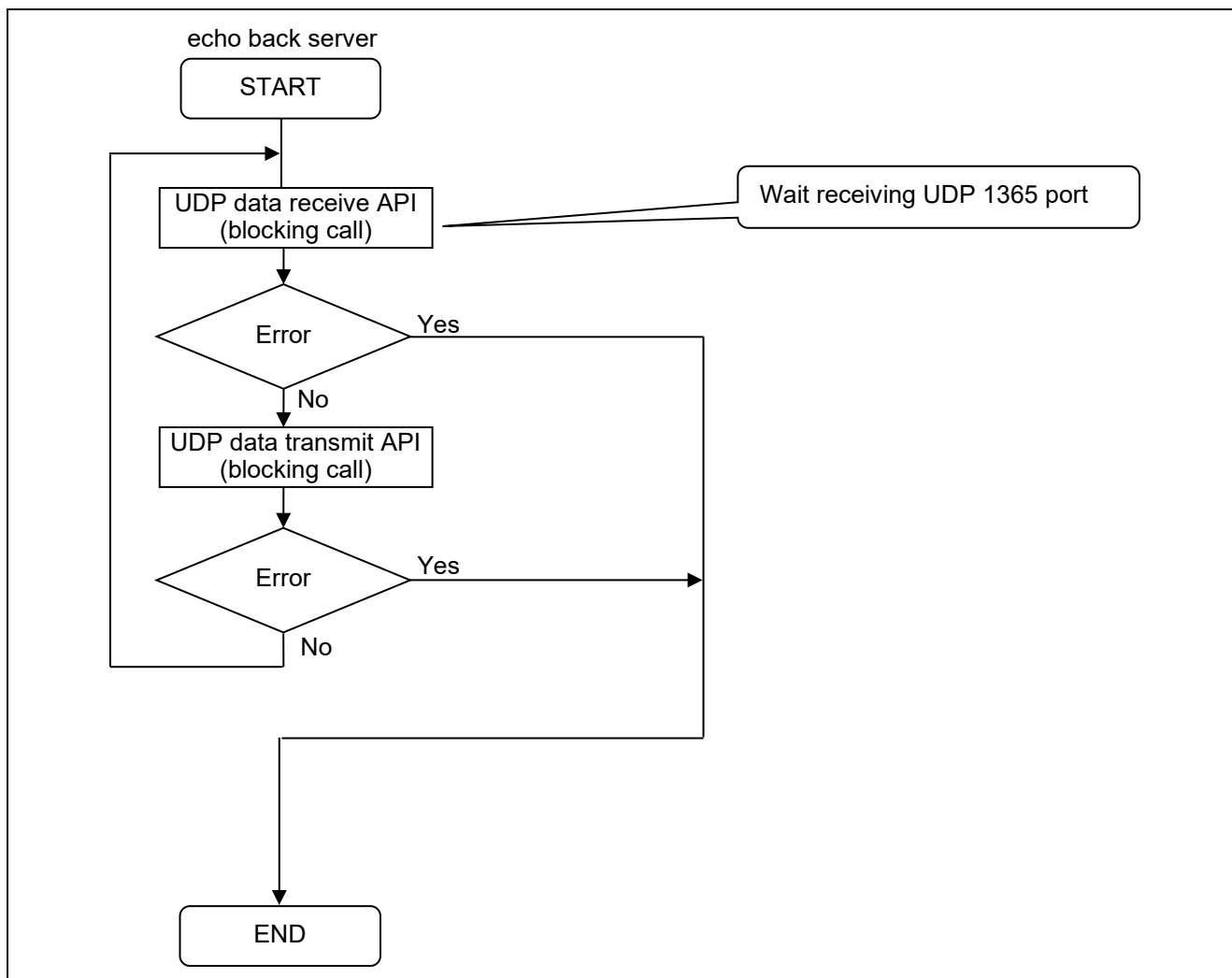


Figure 5.5 Flow of the UDP Echo Back Server Function (for Blocking Call)

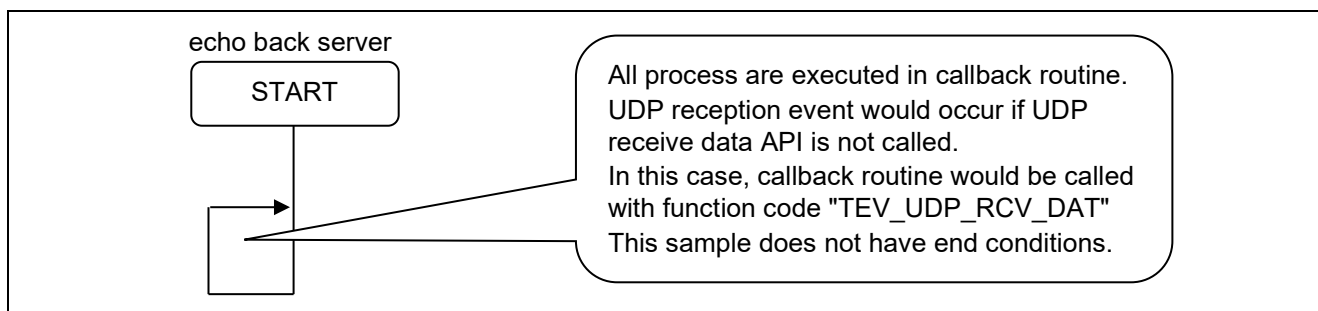


Figure 5.6 Flow of the UDP Echo Back Server Function (for Non-Blocking Call)

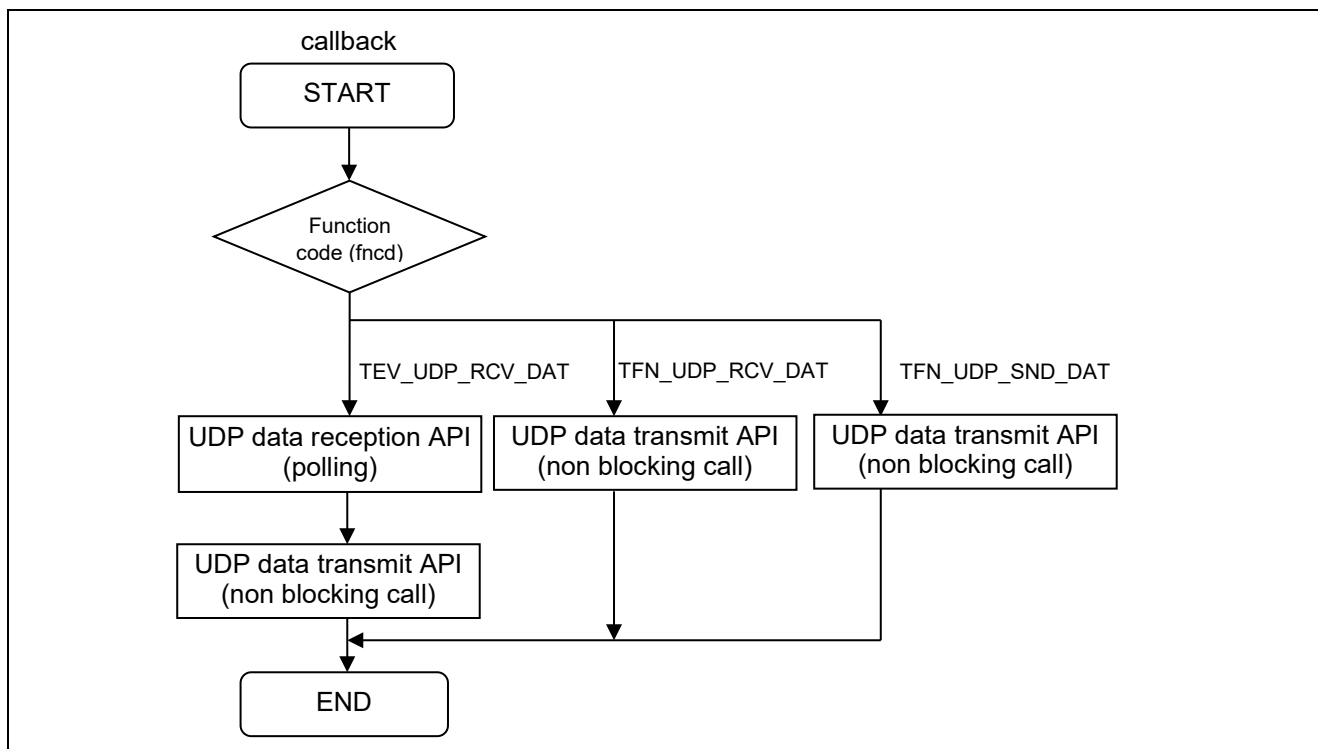


Figure 5.7 Flow of the UDP Callback Function (for Non-Blocking Call)

5.5 Component Setting

When creating projects with Smart Configurator, users use software components.

Table 5.1 shows initial values and setting examples.

In the case of TCP, register the callback function in [TCP CEPIDx callback function name].

For UDP, register the callback function in [UDP CEPIDy callback function name].

- x is an integer from 1 to 6. y is an integer from 1 to 6.

Table 5.1 Component Setting

Configurations	Initial setting	Setting Example	
		UDP Non-blocking	TCP Non-blocking
Channel number your system has.	1	2	2
Enable/Disable DHCP function.	1	1	1
IP address for ch0, when DHCP disable.	192,168,0,3	-	-
Subnet mask for ch0, when DHCP disable.	255,255,255,0	-	-
Gateway address for ch0, when DHCP disable.	0,0,0,0	-	-
IP address for ch1,when DHCP disable.	192,168,0,10	-	-
Subnet mask for ch1, when DHCP disable.	255,255,255,0	-	-
Gateway address for ch1, when DHCP disable.	0,0,0,0	-	-
Ether ch0 MAC address.	0x74,0x90,0x50 ,0x10,0xFE,0x77	0x74,0x90,0x50 ,0x10,0xFE,0x77	0x74,0x90,0x50 ,0x10,0xFE,0x77
Ether ch1 MAC address.	0x74,0x90,0x50 ,0x00,0x79,0x79	-	-
SYSTEM callback function use.	1	1	1
SYSTEM callback function name.	system_callback	system_callback	system_callback
TCP REPID1 port number.	1024	1024	1024
TCP REPID2 port number.	1025	1025	1025
TCP REPID3 port number.	1026	1026	1026
TCP REPID4 port number.	1027	1027	1027
TCP CEPID1 channel number.	0	0	0
TCP CEPID1 receive window size.	1460	1460	1460
TCP CEPID1 callback function use.	0	0	0
TCP CEPID1 callback function name.	0	-	-
TCP CEPID1 Keep-alive enable.	0	0	0
TCP CEPID2 channel number.	0	0	0
TCP CEPID2 receive window size.	1460	1460	1460
TCP CEPID2 callback function use.	0	0	0
TCP CEPID2 callback function name.	0	-	-
TCP CEPID2 Keep-alive enable.	0	0	0
TCP CEPID3 channel number.	0	0	0
TCP CEPID3 receive window size.	1460	1460	1460
TCP CEPID3 callback function use.	0	0	1
TCP CEPID3 callback function name.	0	-	tcp_nonblocking_callback
TCP CEPID3 Keep-alive enable.	0	0	0
TCP CEPID4 channel number.	0	0	0
TCP CEPID4 receive window size.	1460	1460	1460
TCP CEPID4 callback function use.	0	0	1
TCP CEPID4 callback function name.	0	-	tcp_nonblocking_callback
TCP CEPID4 Keep-alive enable.	0	0	0
TCP CEPID5 channel number.	0	0	1
TCP CEPID5 receive window size.	1460	1460	1460
TCP CEPID5 callback function use.	0	0	1

TCP CEPID5 callback function name.	0	-	tcp_nonblocking_callback
TCP CEPID5 Keep-alive enable.	0	0	0
TCP CEPID6 channel number.	0	0	1
TCP CEPID6 receive window size.	1460	1460	1460
TCP CEPID6 callback function use.	0	0	1
TCP CEPID6 callback function name.	0	-	tcp_nonblocking_callback
TCP CEPID6 Keep-alive enable.	0	0	0
TCP max segment size.	1460	1460	1460
TCP max segment life time.	60	60	60
TCP max timeout period time.	600	600	600
TCP divide the sending packet.	1	1	1
UDP CEPID1 channel number.	0	0	0
UDP CEPID1 port number.	1365	1365	1365
UDP CEPID1 callback function use	0	0	0
UDP CEPID1 callback name.	0	-	-
UDP CEPID2 channel number.	0	0	0
UDP CEPID2 port number.	1366	1366	1366
UDP CEPID2 callback function use.	0	0	0
UDP CEPID2 callback name.	0	-	-
UDP CEPID3 channel number.	0	0	0
UDP CEPID3 port number.	1367	1367	1367
UDP CEPID3 callback function use.	0	1	0
UDP CEPID3 callback name.	0	udp_nonblocking_callback	-
UDP CEPID4 channel number.	0	1	0
UDP CEPID4 port number.	1368	1368	1368
UDP CEPID4 callback function use.	0	1	0
UDP CEPID4 callback name.	0	udp_nonblocking_callback	-
UDP CEPID5 channel number.	0	1	0
UDP CEPID5 port number.	1369	1369	1369
UDP CEPID5 callback function use.	0	1	0
UDP CEPID5 callback name.	0	udp_nonblocking_callback	-
UDP CEPID6 channel number.	0	1	0
UDP CEPID6 port number.	1370	1370	1370
UDP CEPID6 callback function use	0	1	0
UDP CEPID6 callback name.	0	udp_nonblocking_callback	-
UDP multicast TTL value.	1	1	1
UDP behavior of received zero checksum.	0	0	0
IP ARP cache table.	3	3	3

5.6 Environment for RX65N sample program

Sample program projects for RX65N are built using FIT module plug-in function included in e² studio.

5.6.1 Using FIT Modules

This sample program use following FIT Modules.

- RX Family TCP/IP for Embedded system M3S-T4-Tiny Introduction Guide Firmware Integration Technology (R20AN0051)
RX Family Interface conversion module for Ethernet Driver and Embedded system M3S-T4-Tiny Firmware Integration Technology (R20AN0311)
- RX Family Ethernet Module Using Firmware Integration Technology (R01AN2009)
- RX Family System Timer Module Firmware Integration Technology (R20AN0431)
- RX Family CMT Module Using Firmware Integration Technology (R01AN1856)
- RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685)

Please refer to the document included each FIT modules when user needs more details information about each FIT modules.

5.6.2 Software Structure

Show software structure and FIT modules structure of sample program.

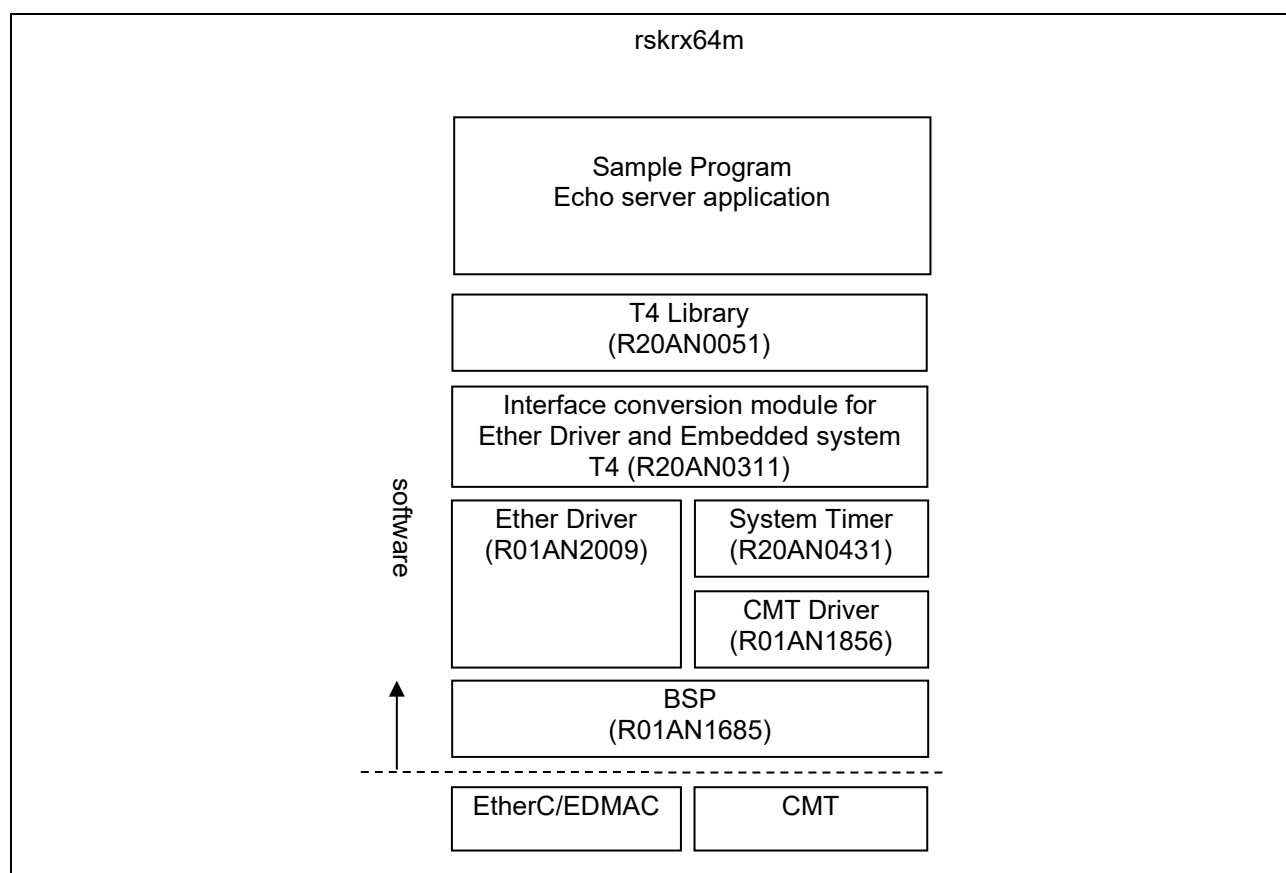


Figure 5.8 FIT modules structure of sample program

5.6.3 Scheme of link the Library file

The followings are how to link T4 library files.

CC-RX:

When the FIT module of the T4 library is read from the smart configurator, the link setting of the T4 library file is performed automatically.

GCC:

Settings for T4 Library in manually after the T4 FIT module.

- Right click on the project name in "Project Explorer" window, and click "Properties".
- In the status, selected "settings" in the "Property" dialog in the left side tree, after selecting "Linker" -> "Source", then click the "Add" icon of "Additional Input files".
- Click the "Workspace" in the "Add File path" dialog.
- Click "OK" in the "Select File" dialog with selecting `r_t4_rx -> lib -> gcc -> libT4_Library_ether_gcc_rxv1_little.a`. Then click "OK", for back the "Property" dialog.
- In the status, selected "settings" in the "Property" dialog in the left side tree, click "Linker" -> "Archive". If T4 libraries (regardless of compiler) are added in "User defined archive (library) files", click "delete" icon to delete them. Note that you must need this operation every time after executing code generation by the Smart Configurator V.20.10.0, because the Smart Configurator automatically add to the T4 library in "User defined archive (library) files".
- Click "OK" in the "Property" dialog.

IAR

Settings for T4 Library in manually after the T4 FIT module.

- Right click on the project name in "Project Explorer" window, and click "Options...".
- In the status, selected "Linker" in the "Category" dialog in the left side tree, After selecting "Library" tab, and add `r_t4_rx -> lib -> gcc -> T4_Library_ether_iar_rxv1_little.a` in -> "Additional libraries" text box.
- Click "OK"

5.6.4 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 RX64M).

- 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 "RX64M" -> "R5F564MLDxFC" 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.

5.7 Confirm sample program

How to confirm Ethernet sample program

5.7.1 Environment

(1) Connecting the hardware

Setup Hardware connections

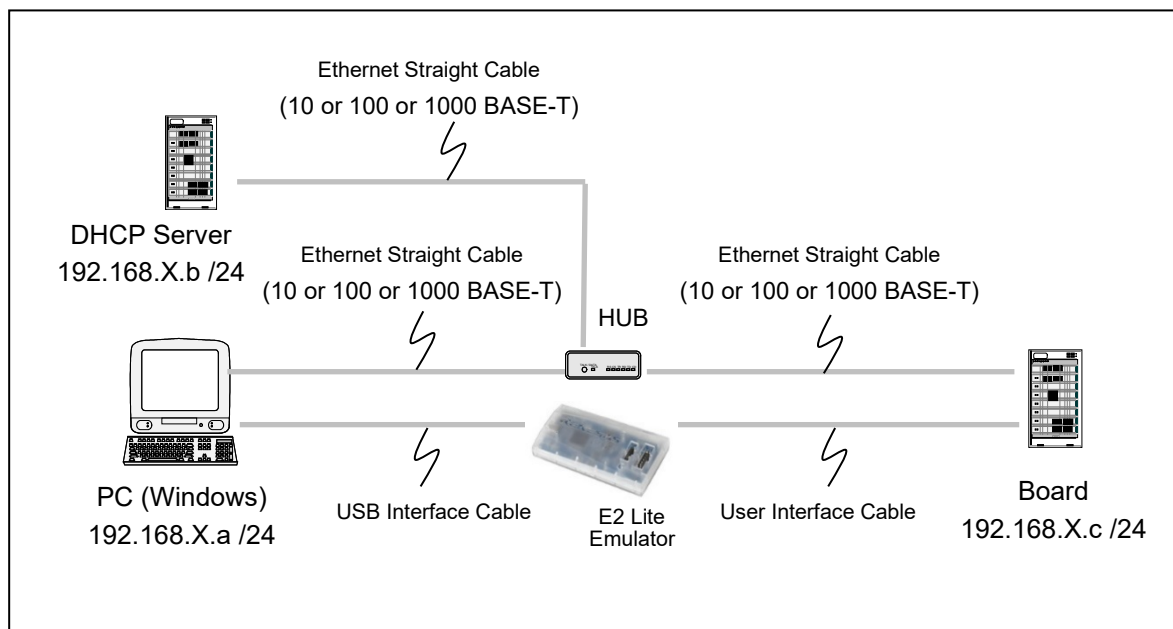


Figure 5.9 Ethernet sample program environment

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.

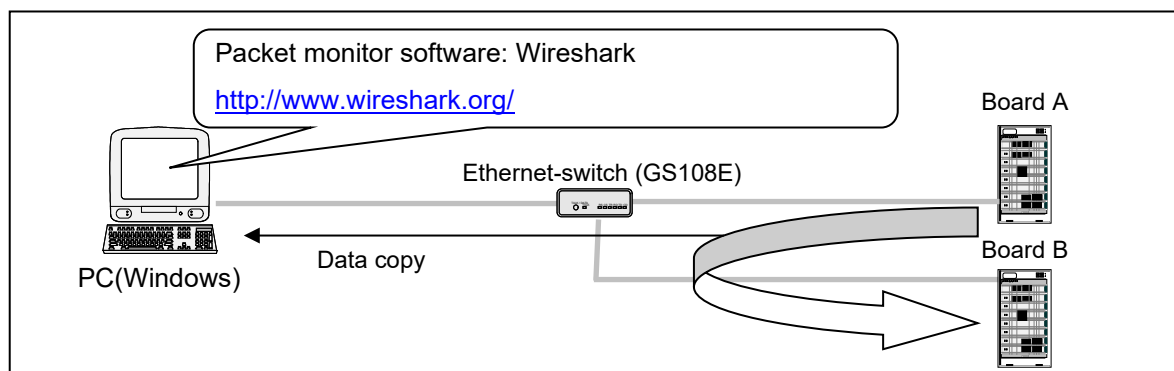


Figure 5.10 Ethernet sample program confirmation environment

(2) PC setting

Windows 10:

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.

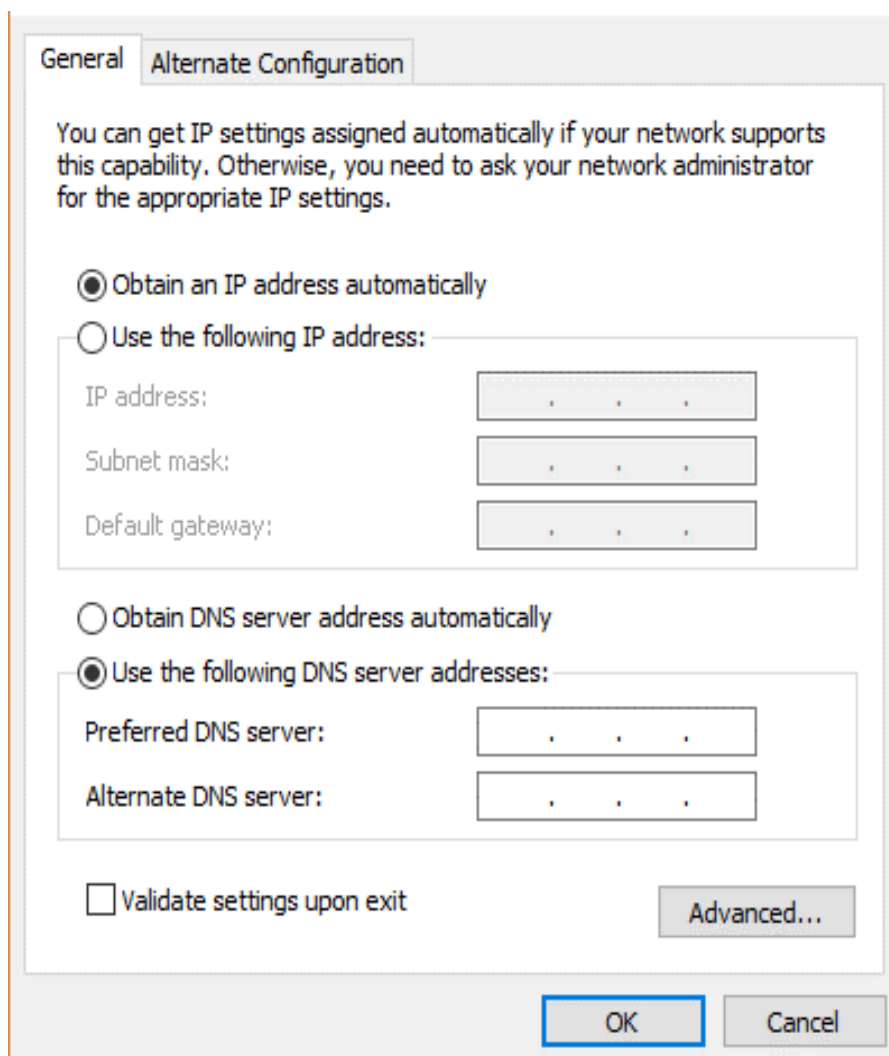



Figure 5.11 Internet Protocol Version 4 (TCP/IPv4) Property

After setting, please push OK button.

(3) Start Sample program (sample folder)

- Start e²studio 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.

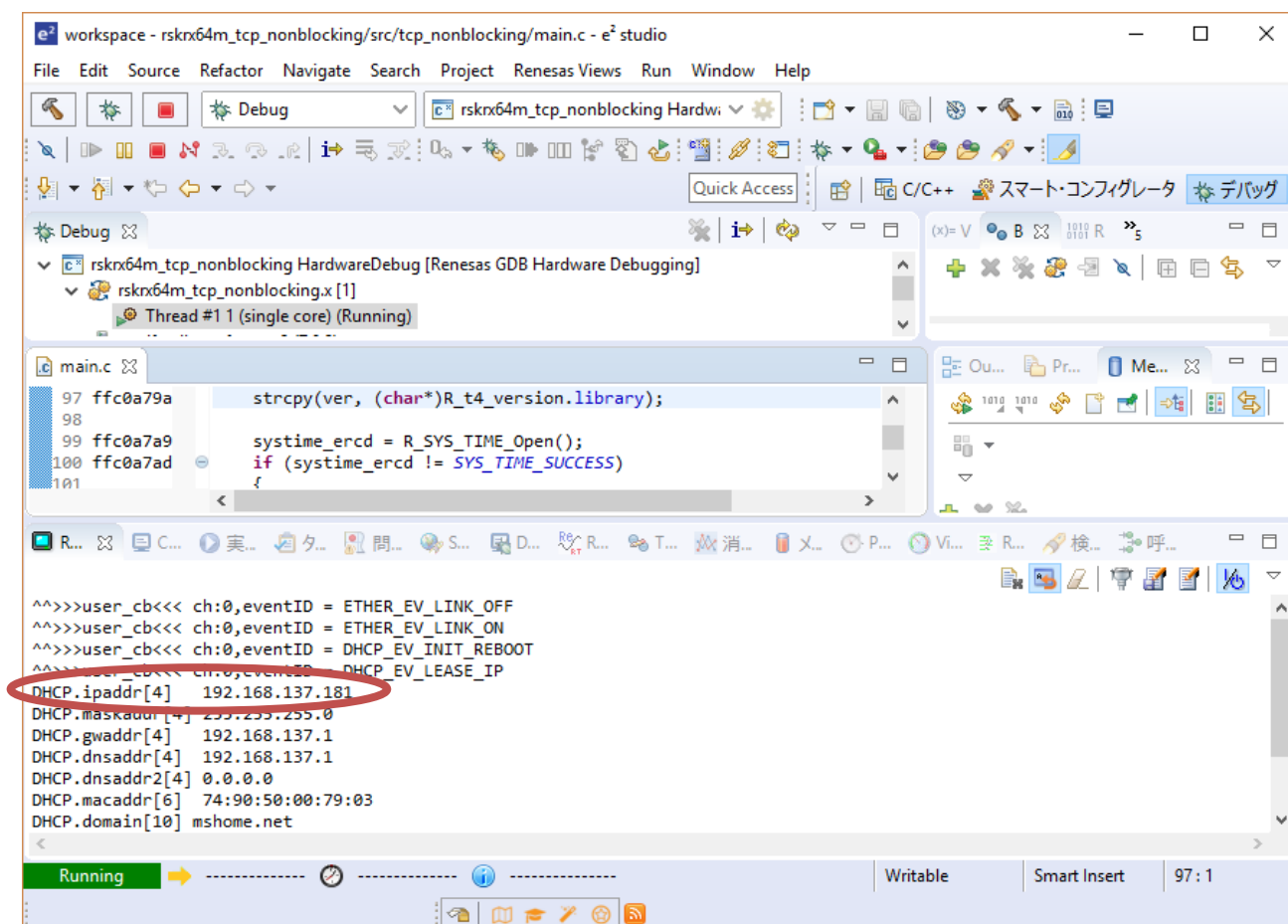


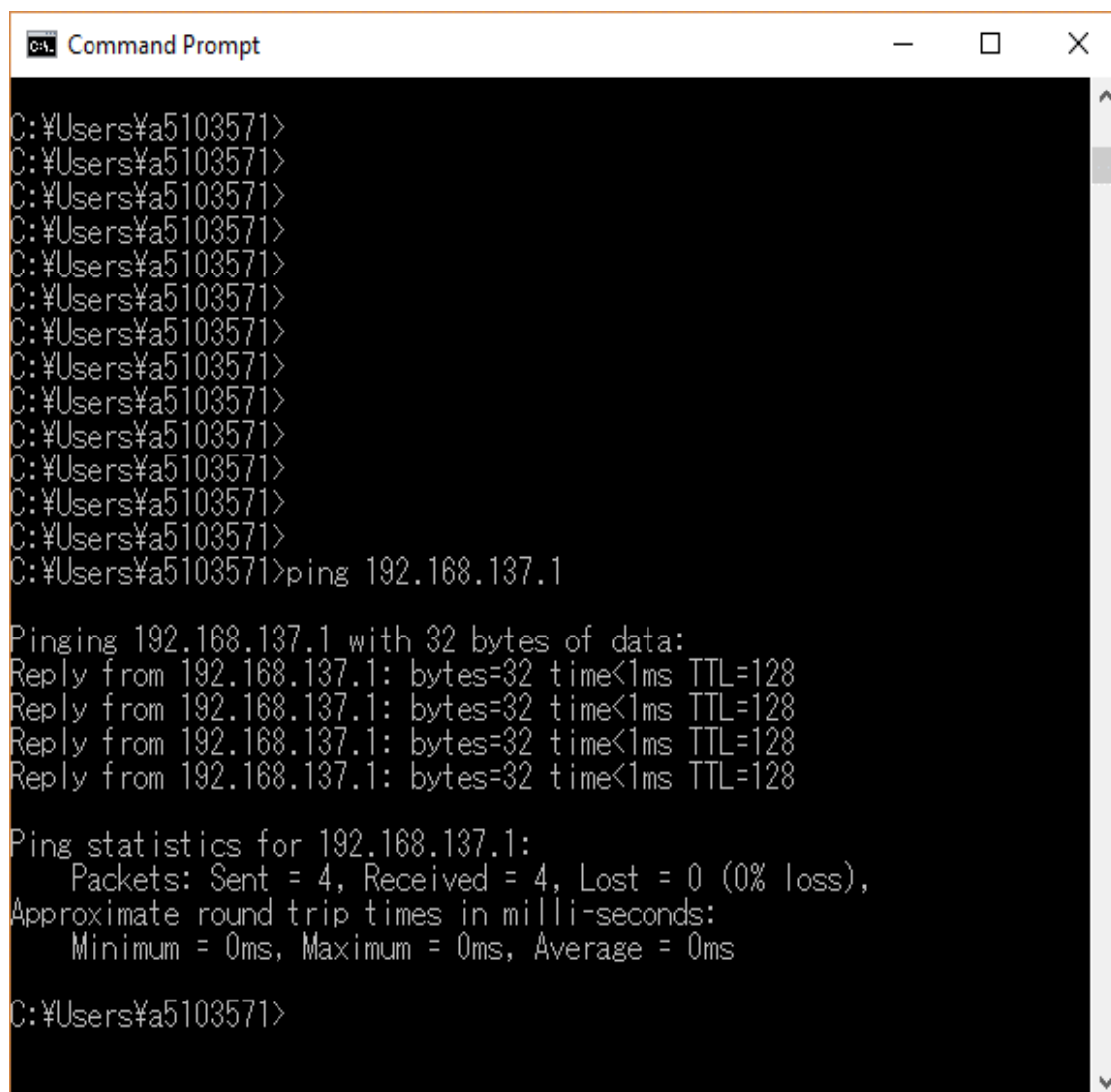
Figure 5.12 Example about Display of IP address

Figure Example about display of IP Address shows IP address is 192.168.137.181 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.



```
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>
C:\Users\Ya5103571>ping 192.168.137.1

Pinging 192.168.137.1 with 32 bytes of data:
Reply from 192.168.137.1: bytes=32 time<1ms TTL=128
Reply from 192.168.137.1: bytes=32 time<1ms TTL=128
Reply from 192.168.137.1: bytes=32 time<1ms TTL=128
Reply from 192.168.137.1: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.137.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\Ya5103571>
```

Figure 5.13 Example about Execution of ping

5.7.2 Confirm TCP connection

Execute telnet in command prompt

(1) Enable telnet(Windows10 only)

- Start -> Windows System Tool -> Control Panel -> Program and Function

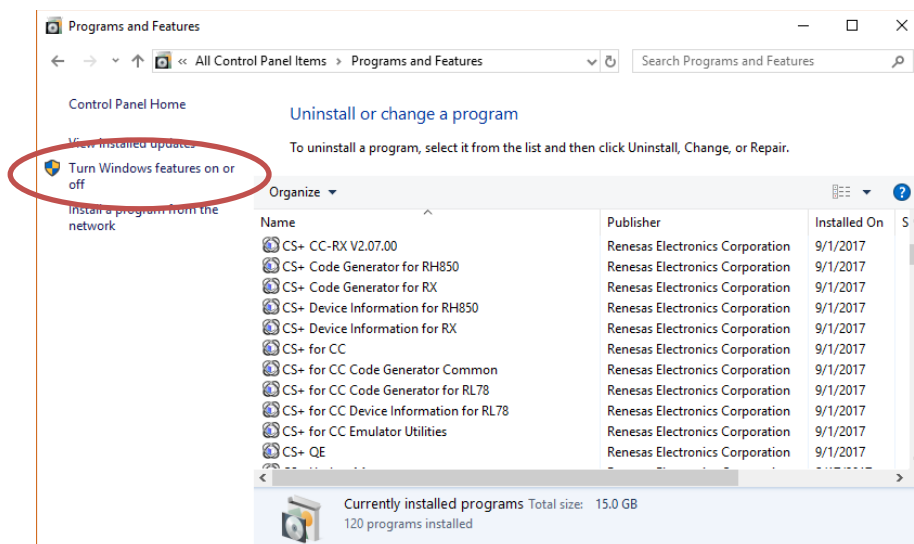


Figure 5.14 Program and Function

- Please check Telnet client

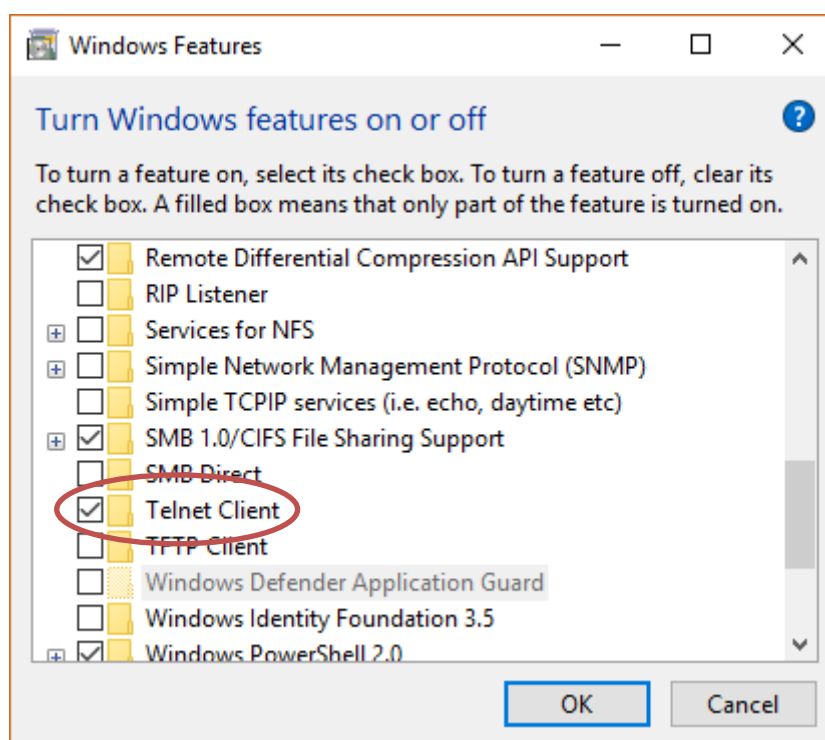


Figure 5.15 Telnet Client

(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.

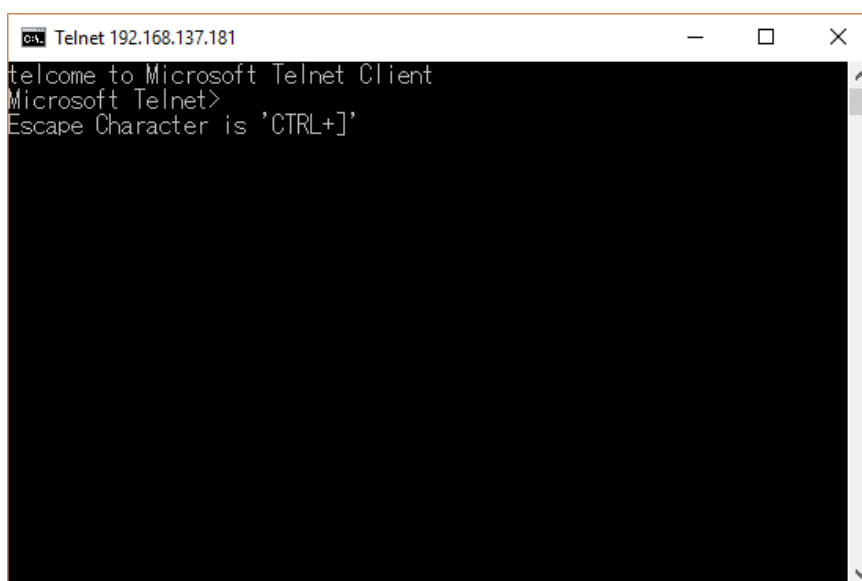


Figure 5.16 Disconnect

5.7.3 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.

6. Notes

6.1 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()`, `tcp_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`.

6.2 Smart Configurator

You can customize 6 endpoints on the T4 software component setting screen.

To change the number of endpoints themselves, you need to edit `r_t4_rx_config.h` and `config_tcpudp.c`.

6.3 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 it's necessary according to the system.
- (3) The sample programs are generated for RX65N without TSIP, but you can customize them for RX65N equipped TSIP. There are two methods of customize:
 - By using the Smart Configurator
 - a. Change board and device type for RX65N equipped TSIP in the board tab of the Smart Configurator.
 - b. Add the TSIP FIT module in the component tab of the Smart Configurator.
 - c. Execute code generation by the Smart Configurator.
 - By editing manually ^(Note 1)
 - a. Change value of "BSP_CFG_MCU_PART_ENCRYPTION_INCLUDED" macro ^(Note 2) to "true".
 - b. Add TSIP FIT module, which is downloaded from Web page, in the e² studio project manually.

Note 1: When executing code generation by the Smart Configurator again, the range of edit may be reset.

Note 2: For more information about a target macro, please refer to 64 ~ 76 lines of `t4_drvr.c` in `r_t4_driver_rx Rev.1.09`.

7. Appendices

7.1 Confirmed Operation Environment

This section describes confirmed operation environment for the T4 FIT module.

Table 7.1 Confirmed Operation Environment (Rev. 2.09)

Item	Contents
Integrated development environment	Renesas Electronics e ² studio Version 7.3.0 Renesas Electronics CS+ V8.01.00 (T4 Library make environment) IAR Embedded Workbench for Renesas RX 4.11.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
	GCC for Renesas RX 4.8.4.201801 Compiler option: The following option is added to the default settings of the integrated development environment. -std=gnu99
	IAR C/C++ Compiler for Renesas RX version 4.11.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.2.09
Software tool (optional)	QE for TCP/IP V1.0.1
Board used	Renesas Starter Kit+ for RX65N-2MB (RTK50565Nxxxxxx) Renesas Starter Kit+ for RX64M (R0K50564Mxxxxxx)

Table 7.2 Confirmed Operation Environment (Rev. 2.10)

Item	Contents
Integrated development environment	Renesas Electronics e ² studio V20.10.0 Renesas Electronics CS+ V8.04.00 (T4 Library make environment) IAR Embedded Workbench for Renesas RX 4.14.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.02.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = c99
	GCC for Renesas RX 4.8.4.201803 Compiler option: The following option is added to the default settings of the integrated development environment. -std=gnu99
	IAR C/C++ Compiler for Renesas RX version 4.14.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.2.10
Software tool (optional)	QE for TCP/IP V1.0.1
Board used	Renesas Starter Kit+ for RX65N-2MB (RTK50565Nxxxxxx) Renesas Starter Kit+ for RX64M (R0K50564Mxxxxxx)

7.2 Software update information

Software version	change	release date
V.2.10	Countermeasures for more security: - Changed generation algorithm of Initial Sequence Number (ISN) for defending against sequence number attacks. Limitation: - Added limitation about GCC version.	Apr.01.21
V.2.09	Added Target Compiler: GCC for Renesas RX, IAR C/C++ Compiler for Renesas RX Added function: Added TCP keep-Alive Bug Fix: - Fixed a problem that TCP / UDP communication can not be performed for some destination IP addresses when the gateway address is valid. - Fixed a problem that E_QOVR might return when calling API from callback routine. - When DHCP is enabled, the problem that UDP callback routine is invoked when UDP packet addressed to multicast address or broadcast address is received before the IP address is confirmed has been corrected.	Jun.28.19
V.2.08	Bug Fix: Fixed a problem that IP information of some channels in the work area was broken when DHCP was disabled and more than 3 channels were used. Fixed a problem that IP can not be acquired when DHCP is enabled and multiple boards are restarted at the same time. Fixed a problem that the callback function might not be called when UDP transmission is completed. Fixed an issue where IP addresses conflict when connecting a repeater hub.	Dec,10,18
V.2.07	Bug Fix: Fixed a problem that T4 Tiny sent an illegal Ping reply packet. Fixed the problem that UDP transmission / reception processing can't be performed when UDP transmission / reception processing is canceled.	Dec,31,17
V.2.06 Release 00	Add function: Added DHCP function. Bug Fix: Receive API cancel will not be accepted when TCP receive window data is remaining. SYN/ACK will not return when several SYNs come from by peer in same time. Discard the received packet when re-transmitted data includes additional payload. Illegal transmit data will be generated when T4 re-transmit and receive are occurred in same time.	Dec,15,16
V.2.05 Release 00	Add Function: Add IGMP functions Add igmp_join_group() function and igmp_leave_group() function. Join to the Multicast group using igmp_join_group() function. Leave from the Multicast group using igmp_leave_group() function.	Dec,01,15
V.2.03 Release 00	Bug Fix: Fixed wrong 2 behavior of FIN packet crossing between T4 and peer. Fixed code for peer that is not sending ACK	Aug,07,15

	Fixed code for internal state that detect wrong zero-window status after closing, the next connection will fail.	
V.2.02 Release 00	Changed library make Integrated Development Environment for RX Family	Jan,05,15
V.2.01 Release 00	Applied the T4 source code to Renesas coding rule	Jul,01,14
V.2.00 Release 00	<p>Add Function:</p> <p>Support Several LAN ports. Each LAN ports can have the MAC address and IP address. Opened source code. Supported FIT(Firmware Integration Technology). Supported Hokuto-Denshi RX63N board.</p> <p>Bug Fix:</p> <p>Fixed error code fitting to ITRON V.4 Fixed behavior when LAN cable is disconnected, cannot cancel tcp_cls_cep(). Fixed behavior when UDP transmitting and not resolve the ARP sequence, cannot complete UDP transmitting. Fixed tcp_sht_cep() can be canceled with specifying TFN_TCP_ALL .</p>	Apr,01,14
V.1.06 Release 00	<p>Add Function:</p> <p>UDP broadcast receive function (destination IP address 255.255.255.255) UDP directed-broadcast receive function (destination IP address example: network address = 192.168.0.0/24 -> broadcast address 192.168.0.255) UDP broadcast send function (destination IP address 255.255.255.255) UDP directed-broadcast send function (destination IP address example: network address = 192.168.0.0/24 -> broadcast address 192.168.0.255)</p> <p>Bug Fix:</p> <p>When user use RI600/4(Renesas uITRON) with T4, conflict r_t4_itcpip.h and itron.h. Receiving TCP window size is 0 packet, incorrect ACK would be sent from T4 Incorrect return value from tcp_acp_cep() that is in state of accepting. There is incorrect combination about IP address and subnet mask. This combination makes the packets not to transmit. Failure to re-allocate IP address from PPP server when PPP re-connection Incorrect setting for SCI channel 1 for RX210 PPP driver.</p>	Jun,21,13
1.05	<p>Add Function:</p> <p>Add T4 Library for PPP Divide APIs api_wup() to tcp_api_wup() and udp_api_wup() Divide APIs api_slp() to tcp_api_slp() and udp_api_slp()</p> <p>Improve Performance:</p> <p>Optimize checksum calculation. Enable Ethernet transmit interrupt</p> <p>Bug Fix:</p> <p>In case, result of calculating UDP checksum is ZERO, T4 stores temporary value to received UDP checksum area. In case, receiving broadcast packet before sending ARP response, T4 sends illegal packet.</p>	Apr,01,12
1.04	<p>Add Function:</p> <p>Add Etherent driver function "report_error". Add variable "_udp_enable_zerochecksum" for behavior of UDP sum check.</p>	Aug.30.11

	Bug Fix: Correct "t4_driver.c" to fix FR flag clear timing. This fixes wrong operation that EDMAC stops incorrectly.	
1.03	Bug Fix: When user use RI600/4(Renesas uITRON) with T4, User definition function "api_wup()" has no way to know which communication endpoint is ended. Change "api_wup()" argument. To know which communication endpoint is ended.	Feb.02.11
1.02	Bug Fix: When user use RI600/4(Renesas uITRON) with T4, conflict r_t4_itcpip and itron.h. Fixed r_t4_itcpip.h	internal use
1.01	Bug Fix: When T4 uses API "tcp_snd_dat" with condition that other endpoint becomes zerowindow, and other endpoint returns ACK with enough window size. T4 (sender) continues zerowindow probe, and other endpoint returns ACK with enough window size. This condition makes T4 not to be able to update remote window size and hung-up. When T4 judges "other endpoint is zerowindow", and other endpoint returns ACK with enough window size, T4 retransfers previous data. (not zerowindow probe)	Nov.10.10
1.00	first release	Oct.09.10

Revision History

Rev.	Date	Description	
		Page	Summary
2.10	Apr.01.21	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.10 Update following things. - Introduction - Added the TSIP FIT module in software stack. - Deleted contents about PPP feature. - Deleted contents about RX62N and RX63N that are not supported by FIT module. - 5.6 - Deleted contents about RX64M sample program. Add following things. - 1.6 - Added limitation about GCC version.
2.09	Jun.28.19	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.09 Update following things. 1. Overview 2. API Information 3. T4 Library Information 7. Appendices
2.08	Dec.10.18	-	Added 4.Sample Program from r20uw0031. Integrated r20an0312 into 4.Sample Program. Added 4.6 Component settings.
2.07	Dec.31.17	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.07 ->Changed from "Package version" to "Ver". Update following things. 1. Overview 2.3 T4 Ethernet Sample Application ROM / RAM / Stack Size 2.4 Version information 4. Notes 5. Appendices Adding following thing. 2.5 Adding the FIT Module to Your Project 5.1 Development Environment
2.06	Dec.15.16	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.06 Release 00 Added Support RX65N group. Added Chapter7. Added library with debug information. Changed library Compiler option. Changed library make Integrated Development Environment
2.05	Dec.01.15	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.05 Release 00 Update following things. Outline Remove rxv2 core library from the file structure. 4. Development Environment compiler version

			5. T4 Ethernet sample Application ROM / RAM / Stack Size 6. Version information
2.03	Aug.07.15	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.03 Release 00
2.02	Jan.05.15	- p4,p5 p6	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.02 Release 00 Added Support RX71M group. - Changed library file name and Compiler option. - Changed library make Integrated Development Environment
2.01	Jul.01.14	- p1 p2	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.01 Release 00 - Changed FIT Module URL. - Figure 1 T4 Software Stack
2.00	Apr.01.14	-	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.2.00 Release 00 - Added Hokuto Denshi RX63N board for environment - Changed stack size table. - Changed stack size table. - Changed stack size value.
1.06	Jun.21.13	- p6 p10 p12 p13 p14 p15	Release with TCP/IP for Embedded system M3S-T4-Tiny for the RX Family V.1.06 Release 00E - Changed form "Library version information" to "Software update information". ->Changed from "Ver" to "Package version" - Added Hokuto Denshi RX62N board for environment - Added Gadget Renesas RX63n board for environment - Changed stack size table. - Changed stack size table. - Changed stack size value. - Added section for Ethernet sample driver patch program - Added How to confirm sample program sections
1.05	Nov.09.12	p1 p4	Release with M3S-T4-Tiny for the RX Family V.1.05 Release01 Added RX63N to introduction Added RX63N to Development Environment
1.04	Sep.30.12	all	Release for internal use. Added RX63N sample program. Updated RX62N sample program. Updated RX62N Ether driver Applied Zero-copy API, and Improve performance. Added function, LAN cable hotswap. Added function, wake on LAN.
1.03	Apr.01.12	all p2 p6	Release with M3S-T4-Tiny for the RX Family V.1.05 Release00E Add information about T4 PPP. Add description for word that "HEW". Add notes for sample program. Add notes for multicast
1.02	Aug.30.11	all	Release with T4 library ver 1.04
1.01	Feb.02.11	all	Release with T4 library ver 1.03
1.00	Nov.10.10	-	First edition issued

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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

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

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

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics 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.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

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

Contact information

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

www.renesas.com/contact/.