Smart Configurator for RL78 Plug-in in e^2 studio 2023-01
Smart Configurator for RL78 V1.5.0

Release Note

Introduction

Thank you for using the Smart Configurator for RL78.
This document describes the restrictions and points for caution. Read this document before using the product.

Contents

1. Introduction....3
1.1 System requirements....3
1.1.1 Windows PC....3
1.1.2 Linux PC....3
1.1.3 Development Environments....3

2. Support List....4
2.1 Support Devices List....4
2.2 Support Components List....5
2.3 New support....7
2.3.1 Support RL78/F23 and RL78/G22 devices....7
2.3.2 BSP (Board Support Package) revision update....7
2.3.1 User code protection feature for Smart Configurator Code Generation component....7
2.3.2 Support CC-RL C++ project....9
2.3.3 New support of Linux OS....10

3. Changes....11
3.1 Correction of issues/limitations....11
3.1.1 Fixed the issue of BSP or RL78 SIS modules files are not re-generated when changing version....11
3.1.2 Fixed the issue of UARTA reception error interrupt generation....11
3.1.3 Fixed the issue of ROM overflow build error for LLVM project without any component added....11
3.1.4 Fixed the issue of the r_bsp_config.h console message is not generated after changing generated location....12
3.2 Specification changes....13
3.2.1 Improvement for layout of Smart Configurator Component Preference page for better user experience....13
3.2.2 Improvement for “Flash serial programming operation setting” default setting....13
3.2.3 Improvement for MDIV divider setting....14
3.2.4 Improvement for keeping the user defined interrupt code in "r_cg_inthandler.c" file....15
3.2.5 Improvement for RTC wait time is not enough issue....15

4. List of RENESAS TOOL NEWS AND TECHNICAL UPDATE 16

5. Points for Limitation 17
5.1 List of Limitation 17
5.2 Details of Limitation 17
5.2.1 Note on extra help document issue 17
5.2.2 Note on ELCL D flip flop component GUI warning display incorrectly 18
5.2.3 Note on the unsupported setting items for some ELCL components 19
5.2.4 Note on the “Show view” dialog in Standalone Smart Configurator 19
5.2.5 Note on e^2 studio becomes unresponsive when Developer Assistance is loading 19

6. Points for Caution 20
6.1 List of Caution 20
6.2 Details of Caution 21
6.2.1 Note on the build error message such as “section .bss virtual address range overlaps with .dtc_vectortable” 21
6.2.2 Note on the installation of the Smart Configurator 22
6.2.3 Note on using TRDIOA0 for Input capture and TRDIOB0 for Output compare at same time 22
6.2.4 Note on pulse width calculation of Timer RD input capture function 22
6.2.5 Note on using Touch middleware and UART communication components 23
6.2.6 Note on the include path update issue when renaming the component’s configuration name 23
6.2.7 Note on TAU Input Signal High/Low level Measurement component 25
6.2.8 Note on ADCKL setting when ADCEN is set to 0 25
6.2.9 Note on CC-RL V1.12 C++ language project 26
6.2.10 Note on browsing “Release Notes” and “Tool News” URL from the help menu 26
6.2.11 Note on using user code protection feature 27

Revision History 28

1. Introduction

Smart Configurator is a utility for combining software to meet your needs. It supports the following three functions related to the embedding of Renesas drivers in your systems: importing middleware, generating driver code, and setting pins.
Smart Configurator for RL78 V1.5.0 is equivalent to Smart Configurator for RL78 Plugin in e^2 studio 2023-01.

1.1 System requirements

The operating environment is as follows.

1.1.1 Windows PC

- System: x64/x86 based processor
  Windows® 11
  Windows® 10 (64-bit version)
  Windows® 8.1 (64-bit version)
- Memory capacity: We recommend 4 GB or more
- Capacity of hard disk: At least 300 MB of free space.
- Display: Graphics resolution should be at least 1024 x 768, and the mode should display at least 65,536 colors.
- Processor: 1 GHz or higher (must support hyper-threading, multi-core CPUs)

1.1.2 Linux PC

Only Smart Configurator for RL78 plug-in in e2 studio 2023-01 is supported on Linux OS.

- System: x64 based processor, 2 GHz or faster (with multicore CPUs)
  Ubuntu 22.04 LTS Desktop (64-bit version)
  Ubuntu 20.04 LTS Desktop (64-bit version)
- Memory capacity: We recommend 2 GB or more.
- Capacity of hard disk: At least 2 GB of free space.

1.1.3 Development Environments

- Renesas Electronics Compiler for RL78 [CC-RL] V1.12 or later
- LLVM for Renesas RL78 10.0.0.202209 or later
- IAR Embedded Workbench for Renesas RL78 V4.21.3 or later
- SMS Assembler $^{\text{Note}}$ V1.00.00 or later

Note:

If you want to add SMS Assembler to e^2 studio, install it from the integrated installer of e^2 studio 21-04 or later. (e^2 studio)

As with other compilers, select and install from the [Additional Software] - [Renesas Toolchains & Utilities] tab of the e^2 studio setup wizard.

2. Support List

2.1 Support Devices List

Below is a list of devices supported by the Smart Configurator for RL78 V1.5.0.

Table 2-1 Support Devices

Group (HW Manual number)	PIN	Device name
RL78/G23 Group (R01UH0896EJ0120)	30pin	R7F100GAFxSP, R7F100GAGxSP, R7F100GAHxSP, R7F100GAJxSP
	32pin	R7F100GBFxNP, R7F100GBGxNP, R7F100GBHxNP, R7F100GBJxNP, R7F100GBFxFP, R7F100GBGxFP, R7F100GBHxFP, R7F100GBJxFP
	36pin	R7F100GCFxLA, R7F100GCGxLA, R7F100GCHxLA, R7F100GCJxLA
	40pin	R7F100GEFxNP, R7F100GEGxNP, R7F100GEHxNP, R7F100GEJxNP
	44pin	R7F100GFFxFP, R7F100GFGxFP, R7F100GFHxFP, R7F100GFJxFP, R7F100GFKxFP, R7F100GFLxFP, R7F100GFNxFP
	48pin	R7F100GGFxFB, R7F100GGGxFB, R7F100GGHxFB, R7F100GGJxFB, R7F100GGKxFB, R7F100GGLxFB, R7F100GGNxFB, R7F100GGFxNP, R7F100GGGxNP, R7F100GGHxNP, R7F100GGJxNP, R7F100GGKxNP, R7F100GGLxNP, R7F100GGNxNP
	52pin	R7F100GJFxFA, R7F100GJGxFA, R7F100GJHxFA, R7F100GJJxFA, R7F100GJKxFA, R7F100GJLxFA, R7F100GJNxFA
	64pin	R7F100GLFxFA, R7F100GLGxFA, R7F100GLHxFA, R7F100GLJxFA, R7F100GLKxFA, R7F100GLLxFA, R7F100GLNxFA, R7F100GLFxFB, R7F100GLGxFB, R7F100GLHxFB, R7F100GLJxFB, R7F100GLKxFB, R7F100GLLxFB, R7F100GLNxFB, R7F100GLFxLA, R7F100GLGxLA, R7F100GLHxLA, R7F100GLJxLA, R7F100GLKxLA, R7F100GLLxLA, R7F100GLNxLA
	80pin	R7F100GMGxFA, R7F100GMHxFA, R7F100GMJxFA, R7F100GMKxFA, R7F100GMLxFA, R7F100GMNxFA, R7F100GMGxFB, R7F100GMHxFB, R7F100GMJxFB, R7F100GMKxFB, R7F100GMLxFB, R7F100GMNxFB
	100pin	R7F100GPGxFB, R7F100GPHxFB, R7F100GPJxFB, R7F100GPKxFB, R7F100GPLxFB, R7F100GPNxFB, R7F100GPGxFA, R7F100GPHxFA, R7F100GPJxFA, R7F100GPKxFA, R7F100GPLxFA, R7F100GPNxFA
	128pin	R7F100GSJxFB, R7F100GSKxFB, R7F100GSLxFB, R7F100GSNxFB
RL78/F24 Group (R01UH0944EJ0100)	32pin	R7F124FBJ3xNP, R7F124FBJ4xNP, R7F124FBJ5xNP
	48pin	R7F124FGJ3xFB, R7F124FGJ4xFB, R7F124FGJ5xFB
	64pin	R7F124FLJ3xFB, R7F124FLJ4xFB, R7F124FLJ5xFB
	80pin	R7F124FMJ3xFB, R7F124FMJ4xFB, R7F124FMJ5xFB
	100pin	R7F124FPJ3xFB, R7F124FPJ4xFB, R7F124FPJ5xFB
RL78/G15 Group (R01UH0959EJ0100)	8pin	R5F12008xNS, R5F12007xNS
	10pin	R5F12018xSP, R5F12017xSP
	16pin	R5F12048xNA, R5F12047xNA, R5F12048xSP, R5F12047xSP
	20pin	R5F12068xSP, R5F12067xSP
RL78/F23 Group (R01UH0944EJ0100)	32pin	R7F123FBG3xNP, R7F123FBG4xNP, R7F123FBG5xNP
	48pin	R7F123FGG3xFB, R7F123FGG4xFB, R7F123FGG5xFB
	64pin	R7F123FLG3xFB, R7F123FLG4xFB, R7F123FLG5xFB
	80pin	R7F123FMG3xFB, R7F123FMG4xFB, R7F123FMG5xFB
RL78/G22 Group (R01UH0978EJ0100)	16pin	R7F102G4ExNP, R7F102G4CxNP
	20pin	R7F102G6ExSP, R7F102G6CxSP
	24pin	R7F102G7ExNP, R7F102G7CxNP
	25pin	R7F102G8ExLA, R7F102G8CxLA
	30pin	R7F102GAExSP, R7F102GACxSP
	32pin	R7F102GBExNP, R7F102GBCxNP, R7F102GBExFP, R7F102GBCxFP
	36pin	R7F102GCExLA, R7F102GCCxLA
	40pin	R7F102GEExNP, R7F102GECxNP
	44pin	R7F102GFExFP, R7F102GFCxFP
	48pin	R7F102GGExFB, R7F102GGExNP, R7F102GGCxFB, R7F102GGCxNP

2.2 Support Components List

Below is a list of Components supported by the Smart Configurator for RL78 V1.5.0.

Table 2-2 Support Components (1/2) ✓: Support, -: Non-support

No	Components	Mode	RL78/G23	RL78/F24	RL78/G15	RL78/F23	RL78/G22	Remarks
1	12 Bit A/D Single Scan	-	-	✓	-	✓	-	
2	12 Bit A/D Continuous Scan	-	-	✓	-	✓	-	
3	12 Bit A/D Group Scan	-	-	✓	-	✓	-	
4	A/D Converter	-	✓	-	✓	-	✓	
5	Clock Output/Buzzer Output Controller	-	✓	✓	✓	✓	✓	
6	Comparator	-	✓	✓	✓	-	-	
7	D/A Converter	-	✓	✓	-	-	-	
8	Data Transfer Controller	-	✓	✓	-	✓	✓	
9	Delay Counter	-	✓	✓	✓	✓	✓	
10	Divider Function	-	✓	✓	✓	✓	✓	
11	Event Link Controller	-	−	✓	−	-	✓	
12	External Event Counter	-	✓	✓	✓	✓	✓	
13	IIC Communication (Master mode)	-	✓	✓	✓	✓	✓	
14	IIC Communication (Slave mode)	-	✓	✓	✓	✓	✓	
15	Input Capture Function	-	−	✓	−	✓	−	
16	Input Pulse Interval/Period Measurement	-	✓	✓	✓	✓	✓	
17	Input Signal High-/Low-Level Width Measurement	-	✓	✓	✓	✓	✓	
18	Interrupt Controller	-	✓	✓	✓	✓	✓	
19	Interval Timer	8 bit count mode	✓	✓	✓	✓	✓	
		12 bit count mode	−	−	✓	-	−	
		16 bit count mode	✓	✓	✓	✓	✓	
		16 bit capture mode	✓	-	-	-	✓	
		32 bit count mode	✓	-	-	-	✓	
20	Key Interrupt	-	✓	✓	-	✓	✓	
21	One-Shot Pulse Output	One-Shot Pulse Output	✓	✓	✓	✓	✓	
		Two-Channel Input with One-Shot Pulse Output	−	−	✓	−	−	
22	Output Compare Function	-	−	✓	−	✓	−	
23	Ports	-	✓	✓	✓	✓	✓	

Table 2-3 Support Components (2/2) ✓: Support, -: Non-support

No	Components	Mode	RL78/G23	RL78/F24	RL78/G15	RL78/F23	RL78/G22	Remarks
24	PWM Option Unit A	-	−	✓	−	✓	−	
25	PWM Output	PWM mode	✓	✓	✓	✓	✓	
		PWM3 mode	−	✓	−	✓	−	
		Extended PWM mode	−	✓	−	✓	−	
26	Real-Time Clock	-	✓	✓	−	✓	✓	
27	Remote Control Signal Receiver	-	✓	-	−	-	-	
28	SNOOZE Mode Sequencer	-	✓	-	−	-	✓	
29	SPI (CSI) Communication	Transmission	✓	✓	✓	✓	✓	
		Reception	✓	✓	✓	✓	✓	
		Transmission/reception	✓	✓	✓	✓	✓	
30	Square Wave Output	-	✓	✓	✓	✓	✓	
31	Three-phase PWM Output	Reset Synchronous PWM Mode	−	✓	−	✓	−	
		Complementary PWM Mode	−	✓	−	✓	−	
		Extended Complementary PWM Mode	−	✓	−	✓	−	
32	UART Communication	Transmission	✓	✓	✓	✓	✓	
		Reception	✓	✓	✓	✓	✓	
		Transmission/reception	✓	✓	✓	✓	✓	
33	Voltage Detector	-	✓	✓	-	✓	✓	
34	Watchdog Timer	-	✓	✓	✓	✓	✓	
35	Logic & Event Link Controller	-	✓	-	-	-	-	Need download in Smart Configurator RL78

2.3 New support

2.3.1 Support RL78/F23 and RL78/G22 devices

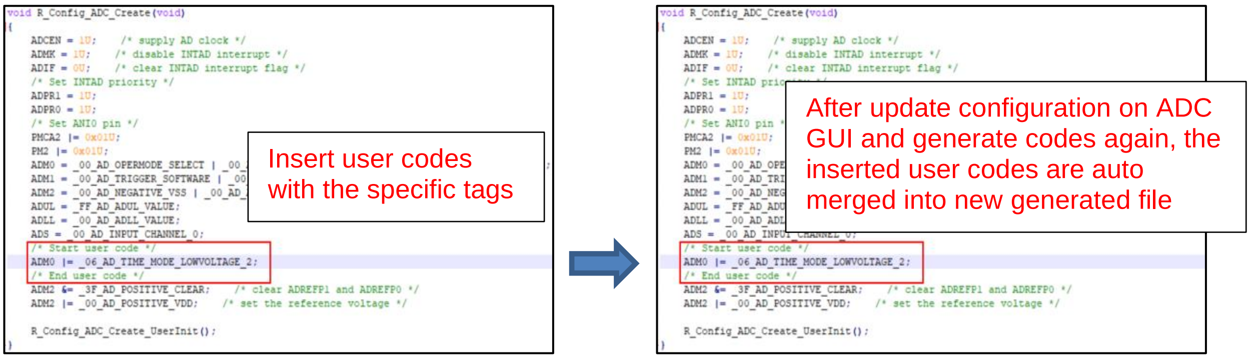
See 2.1 Support Devices List for details on supported packages.

2.3.2 BSP (Board Support Package) revision update

BSP rev1.40 is supported and will be added as default BSP when creating Smart Configurator project.

2.3.1 User code protection feature for Smart Configurator Code Generation component

From Smart Configurator for RL78 V1.4.0 or previous version, user code can be protected when user insert the user code into the fixed location defined inside the generated files of Code Generation component.
From Smart Configurator for RL78 V1.5.0, an enhanced user code protection feature has been implemented which allows user codes to be added to any locations with the specific tags (/* Start user code */ … /* End user code */) as shown in **Figure 2-1**. Inserted user codes will be protected and automatically merged into generated files in the next code generation.
This feature will be available from Smart Configurator for RL78 Plug-in in e^2 studio 2023-01.



```
void R_Config_ADC_Create(void)
{
    ADCEN = 1U;    /* supply AD clock */
    ADMK = 1U;    /* disable INTAD interrupt */
    ADIF = 0U;    /* clear INTAD interrupt flag */
    /* Set INTAD priority */
    ADPR1 = 1U;
    ADPR0 = 1U;
    /* Set ANI0 pin */
    PMCA2 |= 0x01U;
    PM2 |= 0x01U;
    ADM0 = _00_AD_OPERMODE_SELECT | _00_
    ADM1 = _00_AD_TRIGGER_SOFTWARE | _00
    ADM2 = _00_AD_NEGATIVE_VSS | _00_AD_
    ADUL = _FF_AD_ADUL_VALUE;
    ADLL = _00_AD_ADLL_VALUE;
    ADS = _00_AD_INPUT_CHANNEL_0;
    /* Start user code */
    ADM0 |= _06_AD_TIME_MODE_LOWVOLTAGE_2;
    /* End user code */
    ADM2 &= _3F_AD_POSITIVE_CLEAR;    /* clear ADREFP1 and ADREFP0 */
    ADM2 |= _00_AD_POSITIVE_VDD;    /* set the reference voltage */

    R_Config_ADC_Create_UserInit();
}
```

Insert user codes with the specific tags

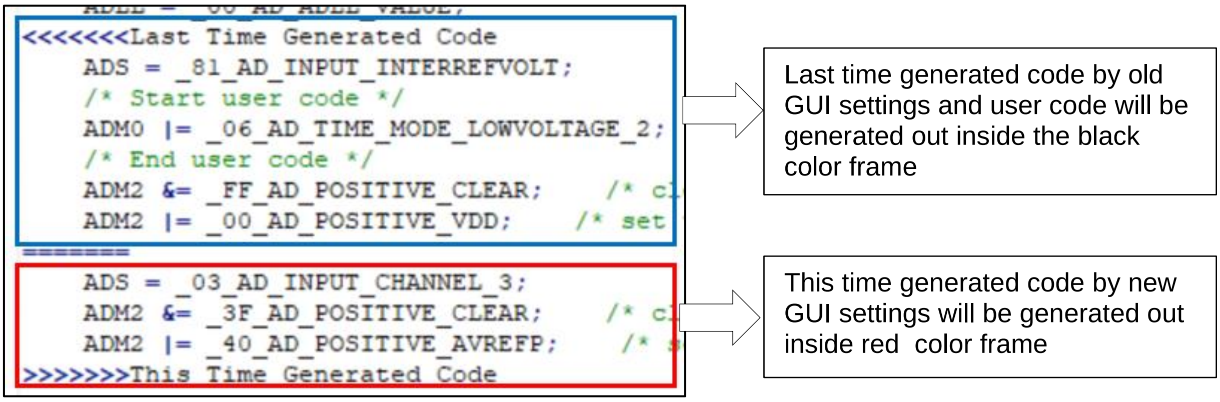
```
void R_Config_ADC_Create(void)
{
    ADCEN = 1U;    /* supply AD clock */
    ADMK = 1U;    /* disable INTAD interrupt */
    ADIF = 0U;    /* clear INTAD interrupt flag */
    /* Set INTAD prio
    ADPR1 = 1U;
    ADPR0 = 1U;
    /* Set ANI0 pin *
    PMCA2 |= 0x01U;
    PM2 |= 0x01U;
    ADM0 = _00_AD_OPE
    ADM1 = _00_AD_TRI
    ADM2 = _00_AD_NEG
    ADUL = _FF_AD_ADU
    ADLL = _00_AD_ADL
    ADS = _00_AD_INPU
    /* Start user code */
    ADM0 |= _06_AD_TIME_MODE_LOWVOLTAGE_2;
    /* End user code */
    ADM2 &= _3F_AD_POSITIVE_CLEAR;    /* clear ADREFP1 and ADREFP0 */
    ADM2 |= _00_AD_POSITIVE_VDD;    /* set the reference voltage */

    R_Config_ADC_Create_UserInit();
}
```

After update configuration on ADC GUI and generate codes again, the inserted user codes are auto merged into new generated file

Figure 2-1 User code protection with auto merge

However, if the lines of generated codes before and after the inserted user codes are updated due to the new GUI configuration, merge conflict code will be generated out as shown in **Figure 2-2.**

(By upgrading the version of Smart Configurator, merge conflicts will be generated if the GUI configuration in new environment updated the lines of generated codes before and after the inserted user code.)



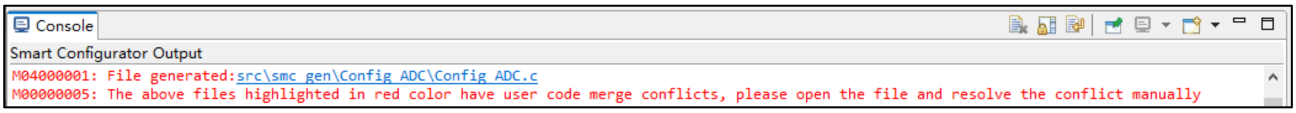
```
    ADLL = _00_AD_ADLL_VALUE;
<<<<<<<Last Time Generated Code
    ADS = _81_AD_INPUT_INTERREFVOLT;
    /* Start user code */
    ADM0 |= _06_AD_TIME_MODE_LOWVOLTAGE_2;
    /* End user code */
    ADM2 &= _FF_AD_POSITIVE_CLEAR;     /* cl
    ADM2 |= _00_AD_POSITIVE_VDD;     /* set
=======
    ADS = _03_AD_INPUT_CHANNEL_3;
    ADM2 &= _3F_AD_POSITIVE_CLEAR;     /* cl
    ADM2 |= _40_AD_POSITIVE_AVREFP;     /* s
>>>>>>>This Time Generated Code
```

Last time generated code by old GUI settings and user code will be generated out inside the black color frame

This time generated code by new GUI settings will be generated out inside red color frame

Figure 2-2 User code protection with merge conflict

As shown in **Figure 2-3**, conflict message will be generated out to the Smart Configurator console when the merge conflict occurs.

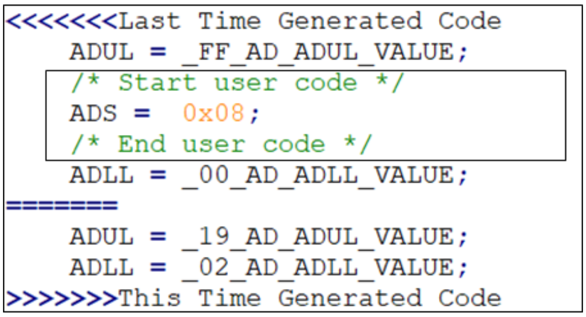


```
M04000001: File generated:src\smc_gen\Config_ADC\Config_ADC.c
M00000005: The above files highlighted in red color have user code merge conflicts, please open the file and resolve the conflict manually
```

Figure 2-3 The merge conflict message outputted in the Smart Configurator console

To resolve this merge conflict, open the highlighted conflict files and follow the steps below to solve the merge conflicts manually.

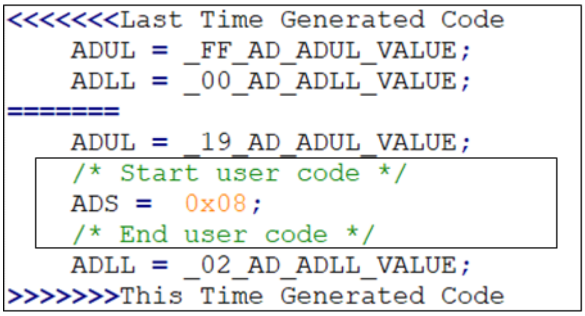
1) Copy the user code that was generated in **Figure 2-4**.



```
<<<<<<<Last Time Generated Code
    ADUL = _FF_AD_ADUL_VALUE;
    /* Start user code */
    ADS =  0x08;
    /* End user code */
    ADLL = _00_AD_ADLL_VALUE;
=======
    ADUL = _19_AD_ADUL_VALUE;
    ADLL = _02_AD_ADLL_VALUE;
>>>>>>>This Time Generated Code
```

Figure 2-4 Generated conflict code

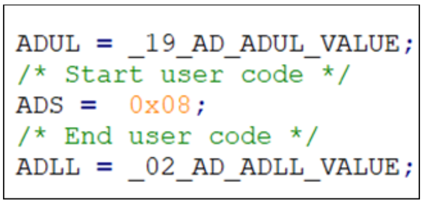
2) Paste the user code into the new position (inside the code that was generated by new GUI settings) as shown in **Figure 2-5**.



```
<<<<<<<Last Time Generated Code
    ADUL = _FF_AD_ADUL_VALUE;
    ADLL = _00_AD_ADLL_VALUE;
=======
    ADUL = _19_AD_ADUL_VALUE;
    /* Start user code */
    ADS =  0x08;
    /* End user code */
    ADLL = _02_AD_ADLL_VALUE;
>>>>>>>This Time Generated Code
```

Figure 2-5 Generated conflict code

3) Remove last time generated code and the conflicts commend (<<<<<<<Last Time Generated Code, ======= and >>>>>>>This Time Generated Code) as shown in **Figure 2-6**.



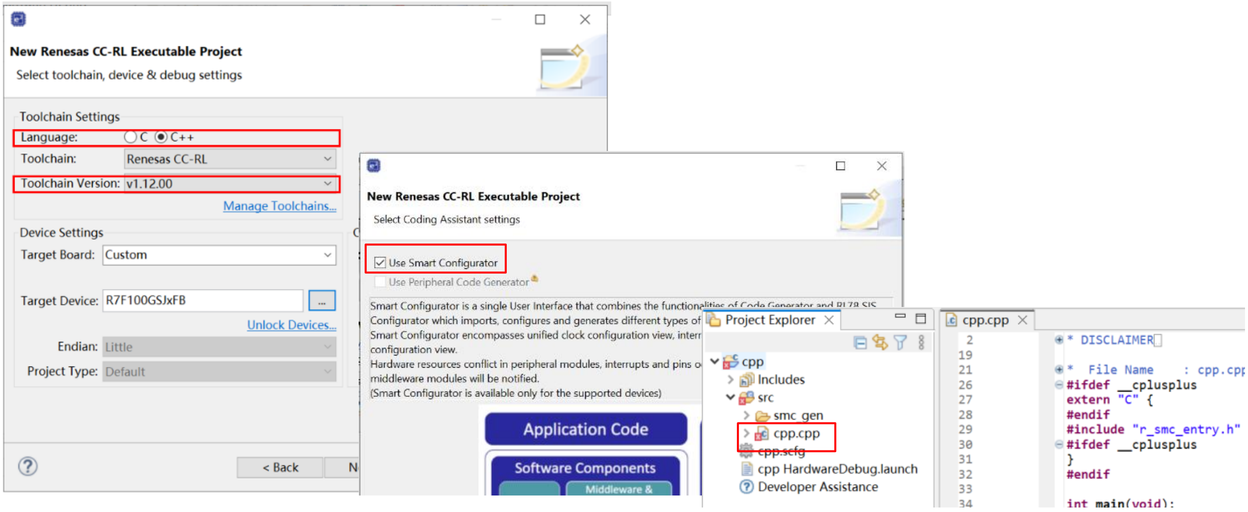
```
ADUL = _19_AD_ADUL_VALUE;
/* Start user code */
ADS =  0x08;
/* End user code */
ADLL = _02_AD_ADLL_VALUE;
```

Figure 2-6 The codes after resolving the merge conflict

2.3.2 Support CC-RL C++ project

From Smart Configurator for RL78 V1.5.0, it can be used in CC-RL C++ project.

- In e^2 studio[*1], when C++ language is selected, Smart Configurator can be selected and generated in this project.



New Renesas CC-RL Executable Project
Select toolchain, device & debug settings

Toolchain Settings
Language: ○ C ● C++
Toolchain: Renesas CC-RL
Toolchain Version: v1.12.00
Manage Toolchains...

Device Settings
Target Board: Custom
Target Device: R7F100GSJxFB
Unlock Devices...
Endian: Little
Project Type: Default

New Renesas CC-RL Executable Project
Select Coding Assistant settings

☑ Use Smart Configurator
□ Use Peripheral Code Generator

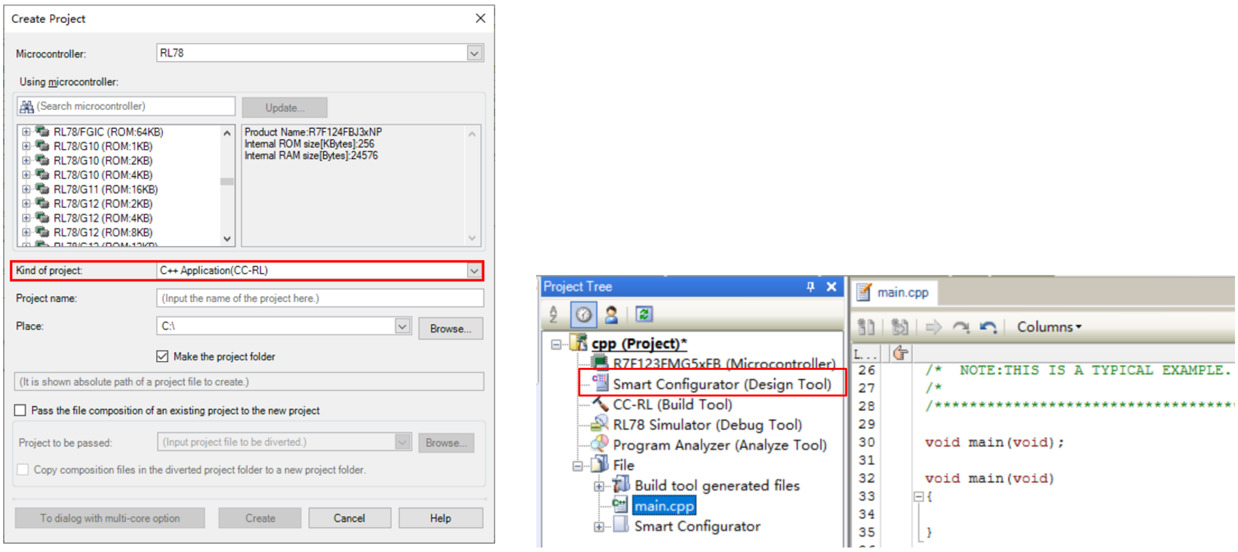
Smart Configurator is a single User Interface that combines the functionalities of Code Generator and RL78 SIS Configurator which imports, configures and generates different types of Smart Configurator encompasses unified clock configuration view, interr configuration view.
Hardware resources conflict in peripheral modules, interrupts and pins o middleware modules will be notified.
(Smart Configurator is available only for the supported devices)

```
* DISCLAIMER
*  File Name    : cpp.cpp
#ifdef __cplusplus
extern "C" {
#endif
#include "r_smc_entry.h"
#ifdef __cplusplus
}
#endif

int main(void);
```

Figure 2-7 C++ language project in e^2 studio

Note 1: C++ language project only be supported by CC-RL V1.12.00 in e^2 studio.

- In CS+, after C++ Application (CC-RL) project is selected, Smart Configurator is generated together in the CC-RL project tree.



Create Project

Microcontroller: RL78
Using microcontroller:
(Search microcontroller) Update...

Product Name:R7F124FBJ3xNP
Internal ROM size[KBytes]:256
Internal RAM size[Bytes]:24576

Kind of project: C++ Application(CC-RL)
Project name: (Input the name of the project here.)
Place: C:\ Browse...
☑ Make the project folder
(It is shown absolute path of a project file to create.)
□ Pass the file composition of an existing project to the new project
Project to be passed: (Input project file to be diverted.) Browse...
□ Copy composition files in the diverted project folder to a new project folder.

```
/*  NOTE:THIS IS A TYPICAL EXAMPLE.
/*
/*************************************

void main(void);

void main(void)
{

}
```

Figure 2-8 C++ Application project in CS+

- Please manually add the following highlighted lines of code into main.cpp

```
void main(void);
#ifdef __cplusplus
extern "C" {
#include "r_smc_entry.h"
}
#endif

void main(void)
{

}
```

2.3.3 New support of Linux OS

From Smart Configurator for RL78 V1.5.0, Smart Configurator will be supported on Linux OS.
Please refer to 1.1.2 Linux PC for the information of supported Linux.

3. Changes

This chapter describes changes to the Smart Configurator for RL78 V1.5.0.

3.1 Correction of issues/limitations

Table 3-1 List of Correction of issues/limitations ✓: Applicable, -: Not Applicable

No	Description	RL78/G23	RL78/F24	RL78/G15	RL78/F23	RL78/G22	Remarks
1	Fixed the issue of BSP or RL78 SIS modules files are not re-generated when changing version	✓	✓	✓	−	−	
2	Fixed the issue of UARTA reception error interrupt generation	✓	−	−	−	−	
3	Fixed the issue of ROM overflow build error for LLVM project without any component added	−	−	✓	−	−	
4	Fixed the issue of the r_bsp_config.h console message is not generated after changing generated location	✓	✓	✓	−	−	

3.1.1 Fixed the issue of BSP or RL78 SIS modules files are not re-generated when changing version

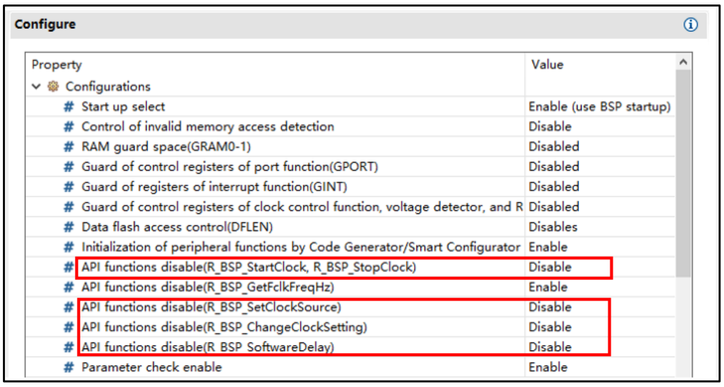
When performing [Change version…] for BSP or RL78 SIS modules, version seems be changed successfully in the "Overview" summary. But some of the source code files are still the old version. As result, user can't aware the version unchanged and it may cause build error in some cases.
This issue has been fixed from Smart Configurator for RL78 V1.5.0.

3.1.2 Fixed the issue of UARTA reception error interrupt generation

When using UARTA component with reception mode and transmission/reception mode, Smart Configuration generation code only supports to handle INTUREn (n = 0, 1) interrupt when a reception error occurs. Even if user select "INTUR interrupt occurs" on UI, the generation code still only handle INTUREn (n = 0, 1) interrupt.
This issue has been fixed from Smart Configurator for RL78 V1.5.0.

3.1.3 Fixed the issue of ROM overflow build error for LLVM project without any component added

When using RL78/G15 8pin and 4KB ROM product, there is ROM overflow build error in case of a LLVM project without adding any component. This is issue has been fixed by disabling BSP API function in default.



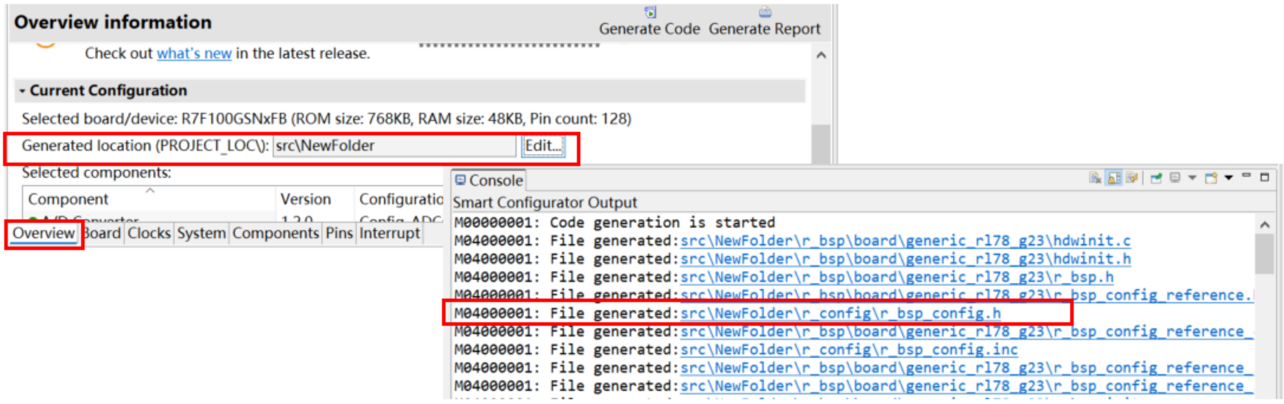
Configure

Property	Value
Configurations	
# Start up select	Enable (use BSP startup)
# Control of invalid memory access detection	Disable
# RAM guard space(GRAM0-1)	Disabled
# Guard of control registers of port function(GPORT)	Disabled
# Guard of registers of interrupt function(GINT)	Disabled
# Guard of control registers of clock control function, voltage detector, and R	Disabled
# Data flash access control(DFLEN)	Disables
# Initialization of peripheral functions by Code Generator/Smart Configurator	Enable
# API functions disable(R_BSP_StartClock, R_BSP_StopClock)	Disable
# API functions disable(R_BSP_GetFclkFreqHz)	Enable
# API functions disable(R_BSP_SetClockSource)	Disable
# API functions disable(R_BSP_ChangeClockSetting)	Disable
# API functions disable(R_BSP_SoftwareDelay)	Disable
# Parameter check enable	Enable

Figure 3-1 BSP API function

3.1.4 Fixed the issue of the r_bsp_config.h console message is not generated after changing generated location

If changing the generated location to a new folder at Overview page, then click [Generate Code], the console message of r_bsp_config.h is not generated. This issue has been fixed from Smart Configurator for RL78 V1.5.0



Selected board/device: R7F100GSNxFB (ROM size: 768KB, RAM size: 48KB, Pin count: 128)
Generated location (PROJECT_LOC\): src\NewFolder Edit...
Selected components:

Smart Configurator Output

```
M00000001: Code generation is started
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\hdwinit.c
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\hdwinit.h
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\r_bsp.h
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\r_bsp_config_reference.
M04000001: File generated:src\NewFolder\r_config\r_bsp_config.h
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\r_bsp_config_reference_
M04000001: File generated:src\NewFolder\r_config\r_bsp_config.inc
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\r_bsp_config_reference_
M04000001: File generated:src\NewFolder\r_bsp\board\generic_rl78_g23\r_bsp_config_reference_
```

Figure 3-2 The console message of r_bsp_config.h

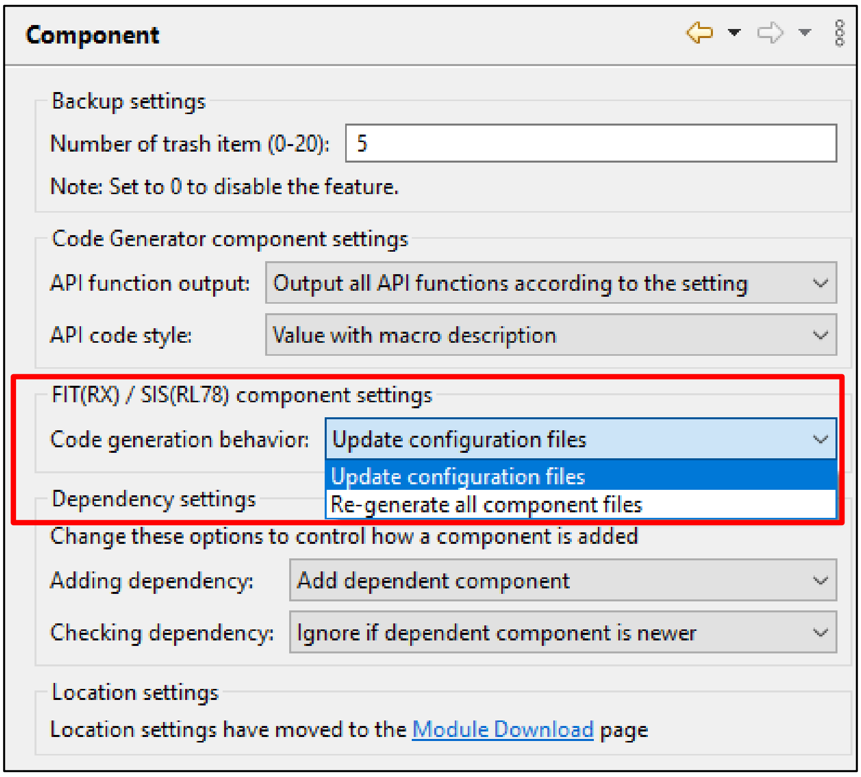
3.2 Specification changes

Table 3-2 List of Specification changes ✓: Applicable, -: Not Applicable

No	Description	RL78/G23	RL78/F24	RL78/G15	RL78/F23	RL78/G22	Remarks
1	Improvement for layout of Smart Configurator Component Preference page for better user experience	✓	✓	✓	✓	✓	
2	Improvement for “Flash serial programming operation setting” default setting	–	✓	–	✓	–	
3	Improvement for MDIV divider setting	–	✓	–	✓	–	
4	Improvement for keeping the user defined interrupt code in "r_cg_inthandler.c" file	✓	✓	✓	✓	✓	
5	Improvement for RTC wait time is not enough issue	✓	–	–	–	✓	

3.2.1 Improvement for layout of Smart Configurator Component Preference page for better user experience

From Smart Configurator for RL78 V1.5.0, the layout for Smart Configurator Component Preferences page has been improved for better user experience of the "code generation behavior" feature. "Update configuration files" is selected by default.


Component

Backup settings
Number of trash item (0-20): 5
Note: Set to 0 to disable the feature.

Code Generator component settings
API function output: Output all API functions according to the setting
API code style: Value with macro description

FIT(RX) / SIS(RL78) component settings
Code generation behavior: Update configuration files
- Update configuration files
- Re-generate all component files

Dependency settings
Change these options to control how a component is added
Adding dependency: Add dependent component
Checking dependency: Ignore if dependent component is newer

Location settings
Location settings have moved to the Module Download page

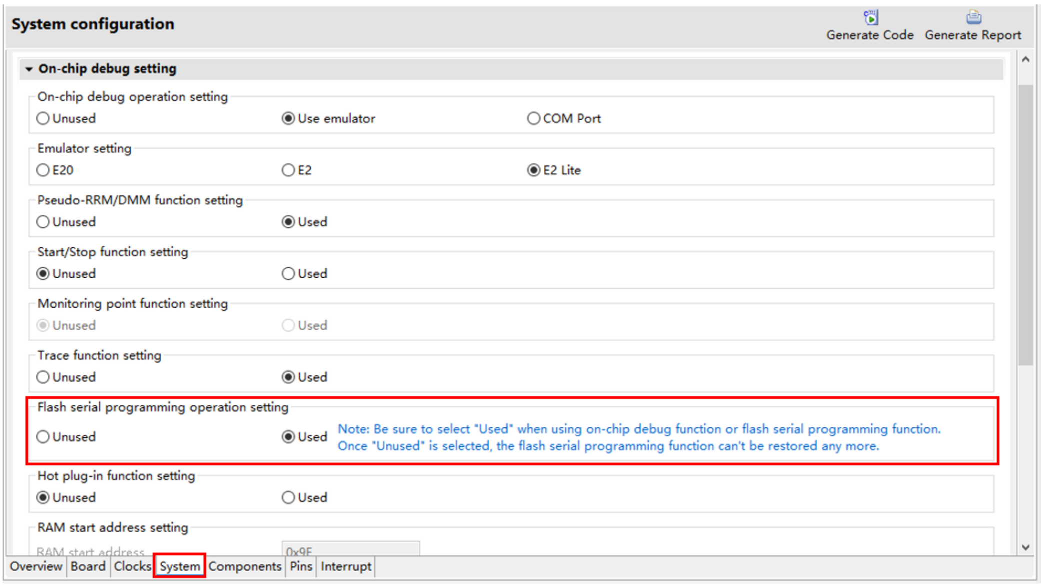
Figure 3-3 New layout for Smart Configurator Component Preference page

If "Update configuration files" is being selected and generate code, Smart Configurator will check whether the files are existing inside the user project. If the file exists, the file will not be overwritten. However, configuration files (e.g., xxx_config.h) will still be refreshed when code is generated.

If "Re-generate all component files" being selected and generate code, Smart Configurator does not check the existence of the file and the file will always be overwritten.

3.2.2 Improvement for “Flash serial programming operation setting” default setting

From Smart Configurator for RL78 V1.5.0, “Flash serial programming operation setting” default selection is changed from “Unused” to “Used”. And a note about “Used” selection is added.



System configuration

On-chip debug setting

On-chip debug operation setting
○ Unused ● Use emulator ○ COM Port

Emulator setting
○ E20 ○ E2 ● E2 Lite

Pseudo-RRM/DMM function setting
○ Unused ● Used

Start/Stop function setting
● Unused ○ Used

Monitoring point function setting
● Unused ○ Used

Trace function setting
○ Unused ● Used

Flash serial programming operation setting
○ Unused ● Used

Note: Be sure to select "Used" when using on-chip debug function or flash serial programming function.
Once "Unused" is selected, the flash serial programming function can't be restored any more.

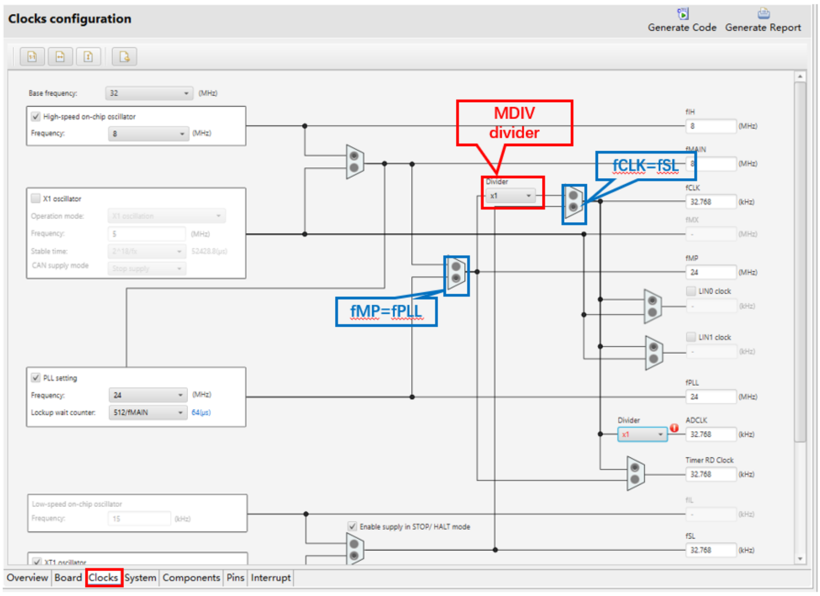
Hot plug-in function setting
● Unused ○ Used

RAM start address setting
RAM start address 0x9F

Figure 3-4 Flash serial programming operation setting

3.2.3 Improvement for MDIV divider setting

When fMP is selected fPLL and fCLK is selected fSL, after connecting the debugger, sometimes, the debugger will have "the debugger does not respond" issue. To solve this issue, from Smart Configurator for RL78 V1.5.0, the MDIV divider setting becomes enable in this case, so that user can adjust it to a correct value for debugger.



Clocks configuration

Base frequency: 32 (MHz)

☑ High-speed on-chip oscillator
Frequency: 8 (MHz)

□ X1 oscillator
Operation mode: X1 oscillation
Frequency: 5 (MHz)
Stable time: 2^18/fx 52428.8(μs)
CAN supply mode Stop supply

☑ PLL setting
Frequency: 24 (MHz)
Lockup wait counter: 512/fMAIN 64(μs)

Low-speed on-chip oscillator
Frequency: 15 (kHz)

☑ XT1 oscillator

☑ Enable supply in STOP/ HALT mode

MDIV divider

Divider x1

fCLK=fSL

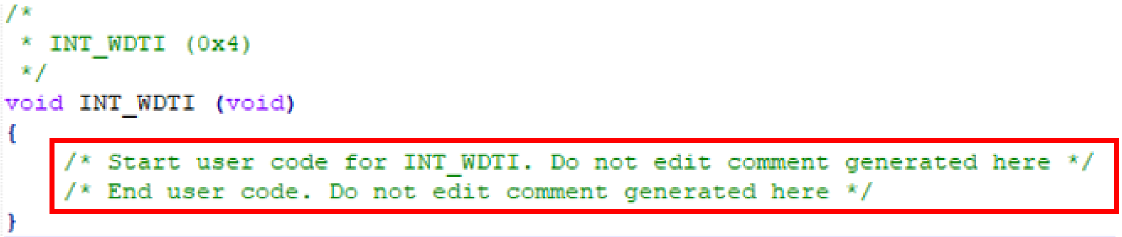
fMP=fPLL

fIH 8 (MHz)
fMAIN 8 (MHz)
fCLK 32.768 (kHz)
fMX - (MHz)
fMP 24 (MHz)
□ LIN0 clock - (kHz)
□ LIN1 clock - (kHz)
fPLL 24 (MHz)
Divider x1
ADCLK 32.768 (kHz)
Timer RD Clock 32.768 (kHz)
fIL - (kHz)
fSL 32.768 (kHz)

Figure 3-5 MDIV divider setting

3.2.4 Improvement for keeping the user defined interrupt code in "r_cg_inthandler.c" file

From Smart Configurator for RL78 V1.5.0, in LLVM project the user code in "r_cg_inthandler.c" file which written in user codes comment blocks can be kept.



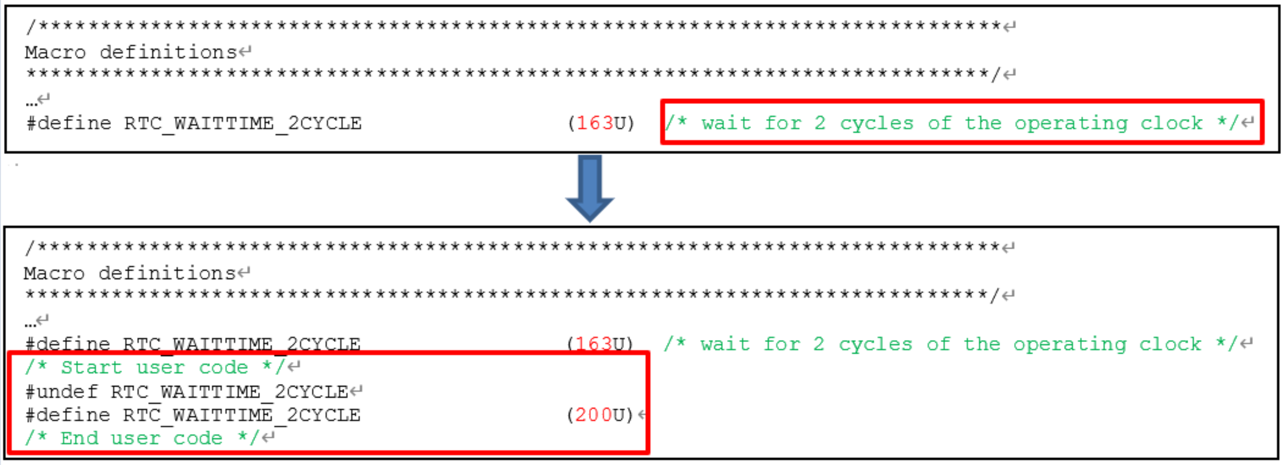
```
/*
 * INT_WDTI (0x4)
 */
void INT_WDTI (void)
{
    /* Start user code for INT_WDTI. Do not edit comment generated here */
    /* End user code. Do not edit comment generated here */
}
```

Figure 3-6 User codes comment blocks

3.2.5 Improvement for RTC wait time is not enough issue

When R_Config_RTC_Set_CounterValue() function returns MD_BUSY1, it means the wait time is not enough.

From Smart Configurator for RL78 V1.5.0, a comment is added in “Config_RTC.h" file, which indicates the necessary waiting time. If the actual wait time is not enough, user should manually modify the macro "RTC_WAITTIME_2CYCLE" value to a larger one.



```
/***********************************************************************************
Macro definitions
***********************************************************************************/
…
#define RTC_WAITTIME_2CYCLE                    (163U)  /* wait for 2 cycles of the operating clock */
```

```
/***********************************************************************************
Macro definitions
***********************************************************************************/
…
#define RTC_WAITTIME_2CYCLE                    (163U)  /* wait for 2 cycles of the operating clock */
/* Start user code */
#undef RTC_WAITTIME_2CYCLE
#define RTC_WAITTIME_2CYCLE                    (200U)
/* End user code */
```

Figure 3-7 Manually modified macro value

Note: Adding user codes with the specific tags is only available on e² studio Smart Configurator.

4. List of RENESAS TOOL NEWS AND TECHNICAL UPDATE

Below is a list of notifications delivered by RENESAS TOOL NEWS and TECHNICAL UPDATE.

Issue date	Document No.	Description	Applicable MCUs	Fixed version
Oct. 01, 2021	R20TS0757	1. Notes on creating LLVM for Renesas RL78 C/C++ Executable Project 2. Notes on using Port Input buffer function https://www.renesas.com/document/tnn/notes-e-studio-smart-configurator-plug-smart-configurator-rl78	RL78/G23	V1.2.0
Mar. 16, 2022	R20TS0822	1. Notes when build or clean e^2 studio Smart Configurator project https://www.renesas.com/document/tnn/notes-e-studio-smart-configurator-plug-smart-configurator-rl78-0	RL78/G23	V1.3.0
Dec. 01, 2022	R20TS0895	1. Notes when changing version of Board Support Program (BSP) or RL78 Software Integration System (SIS) modules https://www.renesas.com/us/en/document/tnn/notes-e-studio-smart-configurator-rl78-plug-smart-configurator-rl78	RL78/G23 RL78/F24 RL78/G15	V1.5.0

5. Points for Limitation

This section describes points for limitation regarding the Smart Configurator for RL78 Smart Configurator for RL78 V1.5.0.

5.1 List of Limitation

Table 5-1 List of Limitation ✓: Applicable, -: Not Applicable

No	Description	RL78/G23	RL78/F24	RL78/G15	RL78/F23	RL78/G22	Remarks
1	Note on extra help document issue	✓	✓	✓	✓	✓	
2	Note on ELCL D flip flop component GUI warning display incorrectly	✓	−	−	−	−	
3	Note on the unsupported setting items for some ELCL components	✓	−	−	−	−	
4	Note on the “Show view” dialog in Standalone Smart Configurator	✓	✓	✓	✓	✓	
5	Note on e2 studio becomes unresponsive when Developer Assistance is loading	✓	−	−	−	−	

5.2 Details of Limitation

5.2.1 Note on extra help document issue

For Smart Configurator, there is an extra help "Smart Browser" under "[Help] > [Help Contents]". Please ignore it.

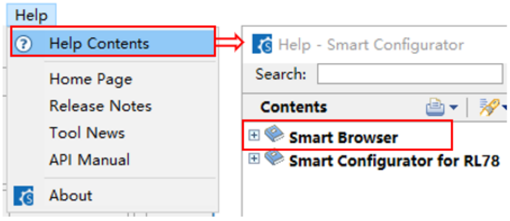


Figure 5-1 Extra help issue

5.2.2 Note on ELCL D flip flop component GUI warning display incorrectly

When selecting the event signal in ELCL D flip flop component, even if the selected signal consist with the hardware specification, there still displays the warning on the GUI.

[Avoidance measure]
Make reference to the hardware manual and set the selectable event signal though warning appeared in GUI, the waring is no impact for the code generation.

The following is example of using flip-flop 0 and flip-flop 1 in ELCL logic cell block L1.

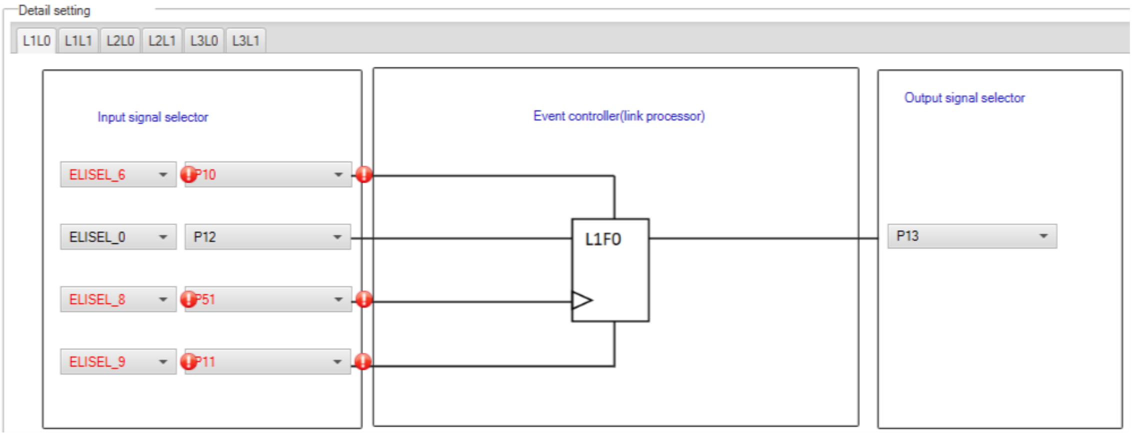


Figure 5-2 The flip-flop 0 in ELCL logic cell block L1 usage example

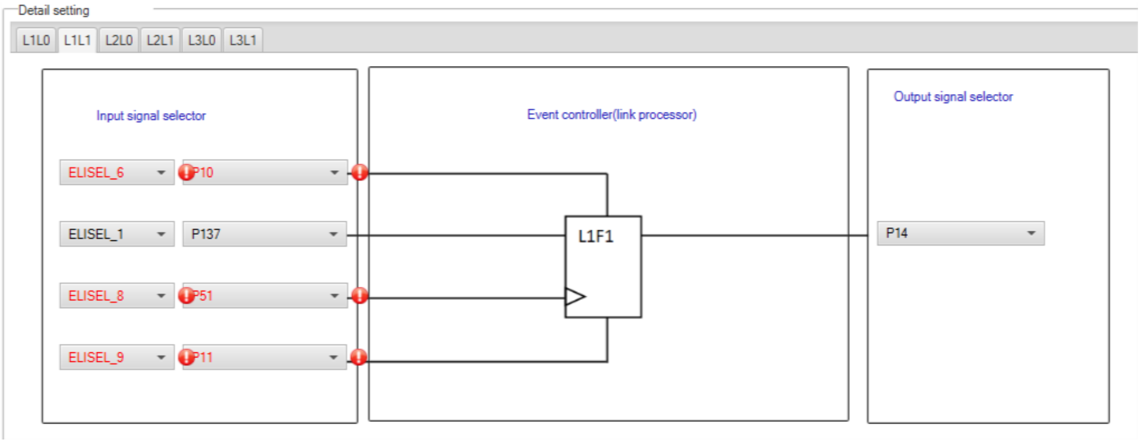


Figure 5-3 The flip-flop 1 in ELCL logic cell block L1 usage example

5.2.3 Note on the unsupported setting items for some ELCL components

In the following ELCL modules, it is not possible to set "no selection (fixed to 0)" as the input signal of the logic cell block and "negative logic output (inverted)" as the output level of the event signal.

- ELCL AND
- ELCL D flip flop
- ELCL EXOR
- ELCL selector
- ELCL Through

[Avoidance measure] None

5.2.4 Note on the “Show view” dialog in Standalone Smart Configurator

When configuring the "Show View" dialog in Standalone Smart Configurator, although the following items will be shown inside the menu, but do not configure or use it. This issue will be fixed from the next release.

・Bookmarks
・Markers
・Navigator (Deprecated)
・Problems
・Progress
・Tasks
・Welcome
・Cheat Sheets

5.2.5 Note on e^2 studio becomes unresponsive when Developer Assistance is loading

For there have multiple projects and at least 1 scfg editor opened case, when opening another *.scfg file containing any of the following ELCL module during the Developer Assistance node on loading status, the e^2 studio becomes unresponsive.

- ELCL AND
- ELCL D flip flop
- ELCL EXOR
- ELCL OR
- ELCL selector
- ELCL Through

[Avoidance measure] Please open *.scfg file after Developer Assistance node finish loading.

6. Points for Caution

This section describes points for caution regarding the Smart Configurator for RL78 V1.5.0.

6.1 List of Caution

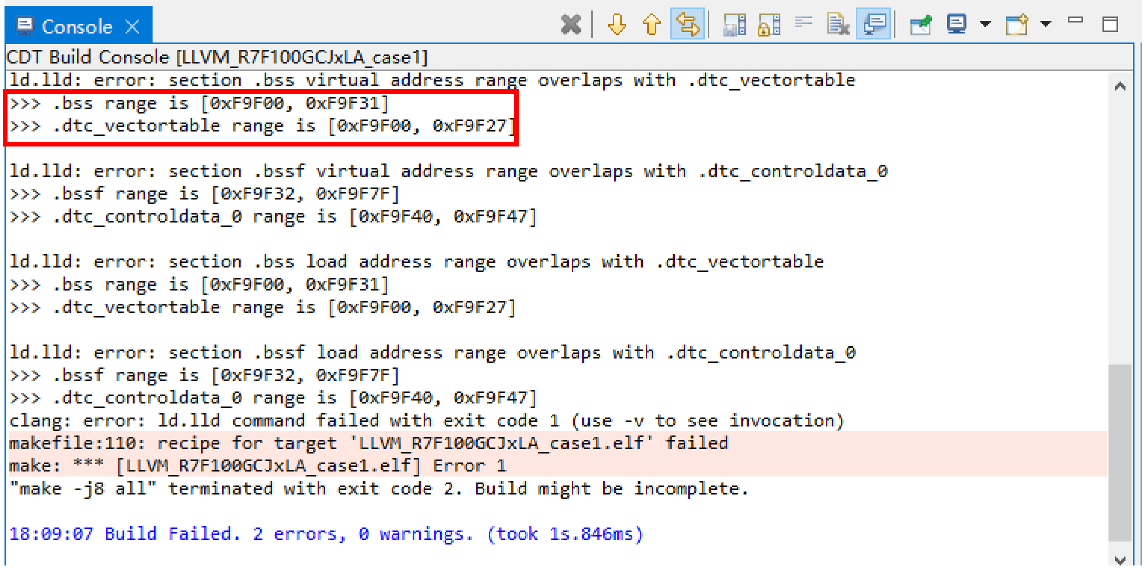
Table 6-1 List of Caution **✓: Applicable, -: Not Applicable**

No	Description	RL78/G23	RL78/F24	RL78/G15	RL78/F23	RL78/G22	Remarks
1	Note on the build error message such as “section .bss virtual address range overlaps with .dtc_vectortable”	✓	✓	−	✓	✓	
2	Note on the installation of the Smart Configurator	✓	✓	✓	✓	✓	
3	Note on using TRDIOA0 for Input capture and TRDIOB0 for Output compare at same time	−	✓	−	−	−	
4	Note on pulse width calculation of Timer RD input capture function	−	✓	−	−	−	
5	Note on using Touch middleware and UART communication components	✓	−	−	−	−	
6	Note on the include path update issue when renaming the component’s configuration name	✓	✓	✓	✓	✓	
7	Note on TAU Input Signal High/Low level Measurement components.	✓	✓	✓	✓	✓	
8	Note on ADCKL setting when ADCEN is set to 0.	−	✓	−	✓	−	
9	Note on C++ project of CC-RL V1.12	✓	✓	✓	✓	✓	
10	Note on browsing “Release Notes” and “Tool News” URL from the help menu	✓	✓	✓	−	−	
11	Note on using user code protection feature	✓	✓	✓	✓	✓	

6.2 Details of Caution

6.2.1 Note on the build error message such as “section .bss virtual address range overlaps with .dtc_vectortable”

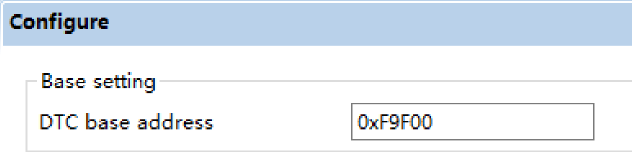
When user use many components and DTC component together, the generated code build might fail due to some section address overlaps.



```
CDT Build Console [LLVM_R7F100GCJxLA_case1]
ld.lld: error: section .bss virtual address range overlaps with .dtc_vectortable
>>> .bss range is [0xF9F00, 0xF9F31]
>>> .dtc_vectortable range is [0xF9F00, 0xF9F27]

ld.lld: error: section .bssf virtual address range overlaps with .dtc_controldata_0
>>> .bssf range is [0xF9F32, 0xF9F7F]
>>> .dtc_controldata_0 range is [0xF9F40, 0xF9F47]

ld.lld: error: section .bss load address range overlaps with .dtc_vectortable
>>> .bss range is [0xF9F00, 0xF9F31]
>>> .dtc_vectortable range is [0xF9F00, 0xF9F27]

ld.lld: error: section .bssf load address range overlaps with .dtc_controldata_0
>>> .bssf range is [0xF9F32, 0xF9F7F]
>>> .dtc_controldata_0 range is [0xF9F40, 0xF9F47]
clang: error: ld.lld command failed with exit code 1 (use -v to see invocation)
makefile:110: recipe for target 'LLVM_R7F100GCJxLA_case1.elf' failed
make: *** [LLVM_R7F100GCJxLA_case1.elf] Error 1
"make -j8 all" terminated with exit code 2. Build might be incomplete.

18:09:07 Build Failed. 2 errors, 0 warnings. (took 1s.846ms)
```

Figure 6-1 Build error message

[Workaround]
The Smart Configurator cannot set “.bss” and “.bssf” section address. So user should consider to modify “.bss” and “.bssf” section address manually in “linker_script.ld” file or change the DTC base address to avoid such section overlap error.



Configure

Base setting

DTC base address: 0xF9F00

Figure 6-2 DTC base address setting

6.2.2 Note on the installation of the Smart Configurator

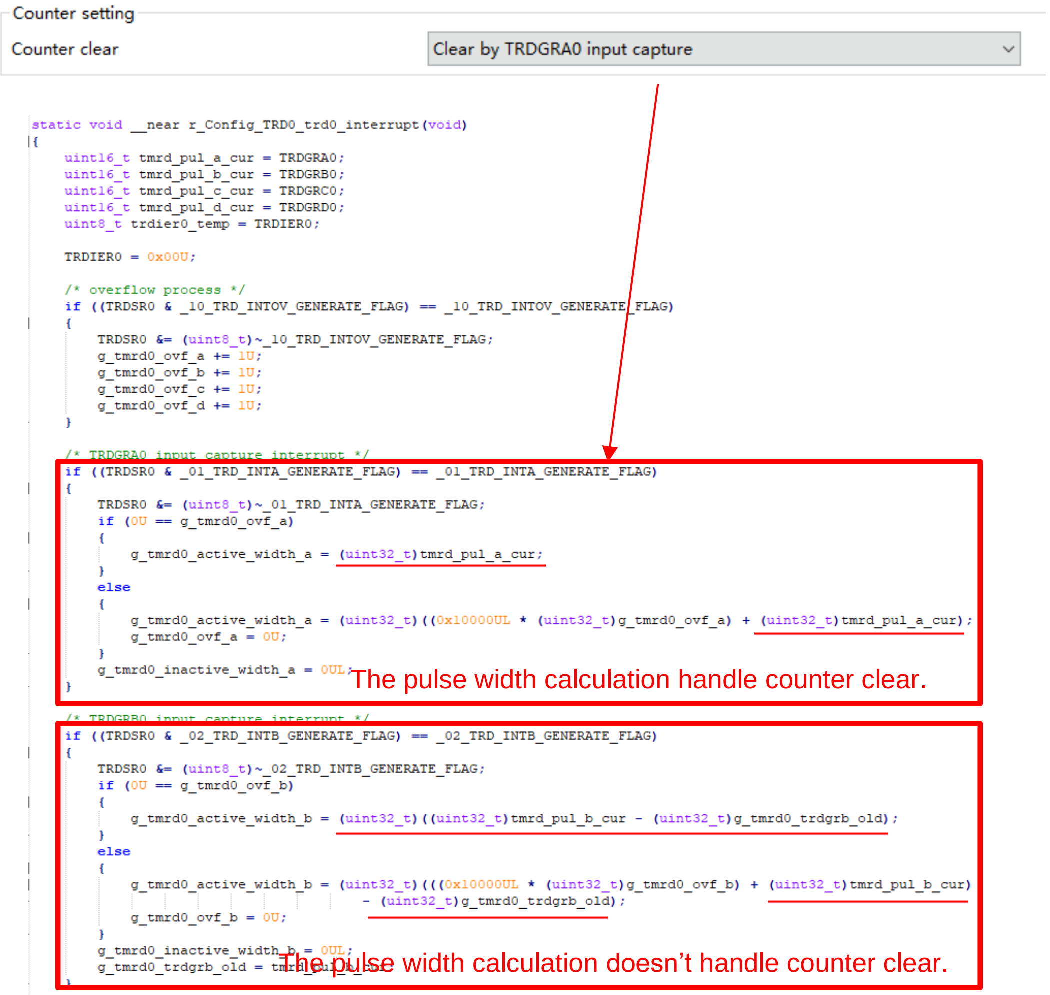
Do not set more than 64 characters for the installation directory.
You might see an error message "The specified path is too long" and will not be able to install Smart Configurator.

6.2.3 Note on using TRDIOA0 for Input capture and TRDIOB0 for Output compare at same time

If user sets up TRDIOA0 for Input capture and TRDIOB0 for Output compare at the same time. Smart Configurator will output a Peripheral conflict error.
User can ignore this Smart Configurator error message and use these two functions at the same time.

6.2.4 Note on pulse width calculation of Timer RD input capture function

The pulse width calculation code is with the assumption that the counter is not cleared between two interrupts occurrence, except the input pulse width which is selected as counter clear trigger on GUI.
For example, when “Clear by TRDGRAn input capture” is selected, only TRDIOAn pulse width calculation handle counter clear, other input pulse width calculation doesn’t handle counter clear.



Counter setting
Counter clear: Clear by TRDGRA0 input capture

```
static void __near r_Config_TRD0_trd0_interrupt(void)
{
    uint16_t tmrd_pul_a_cur = TRDGRA0;
    uint16_t tmrd_pul_b_cur = TRDGRB0;
    uint16_t tmrd_pul_c_cur = TRDGRC0;
    uint16_t tmrd_pul_d_cur = TRDGRD0;
    uint8_t trdier0_temp = TRDIER0;

    TRDIER0 = 0x00U;

    /* overflow process */
    if ((TRDSR0 & _10_TRD_INTOV_GENERATE_FLAG) == _10_TRD_INTOV_GENERATE_FLAG)
    {
        TRDSR0 &= (uint8_t)~_10_TRD_INTOV_GENERATE_FLAG;
        g_tmrd0_ovf_a += 1U;
        g_tmrd0_ovf_b += 1U;
        g_tmrd0_ovf_c += 1U;
        g_tmrd0_ovf_d += 1U;
    }

    /* TRDGRA0 input capture interrupt */
    if ((TRDSR0 & _01_TRD_INTA_GENERATE_FLAG) == _01_TRD_INTA_GENERATE_FLAG)
    {
        TRDSR0 &= (uint8_t)~_01_TRD_INTA_GENERATE_FLAG;
        if (0U == g_tmrd0_ovf_a)
        {
            g_tmrd0_active_width_a = (uint32_t)tmrd_pul_a_cur;
        }
        else
        {
            g_tmrd0_active_width_a = (uint32_t)((0x10000UL * (uint32_t)g_tmrd0_ovf_a) + (uint32_t)tmrd_pul_a_cur);
            g_tmrd0_ovf_a = 0U;
        }
        g_tmrd0_inactive_width_a = 0UL;
    }

    /* TRDGRB0 input capture interrupt */
    if ((TRDSR0 & _02_TRD_INTB_GENERATE_FLAG) == _02_TRD_INTB_GENERATE_FLAG)
    {
        TRDSR0 &= (uint8_t)~_02_TRD_INTB_GENERATE_FLAG;
        if (0U == g_tmrd0_ovf_b)
        {
            g_tmrd0_active_width_b = (uint32_t)((uint32_t)tmrd_pul_b_cur - (uint32_t)g_tmrd0_trdgrb_old);
        }
        else
        {
            g_tmrd0_active_width_b = (uint32_t)(((0x10000UL * (uint32_t)g_tmrd0_ovf_b) + (uint32_t)tmrd_pul_b_cur)
                                      - (uint32_t)g_tmrd0_trdgrb_old);
            g_tmrd0_ovf_b = 0U;
        }
        g_tmrd0_inactive_width_b = 0UL;
        g_tmrd0_trdgrb_old = tmrd_pul_b_cur;
```

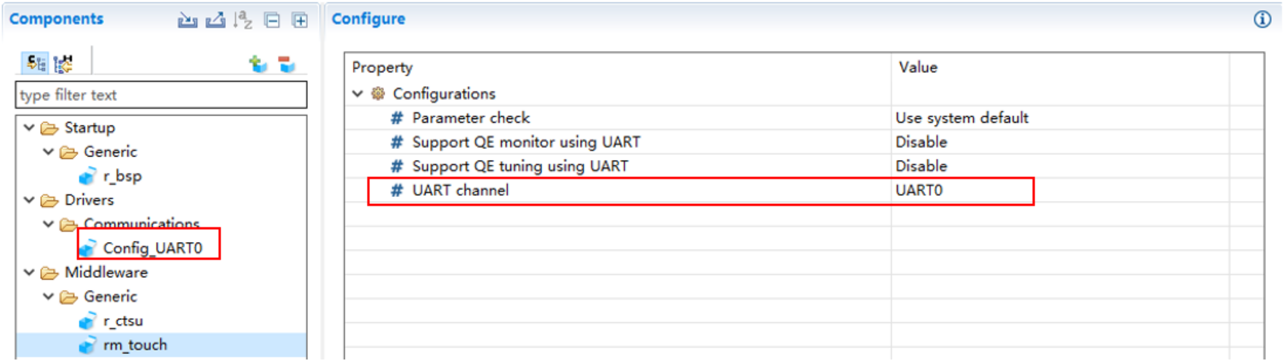
The pulse width calculation handle counter clear.

The pulse width calculation doesn’t handle counter clear.

Figure 6-3 Counter clear setting in Input capture function

6.2.5 Note on using Touch middleware and UART communication components

When use Touch middleware, please do not change the name of UART components. Otherwise, due the file name mismatch will bring build error.
For example, in touch middleware select UART0 as UART channel, for UART0 component please use Config_UART0.



Components

type filter text

- Startup
  - Generic
    - r_bsp
- Drivers
  - Communications
    - Config_UART0
- Middleware
  - Generic
    - r_ctsu
    - rm_touch

Configure

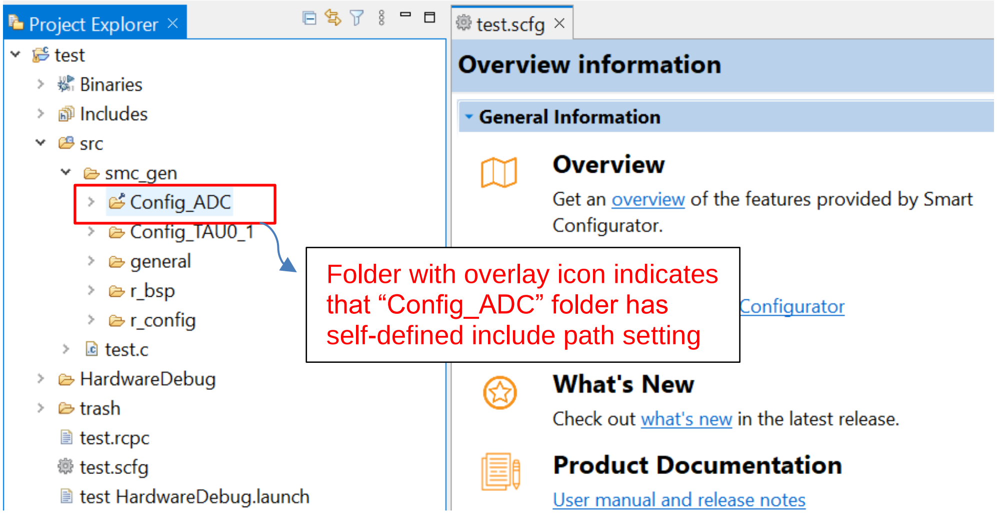
Property	Value
Configurations	
Parameter check	Use system default
Support QE monitor using UART	Disable
Support QE tuning using UART	Disable
UART channel	UART0

Figure 6-4 Touch middleware and UART communication components

6.2.6 Note on the include path update issue when renaming the component’s configuration name

When renaming the added component’s configuration in e^2 studio Smart Configurator project that has self-defined include path setting for any folder or file, include path setting for that folder or file will keep the old name setting after code generation. This will cause build error when compiling the newly generated codes so please manually update the include path.

The folder or file which has self-defined include path setting can be recognized by checking the overlay icon (📂) on that folder or file. Below is an example on how to handle the include path update after renaming Compare Match Timer component configuration.



Project Explorer

- test
  - Binaries
  - Includes
  - src
    - smc_gen
      - Config_ADC
      - Config_TAU0_1
      - general
      - r_bsp
      - r_config
    - test.c
  - HardwareDebug
  - trash
  - test.rcpc
  - test.scfg
  - test HardwareDebug.launch

test.scfg

Overview information

General Information

Overview
Get an overview of the features provided by Smart Configurator.

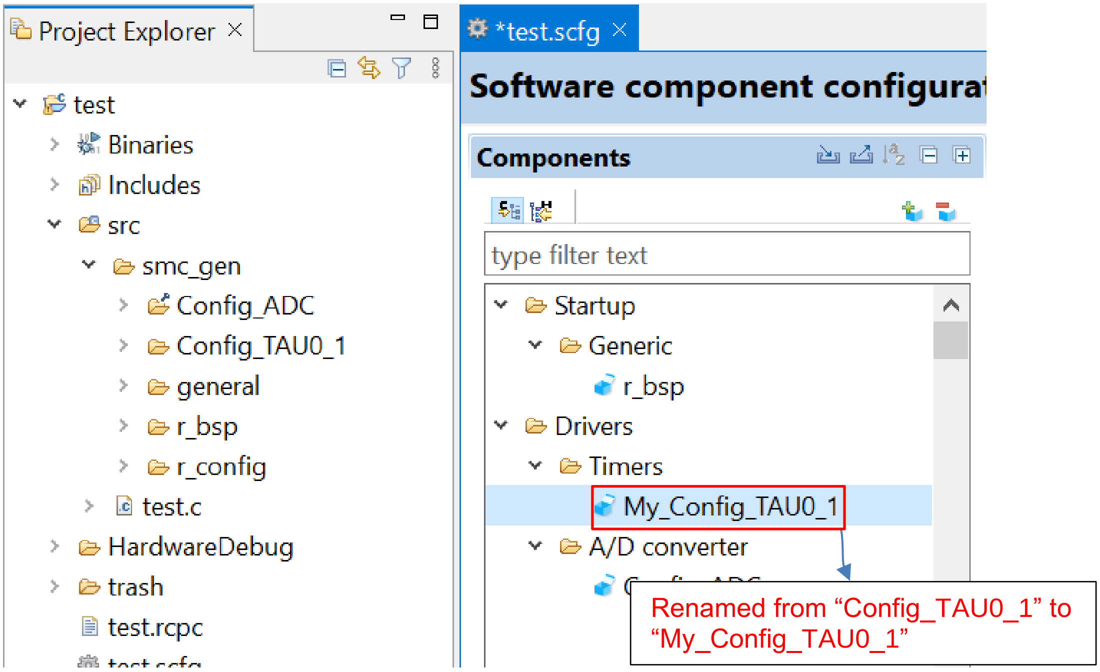
Configurator

What's New
Check out what's new in the latest release.

Product Documentation
User manual and release notes

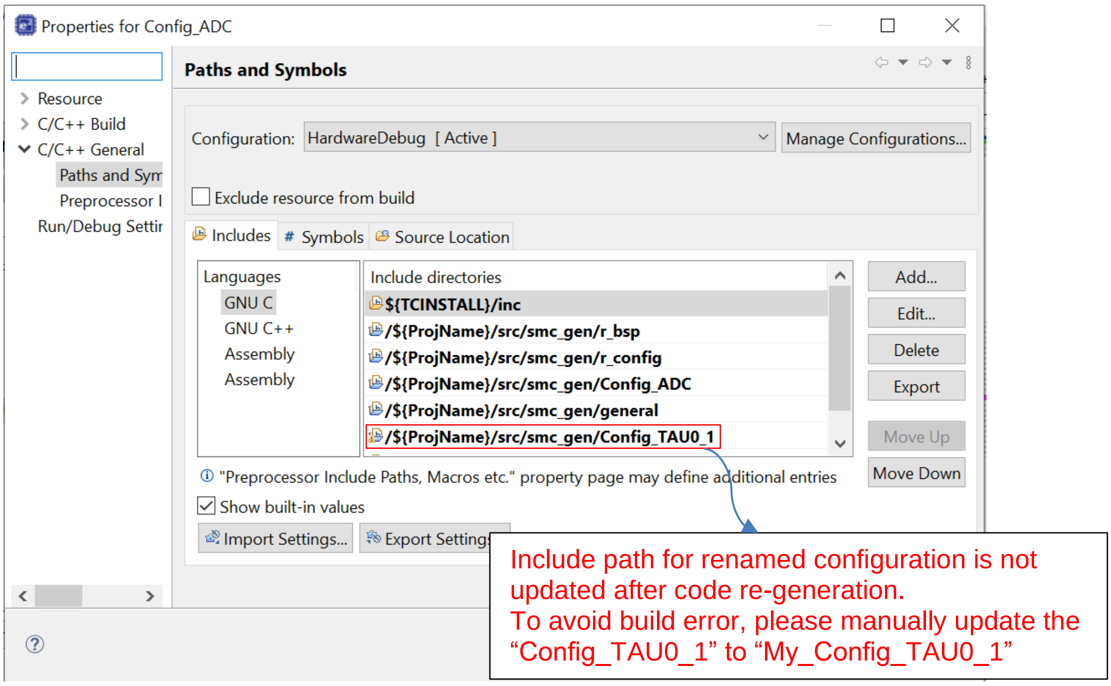
Folder with overlay icon indicates that “Config_ADC” folder has self-defined include path setting

Figure 6-5 Interval Timer component configuration before renaming



Renamed from “Config_TAU0_1” to “My_Config_TAU0_1”

Figure 6-6 The Interval Timer component configuration after renaming



Configuration: HardwareDebug [Active]

□ Exclude resource from build

Include directories
${TCINSTALL}/inc
/${ProjName}/src/smc_gen/r_bsp
/${ProjName}/src/smc_gen/r_config
/${ProjName}/src/smc_gen/Config_ADC
/${ProjName}/src/smc_gen/general
/${ProjName}/src/smc_gen/Config_TAU0_1

"Preprocessor Include Paths, Macros etc." property page may define additional entries

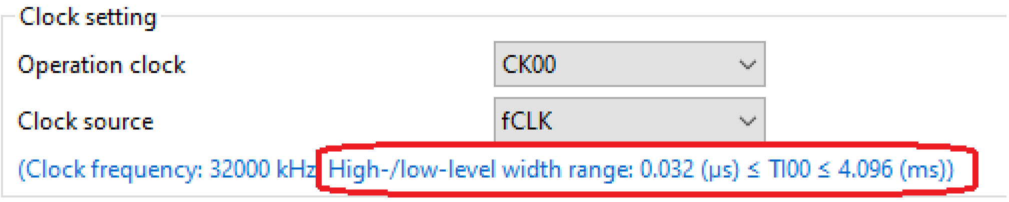
☑ Show built-in values

Include path for renamed configuration is not updated after code re-generation.
To avoid build error, please manually update the “Config_TAU0_1” to “My_Config_TAU0_1”

Figure 6-7 Include path setting for the “Config_ADC” configuration

6.2.7 Note on TAU Input Signal High/Low level Measurement component

When using TAU Input Signal High/Low level Measurement component, after used noise filter function for TImn input pulse, please make sure the High/Low level width min value needs to be greater than two times the minimum value prompted on the UI.
For example, the High/Low level width min value is 0.032us (min value), when use noise filter function, the width min value should be 0.064us.



Clock setting

Operation clock: CK00

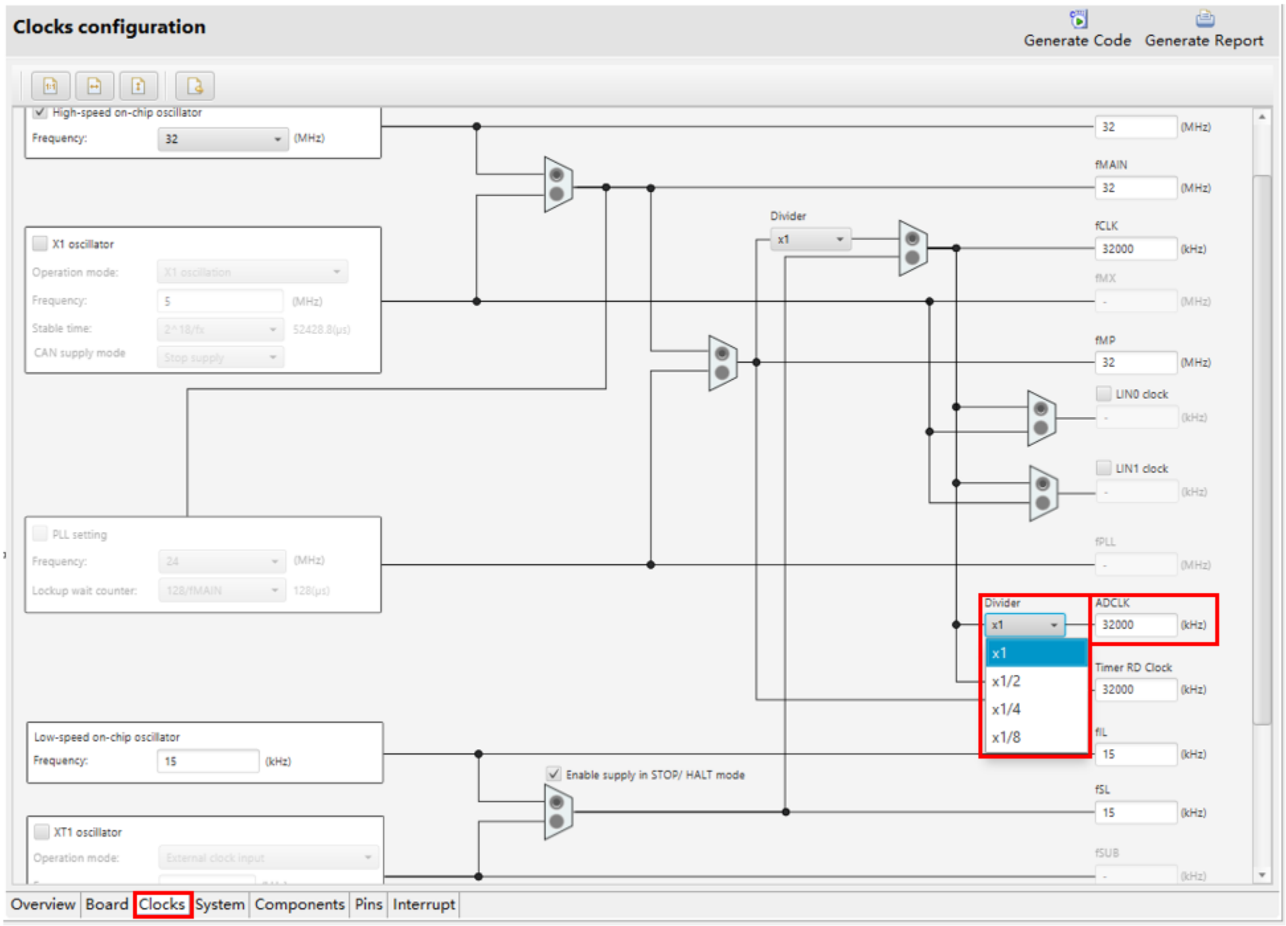
Clock source: fCLK

(Clock frequency: 32000 kHz High-/low-level width range: 0.032 (μs) ≤ TI00 ≤ 4.096 (ms))

Figure 6-8 High/Low level width min value

6.2.8 Note on ADCKL setting when ADCEN is set to 0

When ADCEN is set to 0 in user code, ADCKS register is reset, which causes the actual ADCLK value is not inconsistent with UI selection. Please take this note when adding “ADCEN = 0” or calling API “R_ADC_Set_PowerOff()” in user code .



Clocks configuration

☑ High-speed on-chip oscillator
Frequency: 32 (MHz)

□ X1 oscillator
Operation mode: X1 oscillation
Frequency: 5 (MHz)
Stable time: 2^18/fx 52428.8(μs)
CAN supply mode Stop supply

□ PLL setting
Frequency: 24 (MHz)
Lockup wait counter: 128/fMAIN 128(μs)

Low-speed on-chip oscillator
Frequency: 15 (kHz)

□ XT1 oscillator
Operation mode: External clock input

☑ Enable supply in STOP/ HALT mode

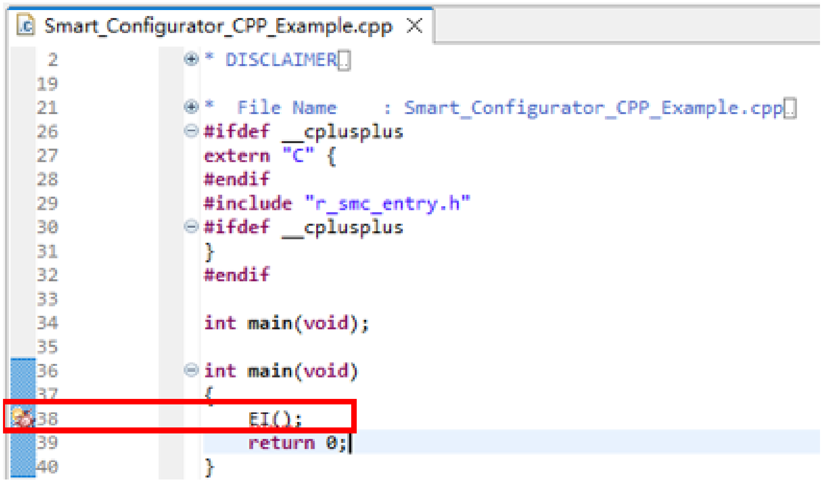
Divider: x1

32 (MHz)
fMAIN: 32 (MHz)
fCLK: 32000 (kHz)
fMX: - (MHz)
fMP: 32 (MHz)
□ LIN0 clock: - (kHz)
□ LIN1 clock: - (kHz)
fPLL: - (MHz)
Divider: x1
ADCLK: 32000 (kHz)
Timer RD Clock: 32000 (kHz)
fIL: 15 (kHz)
fSL: 15 (kHz)
fSUB: - (kHz)

Figure 6-9 ADCLK setting in Clock page

6.2.9 Note on CC-RL V1.12 C++ language project

CC-RL V1.12 C++ language project is a preview function, there are some dummy issues such as “EI()” in editor. However this does not affect the program operation. Please ignore it.



```
* DISCLAIMER
*  File Name    : Smart_Configurator_CPP_Example.cpp
#ifdef __cplusplus
extern "C" {
#endif
#include "r_smc_entry.h"
#ifdef __cplusplus
}
#endif

int main(void);

int main(void)
{
    EI();
    return 0;
}
```

Figure 6-10 CODAN issue in CC-RL V1.12 C++ project

6.2.10 Note on browsing “Release Notes” and “Tool News” URL from the help menu

For Smart Configurator for RL78 V1.4.0 or before version, “Release Notes” and “Tools News” in the help menu cannot access the correct URL. This issue has been fixed from this version.

Please access the URL below directly for Smart Configurator for RL78 V1.4.0 or before version.

Release Notes: https://www.renesas.com/rl78-smart-configurator-release-note

Tool News: https://www.renesas.com/rl78-smart-configurator-tn-notes

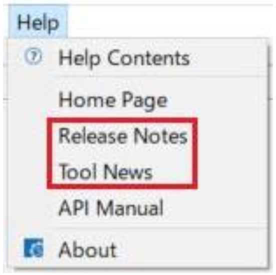


Figure 6-11 Release Notes and Tool News in Smart Configurators

6.2.11 Note on using user code protection feature

From Smart Configurator for RL78 V54.0 onwards, user code protection feature will be supported for all Code Generation components. Please use the following specific tags to add user code when using the user code protection feature. If the specific tags do not match exactly, inserted user code will not be protected after the code generation.

```
/* Start user code */

User code can be added between the specific tags

/* End user code */
```

The user code protection feature will only be supported on the files that are generated by the Code Generation component. Hence, the user code protection feature is not available for non-Code Generation components.

Revision History

Rev.	Section	Description
1.00	-	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

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/.

Trademarks

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