Smart Configurator for RISC-V MCU Plug-in in e² studio 2024-07
Smart Configurator for RISC-V MCU V1.2.0
Release Note

Introduction
Thank you for using the Smart Configurator for RISC-V MCU.
This document describes the restrictions and points for caution. Read this document before using the product.

Contents

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 RISC-V MCU V1.2.0 is equivalent to Smart Configurator for RISC-V MCU Plug-in in e2 studio 2024-07 Smart Configurator for RISC-V MCU V1.2.0.

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

Smart Configurator for RISC-V MCU plug-in in e2 studio 2024-07 or later 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

- Windows

  LLVM for Renesas RISC-V 17.0.2.202401 or later
  IAR Embedded Workbench for RISC-V 3.30.1 or later
  SEGGER Embedded Studio for RISC-V 8.10 or later

- Linux

  LLVM for Renesas RISC-V 17.0.2.202406 or later
  IAR bxriscv-3.30.1 or later
  SEGGER Embedded Studio 8for RISC-V 8.12a or later

2. Support List

2.1 Support Devices List

Below is a list of devices supported by the Smart Configurator for RISC-V MCU Plug-in in e2 studio 2024-07 Smart Configurator for RISC-V MCU V1.2.0.

Table 2-1 Support Devices

Group (HW Manual number)	PIN	Device name
RISC-V/G021 Group (R01UH1036EJ0110)	16pin	R9A02G0214CBY
	24pin	R9A02G0214CNK
	32pin	R9A02G0214CNH
	48pin	R9A02G0214CNE

2.2 Support Components List

Below is a list of Components supported by the Smart Configurator for RISC-V MCU Plug-in in e2 studio 2024-07 Smart Configurator for RISC-V MCU V1.2.0.

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

No	Components	Mode	G021	Remarks
1	A/D Converter	-	✓	
2	Comparator	-	✓	
3	D/A Converter	-	✓	
4	Data Transfer Controller	-	✓	
5	Delay Counter	-	✓	
6	Divider Function	-	✓	
7	Event Link Controller	-	✓	
8	External Event Counter	-	✓	
9	IIC Communication (Master mode)	-	✓	
10	IIC Communication (Slave mode)	-	✓	
11	Input Pulse Interval/Period Measurement	-	✓	
12	Input Signal High-/Low-Level Width Measurement	-	✓	
13	Interrupt Controller	-	✓	
14	Interval Timer	8 bit count mode	✓	
		16 bit count mode	✓	
		16 bit capture mode	✓	
		32 bit count mode	✓	
15	Key Interrupt	-	✓	
16	One-Shot Pulse Output	-	✓	
17	Ports	-	✓	
18	PWM Output	-	✓	
19	Real-Time Clock	-	✓	
20	Remote Control Signal Receiver	-	✓	
21	Simple SPI Communication	Transmission	✓	
		Reception	✓	
		Transmission/reception	✓	
22	Square Wave Output	-	✓	
23	UART Communication	Transmission	✓	
		Reception	✓	
		Transmission/reception	✓	
24	Voltage Detector	-	✓	
25	Watchdog Timer	-	✓	

2.3 New support

2.3.1 BSP (Board Support Package) revision update

BSP rev1.20 is supported and will be added as default BSP when creating Smart Configurator project.

2.3.2 Linux host OS support

From this version, Linux host has been supported.

- For stand-alone version, user can start Smart Configurator for RISC-V as below steps.

1) Download stand-alone Smart Configurator for RISC-V package: SmartConfigurator_for_RISC-V_MCU_V1_2_0.tar.gz from Renesas web and extract to local folder.

2) Go to extracted folder .$DIR/eclipse and run SmartConfigurator

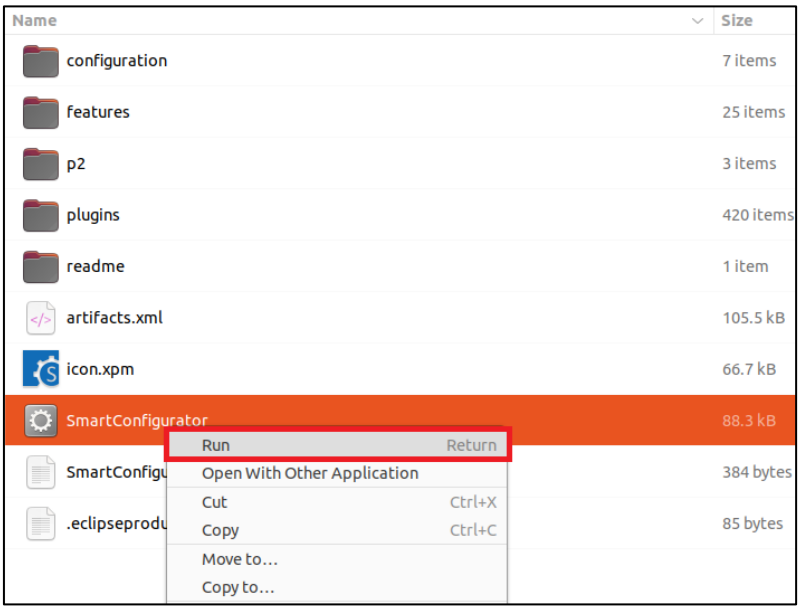

Figure 2-1 Run SmartConfigurator

3) Create Smart Configurator project and start to use

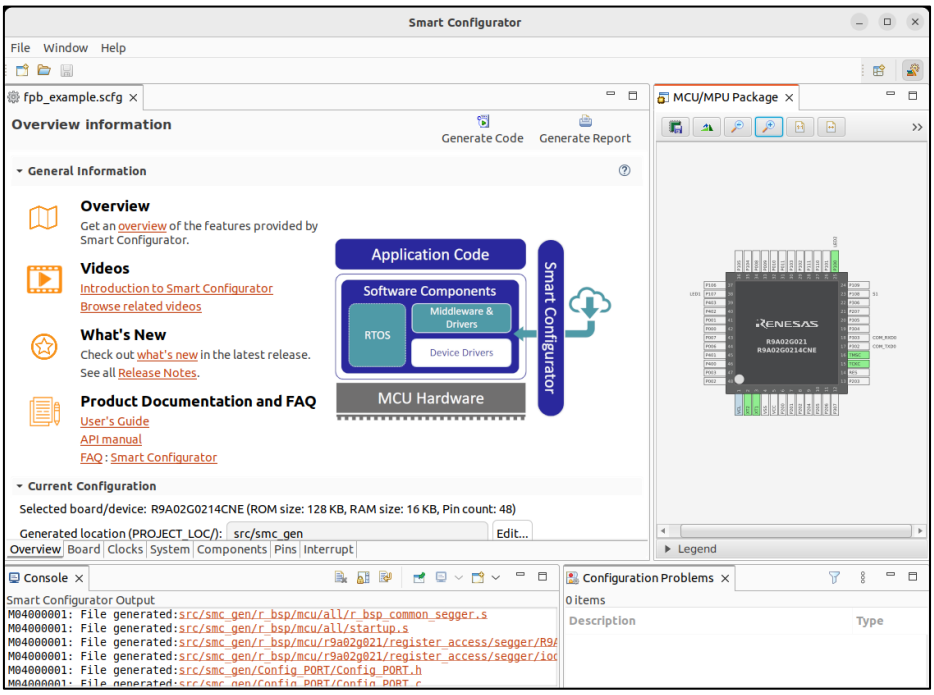

Figure 2-2 Stand-alone SmartConfigurator started

- For e² studio Smart Configurator plugin version, user can start Smart Configurator for RISC-V as below steps.

1) Run e² studio under installation directory: /home/sc/.local/share/renesas/e2_studio

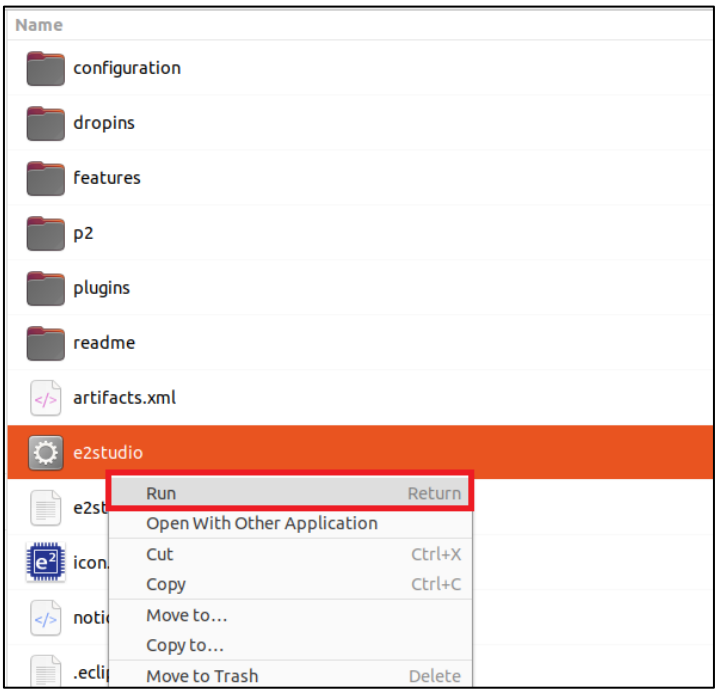

Figure 2-3 Run e² studio

2) Create new project with Renesas RISC-V MCU and select Smart Configurator in project wizard, after project created, Smart Configurator will be started.

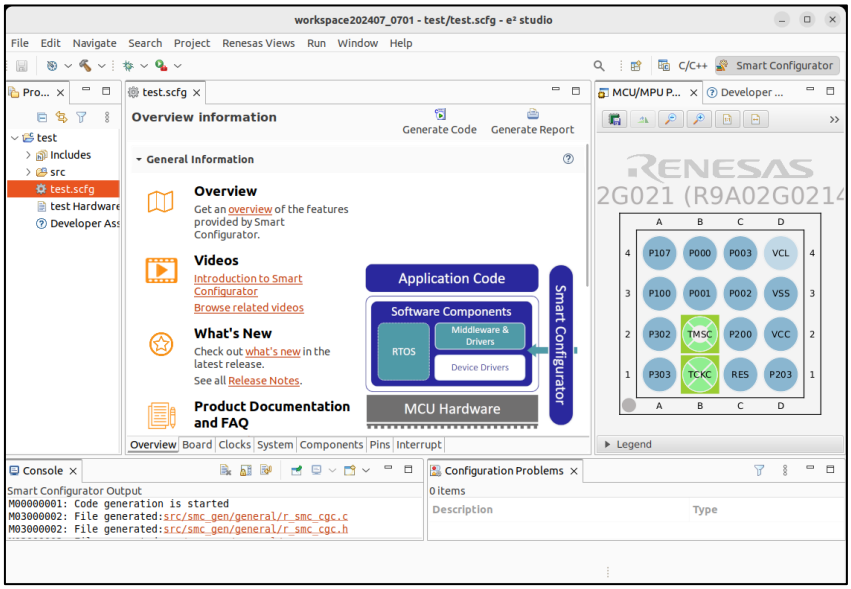

Figure 2-4 Smart Configurator started

2.3.3 Access FAQ from Overview tab

From this version, FAQ can be accessed from Overview tab.

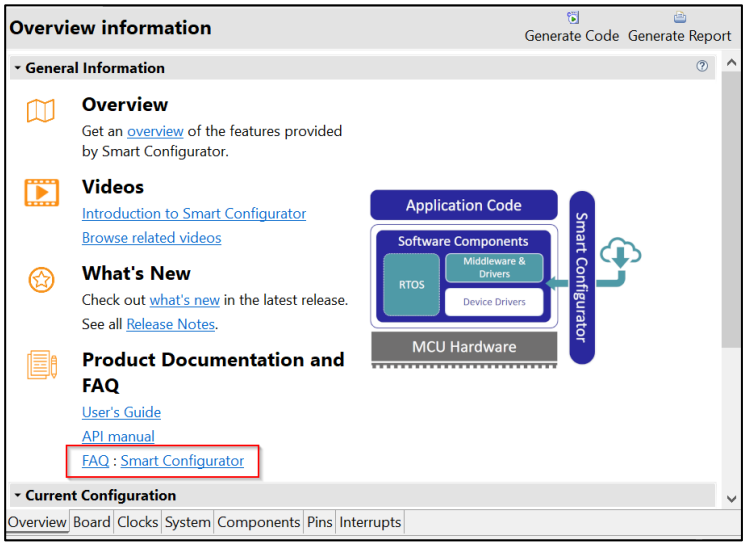

Figure 2-5 FAQ link at Overview tab

2.3.4 Simplify MCU/MPU Package view

From this version, when using QFP device, MCU/MPU Package view is simplified to remove outer pad by default for easy recognition of user changes.

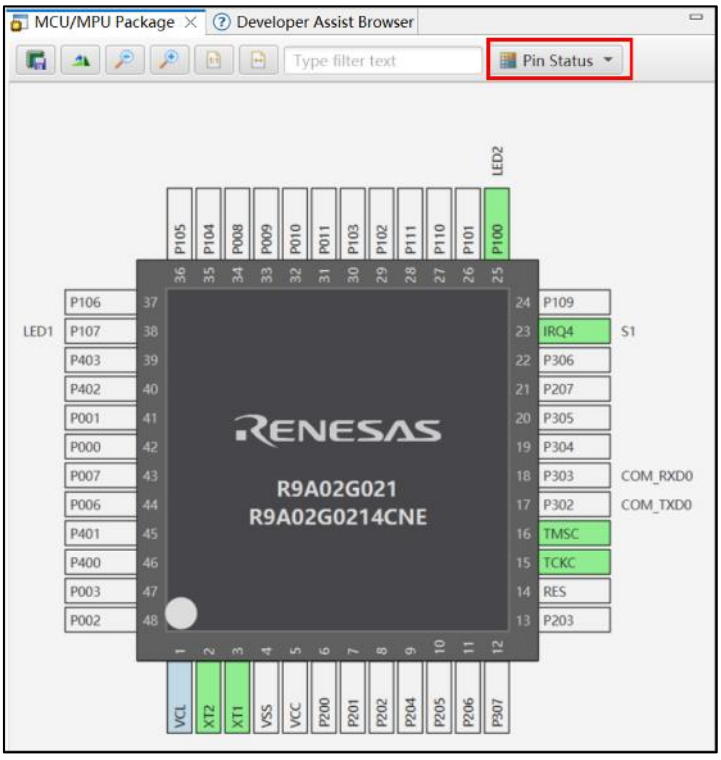

Figure 2-6 MCU/MPU Package view with outer pad setting

2.3.5 Support My Renesas login within Smart Configurator standalone

From this version, users can login My Renesas within Smart Configurator and download SIS modules.

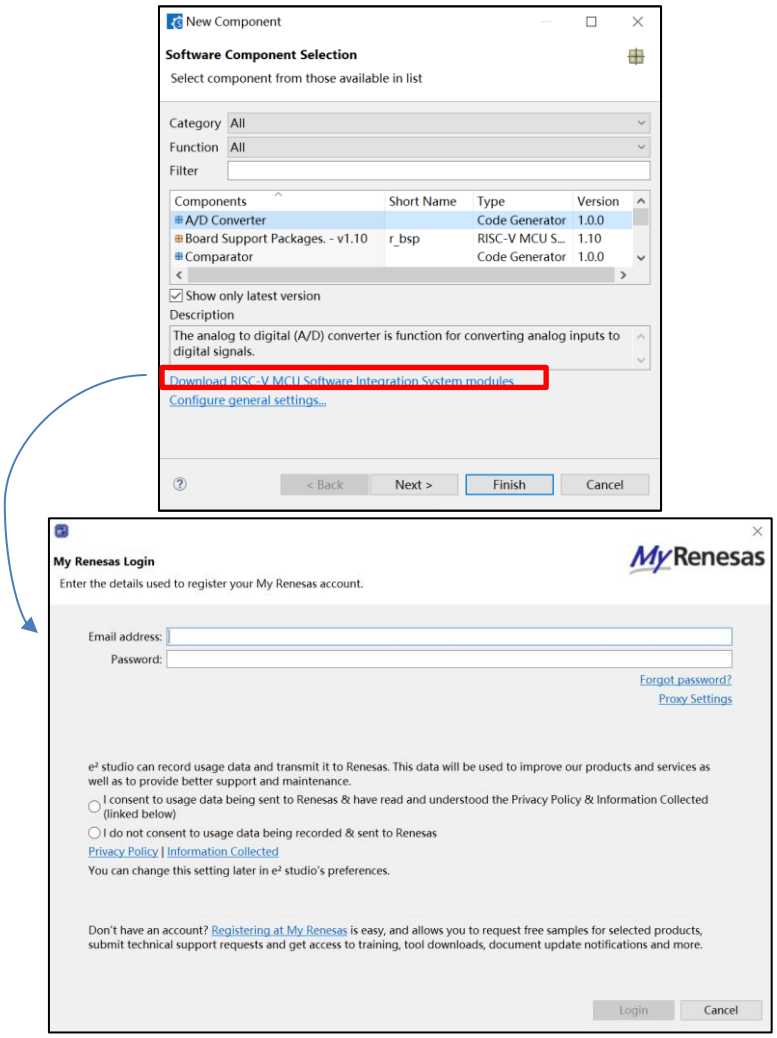

Figure 2-7 My Renesas log from Smart Configurator standalone

Note: Due to some SIS modules don't support IAR/SEGGER toolchain, so they will not be shown in components list when user selected IAR/SEGGER toolchain project.

3. Changes

This chapter describes changes to the Smart Configurator for RISC-V MCU V1.2.0.

3.1 Correction of issues/limitations

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

No	Description	G021	Remarks
1	Fixed the build error in mcu_mapped_interrupts.c	✓	
2	Fixed the issue of developer assistance R_Config_IICA1_Master_Send has wrong parameters	✓	
3	Fixed the issue of the core options in SEGGER Embedded Studio project	✓	
4	Fixed the issue of the description of KEY_INTKR in interrupt page	✓	

3.2 Details of Changes

3.2.1 Fixed the build error in mcu_mapped_interrupts.c

When enabling the [Interrupt setting(bsp_mapped_interrupt_open)] in r_bsp, there will be build errors in mcu_mapped_interrupts.c:
This issue has been fixed in this version.

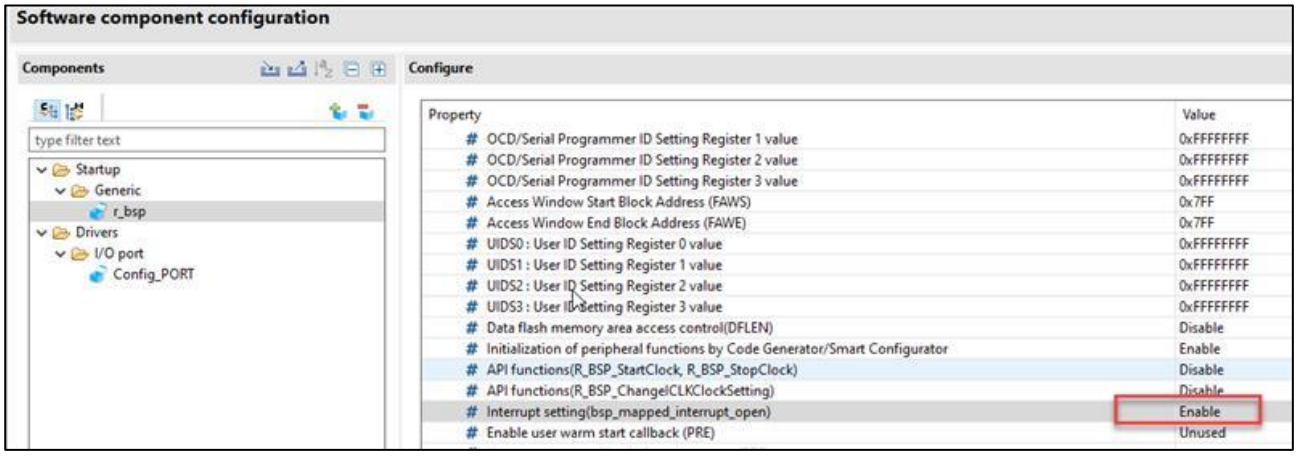

Figure 3-1 Enable the [Interrupt setting(bsp_mapped_interrupt_open)]

3.2.2 Fixed the issue of developer assistance R_Config_IICA1_Master_Send has wrong parameters

When using the developer assistance function, for IICA Master_Send and Master_Receive function, after drag, the parameters are wrong.
This issue has been fixed in this version.

3.2.3 Fixed the issue of the core options in SEGGER Embedded Studio project

When using SEGGER toolchain, the following options default vales have been changed:

1) RISC-V ISA Zba Extension: is “Yes”.
2) RISC-V ISA Zbb Extension: is “Yes”.

3) RISC-V ISA Zbs Extension: is “Yes”.

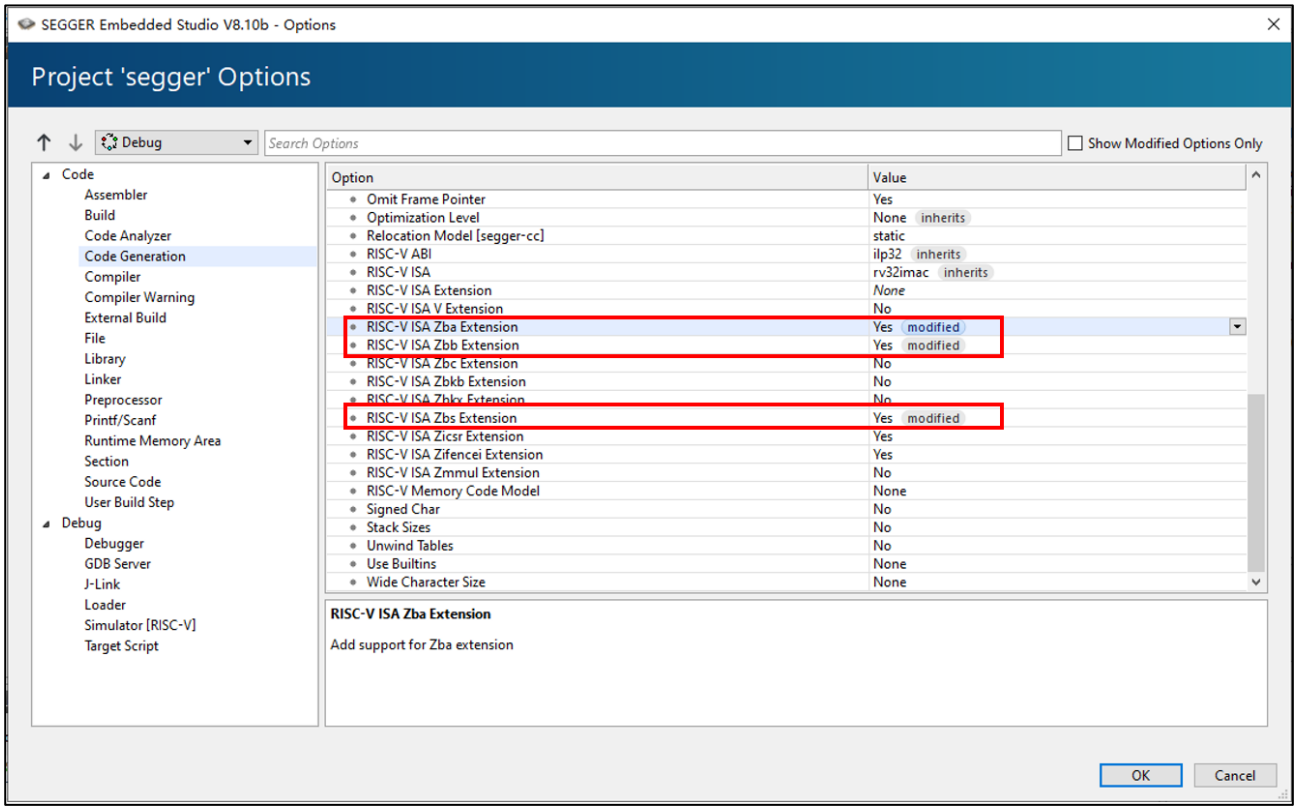

Figure 3-2 Change the Zba/Zbb/Zbs Extension options

3.2.4 Fixed the issue of the description of KEY_INTKR in interrupt page

Add the description of KEY_INTKR interrupt in interrupt page.

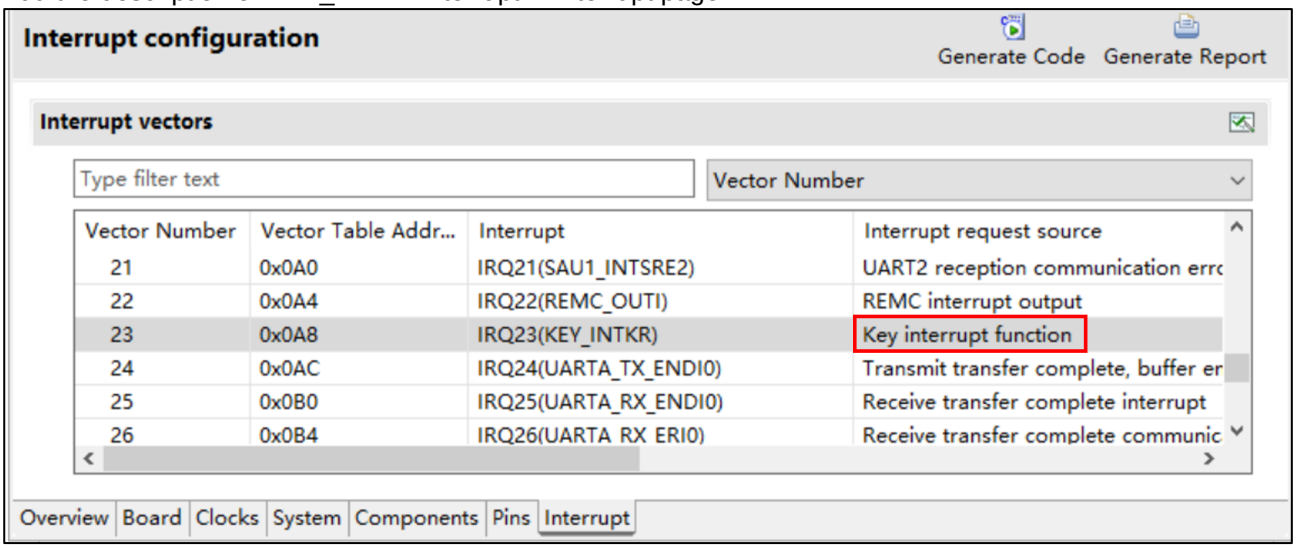

Figure 3-3 KEY_INTKR interrupt description

3.3 Specification changes

Table 3-2 List of Correction of issues/limitations ✓: Applicable, -: Not Applicable

No	Description	G021	Remarks
1	Improvement for changing blinky frequency for Blinky sample project	✓	
2	Improvement for port initialization sequence for I2C to remove glitch on the I2C SDA/SCL line	✓	
3	Improvement for adding common functions node for developer assistance function	✓	
4	Improvement for adding a new interrupt vector routine for NMI interrupt	✓	
5	Improvement for removing PMR setting for ICU pins	✓	

3.3.1 Improvement for changing blinky frequency for Blinky sample project

From this version, Blinky sample project is improved to detect if selected board has switch, users can use switch to change blinky frequency.

```
#include "r_smc_entry.h"

volatile uint32_t blinkDelay = 1000; // Initial blink delay of 1

int main(void);

int main(void)
{
    /* Start SW Interrupt */
    R_Config_ICU_IRQ4_Start();

    while(1)
    {
        /* Toggle LED status */
        PIN_WRITE(LED2) = ~PIN_READ(LED2);

        /* Delay blinkDelay milliseconds before returning */
```

Figure 3-4 Create new project in Smart Configurator

3.3.2 Improvement for port initialization sequence for I2C to remove glitch on the I2C SDA/SCL line

From this version, port initialization sequence has been changed for I2C to remove glitch on the I2C SDA/SCL line.
Remove PODR setting, move NCODR, PMR, PSEL settings after IICE = 1.

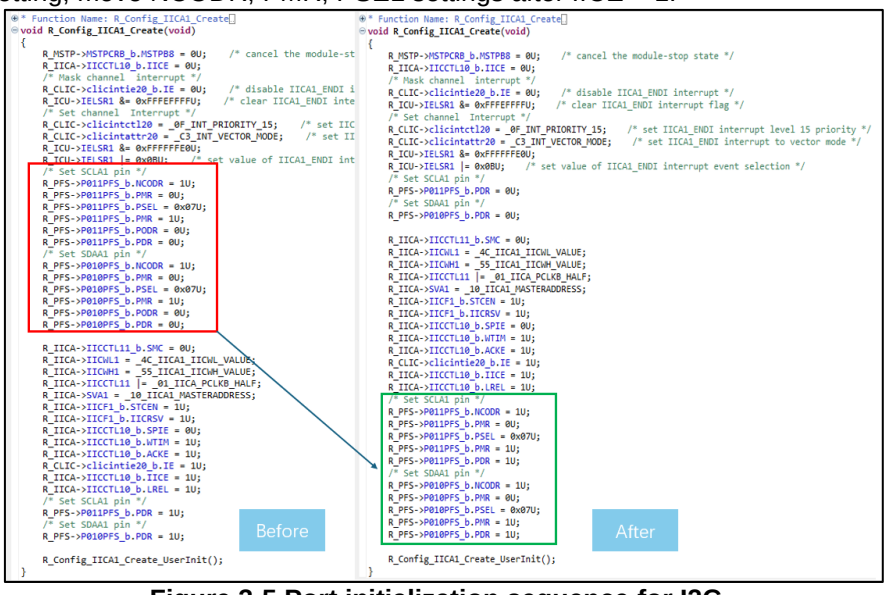

Figure 3-5 Port initialization sequence for I2C

3.3.3 Improvement for adding common functions node for developer assistance function

From this version, the functions in general folder have been added in components node:

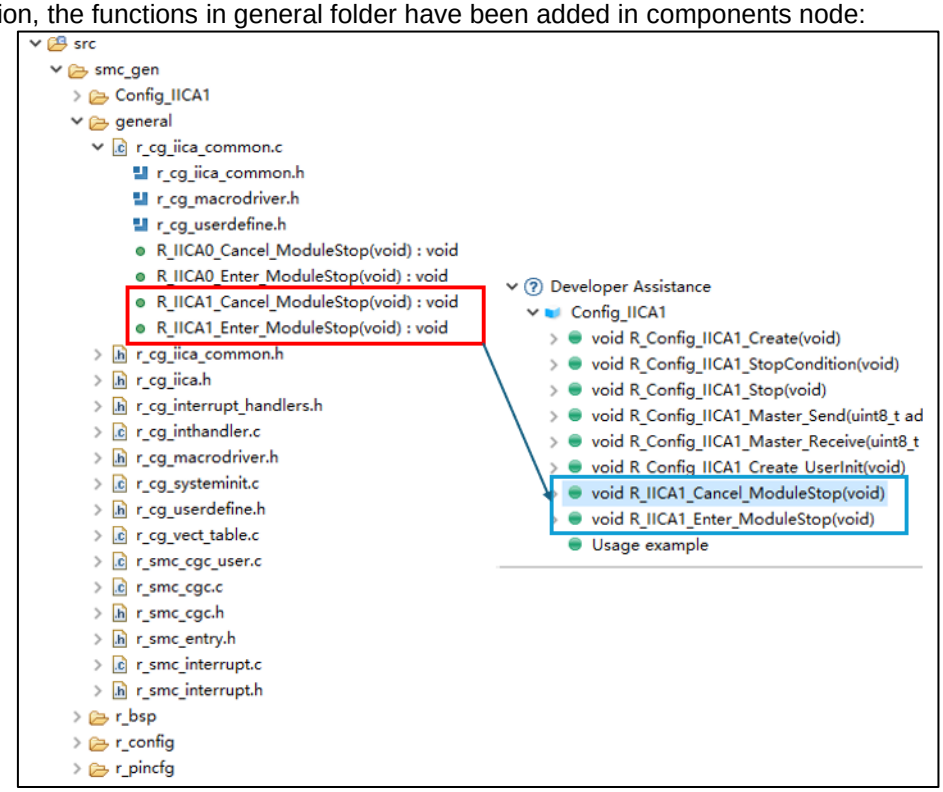

Figure 3-6 Common functions node for developer assistance

3.3.4 Improvement for adding a new interrupt vector routine for NMI interrupt

From this version, the NMI interrupt will be pointed to a new interrupt vector routine nmi_handler() function in r_cg_vect_table.c:

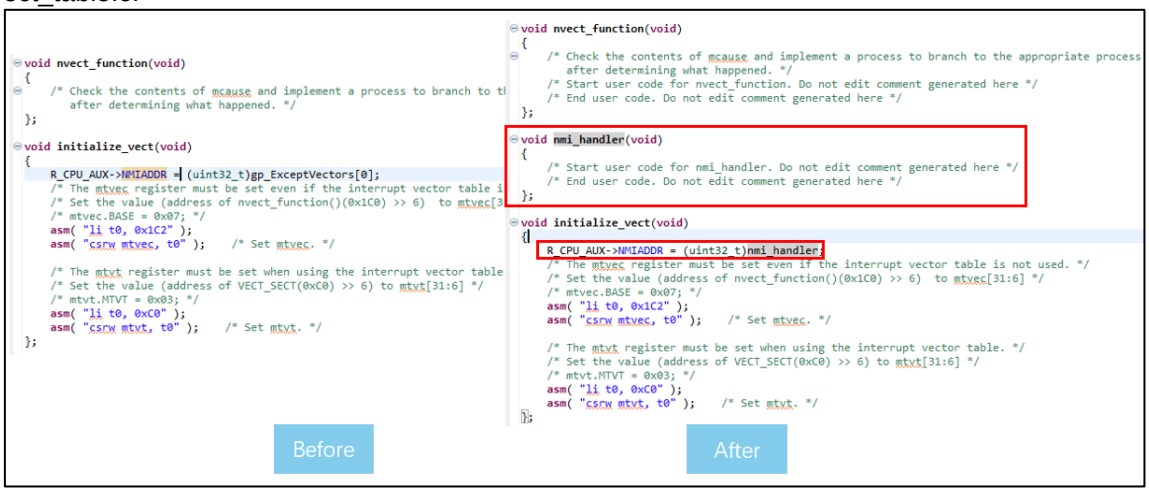

Figure 3-7 New interrupt vector routine for NMI interrupt

3.3.5 Improvement for removing PMR setting for ICU pins

From this version, the pin setting code of ICU will be removed PMR setting code in R_Config_ICU_Create().

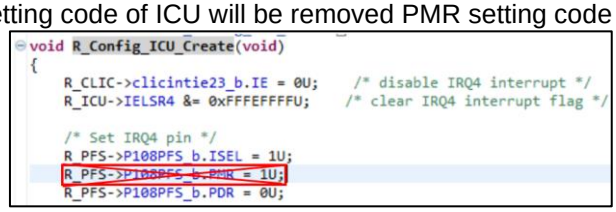

Figure 3-8 Remove PMR setting for ICU pins

4. Points for Limitation

This section describes points for limitation regarding the Smart Configurator for RISC-V MCU V1.2.0.

4.1 List of Limitation

Table 4-1 List of Limitation ✓: Applicable, -: Not Applicable

No	Description	G021	Remarks
1	Note on extra help document issue	✓	
2	Note on PORT functions in developer assistance function cannot be dragged issue	✓	
3	Note on ITL function 16 bit capture mode with channel 0 and 1 can't be used together with 16 bit count mode with channel 2 and 3 issue	✓	
4	Note on BSP machine timer does not support refresh operation issue	✓	

4.2 Details of Limitation

4.2.1 Note on extra help document issue

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

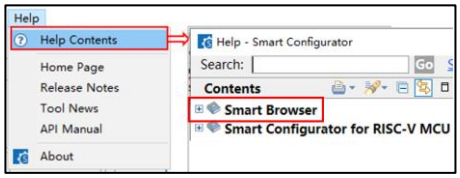

Figure 4-1 Extra help issue

4.2.2 Note on PORT functions in developer assistance function cannot be dragged issue

In e² studio version, when using Developer Assistance function, there are two PORT functions cannot be dragged to the file.
Please add these two functions manually.
This issue will be fixed in next release.

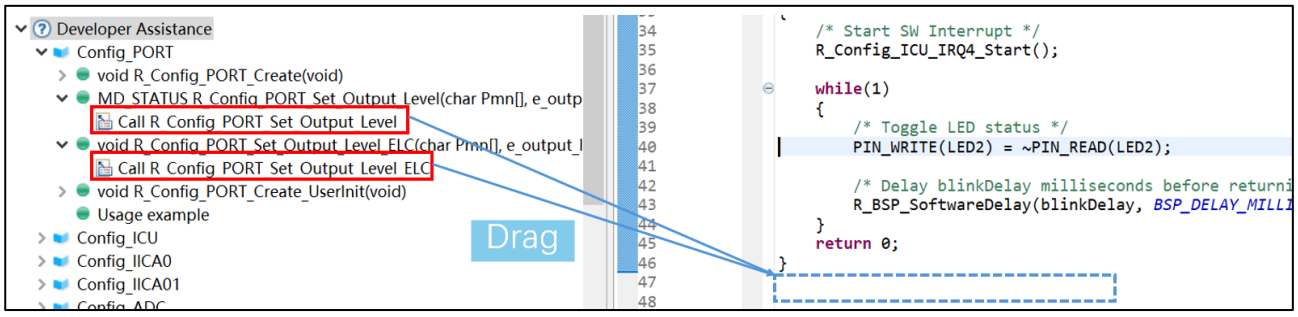

Figure 4-2 PORT functions cannot be dragged issue

4.2.3 Note on ITL function 16 bit capture mode with channel 0 and 1 can't be used together with 16 bit count mode with channel 2 and 3 issue

When using 16 bit capture mode with channel 0 and 1,if user wants to use 16 bit count mode with channel 2 and 3 at same time. Smart Configurator only support selecting "ITLCMP01 compare match interrupt" as "Capture trigger". If user wants to use the "Capture trigger" other than "ITLCMP01 compare match interrupt" for 16 bit count mode with channel 2 and 3, Smart Configurator can't support these settings now.

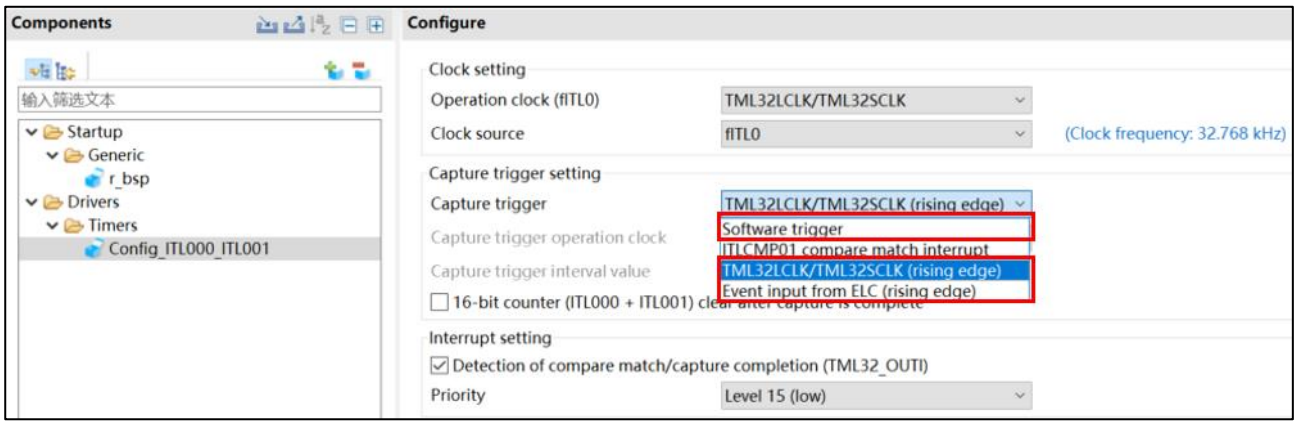

Figure 4-3 "Capture trigger" other than "ITLCMP01 compare match interrupt" settings

As a workaround, please change the code in the following steps manually. If code generation is executed again after changing the code, the code will be conflicted. Please refer to chapter 11.3.2 Steps for Resolving the Merge Conflict of RISC-V MCU Smart Configurator User's Guide: e² studio.

Step1: Create a component - Interval Timer (ITL000, ITL001) as 16bit Capture Mode. Set "ITLCMP01 compare match interrupt" as "Capture trigger" and generate code.

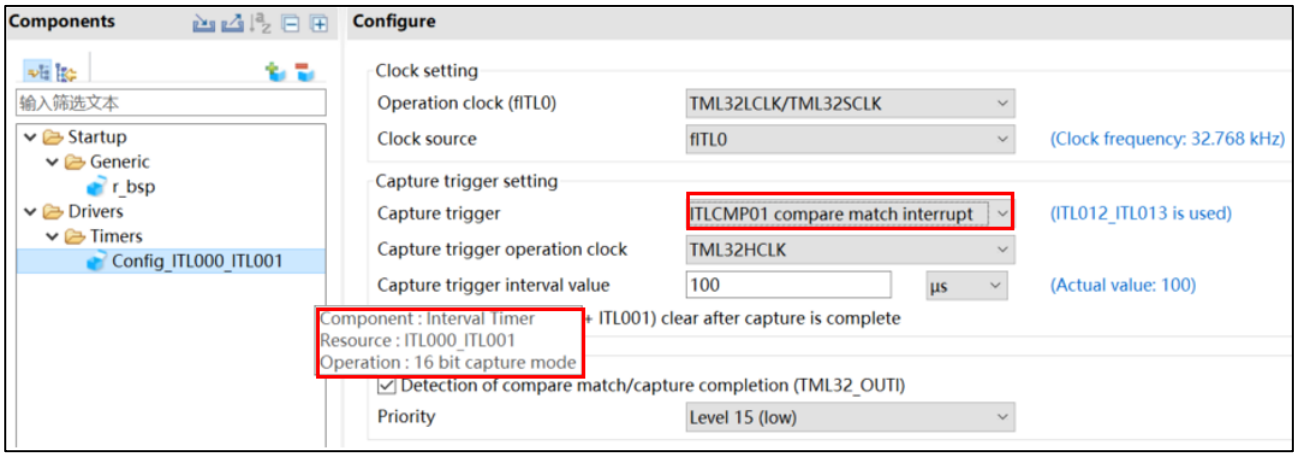

Figure 4-4 Set "ITLCMP01 compare match interrupt" as "Capture trigger"

Step2: Change driver code manually. Using TML32LCLK/TML32SCLK (rising edge) as example:

1) Chang code in Config_ITL000_ITL001.c file: The macro definition can be found in ..\smc_gen\general\r_cg_itl.h.

```
void R_Config_ITL000_ITL001_Create(void)
{
    …
    /* Capture setting */
…
    R_TML32->ITLCC0 |=
_00_ITL_CAPTURE_COUNTER_RETAIN;
    R_TML32->ITLCC0 &=
 FC_ITL_CAPTURE_TRIGGER_CLEAR;
    R_TML32->ITLCC0 |=
_01_ITL_CAPTURE_TRIGGER_INTERNAL;


R_Config_ITL000_ITL001_Create_UserInit();
}
```

```
void R_Config_ITL000_ITL001_Create(void)
{
    …
    /* Capture setting */
…
    R_TML32->ITLCC0 |=
_00_ITL_CAPTURE_COUNTER_RETAIN;
    R_TML32->ITLCC0 &=
_FC_ITL_CAPTURE_TRIGGER_CLEAR;
    /* Start user code */
    R_TML32->ITLCC0 |=
_02_ITL_CAPTURE_TRIGGER_TML32LCLK_TML32SCLK;
    /* End user code */

R_Config_ITL000_ITL001_Create_UserInit();}
```

2) Change code in Config_ITL000_ITL001.c file:

```
void R_Config_ITL000_ITL001_Start(void)
{
    R_TML32->ITLS0 &=
(uint8_t)~_10_ITL_CAPTURE_COMPLETE_DETECT
E;
    R_TML32->ITLMKF0 &=
(uint8_t)~_10_ITL_CAPTURE_COMPLETE_MASK;
    /* Mask channel 2 compare match
status flag */
    R_TML32->ITLMKF0 |=
 04 ITL CHANNEL2 COUNT MATCH MASK;
    R_TML32->ITLCTL0_b.EN0 = 1U;
    R_TML32->ITLCTL0_b.EN2 = 1U;
}
```

```
void R_Config_ITL000_ITL001_Start(void)
{
    R_TML32->ITLS0 &=
(uint8_t)~_10_ITL_CAPTURE_COMPLETE_DETECTE;
    R_TML32->ITLMKF0 &=
(uint8_t)~_10_ITL_CAPTURE_COMPLETE_MASK;
    /* Mask channel 2 compare match status
flag */
    /* Start user code */
    R_TML32->ITLMKF0 &=
(uint8_t)~_04_ITL_CHANNEL2_COUNT_MATCH_MASK;
    /* End user code */
    R_TML32->ITLCTL0_b.EN0 = 1U;
    R_TML32->ITLCTL0_b.EN2 = 1U;
}
```

3) Add code in ..\smc_gen\general\r_cg_itl_common_user.c file:

```
void r_itl_interrupt(void)
{
    R_ICU->IELSR20 &= 0xFFFEFFFFU;    /*
clear TML32_OUTI interrupt flag */
    if (_10_ITL_CAPTURE_COMPLETE_DETECTE
== (R_TML32->ITLS0 &
_10_ITL_CAPTURE_COMPLETE_DETECTE))
    {
        R_TML32->ITLS0 &=
(uint8_t)~_10_ITL_CAPTURE_COMPLETE_DETECT
E;

R_Config_ITL000_ITL001_Callback_Shared_In
terrupt();
    }
    R_TML32->ITLS0 &=
(uint8_t)~_01_ITL_CHANNEL0_COUNT_MATCH_DE
TECTE;
}
```

```
void r_itl_interrupt(void)
{
    R_ICU->IELSR20 &= 0xFFFEFFFFU;    /*
clear TML32_OUTI interrupt flag */
    if (_10_ITL_CAPTURE_COMPLETE_DETECTE ==
(R_TML32->ITLS0 &
_10_ITL_CAPTURE_COMPLETE_DETECTE))
    {
        R_TML32->ITLS0 &=
(uint8_t)~_10_ITL_CAPTURE_COMPLETE_DETECTE;

R_Config_ITL000_ITL001_Callback_Shared_Inter
rupt();
    }
    /* Start user code */
    if (_04_ITL_CHANNEL2_COUNT_MATCH_DETECTE
== (R_TML32->ITLS0 &
_04_ITL_CHANNEL2_COUNT_MATCH_DETECTE))
    {
        R_TML32->ITLS0 &=
(uint8_t)~_04_ITL_CHANNEL2_COUNT_MATCH_DETEC
TE;

R_Config_ITL000_ITL001_Callback_Shared_Inter
rupt();
    }
    /* End user code */
    R_TML32->ITLS0 &=
(uint8_t)~_01_ITL_CHANNEL0_COUNT_MATCH_DETEC
TE;
}
```

4.2.4 Note on BSP machine timer does not support refresh operation issue

When using the machine timer in default r_bsp v1.20, if user wants to use refresh operation, Smart Configurator can't support refresh function. User needs to add code manually.
As a workaround, in r_bsp v1.21 has supported refresh operation.
Please use r_bsp v1.21 by following steps:

Step1: Download r_bsp v1.21 by [Download RISC-V MCU Software Integration System Modules].
① Click [Download RISC-V MCU Software Integration System Modules] link→ ② Click BSP Rev.1.21→

③ Click [Download].

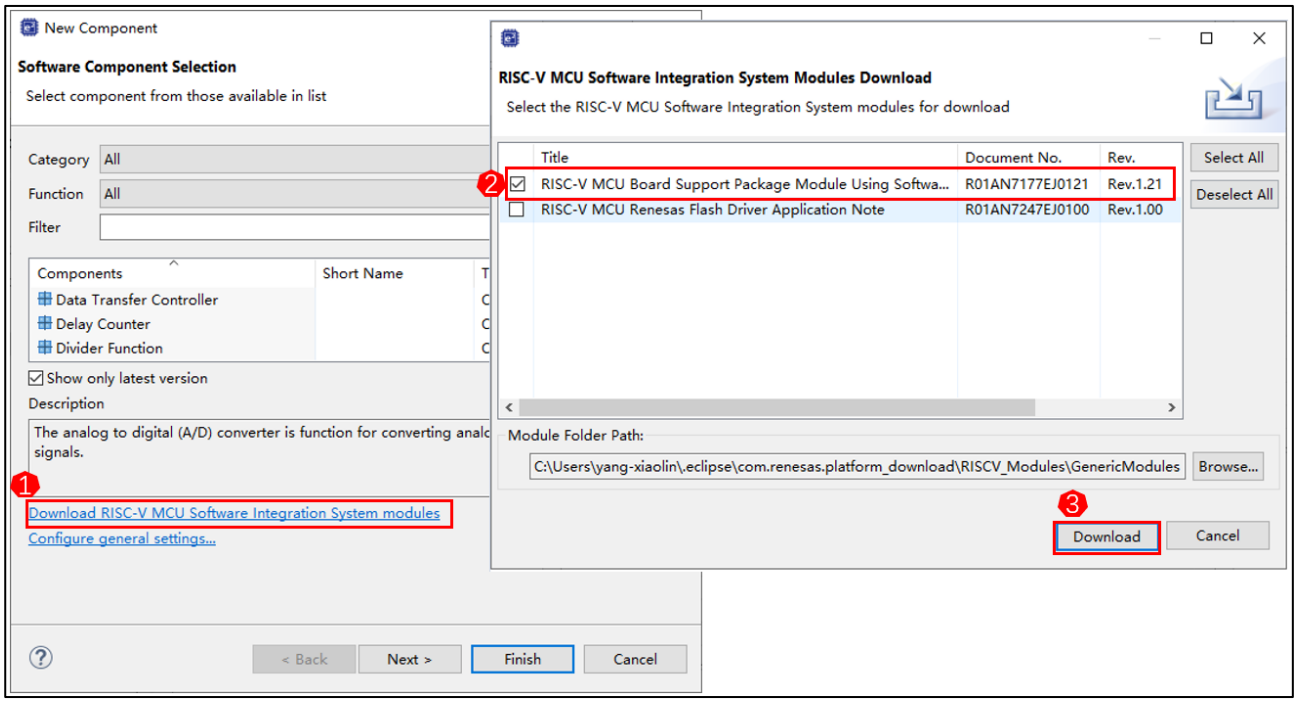

Figure 4-5 Download r_bsp v1.21 by [Download RISC-V MCU Software Integration System Modules]

Step2: Change the [Code generation behavior] setting to "Re-generate all component files" by Click [Configure general setting…] in Step1 "New Component" window.

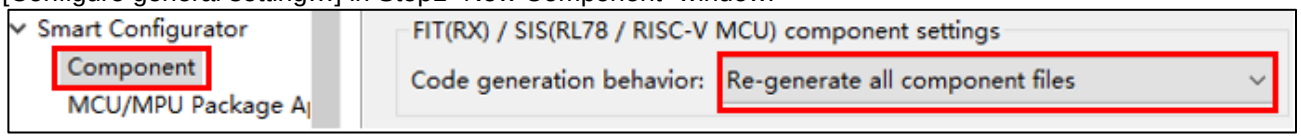

Figure 4-6 [Code generation behavior] setting

Note: If the \smc_gen\r_config\ r_bsp_config.h file cannot be updated correctly, please using following workaround: Remove r_bsp by right-click menu → Click generate code → Add BSP v1.21 by [New Component] dialog → Click generate code again.

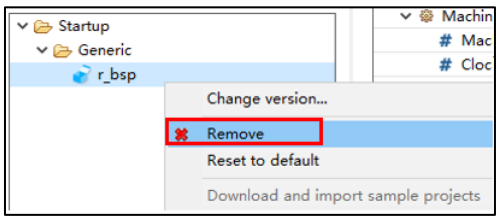

Figure 4-7 Remove r_bsp

Step 3: Change r_bsp version.
① Right-click [Change version] → ② Confirm BSP version is Rev.1.21→ ③ Click [Next]→

④ Click [Finish]→ ⑤ Click [Yes] →⑥ Click [Proceed] .

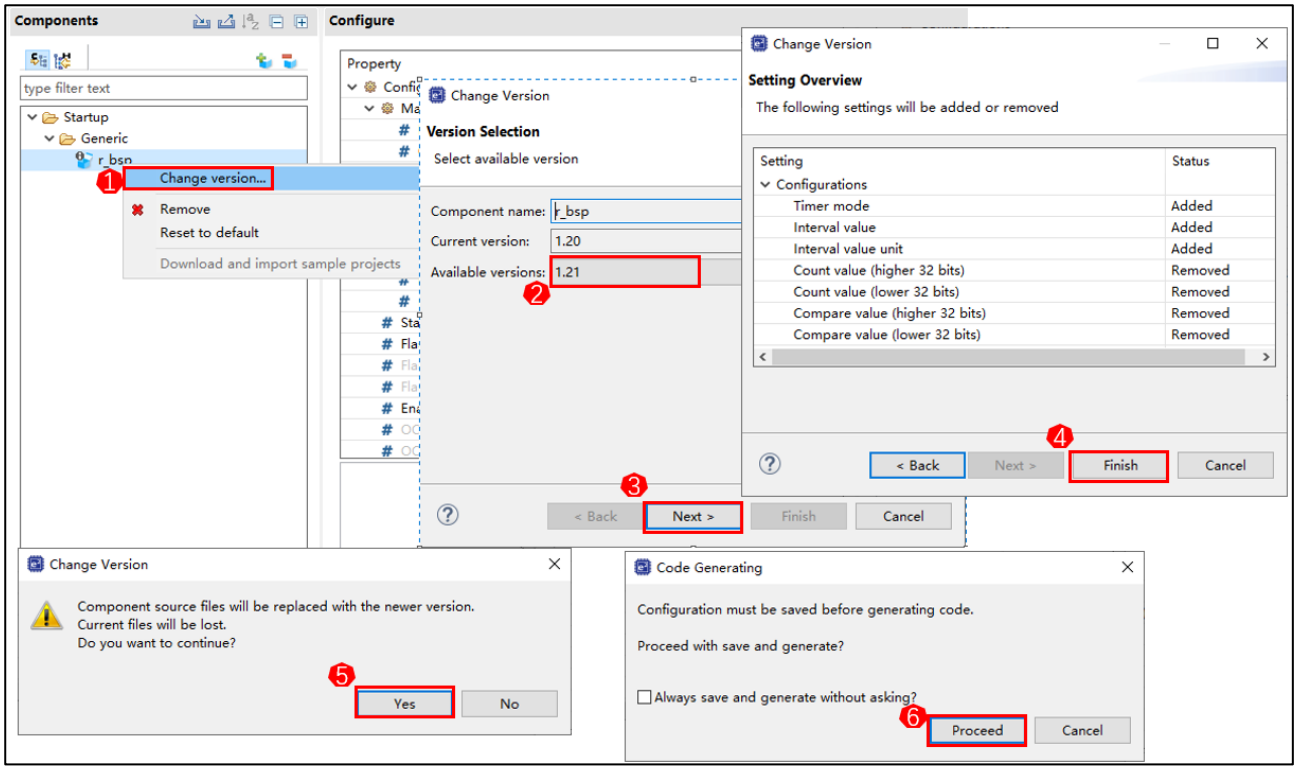

Figure 4-8 Change r_bsp version

Step 4: When using “Periodic” of Timer mode, please add code in ..\smc_gen\general\ r_cg_inthandler.c manually. Detailed information of machine timer, please refer to ..\smc_gen\r_bsp\doc\en\r01an7177ej0121-risc-v-mcu-bsp.pdf chapter 3.2.4 Machine Timer Operation and chapter 5.3 API for Machine Timer.

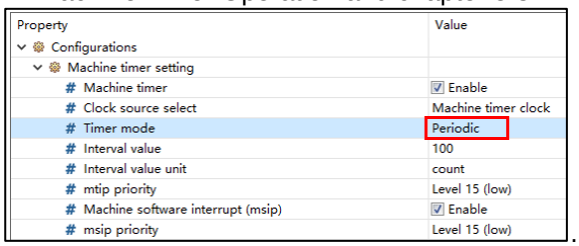

Figure 4-9 “Periodic” of Timer mode

```
#include "r_cg_interrupt_handlers.h"
/* Start user code */
#include "platform.h"
/* End user code */
…
void INT_ACLINT_MTIP(void)
{
      /* Start user code */
#if BSP_CFG_MACHINE_TIMER == 1 && BSP_CFG_MACHINE_TIMER_MODE == 1
      machine_timer_refresh();
#endif
      /* End user code */
    …
}
```

5. Points for Caution

This section describes points for caution regarding the Smart Configurator for RISC-V MCU V1.2.0.

5.1 List of Caution

Table 5-1 List of Caution **✓: Applicable, -: Not Applicable**

No	Description	G021	Remarks
1	Note on the installation of the Smart Configurator	✓	
2	Note on the include path update issue when renaming the component's configuration name	✓	
3	Note on TAU Input Signal High/Low level Measurement components.	✓	
4	Note on using the user code protection feature	✓	
5	Note on the build error when an interrupt is not allocated to any interrupt vector	✓	
6	Note on SIS modules cannot be showed in New Component window	✓	
7	Note on the start address 00000000 is not from start.s file function for Blinky CPP LLVM project	✓	
8	Note on the issue when opening the emProject file in SEGGER	✓	

5.2 Details of Caution

5.2.1 Note on the installation of the Smart Configurator

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

5.2.2 Note on the include path update issue when renaming the component's configuration name

When renaming the added component's configuration in e2 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.

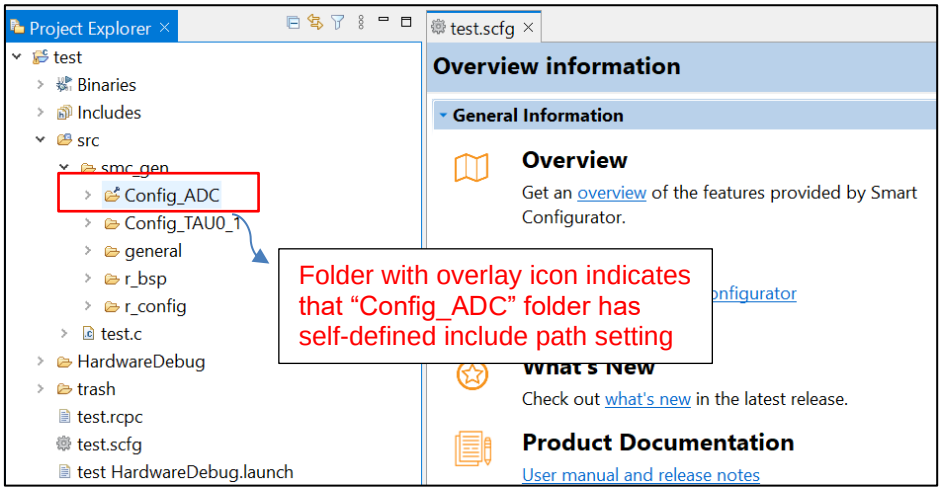

Figure 5-1 Interval Timer component configuration before renaming

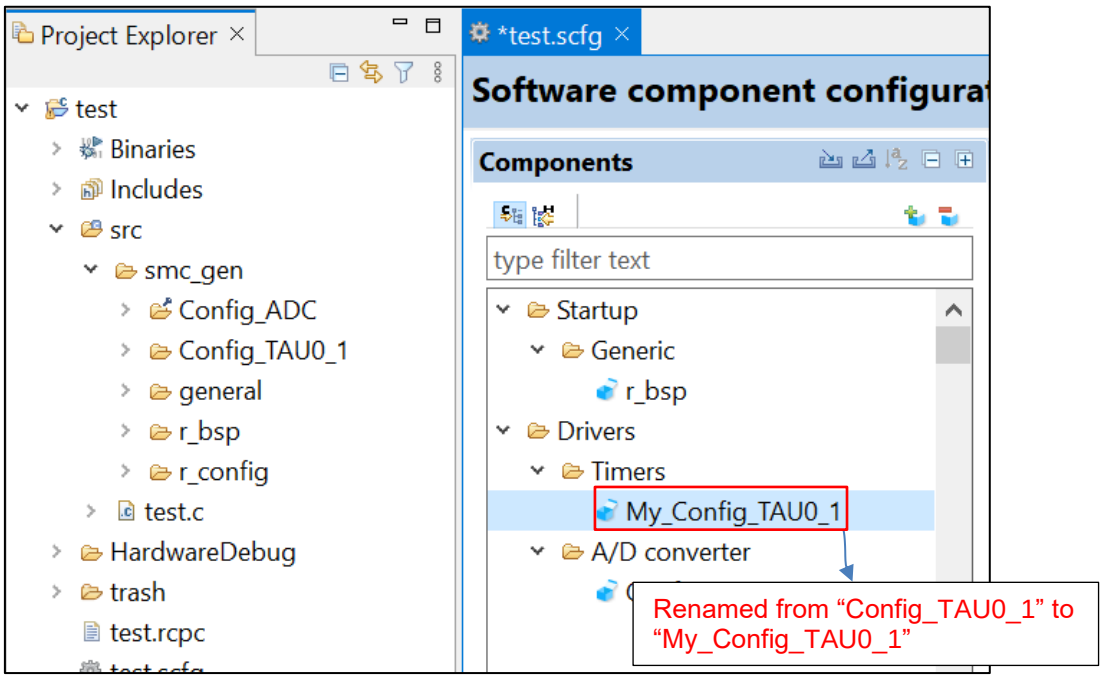

Figure 5-2 The Interval Timer component configuration after renaming

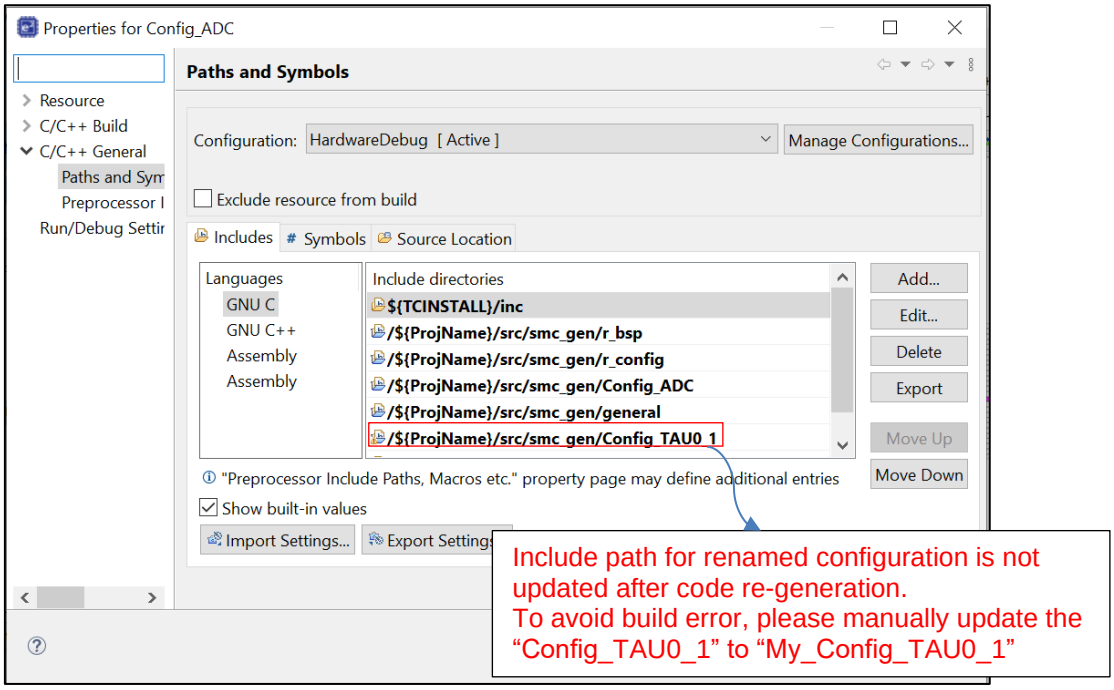

Figure 5-3 Include path setting for the “Config_ADC” configuration

5.2.3 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.

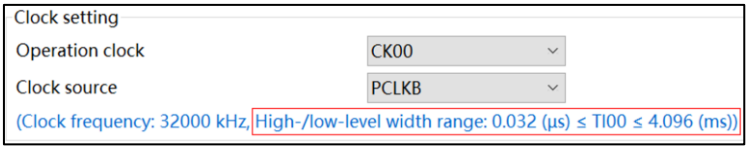

Figure 5-4 High/Low level width min value

5.2.4 Note on using the user code protection feature

The 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 */
```

5.2.5 Note on the build error when an interrupt is not allocated to any interrupt vector

Use IRQ0 as example, when IRQ0 is used in interrupt component, but an interrupt error is shown in “Configuration Problems” window. It means all interrupt vectors which can used by IRQ0 have been used.

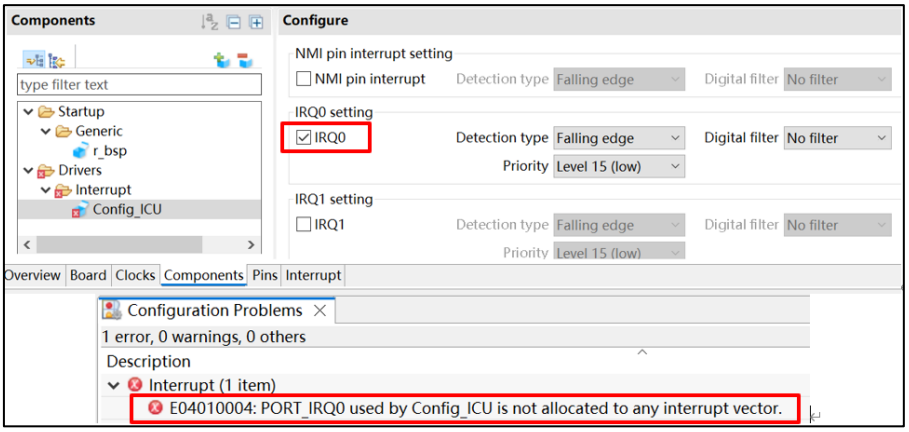

Figure 5-5 An interrupt is not allocated to any interrupt vector

When you generate code, some build error will be shown:

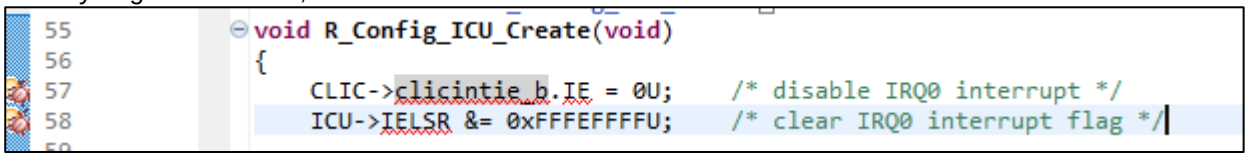

Figure 5-6 Build errors when an interrupt is not allocated to any interrupt vector

To solve these build errors, please refer to Smart Configurator User's Manual e² studio or IAREW, SEGGER Embedded Studio chapter 4.6.4 Resolving Interrupt error.

5.2.6 Note on the start address 00000000 is not from start.s file function for Blinky CPP LLVM project

When using a Blinky CPP LLVM project, the start addresse 00000000 is not from start.s file function.

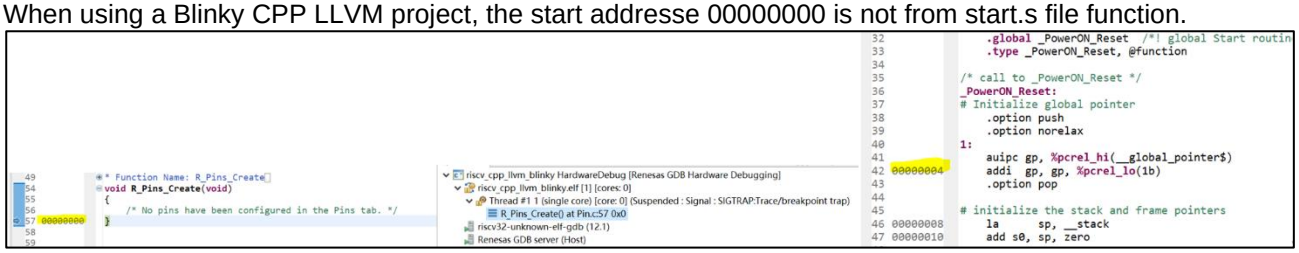

Figure 5-7 Incorrect function of start address 00000000

Correct start function address should in BSP start.s file and *_PowerON_Reset*().

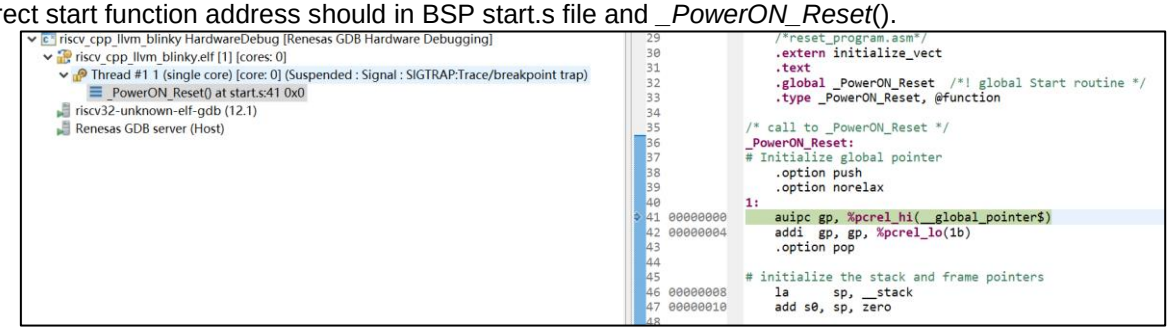

Figure 5-8 Start address 00000000 is from start.s file and _PowerON_Reset()

As a workaround, please changed the [Debug format] to dwarf-2 or dwarf-3:

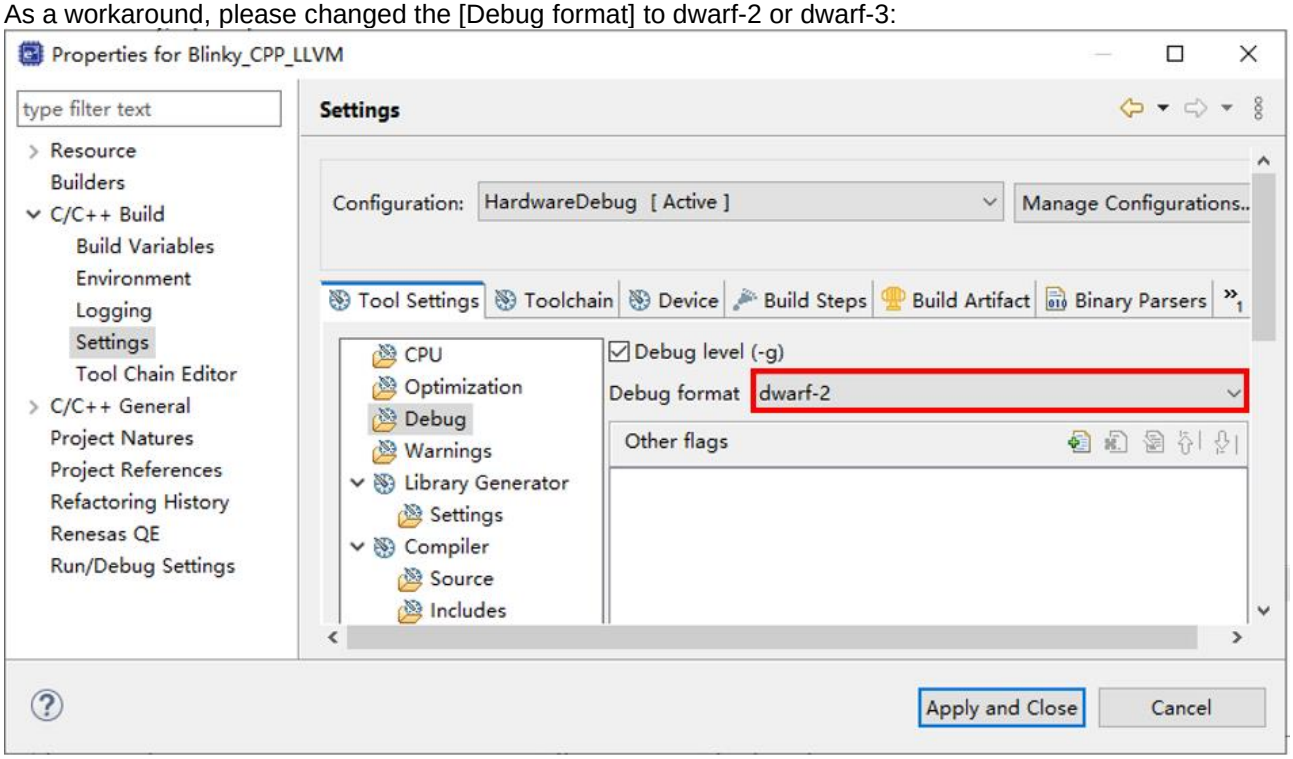

Figure 5-9 Change the [Debug format] CPP LLVM project

5.2.7 Note on the issue when opening the emProject file in SEGGER

When opening the emProject file generated by SEGGER. The following prompt box will be pop up.

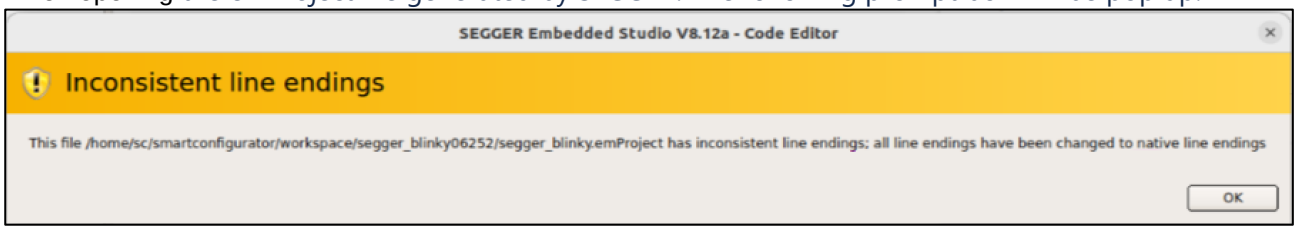

Figure 5-10 The issue when opening the emProject file

As a workaround, please "Ctrl + A" right-click change Line ending to “Unix”.

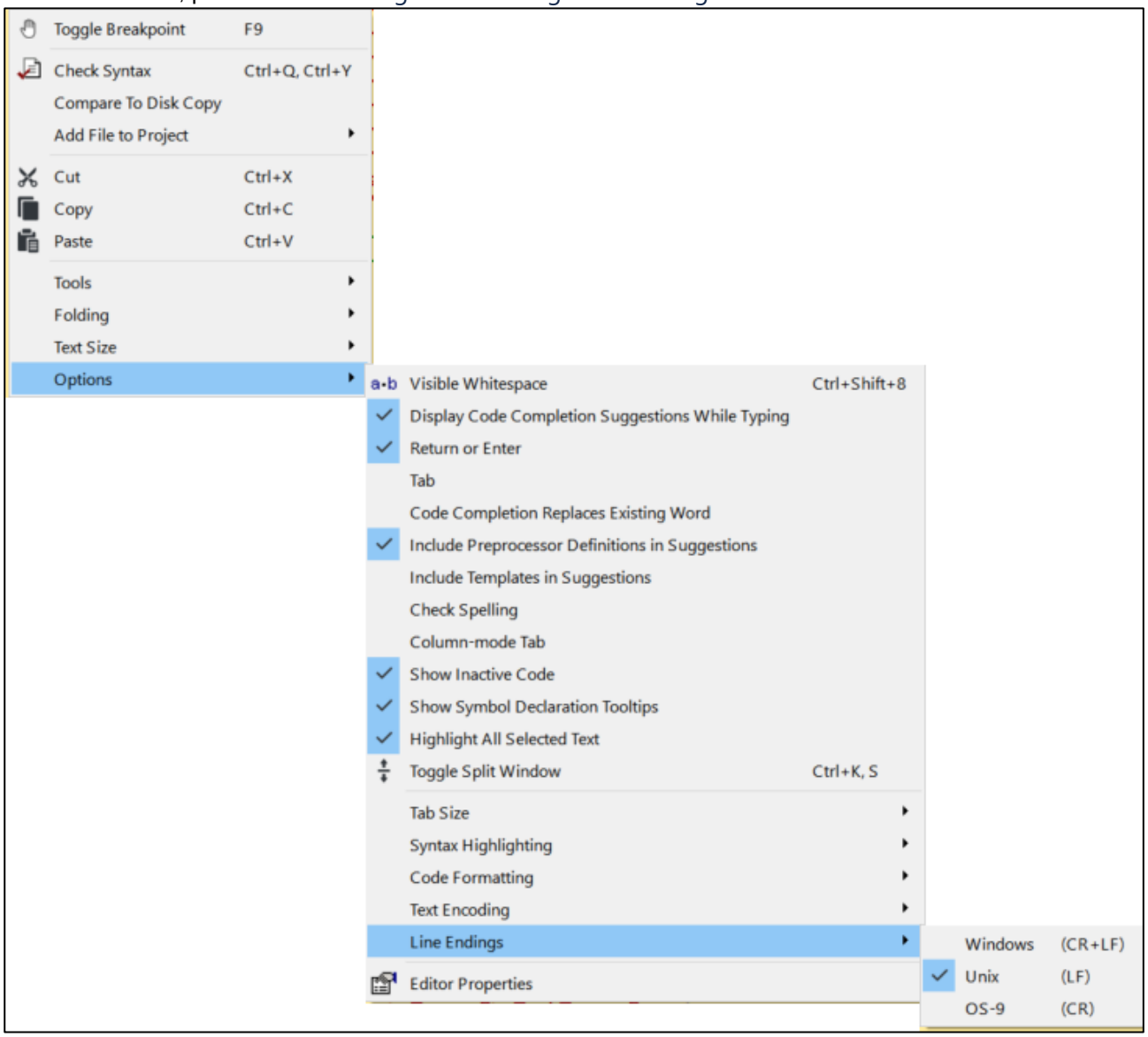

Figure 5-11 Change Line ending to “Unix”

Revision History

Rev.	Section	Description
1.00	-	First edition issued
1.01	4. Points for Limitation	Add 4.2.4 Note on BSP machine timer does not support refresh operation issue

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

2. Processing at power-on

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

3. Input of signal during power-off state

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

4. Handling of unused pins

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

5. Clock signals

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

6. Voltage application waveform at input pin

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

7. Prohibition of access to reserved addresses

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

8. Differences between products

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

Notice

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

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

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

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

Contact information

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