# RENESAS

# Renesas e<sup>2</sup> studio

# Smart Configurator Application Examples: Ethernet

R20AN0495ES0100 Rev.1.00 Mar.27 ,2018

# Introduction

Smart Configurator (SC) is a GUI-based tool that has the functionalities of code generation and configuration for drivers, middleware and pins. SC generates suitable code for each Renesas MCU family and has the functionality to import code generated by FIT modules.

This application note guides user to use SC in the  $e^2$  studio to configure ether component and generate codes. The following operating systems on the host computer are supported:

- Windows 7 32-bit / 64-bit
- Windows 8.1 32-bit / 64-bit
- Windows 10 32-bit / 64-bit

#### **Target Device**

- RX64M Group
- RX65N Group

# **Software Components**

The Smart Configurator supports 2 types of software components: Code Generator (CG) and Firmware Integration Technology (FIT). Drivers and middleware supported by each software type are:

- Basic drivers: CG drivers (CMT, A/D Converter, SCI, etc.)
  - FIT modules (CMT, DTC, DMAC, RSPI, SCIFA, etc.)
- Middleware: FIT modules (USB, Ethernet, Flash Memory (programming the on-chip flash memory), etc.)

The basic driver is a control program for peripheral functions of microcomputer such as CMT, A/D converter, SCI, etc. It is convenient to embed software component (CG driver) using code generation (CG) function. In addition, FIT modules can be embedded for using middleware such as USB, Ethernet, and Flash memory (programming the on-chip flash memory) as software components.

List of abbreviations:

CMT: Compare Match Timer DTC: Data Transfer Controller DMAC: Direct Memory Access Controller RSPI: Serial Peripheral Interface SCIFA: FIFO Embedded Serial Communications Interface



# Contents

1.	Overview3
1.1	Purpose3
1.2	Operating Environment3
1.3	Basic Operation Steps of Smart Configurator Project4
1.4	Module Structure
1.5	Pin Setting for Ethernet Driver6
1.6	Main Clock Source
2.	Application Example (Create an Ethernet Program Using Smart Configurator)8
2.1	Program Work Flowchart
2.2	Creating a Workspace10
2.3	Creating a Project10
2.4	Clock Settings
2.5	Adding Software Components16
2.6	MCU Package
2.7	Generating Codes
2.8	Adding Source File Under src Folder 25
2.9	Adding Application Codes in main()
3.	Verify Operation
3.1	Marco Definition
3.2	Setting on Board and PC 32
3.3	Build and Debug Project
3.4	Echo Server Operation 41
3.5	Additional Debugging Assistance Tool (QE) 42
Web	site and Support43



# 1. Overview

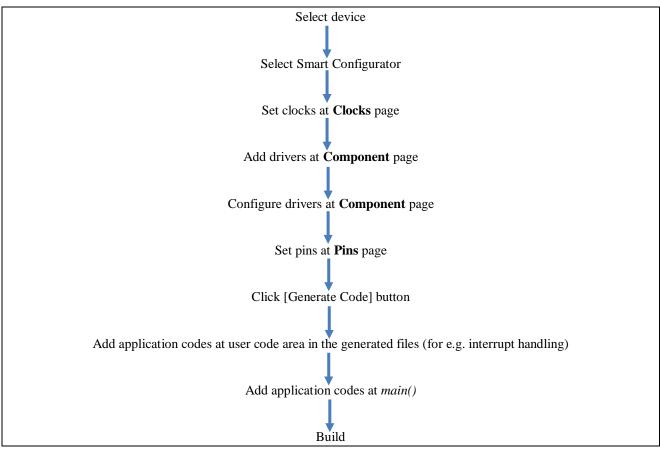
# 1.1 Purpose

This document guides user to create an echo server program using ethernet FIT modules in Smart Configurator.

# 1.2 Operating Environment

Target devices	RX64M Group
-	RX65N, RX651 Group
<b>Evaluation board</b>	Renesas Starter Kit+ for RX65N-2MB
	(RX65N R5F565NEDxFC)
Debugger	E1 /E2 Lite
IDE	e <sup>2</sup> studio v.6.2.0 or above
Toolchains	Renesas C/C++ compiler package for RX family





# **1.3 Basic Operation Steps of Smart Configurator Project**

Figure 1-1 Basic operation

Refer to "Smart Configurator User Guide" document for detailed operations of Smart Configurator.



# 1.4 Module Structure

This section shows the structure of the FIT modules used by Echo Server sample.

Application note that explains the usage of FIT module is available in the project tree under "doc" folder of each module. For example, Application Note for Ethernet Driver, R01AN2009, is located in \src\smc\_gen\r\_ether\_rx\doc folder.

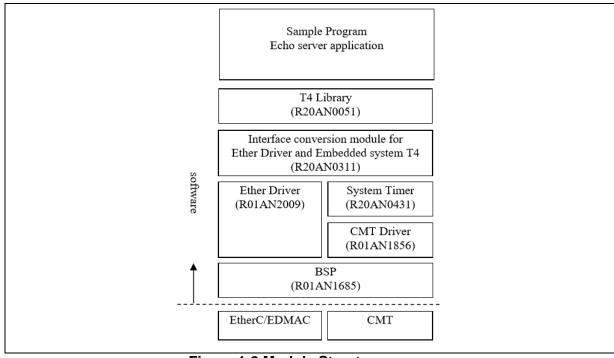


Figure 1-2 Module Structure

Table 1-1 below shows FIT Module to be configured.

Table 1-1 FIT Modules

Туре	Module	SC Software Component Name	Version
Middleware	T4 Library (TCP/IP for Embedded System M3S- T4-Tiny)	r_t4_rx	2.07
Interface	Interface Conversion Module for Ether Driver and Embedded System T4	r_t4_driver_rx	1.06
Device Driver	Ether Driver	r_ether_rx	1.14
Middleware	System Timer	r_sys_time_rx	1.00
Device Driver	CMT Driver (Compare Match Timer)	r_cmt_rx	3.10
BSP	BSP (Board Support Package)	r_bsp	3.60



# **1.5 Pin Setting for Ethernet Driver**

The MII Ethernet control mode is used in this Ethernet application example. An extract from Application Note for Ethernet module using FIT (R01AN2009EJ0114 - located in \src\smc\_gen\r\_ether\_rx\doc) is shown below.

Case of Using MII Mode	Case of Using RMII Mode
ETO TX CLK	
ETO RX CLK	REF50CK0
ETO TX EN	RMII0_TXD_EN
ET0_ETXD3	
ET0_ETXD2	
ET0_ETXD1	RMII0_TXD1
ET0_ETXD0	RMII0_TXD0
ET0_TX_ER	
ET0_RX_DV	
ET0_ERXD3	
ET0_ERXD2	
_ET0_ERXD1	RMII0_RXD1
ET0_ERXD0	RMII0_RXD0
_ET0_RX_ER	RMII0_RX_ER
ET0_CRS	RMII0_CRS_DV
ET0_COL	
ET0_MDC	
ET0_MDIO	
ET0_LINKSTA *1	
ET0_EXOUT *2	
ET0_WOL *2	
	d if the setting of #define ETHER_CFG_USE_LINKSTA is 0.
Notes: 2. Setting is not required	d because these pin are not used in Ethernet FIT module.

Figure 1-3 Pin Usage in MII Ethernet Control Mode

The schematic taken from Renesas Starter Kit+ for RX65N-2MB below shows the corresponding pins.

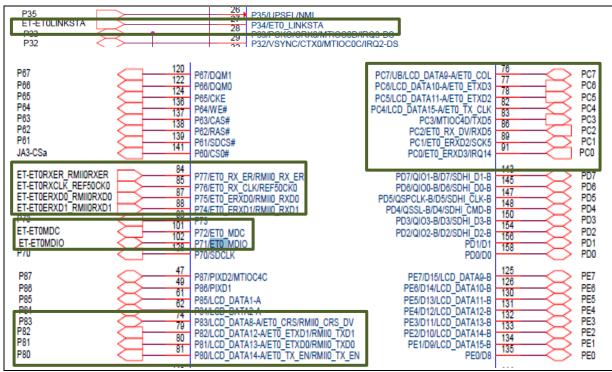


Figure 1-4 MII Ethernet physical pin assignment in Renesas Starter Kit+ for RX65N-2MB

In summary, configure the pins for Renesas Starter Kit+ for RX65N-2MB to operate in MII Ethernet control mode as shown in Table 1-2. This pin assignment will be configured in <u>Chapter 2.5</u> on component pin setting.



	-	
Function	Port Assignment	Pin Number
ET0_COL	PC7	76
ET0_CRS	P83	74
ET0_ERXD0	P75	87
ET0_ERXD1	P74	88
ET0_ERXD2	PC1	89
ET0_ERXD3	PC0	91
ET0_ETXD0	P81	80
ET0_ETXD1	P82	79
ET0_ETXD2	PC5	78
ET0_ETXD3	PC6	77
ET0_LINKSTA	P34	27
ET0_MDC	P72	101
ET0_MDIO	P71	102
ET0_RX_CLK	P76	85
ET0_RX_DV	PC2	86
ET0_RX_ER	P77	84
ET0_TX_CLK	PC4	82
ET0_TX_EN	P80	81
ET0_TX_ER	PC3	83

#### Table 1-2 Pin Assignment

# 1.6 Main Clock Source

Based on the schematic below, RX65N main clock is connected to a 24MHz crystal resonator. This clock will be configured as main clock source in <u>Chapter 3.3</u> on debug configuration setting.

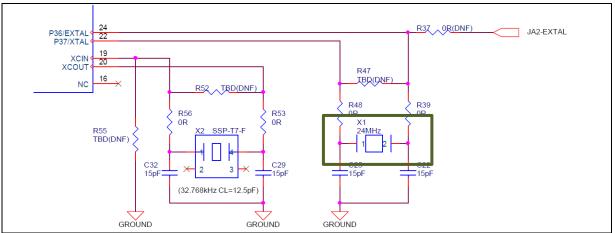


Figure 1-5 Schematic Diagram of Renesas Starter Kit+ for RX65N-2MB

# 2. Application Example (Create an Ethernet Program Using Smart Configurator)

# 2.1 **Program Work Flowchart**

The program flowchart is shown as below: a) Main function:

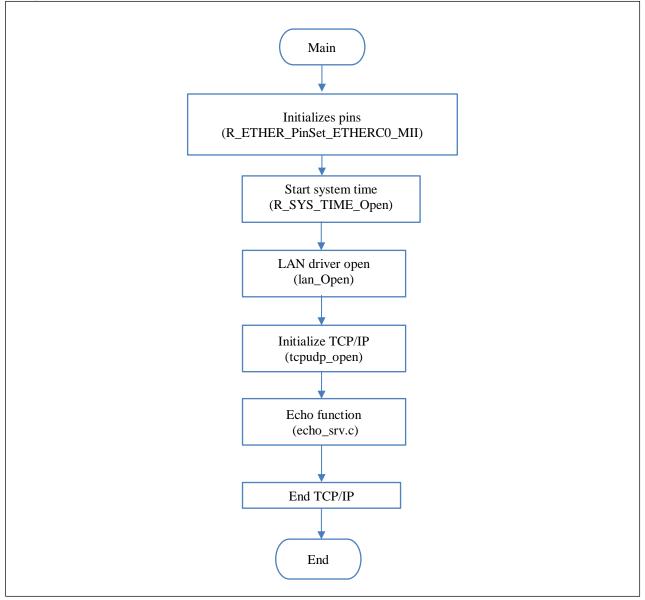


Figure 2-1 Main Flowchart

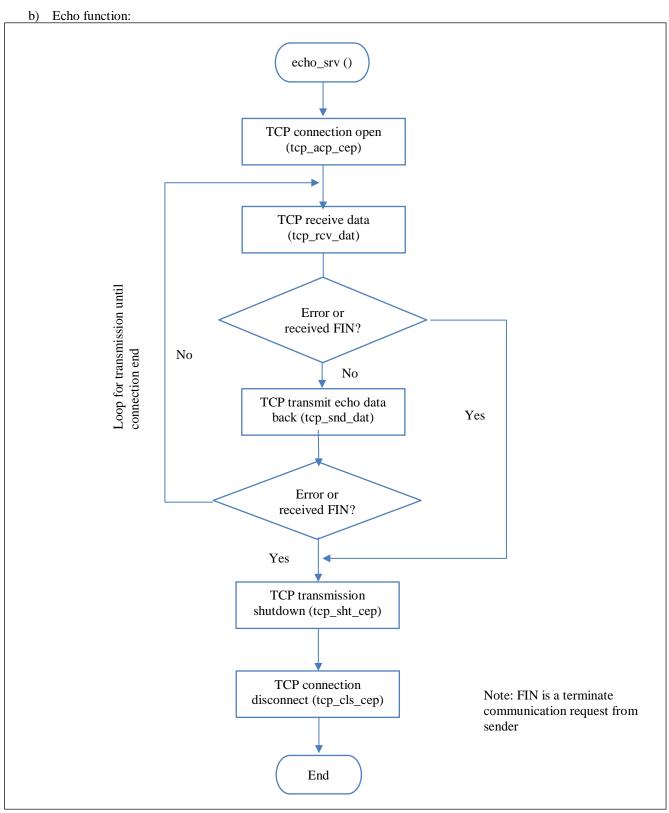


Figure 2-2 Echo Function Flowchart

# 2.2 Creating a Workspace

1) Start e<sup>2</sup> studio from Windows® Start Menu. Use default workspace folder and click [OK].

e <sup>2</sup> Workspace Launcher	2
Select a workspace	
e2 studio stores your projects in a folder called a workspace. Choose a workspace folder to use for this session.	
Workspace:	→ Browse
Use this as the default and do not ask again	

Figure 2-3 Workspace Launcher

# 2.3 Creating a Project

 Create a new C project in e<sup>2</sup> studio. Go to [File] → [New] → [C/C++ Project] to start new project generation

File	Edit Navigate Search Proje	ct Renesas Views	Run	Window	н	elp	
	New		Alt+Sh	ift+N >	Ð	Synergy C/C++ Projec	t
	Open File				СŶ	C/C++ Project	
	Open Projects from File System				Ċ	Project	
	Close		C	trl+W	<b>_</b>	Other	Ctrl+N
	Close All		Ctrl+Shi	ift+W			

Figure 2-4 Creating Project from File Menu



2) Select [Renesas RX]  $\rightarrow$  [Renesas CC-RX C/C++ Executable Project]  $\rightarrow$  [Next]

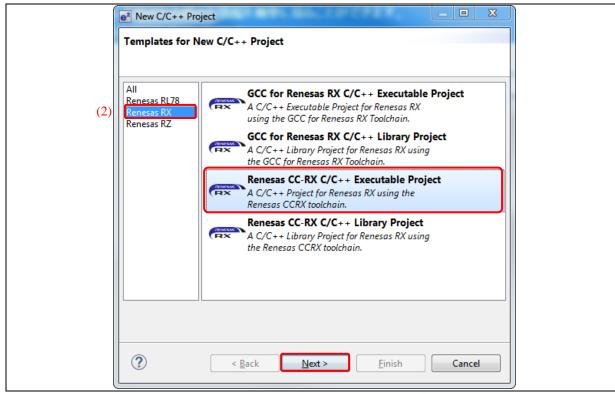


Figure 2-5 Creating Project from File Menu

3) Give an appropriate name to the project, for example "Smart\_Configurator\_Example"  $\rightarrow$  [Next]

	New Renesas CC-RX Executable Project           New Renesas CC-RX Executable Project
(3)	Project name: Smart_Configurator_Example
	Location: C:\Smart_Configurator_Example Browse Create Directory for Project Choose file system: default
	Working sets New
	Working sets: Select
	Image: Second

Figure 2-6 Creating Project from File Menu

- 4) Select "C" as Language
- 5) Select "Renesas CCRX" as Toolchain
- 6) Select Toolchain Version, e.g. "v2.08.00"
- Select Target Device accordingly: For RX65N-2MB, select "RX600 > RX65N > RX65N - 176pin > R5F565NEDxFC" For RX64M, select "RX600 > RX64M > RX64M - 176pin > R5F564MLCxFC"
- 8) Ensure [Create Hardware Debug Configuration] is ticked. Select emulator, e.g. "E1 (RX)".
- 9) Click [Next]

Endian: Little Create Debug Configuration Project Type: Default RX Simulator	Select toolchain, device & debug settings  Toolchain Settings Language: C C ++ (5) Toolchain: Renesas CCRX (6) Manage Toolchains  Device Settings (7) Configurations Target Device: R5F565NEDxFC C Inlian: Little Unlock Devices Endian: Little Project Type: Default (9)	— [				e <sup>2</sup>	$\times$
Toolchain Settings   Language:   C C C++   Toolchain:   Renesas CCRX   (6)   Toolchain Version:   v2.08.00   Manage Toolchains   Device Settings (7)   Target Device:   R5F565NEDxFC     Endian:   Little   Project Type:   Default   Configuration RX Simulator	Toolchain Settings   Language:   Language:   C C + +   Toolchain:   Renesas CCRX   (6)   Toolchain Version:   V2.08.00   Manage Toolchains   Device Settings (7)   Configurations   Target Device:   RSF565NEDxFC   Unlock Devices   Endian:   Little   Project Type:   Default   (9)			cecutable Projec	Renesas CC-RX Ex	New Renesas CC-RX Executable Project	-
Language: (C) C++ (5) Toolchain: Renesas CCRX (6) Toolchain Version: v2.08.00 (6) Device Settings (7) Target Device: R5F565NEDxFC (8) Configurations Target Device: R5F565NEDxFC (8) Create Hardware Debug Configuration Unlock Devices Endian: Little (100 Create Debug Configuration) Project Type: Default (7)	Language: Toolchain: Renesas CCRX Toolchain Version: V2.08.00 Manage Toolchains Device Settings (7) Target Device: R5F565NEDxFC Unlock Devices Endian: Little Project Type: Default (9) (6) Configurations E1 (RX) Create Hardware Debug Configuration RX Simulator Create Release Configuration			debug settings	toolchain, device &	Select toolchain, device & debug settings	
Toolchain: Renesas CCRX   Toolchain Version: v2.08.00   Manage Toolchains   (6) Configurations Target Device: R5F565NEDxFC Unlock Devices Endian: Little    Endian: Little   Project Type: Default   (5) Configuration (6) Configuration (7) Configurat	(5) Toolchain: Renesas CCRX (6) Toolchain Version: v2.08.00 Manage Toolchains (6) Device Settings (7) Target Device: R5F565NEDxFC				nain Settings	Toolchain Settings	
Toolchain Version:       v2.08.00       (6)         Manage Toolchains       Configurations         Device Settings (7)       Configurations         Target Device:       RSF565NEDxFC      (8)         Unlock Devices       Endian:       Little         Project Type:       Default       Create Debug Configuration	Toolchain Version: v2.08.00   Manage Toolchains   Device Settings (7)   Target Device:   R5F565NEDxFC  (8)   Create Hardware Debug Configuration   Endian:   Little   Project Type:   Default   (9)		(5)	)C++	age: 🚺 💽 📿	Language: $O C C++$ (5)	
Toolchain Version:       v2.08.00         Manage Toolchains       Manage Toolchains         Device Settings (7)       Configurations         Target Device:       RSF565NEDxFC      (8)         Unlock Devices       Endian:       Little         Project Type:       Default       Create Debug Configuration	Toolchain Version:       v2.08.00         Manage Toolchains       Manage Toolchains         Device Settings (7)       Configurations         Target Device:       RSF565NEDxFC      (8)         Endian:       Little       Create Hardware Debug Configuration         Project Type:       Default       Create Debug Configuration         RX Simulator       Create Release Configuration		× (6)	is CCRX	iain: Renesas		
Device Settings (7) Target Device: RSF565NEDxFC(8) Create Hardware Debug Configuration Endian: Little Project Type: Default Create Debug Configuration RX Simulator	Device Settings (7)       Configurations         Target Device:       RSF565NEDxFC       (8)         Unlock Devices       E1 (RX)         Endian:       Little       Create Debug Configuration         Project Type:       Default       RX Simulator         Create Release Configuration       Create Release Configuration		~ (0)	.00	nain Version: v2.08.	Toolchain Version: v2.08.00 v	
Target Device:       RSF565NEDxFC      (8)       Create Hardware Debug Configuration         Unlock Devices       E1 (RX)       ~         Endian:       Little       ~       Create Debug Configuration         Project Type:       Default       ~       RX Simulator	Target Device:       R5F565NEDxFC		olchains	Manag		Manage Toolchains	
Target Device:       R5F565NEDxFC      (8)       Create Hardware Debug Configuration         Unlock Devices       E1 (RX)       ~         Endian:       Little       Create Debug Configuration         Project Type:       Default       RX Simulator	Target Device:       R5F565NEDxFC      (8)       Create Hardware Debug Configuration         Endian:       Little       Endian:       E1 (RX)         Project Type:       Default       RX Simulator         Create Release Configuration       Create Release Configuration	ations	Config		e Settings (7)	Device Settings (7) Configurations	
Unlock Devices       E1 (RX)         Endian:       Little         Project Type:       Default         RX Simulator       Image: Configuration	Unlock Devices Endian: Little Project Type: Default (9)	e Hardware Debug Co	(8) Cre	DxFC			guration
Endian:       Little       Create Debug Configuration         Project Type:       Default       RX Simulator	Endian: Little Create Debug Configuration Project Type: Default  (9)						×.
Project Type: Default   RX Simulator	Project Type: Default RX Simulator Create Release Configuration (9)				Endian: Little		
	(9)				ct Type: Default		~
	(9)						
Create Release Configuration		e Release Configuratio	Cre			Create Release Configuration	
				(9)		(9)	
(9)	Cancel	Finish Ca	Next >	< Back			cel

Figure 2-7 C Project - Target Specific Settings Example with E1 Emulator



- 10) In the "Select Coding Assistant settings" dialog, select the checkbox of "Smart Configurator"
- 11) Click [Finish]

	New Renesas CC-RX Executable Project
	Select Coding Assistant settings
(10)	
	Use Peripheral Code Generator
	Use FIT Module Download FIT Modules
	Smart Configurator is a single User Interface that combines the functionalities of Code Generator and FIT Configurator which imports, configures and generates different types of drivers and middleware modules. Smart Configurator encompasses unified clock configuration view, interrupt configuration view and pin configuration view. Hardware resources conflict in peripheral modules, interrupts and pins occurred in different types of drivers and middleware modules will be notified. (Smart Configurator is available only for the supported devices)
	User Application Driver and Middleware Driver Code Configure din GUI and Generated MCU Hardware
	(11) ⑦ < Back Next > Finish Cancel

Figure 2-8 Select Coding Assistant Tool



# 2.4 Clock Settings

Smart Configurator perspective will be launched as shown below.

1) In Smart\_Configurator\_Example.scfg pane, click the [Clocks] page

<ul> <li>General Information</li> </ul>				
This editor allows you to mod	lify the settings stored in config	uration file (.scfg)		
Board				
Allow board and device select	tion			
Clocks			Application under	
Allow clock configuration			development	
,,			Middleware	
Components			Device	
Allow software component se	election and configuration		driver RTOS	
			← Pins	
Pins			+ Pins	
	on and pin configuration for sel	ected software component	← Pins	
Allow general pin configuration	on and pin configuration for sel	ected software component	← Pins	
Allow general pin configuratio		·		
Allow general pin configuratio	on and pin configuration for sel guration and interrupt configura	·		
Allow general pin configuration Interrupt Allow general interrupt config		·		
Allow general pin configuration Interrupt Allow general interrupt configuration Current Configuration	guration and interrupt configura	tion for selected software com		
Allow general pin configuration Interrupt Allow general interrupt configu- Current Configuration Selected board/device: R5556		tion for selected software com		
Allow general pin configuration Interrupt Allow general interrupt configu- Current Configuration Selected board/device: R5F56 Selected components:	guration and interrupt configura	tion for selected software com I size: 640KB, Pin count: 176)		
Allow general pin configuration Interrupt Allow general interrupt configu- Current Configuration Selected board/device: R5F56 Selected components: Component	guration and interrupt configura SSNEDxFC (ROM size: 2MB, RAM Version	tion for selected software com size: 640KB, Pin count: 176) Configuration		
Allow general pin configuration Interrupt Allow general interrupt configu- Current Configuration Selected board/device: R5F56 Selected components:	guration and interrupt configura	tion for selected software com I size: 640KB, Pin count: 176)		
Allow general pin configuration Interrupt Allow general interrupt configu- Current Configuration Selected board/device: R5F56 Selected components: Component	guration and interrupt configura SSNEDxFC (ROM size: 2MB, RAM Version	tion for selected software com size: 640KB, Pin count: 176) Configuration		

Figure 2-9 Smart Configurator Perspective



2) Since a 24MHz crystal resonator is connected to main clock of RX65N (refer to chapter 1.6), check to confirm that the main clock frequency is set to 24 MHZ. Keep the other clock settings as default.

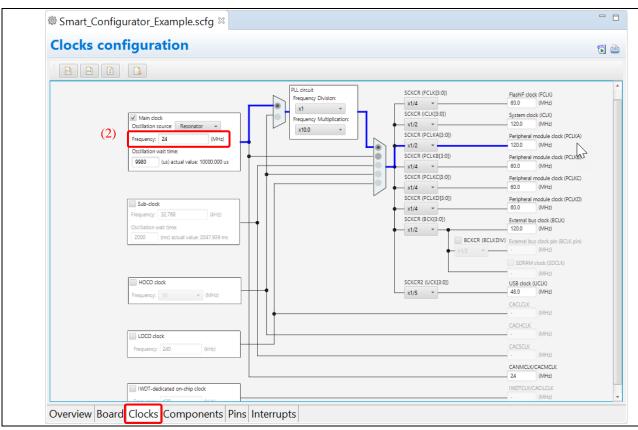


Figure 2-10 Clock Configuration in Smart Configurator



# 2.5 Adding Software Components

1) In Smart\_Configurator\_Example.scfg pane, click the [Components] page

This editor allows you to mo	dify the settings stored in configu	aration file (.scfg)		
Board				
Allow board and device selec	:tion			
Clocks Allow clock configuration			Application under development	
Components			Middleware -Components	
Allow software component s	election and configuration		driver RTOS	
	, , , , , , , , , , , , , , , , , , ,		- Pins	
Pins				
Allow general pin configurat	ion and pin configuration for sele	ected software component		
Interrupt				
	iguration and interrupt configurat	tion for selected software cor	mponent	
Current Configuration				
-	65NEDxFC (ROM size: 2MB, RAM	size: 640KB, Pin count: 176)		
-	65NEDxFC (ROM size: 2MB, RAM	size: 640KB, Pin count: 176)		
Selected board/device: R5F5	65NEDxFC (ROM size: 2MB, RAM Version	size: 640KB, Pin count: 176) Configuration		
Selected board/device: R5F5 Selected components:				
Selected board/device: R5F5 Selected components: Component	Version	Configuration		

# Figure 2-11 Smart Configurator Perspective

# 2) Click to add new component.

# Figure 2-12 Software Component Configuration in Smart Configurator



#### 3) Add *FIT modules* into the project

- a. Navigate the component list and select *r\_cmt\_rx* module
- b. Press and hold Ctrl key, click on *following* modules:

r\_ether\_rx r\_sys\_time\_rx r\_t4\_driver\_rx r\_t4\_rx Note: If above

Note: If above drivers are not available in the list, click [Download more software components] to download the FIT modules.

c. Click [Finish]

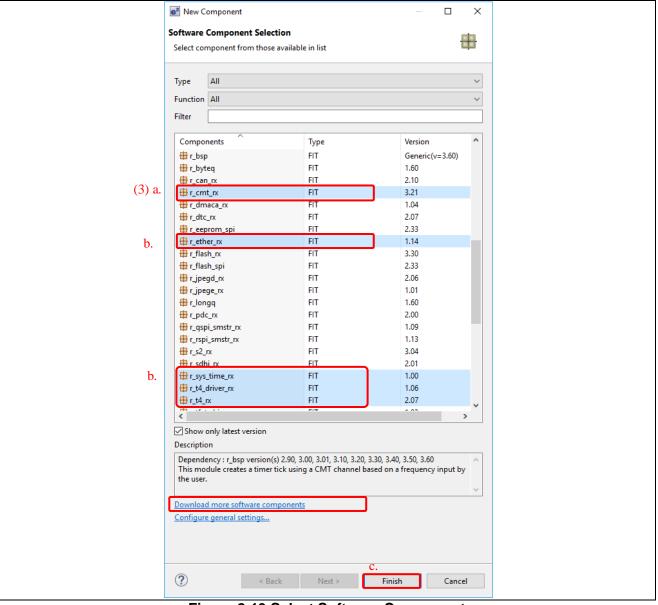


Figure 2-13 Select Software Component

4) New software components are shown in the [Components] page.

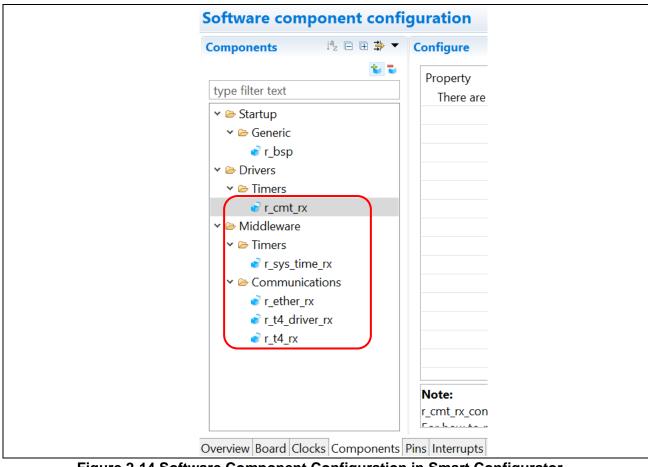


Figure 2-14 Software Component Configuration in Smart Configurator



### 5) In the [Components] page,

*a.* select *r\_t4\_rx* 

b.	set the following settings under 'Configu	urations'
	Channel number your system has.	1
	Enable/Disable DHCP Function.	0
	SYSTEM callback function use	0

Note: These settings can be found in "config\_tcpudp.c" file in  $\rcsmc_gen\r_t4_rx\src$  folder after generating codes.

Components		Configure	•	
	10 T	Proper		Value
type filter text		~ ☺	Configurations	
✓ ➢ Startup		b.	# Channel number your system has.	1
V > Generic		υ.	# Enable/Disable DHCP function.	0
er bsp			# IP address for ch0, when DHCP disable.	192,168,0,3
✓ ▷ Drivers			# Subnet mask for ch0, when DHCP disable.	255,255,255,0
V > Timers			# Gateway address for ch0, when DHCP disable.	0,0,0,0
r cmt rx			# IP address for ch1, when DHCP disable.	192,168,0,10
V A Middleware			# Subnet mask for ch1, when DHCP disable.	255,255,255,0
V C Timers			# Gateway address for ch1, when DHCP disable.	0,0,0,0
_			# Ether ch0 MAC address.	0x74,0x90,0x50,0x00,0x79,0x03
€ r_sys_time_rx			# Ether ch1 MAC address.	0x74.0x90.0x50.0x00.0x79,0x10
✓		b.	# SYSTEM callback function use	0
r_t4_driver_rx			# SYSTEM callback function name.	system_callback
(5) a. <u>r_t4_rx</u>			# TCP REPID1 port number	1024
and r_ether_rx			# TCP REPID2 port number	1025
			# TCP REPID3 port number	1026
			# TCP REPID4 port number	1027
	~	<b>Macro</b> 0 = Unu 1 = Use		LUSE

Figure 2-15 Components Setting



6) In the [Components] page,

а.	select <i>r_ether_rx</i>	
b.	set the following settings under 'Configura	tions'
	Ethernet interface	MII
	PHY-LSI address setting for ETHER 0	30 (For RSK+ 65N-2MB)
		0 (For RSK+ 64M)
	PHY-LSI address setting for ETHER 1	1
	The register bus of PHY0 for ETHER0/1	Use ETHER0
	The register bus of PHY1 for ETHER0/1	Use ETHER1
<u>а</u> , т	base settings can be found in "r other ry co	nfig h" file in \sre\sma gen\r config

Note: These settings can be found in "r\_ether\_rx\_config.h" file in  $\src\smc_gen\r_config folder after generating codes.$ 

Components 👌 🗄 🗄		- C		
		onngure		
type filter text		Property	Value	
✓ ➢ Startup		# Ethernet interface	MII (Media Independent Interface)	
🕆 🗁 Generic	b.	# PHY-LSI address setting for ETHER0	30	
💣 r_bsp		# PHY-LSI address setting for ETHER1	1	
✓ ➢ Drivers		# The number of Rx descriptors	1	
🕆 🗁 Timers		# The number of Tx descriptors	1	
💣 r_cmt_rx		# Transmit and Receive buffer size	1536	
🕆 🗁 Middleware		# EINT interrupt priority level	Level 2	
🕆 🗁 Timers		# Group AL1 interrupt priority level	Level 2	
💣 r_sys_time_rx	b.	# The register bus of PHY0 for ETHER0/1	Use ETHER0	
Y 🗁 Communications	0.	# The register bus of PHY1 for ETHER0/1	Use ETHER1	
(6) a. 💣 r_ether_rx		# The access timing of MII/RMII register	8	
r_t4_driver_rx		# The waiting time for reset completion of PF	HY- 0x00020000L	
r_t4_rx		# The polarity of the link signal output by the	PF Fall -> Rise	
		# The link status is detected.	Used	
		# Use KSZ8041NL of the Micrel Inc	Unused	
		Y 🖲 Resources		
	r	Macro definition: ETHER_CFG_CH0_PHY_ADDRESS	S	

Figure 2-16 Ethernet Pin Setting



#### 7) In the [Components] page,

- *a.* select *r\_ether\_rx*
- b. Under "Resources", click the checkboxes for the resources as shown in the picture below. These selected resources are used in this example code

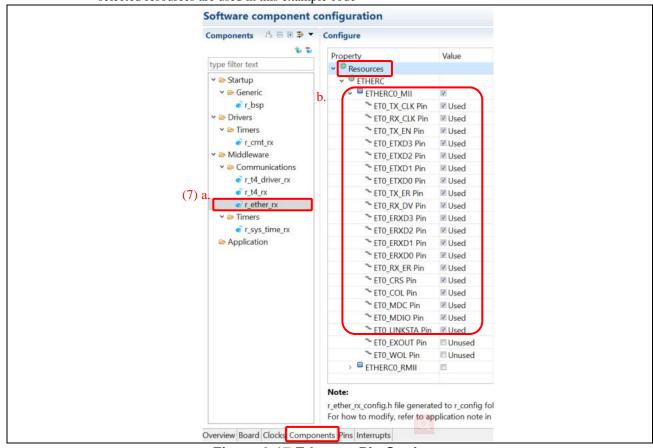


Figure 2-17 Ethernet Pin Setting



8) At [Pins] page, click

button to switch tree to Software Components view

in configuration (8)							۵ 🕑
Hardware Resource 🛛 🕀 🖃 🗸 🗸	Pin Functio	n				2 🖬 è	<u>.</u> 2
Type filter text	how by Hard	ware Resource o	r Software Components				
🚣 All 🔺	Enabled	Function	Assignment	Pin Number	Direction	Remarks	
Clock generator		P00	Not assigned	Not assigned	None		
Clock frequency accuracy measurem		P01	Not assigned	Not assigned	None		
* Buses		P02	Not assigned	Not assigned	None		
EXDMA controller		P03	Not assigned	Not assigned	None		
Interrupt controller unit		P05	Not assigned	Not assigned	None		
Multi-function timer pulse unit 3		P07	Not assigned	Not assigned	None		
MTU0		P10	Not assigned	Not assigned	None		
MTU1		P11	Not assigned	Not assigned	None		
MTU2		P12	Not assigned	Not assigned	None		-
	•		III				Þ.

# Figure 2-18 Pins page

- 9) At the tree, click *r\_ether\_rx* to view pin configurations
- 10) Ensure following functions were assigned to corresponding pins (as explained in chapter 1.5):e.g. Check enable flag and change assignment of Function "ETO\_CRS" from PB7(default) to P83

Software C 🖲 🖻 🖧 🤷 P	'in Fun	iction				
Type filter text Type pin function						
	Ena	Function ET0_COL (10)	Assignment PC7/UB/A23/CS0#/MTIOC3A/MTC	Pin Num	Directi	
✓ Å r cmt rx	$\overline{\mathbf{v}}$	ETO CRS	P83/EDACK1/MTIOC4C/ET0_CRS/R	1.000	1	
r_cmt_rx		ETO ERXDO	P75/CS5#/PO20/ET0_ERXD0/RMII0	1000	1	
✓ ▲ r_sys_time_rx		ETO ERXD1	P74/A20/CS4#/PO19/ET0 ERXD1/		I.	
r_sys_time_rx		ETO ERXD2	PC1/A17/MTIOC3A/TCLKD/PO18/E		1	
✓ ▲ r_t4_driver_rx		ETO_ERXD3	PC0/A16/MTIOC3C/TCLKC/PO17/E	91	1	
r_t4_driver_rx		ETO_ETXD0	P81/EDACK0/MTIOC3D/PO27/ET0	80	0	
✓ ▲ r_t4_rx		ETO_ETXD1	P82/EDREQ1/MTIOC4A/PO28/ET0	79	0	
v_t4_rx	$\checkmark$	ET0_ETXD2	PC5/D3/A21/CS2#/WAIT#/MTIOC	78	0	
✓ ₫ r ether rx	$\checkmark$	ETO_ETXD3	PC6/D2/A22/CS1#/MTIOC3C/MTC	77	0	
) 💣 r_ether_rx		ET0_EXOUT	Not assigned	Not assig	None	
(10	)	ETO_LINKS	P34/MTIOC0A/TMCI3/PO12/POE1	27	1	
	$\checkmark$	ET0_MDC	P72/A19/CS2#/ET0_MDC/LCD_DAT	101	0	
	$\checkmark$	ET0_MDIO	P71/A18/CS1#/ET0_MDIO	102	10	
	$\checkmark$	ETO_RX_CLK	P76/CS6#/PO22/ET0_RX_CLK/REF5	85	1	
	$\checkmark$	ETO_RX_DV	PC2/A18/MTIOC4B/TCLKA/PO21/E	86	1	
	$\checkmark$	ETO_RX_ER	P77/CS7#/PO23/ET0_RX_ER/RMII0	84	1	
	$\checkmark$	ET0_TX_CLK	PC4/A20/CS3#/MTIOC3D/MTCLKC	82	1.	
	$\checkmark$	ETO_TX_EN	P80/EDREQ0/MTIOC3B/PO26/ET0	Subjects .	0	
	$\checkmark$	ETO_TX_ER	PC3/A19/MTIOC4D/TCLKB/PO24/E		0	
		ET0_WOL	Not assigned	Not assig		
		REE50CK0	Not assigned	Not assid	None	

Figure 2-19 Pin configuration of r\_ether\_rx

# 2.6 MCU Package

After completing all pin configurations, the MCU package view also updated the pin assignment automatically as shown in picture below.

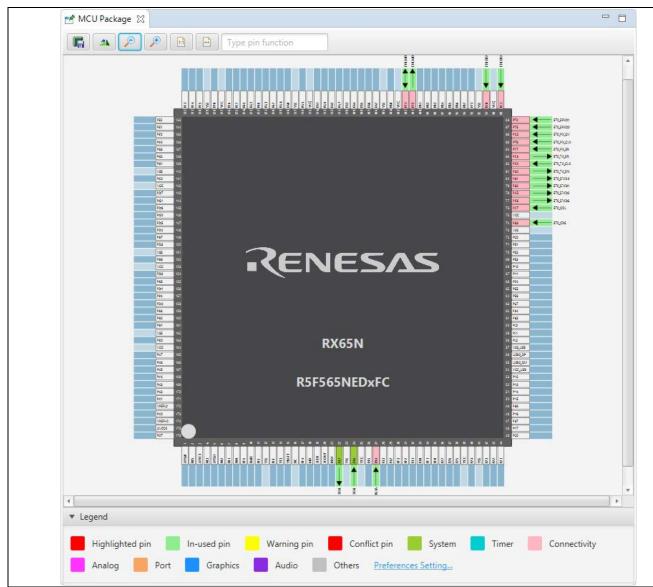


Figure 2-20 Pin Assignment in MCU Package



# 2.7 Generating Codes

1) Click to generate codes

Smart_Configurator_Example.scfg 🔀	
Pins Configuration	(1)
Software Components 🕞 🖃 🖓 🐻 Pin Function	2 📓 2



- 2) Message 'Code generation is successful will be shown at Console
- 3) Files generated into \src\smc\_gen folder of the project

	File Edit Navigate Search Project Renesas	Views Run Window Help	
	🔦 🔯 🔳 🕸 Debug 🛛 🗸 🖻 Smart_Co	onfigurator_Ex; 🗠 🌼 🗄 🔻 🔚 🐚   🗞	- «, - 📾 🗐 🚅 N
	Project Explorer ≅ 🛛 🖻 🕏 ⊽ 🖓 🗖	🔋 🏟 *Smart_Configurator_Example.se	cfg 🛙
	👻 😂 Smart_Configurator_Example	Software component co	onfiguration
	> 🔊 Includes	•	
	× ≌ src	Components 🕴 🗄 🛱 🖛	Configure
(	(3) ັ ັ ັ Smc_gen	🔪 🐨	Property
	> 🗁 general	type filter text	There are
	> 🗁 r_bsp	> 🗁 Startup	
	> 🗁 r_cmt_rx	> 🗁 Drivers	
	> 🍃 r_config	> 🗁 Middleware	
	> 🍅 r_ether_rx	➢ Application	
	> 🗁 r_pincfg		
	> ≽ r_sys_time_rx > ≽ r t4 driver rx		
	$\rightarrow \cong r_1 4 \text{ rx}$		
	> 🗟 Smart Configurator Example.c		
	> > trash		Note:
	Smart Configurator Example Hardware	eD	r_cmt_rx_conf For how to m
	Smart Configurator Example.scfg		
			<
	<	> Overview Board Clocks Compone	ents Pins Interru
	⊑ Console ¤		
	Smart Configurator Output		
	M05000012: File generated:src\smc	_gen\r_pincfg\Pin.c	
	M0600002: File generated:src\smc		
	M06000002: File generated:src\smc M06000002: File generated:src\smc		
(2)	M00000002: Code generation is such		
	M03000004: File modified:src\smc_(		
	<		

Figure 2-22 Successful Code Generation

# 2.8 Adding Source File Under src Folder

1) At Project Explore tree, right click [src] folder, select [New]  $\rightarrow$  [Source file]

🔓 Projec	t Ex	olorer 🛛 🔤 C/C++ Projects					
~ 🔗 Sma	art_	Configurator_Example [HardwareD	ebug]	^			
> 🔊 Ir	nclu	des (2)					
(1), 😰 :		New		>		Project	
<b>~</b> ≨		Go Into			Ľ	File	
		Open in New Window			Ľ	File from Template	
		Show In		>	Ċ	Folder	
	-			-	¢	Class	
		Сору	Ctrl-	(3)	h	Header File	
	E	Paste	Ctrl-		C	Source File	
	×	Delete	Dele	ete	-	Source Folder	
	<u></u>	Remove from Context C	Ctrl+Alt+Shift+Dov	wn			
		Source		>		C/C++ Project	
		Move				RZ/G C/C++ project	
		Rename		F2		Synergy C Project	
					C+	Synergy C++ Project	
	è	Import				Other	

Figure 2-23 Adding Source File

2) Input header file name (e.g. *echo\_srv.c*), click [Finish]

Source File         Create a new source file.         Source folder:       Smart_Configurator_Example/src         Browse         Source file:       echo_srv.c         Template:       Default C source template       ✓	
Source folder: Smart_Configurator_Example/src Browse Source file: echo_srv.c	
Source file: echo_srv.c	
Source file: echo_srv.c	
Template: Default C source template  V Configure	
compute. Denuit e source template	
? Finish Cancel	

Figure 2-24 Adding Source File



 Open "echo\_srv.c" file in \src folder Add below codes in "echo\_srv.c" for this application example:

```
#include "r t4 itcpip.h"
                         ****
Macro definitions
/* Size of Ethernet receive buffer,refer to tcp_ccep[].rbufsz */
#define BUFFER_SIZE
                          (1460)
void echo_srv(void)
{
   ID
             cepid=1; /*ID of a TCP communication end point ("1"~ "30") */
             repid=1; /*ID of a TCP reception point ("1"~ "30") */
   ID
   T IPV4EP
             dst addr; /*destination IP address (PC)*/
   UB
             rcv_buf[BUFFER_SIZE ];/*receive buffer*/
             ercd;/*error code*/
   ER
   /* Make one TCP connection on Ethernet channel.
   Ethernet 0: 192.168.0.3 - Port: 1024 (refer to config_tcpudp.c)*/
    while (1)
   {
       /* TCP connection open */
       ercd = tcp_acp_cep(cepid,repid, &dst_addr,TMO_FEVR);
       if(E_OK == ercd)
       {
             /* process of echo server */
             while(1)
             {
                   /* TCP receive data */
                   ercd = tcp_rcv_dat(cepid, rcv_buf, sizeof(rcv_buf), TMO_FEVR);
                   if(ercd <= 0)</pre>
                   {
                         break;
                   }
                   /* TCP transmit echo data back */
                   ercd = tcp snd dat(cepid, rcv buf, ercd, TMO FEVR);
                   if(ercd < 0)
                   {
                         break;
                   }
             }
             /* Close TCP connection */
             tcp_sht_cep(cepid); /*TCP transmission shutdown */
             tcp_cls_cep(cepid, TMO_FEVR);/*TCP connection disconnect */
          }
      }
} /* End of function echo_srv */
```

Your source file should look like this:

```
* echo_srv.c_
#include "r_t4_itcpip.h"
Macro definitions
   /* Size of Ethernet receive buffer,refer to tcp_ccep[].rbufsz */
#define BUFFER_SIZE (1460)
void echo_srv(void)
{
            cepid=1; /*ID of a TCP communication end point ("1"~ "30") */
   TD
            repid=1; /*ID of a TCP reception point ("1"~ "30") */
   ID
   T_IPV4EP dst_addr; /*destination IP address (PC)*/
   UB
           rcv_buf[BUFFER_SIZE ];/*receive buffer*/
   ER
            ercd;/*error code*/
   /* Make one TCP connection on Ethernet channel.
   Ethernet 0: 192.168.0.3 - Port: 1024 (refer to config_tcpudp.c)*/
   while (1)
   {
       /* TCP connection open */
       ercd = tcp_acp_cep(cepid,repid, &dst_addr,TMO_FEVR);
       if(E_OK == ercd)
       ſ
          /* process of echo server */
          while(1)
          {
              /* TCP receive data */
             ercd = tcp_rcv_dat(cepid, rcv_buf, sizeof(rcv_buf), TMO_FEVR);
             if(ercd <= 0)</pre>
             {
                 break;
             }
             /* TCP transmit echo data back */
             ercd = tcp_snd_dat(cepid, rcv_buf, ercd, TMO_FEVR);
             if(ercd < 0)</pre>
             {
                 break;
             }
          }
          /* Close TCP connection */
          tcp_sht_cep(cepid); /*TCP transmission shutdown */
          tcp_cls_cep(cepid, TMO_FEVR);/*TCP connection disconnect */
         }
      }
} /* End of function echo_srv */
```

Figure 2-25 echo\_srv.c

# 2.9 Adding Application Codes in main()

1) In Smart\_Configurator\_Example.c, add/overwrite below codes after code line [#include "r\_smc\_entry.h"]

```
#include <string.h>
#include "r_t4_itcpip.h"
#include "r_sys_time_rx_if.h"
  Macro definitions
       /* T4 work memory area size is 4.5 KB, refer to application note R20AN0051EJ0206 */
#define T4_WORK_SIZE (4608)
                      *****
Private global variables and functions
                             static UW tcpudp_work[T4_WORK_SIZE / sizeof(UW) + 1];
extern void echo srv(void);
extern void R_ETHER_PinSet_ETHERC0_MII();
void main(void)
{
             /*error code*/
   ER ercd;
   sys_time_err_t systime_ercd; /*system time error code*/
       ver[128];
   char
   /* Initializes pins for r_ether_rx module */
    R_ETHER_PinSet_ETHERC0_MII();
   /* Get the version of T4 */
   strcpy(ver, (char*)R_t4_version.library);
   /* start system time */
   systime_ercd = R_SYS_TIME_Open();
   if (systime_ercd != SYS_TIME_SUCCESS)
   {
      while (1);
   }
   /* start LAN controller */
   ercd = lan_open();
   if (ercd != E_OK)
   {
      while (1)
      {
       /* Cannot open LAN controller */
      };
   }
   /* Initialize the TCP/IP */
   ercd = tcpudp_open(tcpudp_work);
   if (ercd != E_OK)
   {
      while (1)
      {
       /* Cannot open TCP/IP */
      };
   }
   /* start echo server */
   echo_srv();
   /* end TCP/IP */
   tcpudp_close();
   lan_close()
   R_SYS_TIME_Close();
}/* End of function main() */
```



Your source file should look like this:

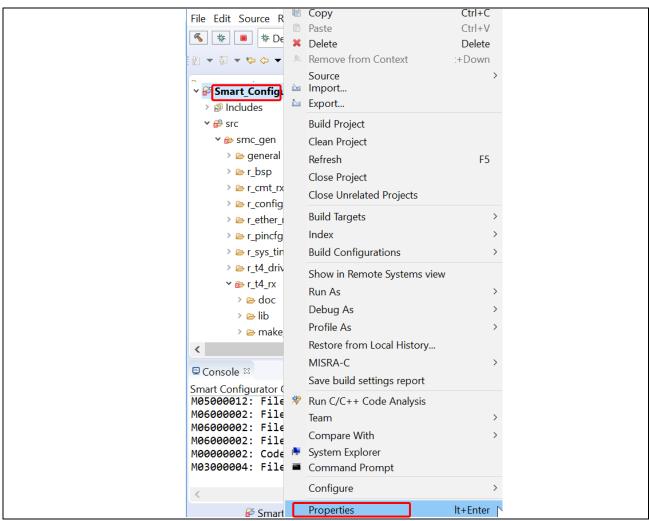
```
#include "r smc entry.h"
#include <string.h>
#include "r_t4_itcpip.h"
#include "r_sys_time_rx_if.h"
Macro definitions
                /* T4 work memory area size is 4.5 KB,refer to application note R20AN0051EJ0206 */
#define T4_WORK_SIZE (4606)
                            ******
Private global variables and functions
                                  static UW tcpudp_work[T4_WORK_SIZE / sizeof(UW) + 1];
Imported global variables and functions (from other files)
                                                  ***********************/
extern void echo srv(void);
extern void R_ETHER_PinSet_ETHERC0_MII();
void main(void)
Ł
   ER ercd; /*error code*/
sys_time_err_t systime_ercd; /*system time error code*/
      ver[128];
char
   /* Initializes pins for r_ether_rx module */
   R_ETHER_PinSet_ETHERC0_MII();
   /* Get the version of T4 */
   strcpy(ver, (char*)R_t4_version.library);
   /* start system time */
   systime_ercd = R_SYS_TIME_Open();
   if (systime_ercd != SYS_TIME_SUCCESS)
   {
       while (1);
   }
   /* start LAN controller */
   ercd = lan_open();
   if (ercd != E OK)
   {
       while (1)
      {
          /* Cannot open LAN controller */
      };
   }
   /* Initialize the TCP/IP */
   ercd = tcpudp_open(tcpudp_work);
   if (ercd != E_OK)
   {
      while (1)
       {
          /* Cannot open TCP/IP */
       };
   }
   /* start echo server */
   echo_srv();
   /* end TCP/IP */
   tcpudp_close();
   lan_close();
   R_SYS_TIME_Close();
  * End of function main() */
```

Figure 2-26 main.c

# 3. Verify Operation

# 3.1 Marco Definition

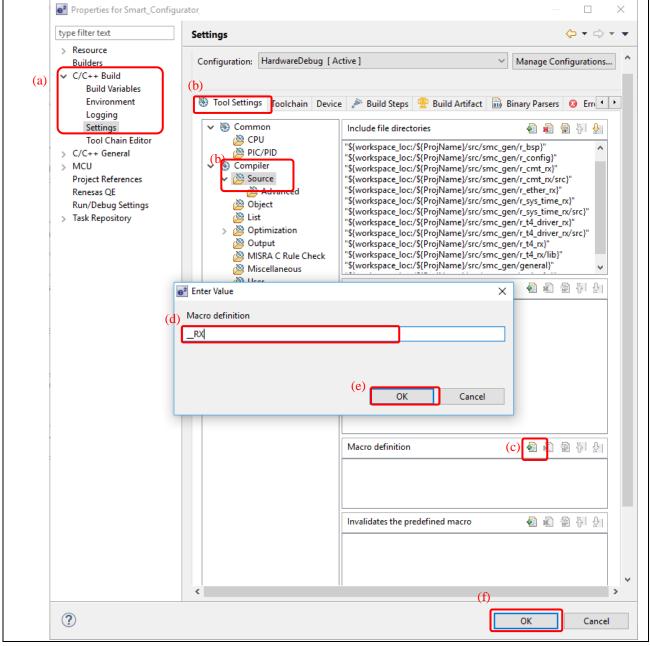
1) Right click [Smart\_Configurator\_Example] in Project Explorer, click [Properties]



**Figure 3-1 Open Properties** 



- 2) Under properties window:
  - a. Click [C/C++ Build]  $\rightarrow$  [Settings]
  - b. Under [Tool Settings] tab, select [Compiler]  $\rightarrow$  [Source]
  - c. Click 🗌 button at [Macro definition] window
  - d. At the pop up window, enter '\_\_RX'
  - e. Click [OK] to close [Enter Value] window
  - f. Click [OK] to confirm all settings



**Figure 3-2 Modify Properties** 

# 3.2 Setting on Board and PC

1) Ensure hardware of target board is setup according to Table 3-1 below.

#### Table 3-1 Setting on Board

Exection		Configuration
Function	RSK+ RX65N-2MB	RSK+ RX64M
LAN cable	ETHERNET socket	ETHERNET0 socket
MII Mode	J8: Open (do not connect)	-
ETOLINKSTA	-	SW6.5: ON; SW6.6: OFF
ETOMDIO	-	J3: Pin1-2
ETOMDC	-	J4: Pin1-2
ET0ERXD1	-	SW6.7: ON; SW6.8: OFF
ETORXCLK	-	J10: Pin1-2; R48 soldered; R57 not soldered
ETORXER	-	SW6.9: ON; SW6.10: OFF
ET0TXEN	-	SW7.1: ON; SW7.2: OFF; SW7.3: OFF
ET0ETXD0	-	SW7.4: ON; SW7.5: OFF; SW7.6: OFF
ET0ETXD1	-	SW7.7: ON; SW7.8: OFF
ETOCRS	-	SW7.9: ON; SW7.10: OFF
ET0ERXD3	-	J11: Pin2-3; SW5.1: ON
ET0ERXD2	-	SW5.2: ON; SW5.3: OFF
ETORXDV	-	SW5.4: ON; SW5.5: OFF
ET0TXER	-	SW5.6: ON; SW5.7: OFF; SW5.8: OFF
ET0TXCLK	-	SW5.9: ON; SW5.10: OFF
ET0ETXD2	-	J13: Pin2-3; SW6.1: ON
ET0ETXD3	-	J14: Pin2-3; SW6.2: ON
ET0COL	-	J12: Pin2-3; SW6.3: ON; SW6.4: OFF

- 2) Connect the target board to PC

   a. Connect RSK+ RX65N-2MB board to E1/ E2 Lite emulator and connect the E1/ E2 Lite emulator to PC
  - b. Use 1 LAN cable (cross or straight) to connect RSK+ RX65N 2MB board to PC

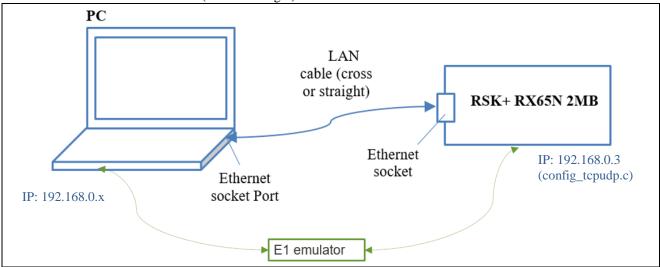


Figure 3-3 Connection between PC and RSK+ RX65N 2MB

- 3) Set IP address on PC
  - a. In Windows 10 OS environment, under [Network & Internet] setting, (a.1) Click [Proxy] to open setup page
    - (a.2) Ensure that [Manual proxy setup] is set to off.

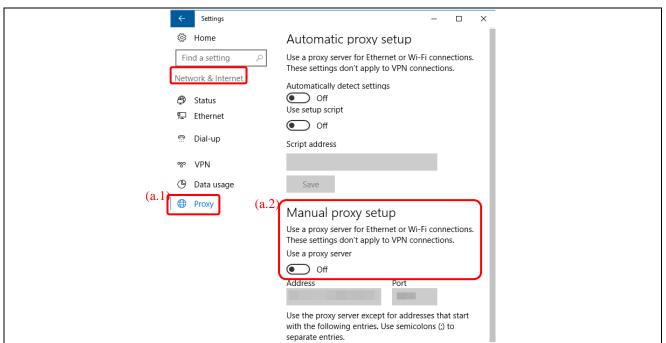


Figure 3-4 Setting IP address on PC

- b. Under [Network & Internet] setting:(b.1) Click [Ethernet] to open setup page(b.2) Click [Change adapter options]
  - (b.3) In the pop up window, right click the 'Ethernet' and select [Properties]

← Settings		- 🗆 ×
O Home	Ethernet	
Find a setting		
Network & Intern	Connected	
🖨 Status	(b.2) Related settings	
(b.1) 🛛 🗜 Ethernet	Change adapter options	
ි Dial-up	🛬 Network Connections	- 🗆 ×
	$\leftarrow \  ightarrow \$	ns ∨ Ö Search Net ₽
% VPN	Organize   Disable this network device   *	P • 🔳 🕐
🕒 Data usage	Ethernet 3 adwin.ren	
Proxy	Intel(R) Et 🗣 Disable Status	
	Diagnose	
	Bridge Connections	
	Create Shortcut	
	(b.3) Rename	
	Properties	
	1 item 1 item selected	E 📰

Figure 3-5 Setting IP address on PC

c. In the pop up window:
(c.1) Select [Internet Protocol Version 4 (TCP/IPv4)]
(c.2) Click [Properties]

	Ethernet 3 Properties	×
	Networking	
	Connect using:          Intel(R) Ethemet Connection (2) I219-LM         Configure         This connection uses the following items:         Image: Client for Microsoft Networks	
(c.		(c.2)
	OK Cancel	al

Figure 3-6 Setting IP address on PC

d. In the pop up window:

- (d.1) Select button for [Use the following IP address] and set the following:
  - IP address: 192.168.0.100
    - (192.168.0.x in which  $x = 1 \sim 254$  except 3, to avoid conflict with IP address of
    - the target board)
- Subnet mask: 255.255.255.0 (d.2) Select button for [Use the following DNS server address]
- (d.3) Click [OK] for confirm the settings for the TCP/IPv4 properties.

	Internet Protocol Version 4 (TCP/IPv4) Properties	×	
	General		
	You can get IP settings assigned automatically if your network support this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.	s	
(d.	Obtain an IP address automatically		
	Use the following IP address:		
	Subnet mask:         255 . 255 . 255 . 0           Default gateway:		
(d.2	Obtain DNS server address automatically		
	Preferred DNS server:     .       Alternate DNS server:     .		
	Validate settings upon exit (d.3)		
	OK Can	cel	

Figure 3-7 Setting IP address on PC



4) Enable "Telnet Client" on PC

- In Windows 10 OS environment, under [Programs & Features] setting,
  - a. Click [Turn Windows features on or off] to open setup page
  - b. Tick the checkbox for [Telnet Client] in the pop up window
  - c. Click [OK] to confirm the settings.

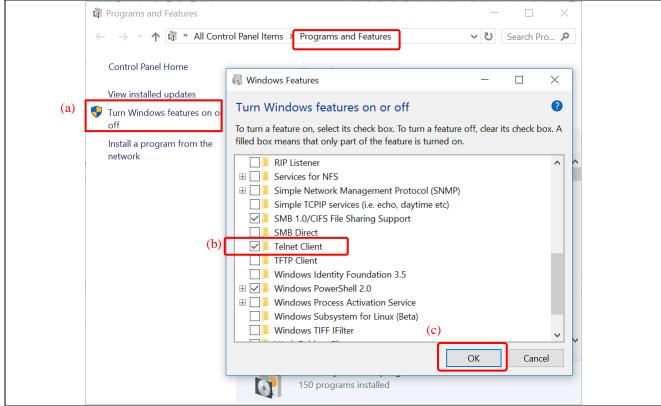


Figure 3-8 Enable "Telnet Client" on PC



# 3.3 Build and Debug Project

1) Build the project, click [Project]  $\rightarrow$  [Build Project]

File Edit Source Refactor Navigate Se	earch [	Project Renesas Views	Run Window Help
🐔 🔯 🔳 🌾 Debug 🗸 🗹 🖾 Sma	art_Co	Open Project	
🕒 Project Explorer 🛛 🖻 😫 🐲 🔻 🗖	₿ Sn	Close Project	
✓  Smart_Configurator_Example [Ha		Open Synergy Confi	guration
> 🕷 Binaries	10	Build All	Ctrl+Alt+B
> 🔊 Includes	11	Build Configurations	; >
✓ <sup>™</sup> src	13	Build Project	Ctrl+B
Y ≽ smc_gen	14	Build Working Set	\ <u>\</u>



2) Debug the code

[Run]  $\rightarrow$  [Debug Configurations...] or the downward arrow by the side of icon  $\rightarrow$  [Debug Configurations...] to open the "Debug Configurations" window

💕 workspace - Smart Configurator - e2 studio					_	
File Edit Navigate Search Project Renesa	as Views Ru	n W	indow Help			
🌾 🌾 🔳 🕸 🗸 🕹	C <sup>™</sup> Smart	_Confi	igurator_Example1 Deb 🗸 🔅			
📑 🗝 🔚 🐚   🥸 🕶 🗞 🕶 🔛 🧰 ! 💆	Ø 8	*	🖸 🗸 💊 🖬 🔗 🖬 🖢 🖓	- *> <> -	-	
	(2	)T	1 Smart_Configurator_Example1 Deb	bug	9   🛱	☆
ြာ Project Explorer 🕴 📄 🔄 🔽 🗖		_	Debug As	>	J Pa S	x
✓ ➢ Smart_Configurator_Example [Debug]			Debug Configurations	2		
> 🔊 Includes			Organize Favorites	-0		
> 😂 src						
> 🔁 Debug > 🗁 trash						
custom.bat						
makefile.init						
Smart_Configurator_Example.scfg						

Figure 3-10 Open Debug Configurations Window



- 3) Expand [Renesas GDB Hardware Debugging] and click on [Smart\_Configurator\_Example\_HardwareDebug]. Click on the [Debugger] tab, click on [Connection Settings] tab and set the following settings:
  - a. Debug Hardware: Select E1 (RX) or E2 Lite (RX) Target Device: R5F565NE (For RSK+65N-2MB)
    - R5F564ML (For RSK+64M)
  - b. Main Clock Source: EXTAL
  - c. Extal Frequency[MHz]: 24MHz
  - d. Permit Clock Source Change On Writing Internal Flash Memory: Yes
  - e. Power Target From The Emulator (MAX 200mA): Yes

Debug Configurations		×
eate, manage, and run configurations		1
		26
🛅 🗙   🖻 ‡ ▼	Name: (3n) art_Configurator_Example HardwareDebug	
/pe filter text	📄 Main 🚺 Debugger 🔪 🕨 Startup 🔲 <u>C</u> ommon 🧤 Sou	urce
C/C++ Application		
C/C++ Remote Application	Debug hardware E1 (RX) Target Device R	RSF565NE b.
EASE Script		
GDB Hardware Debugging	GDB Setting: Connection Settings Debug Tool Settings	
GDB Simulator Debugging (RH850) Java Applet	⊿ Clock	
Java Application	Main Clock Source	EXTAL C.
Launch Group	Extal Frequency[MHz]	<sup>24,0000</sup> d.
Remote Java Application	Permit Clock Source Change On Writing Internal Fla	ish Yes e.
Renesas GDB Hardware Debugging	▲ Connection with Target Board	(1.1.)
3) * Smart_Configurator_Example HardwareDebug [local]	Emulator	(Auto)
C <sup>*</sup> Renesas Simulator Debugging (RX, RL78)	Connection Type	JTag 👻
e heneses simulator blebagging (rot, hero)	JTag Clock Frequency[MHz]	6.00 -
	Fine Baud Rate[Mbps]	No
	Hot Plug	NO
	Power Target From The Emulator (MAX 200mA)	Yes f.
	Supply Voltage	3.3V
	CPU Operating Mode	551
	Register Setting	Single Chip 👻
	Mode pin	Single-chip mode v
	Change startup bank	No
	Startup bank	Bank 0 v
	Communication Mode	
	Mode	Debug Mode 👻
	Execute The User Program After Ending The Debugg	ger No 👻
	⊿ Flash	
	ID Code	FFFFFFFFFFFFFFFFFFFFFFF
		Reyert Apply
ter matched 13 of 15 items		Kever Apply
		Debug Close

Figure 3-11 Debug Configurations Example When Connecting to E1 Emulator



- 4) Click on the Startup tab, uncheck "Set breakpoint at: main"
- 5) Click [Debug] to start debugging.

ate, manage, and run configurations							Ŕ	Ì
Image: Second system         Image: Second system <th>ame: Smart_Configurator_Exar Main 🔅 Debugger 🕨 Star Initialization Commands Reset and Delay (seconds): Halt</th> <th>tup</th> <th>Source</th> <th></th> <th></th> <th></th> <th></th>		ame: Smart_Configurator_Exar Main 🔅 Debugger 🕨 Star Initialization Commands Reset and Delay (seconds): Halt	tup	Source				
Java Application Launch Group Remote Java Application Remote Java Application Remote Java Application		- Load image and symbols						
🔄 * Smart_Configurator_Example HardwareDebug [loca	0	Filename	Load type	Offset (hex)	On connect		Add	
Renesas Simulator Debugging (RX, RL78)		Program Binary [Smart	Image and Symbols	0	Yes		Edit	
							Remove	1
							Move up	1
							Move down	) 1
							Iviove down	
		Runtime Options						
		Set program counter at (hex	):					
(	4)	Set breakpoint at:	main					
		_						
		Run Commands						
er matched 13 of 15 items						Revert	Apply	1

Figure 3-12 Remove Breakpoint at Main



- 6) 'Confirm Perspective Switch' dialog may pop up, click [Yes] to continue.
- 7) Click **I** to start project execution



The following are information on how to suspend and stop the program in debug window. Do not execute it now.

1) To suspend the program execution, click suspend button  $\square$ 



2) To stop the program execution, click disconnect button M to end debug session.

File Edit Source Refact	or Navigate Search Project	Renesas Views Run	Window Help	(9)		
🐔 🔅 🔳 🔅 Debug	✓ Smart_Configurato	r_Exi 🗠 🌞 🗄 👻 🖫	💿   🛞 🕶 🍕 🕶 🗟   📮   🗙   i	▶ II <b>■ M</b> R.		
* D L M			🍇   i+	🗞 ▽ᄆ目		
† Debug ⊠			~ 1 -			
✓						
	<b>E</b> !	-15 Ston Debu				

Figure 3-15 Stop Debug



#### 3.4 Echo Server Operation

- 1) Open [Command Prompt] terminal in Windows OS environment
  - a. Enter command: 'Ping 192.168.0.3'
  - b. Reply from target board (192.168.0.3) can be observed. This indicates that the Ethernet connection between the board and PC is successful.

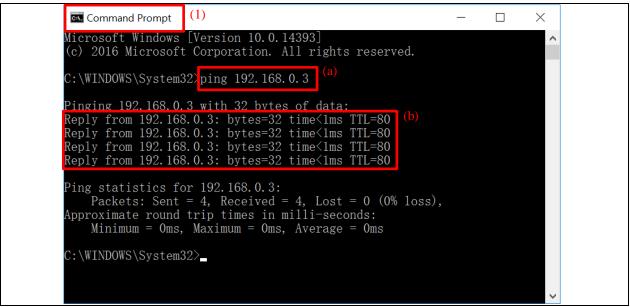


Figure 3-16 Command Prompt

2) Next, enter command: 'telnet 192.168.0.3 1024'

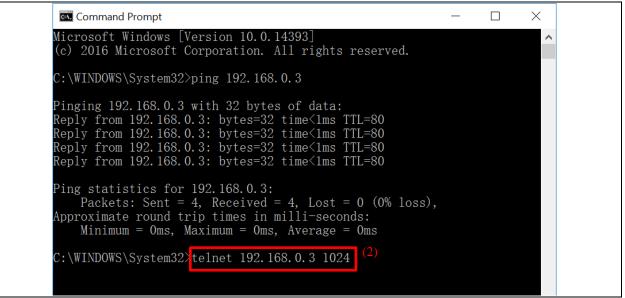


Figure 3-17 Command Prompt



- 3) In the pop up window [Telnet 192.168.0.3] :
  - a. Enter any characters and observed the echo back messages

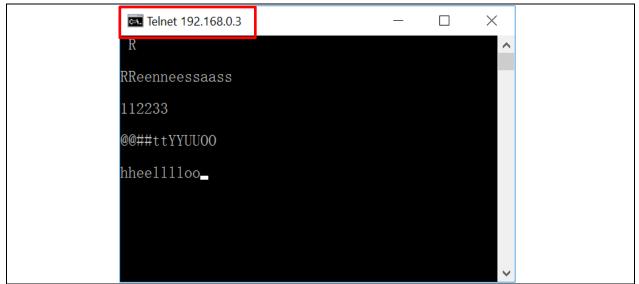


Figure 3-18 Telnet 192.168.0.3

# 3.5 Additional Debugging Assistance Tool (QE)

Renesas has a range of Quick and Effective Tool Solutions (QEs) as development tools for particular applications to assist and improve efficiency during development. Specific to this example of integrating TCP/IP function, QE for TCP/IP is recommended as it has several features to assist in debugging application based on TCP/IP function. Please refer to the following link for more information on QE and the type of supported applications for Renesas IDE:

QE: https://www.renesas.com/qe QE for TCP/IP: https://www.renesas.com/qe-tcpip



# Website and Support

• Renesas Electronics Website http://www.renesas.com/

• Inquiries

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

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



# **Revision History**

		Descript	ion	
Rev.	Date	Page	Summary	
1.00	Mar. 27, 2018	-	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. Handling of Unused Pins
  - Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.
  - The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
- 2. Processing at Power-on

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

- The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
  - In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

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

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

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

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

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

#### Notice

- Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for
  the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by
  you or third parties arising from the use of these circuits, software, or information.
- Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
- 3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
- 4. You shall 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; underse repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics data sheet, use's manual or other Renesas Electronics document.

- 6. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
- 7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
- 8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
- 9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
- 10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
- 11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
- 12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.
- (Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries
- (Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

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

(Rev.4.0-1 November 2017)



#### SALES OFFICES

#### **Renesas Electronics Corporation**

http://www.renesas.com

Renease Electronics America Inc.1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.Tel: +1-408-432-8888, Fax: +1-408-434-5351Renease Electronics Canada Limited251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3Tel: +1-905-237-2004Renease Electronics Europe LimitedDukes Meadow, Milboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.KTel: +44-1628-651-700, Fax: +44-1628-661-804Renease Electronics Europe GmbHArcadiastrasse 10, 40472 Düsseldorf, Germany1e1: +49-21-6503-0, Fax: +49-211-6503-1327Renease Electronics (China) Co., Ltd.Room 7109 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China1e1: +86-10-9235-1155, Fax: +88-10-8235-7679Renease Electronics (Shanghai) Co., Ltd.Unit 301, Tower A, Central Towers, S55 Langao Road, Putuo District, Shanghai, 200333 P. R. ChinaTel: +86-12226-0888, Fax: +88-21-2226-0999Renease Electronics (Shanghai) Co., Ltd.Unit 101-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong KongTel: +86-21-2226-088, Fax: +882-218-2902Renease Electronics Taiwan Co., Ltd.137. No. 363, Fu Shing North Road, Taipei 10543, TaiwanTel: +882-2455-4175-9807, Fax: +882-518-59870Renease Electronics Malaysia Sch.Bhd.Unit 1207, Block B, Meara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, MalaysiaTel: +86-73-930, Fax: +60-3-7955-9510Reneases Electronics Korea Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia</t