

RL78 Family

RL78 Hardware CRC functions

Introduction

Many applications need to check the integrity of a code image or data communication stream by using a CRC function to verify data errors have not occurred. Most RL78 MCUs have a built-in Hardware CRC function that can be used for this purpose

Target Device

RL78 MCUs having the S-2 or S-3 CPU core. The RL78 S-1 core (such as on RL78/G10) does not have the built-in HW CRC functions. However, any RL78 with S-1 CPU core can utilize Software CRC functions to emulate the RL78 High Speed and General-purpose CRC functions.

Contents

1. Overview	3
1.1 RL78 High Speed Hardware CRC Function	3
1.2 RL78 General Purpose ("Low Speed") Hardware CRC Function.....	3
2. Software Environment	4
3. Application Note CRC functions implemented.....	4
3.1 Operating the RL78 High Speed HW CRC function	4
3.2 Operating the RL78 General Purpose ("Low Speed") HW CRC function	5
4. Windows-based CRC calculation Software utility (TestCRC).....	5
4.1 Operating Windows based TestCRC.....	6
4.2 Running "TestCRC Low Speed CRC (LSB)" Mode.....	6
4.3 Running "TestCRC High Speed CRC (MSB)" Mode	6
4.4 Running "Low Speed As High Speed" Mode:	6
5. Operating the RL78 Sample Projects	6
5.1 Running the application note sample code projects as-is	7
5.2 Implementing the CRC code example in User Code projects	7
6. Using IAR EWRL78 sample software project	7
6.1 Linker settings for High Speed Hardware CRC use (EWRL78)	7
6.2 Linker settings for General Purpose Hardware CRC use (EWRL78)	8
6.3 Settings for both High Speed CRC and General Purpose CRC use (EWRL78).....	9
7. Using e2studio with IAR compiler sample software project.....	9
7.1 Linker settings for High Speed Hardware CRC (e2studio/IAR)	9
7.2 Linker settings for General Purpose Hardware CRC (e2studio/IAR)	10
7.3 Settings for both High Speed CRC and General Purpose CRC use (e2studio/IAR).....	11

- 8. Using e2studio with CCRL compiler sample software project..... 13
 - 8.1 Settings for both High Speed Hardware and General Purpose CRC (e2studio/CCRL)..... 13
 - 8.2 Converter CRC Operation settings for High Speed Hardware CRC (e2studio/CCRL) 15
 - 8.3 Converter CRC Operation settings for General Purpose Hardware CRC (e2studio/CCRL)
..... 15
- 9. Hardware platform used for RL78 CRC sample Software program 16

1. Overview

Users often want to check for run-time integrity of code images or data communication packets by using the built-in RL78 CRC functions in hardware. Specific examples include:

- When checking existing Code Flash image for errors on each MCU Power-ON RESET sequence
- When checking new Code flash blocks after they are received during a Boot loader sequence
- When checking for communication channel errors – especially when transmitted over an RF link, such as Wi-Fi or Blue-Tooth network.

The RL78 has two built-in CRC Hardware calculation functions; (1) High Speed CRC, and (2) General Purpose CRC, also referred to as “Low-speed” CRC function. It is also sometimes desirable to emulate the RL78 Hardware CRC functions in Software, to perform a run-time cross-check, or if implementing CRC calculations in another MCU system not having RL78 Hardware CRC function.

This application note Software project implements CRC calculation methods described in sections 1.1, and 1.2 using built-in RL78 CRC Hardware functions.

1.1 RL78 High Speed Hardware CRC Function

The RL78 High Speed Hardware CRC function is designed to be used on the Code Flash space only. The High Speed CRC operates in HALT mode, after copying 2 machine-level instruction (HALT and RETURN) into RAM memory and calling a function to that code in RAM. The execution time is one CPU/SYSTEM cycle per 4-Byte code flash Word. (for example, execution time is 512 uSEC @32 MHz with 64-KB flash memory size). Since the High Speed CRC function is run during HALT mode, it can't be run concurrently with Application code, and must complete operation before any User code operation can be resumed.

The High Speed Hardware CRC uses the CCITT-16 polynomial of 0x11021, and operates on MSB first order from bit 31 to bit 0. That means the input is NOT bit-reversed. The High Speed Hardware CRC output is a 16-bit result, also NOT bit-reversed. Since the memory architecture is “little-endian” the input of 4 code flash bytes is byte3, byte2, byte1, and byte0, where byte3 is the highest order (MSB) byte in the code flash word. Also the CRC result is in low byte, then high byte order in the CRC

When using the RL78 High Speed HW CRC, a compiler/linker-generated 16-bit CRC would normally be stored at the end code space being checked, usually in the last 4bytes not included in High Speed HW CRC checking range, then referenced and compared with results of the High Speed HW CRC value to see if they match.

1.2 RL78 General Purpose (“Low Speed”) Hardware CRC Function

The RL78 General Purpose Hardware CRC function can be used on any range of code memory or data space, including data Flash and RAM. In fact, the General Purpose Hardware CRC can calculate CRC on a non-contiguous range of data values, one byte at a time. The General Purpose Hardware CRC function operates in CPU RUN mode, and takes at least 2 CPU cycles per each data byte, not counting any User indexing code to point to the next byte. Since the General Purpose Hardware CRC function works with CPU RUN mode it can be multiplexed with Application code. The General Purpose HW CRC result is available after each byte is fed into the CRC input register.

The General Purpose Hardware CRC also uses the CCITT-16 polynomial of 0x11021, but operates on LSB first order from bit 0 to bit 7. That means the input IS bit-reversed. The General Purpose Hardware CRC 16bit output result is also bit-reversed.

Therefore the General Purpose Hardware CRC function will NOT generate the same CRC result, even if run on the same code flash space as the High Speed CRC function. However, the General Purpose Hardware CRC and software can be used to emulate the High-speed Hardware CRC by doing the following:

- Inputting 4byte sizes on N x 16KB (minus last 4 bytes) block sizes, N = 1 to 32**
- Input bytes to General Purpose Hardware CRC are bit-reversed and byte-reversed per each 4-byte input**
- The 16bit General Purpose Hardware CRC result should be bit reversed.**

This application note sample Software does NOT include this emulation software, but the included Windows TestCRC utility does.

2. Software Environment

The sample Software accompanying this application note contains 3 different SW projects, using the SW environment as follows:

Item	Description
Microcontroller used	RL78/G13(R5F100LE)
Operating frequency	- High-speed on-chip oscillator (HOCO) clock: 32 MHz - CPU/peripheral hardware clock: 32 MHz
Operating voltage	5.0 V (can run on a voltage range of 2.9 V to 5.5 V.) LVD operation (VLVD): Reset mode 2.81 V (2.76 V to 2.87 V)
Integrated development environment (e2studio)	e2studio V5.2.2.010 from Renesas Electronics Corp.
C compiler (e2studio)	CC-RL V1.03.00 from Renesas Electronics Corp.
Integrated development environment (IAR)	IAR Systems IAR Embedded Workbench for Renesas RL78 V4.21.3
C compiler (IAR)	IAR Systems IAR C/C++ Compiler for Renesas RL78 V4.21.3.2447

Table 1: Sample Software projects versus environment

3. Application Note CRC functions implemented

Table 2 summarizes the two different Hardware CRC functions implemented in this application note Software project. Of course RL78 CRC Hardware functions are called by User software.

CRC type:	Function Call	Data Input Size	Address range	Input Data Bit Order	Output Data Bit Order
RL78 HIGH SPEED (Background) CRC	r_crc_fast_hardware(...)	32 bit	Code Flash Only: 00000H to N (*1) x 16KB - last 4 bytes	MSB first	Normal
RL78 General Purpose CRC	r_crc_general_hardware(...)	8 bit	Any data bytes	LSB first *2	Reversed *2

Note *1: N = 1 (16KB) to 32 (512KB)

Note *2: General Purpose HW CRC bit reversal on input and output performed automatically

Table 2: Available Sample Software CRC functions implemented

3.1 Operating the RL78 High Speed HW CRC function

The RL78 High Speed HW CRC function requires 13 bytes of instruction code be written to RAM memory space (HALT, RETURN, and 10 bytes of NOPs). Then all interrupts are Disabled globally, the **Flash Memory CRC Control Register (CRC0CTL) is set**, and the RAM code is called as a function. The High Speed HW CRC calculation result is available after approximately 4096 CPU clock cycles per 16KB code space size.

The memory size checked by the High Speed CRC function is controlled by CRC0CTL register, always starts at address 00000H, and ends at an integer number of 16KB code blocks – 4 bytes.

CRC0CTL register:

CRC0EN	0	FEA5	FEA4	FEA3	FEA2	FEA1	FEA0
--------	---	------	------	------	------	------	------

FEA5	FEA4	FEA3	FEA2	FEA1	FEA0	High-speed CRC operation range
0	0	0	0	0	0	00000H to 03FFBH (16 Kbytes - 4 bytes)
0	0	0	0	0	1	00000H to 07FFBH (32 Kbytes - 4 bytes)
0	0	0	0	1	0	00000H to 0BFFBH (48 Kbytes - 4 bytes)
0	0	0	0	1	1	00000H to 0FFFBH (64 Kbytes - 4 bytes)
0	0	0	1	0	0	00000H to 13FFBH (80 Kbytes - 4 bytes)
0	0	0	1	0	1	00000H to 17FFBH (96 Kbytes - 4 bytes)
0	0	0	1	1	0	00000H to 1BFFBH (112 Kbytes - 4 bytes)
0	0	0	1	1	1	00000H to 1FFFBH (128 Kbytes - 4 bytes)
0	0	1	0	0	0	00000H to 23FFBH (144 Kbytes - 4 bytes)
0	0	1	0	0	1	00000H to 27FFBH (160 Kbytes - 4 bytes)
0	0	1	0	1	0	00000H to 2BFFBH (176 Kbytes - 4 bytes)
0	0	1	0	1	1	00000H to 2FFFBH (192 Kbytes - 4 bytes)
0	0	1	1	0	0	00000H to 33FFBH (208 Kbytes - 4 bytes)
0	0	1	1	0	1	00000H to 37FFBH (224 Kbytes - 4 bytes)
0	0	1	1	1	0	00000H to 3BFFBH (240 Kbytes - 4 bytes)
0	0	1	1	1	1	00000H to 3FFFBH (256 Kbytes - 4 bytes)
0	1	0	0	0	0	00000H to 43FFBH (272 Kbytes - 4 bytes)
0	1	0	0	0	1	00000H to 47FFBH (288 Kbytes - 4 bytes)
0	1	0	0	1	0	00000H to 4BFFBH (304 Kbytes - 4 bytes)
0	1	0	0	1	1	00000H to 4FFFBH (320 Kbytes - 4 bytes)
0	1	0	1	0	0	00000H to 53FFBH (336 Kbytes - 4 bytes)
0	1	0	1	0	1	00000H to 57FFBH (352 Kbytes - 4 bytes)
0	1	0	1	1	0	00000H to 5BFFBH (368 Kbytes - 4 bytes)
0	1	0	1	1	1	00000H to 5FFFBH (384 Kbytes - 4 bytes)
0	1	1	0	0	0	00000H to 63FFBH (400 Kbytes - 4 bytes)
0	1	1	0	0	1	00000H to 67FFBH (416 Kbytes - 4 bytes)
0	1	1	0	1	0	00000H to 6BFFBH (432 Kbytes - 4 bytes)
0	1	1	0	1	1	00000H to 6FFFBH (448 Kbytes - 4 bytes)
0	1	1	1	0	0	00000H to 73FFBH (464 Kbytes - 4 bytes)
0	1	1	1	0	1	00000H to 77FFBH (480 Kbytes - 4 bytes)
0	1	1	1	1	0	00000H to 7BFFBH (496 Kbytes - 4 bytes)
0	1	1	1	1	1	00000H to 7FFFBH (512 Kbytes - 4 bytes)
Other than the above						Setting prohibited

Table 3: CRC0CTL settings per Code flash memory size

Note: Allowed CRC0CTL settings of FEA0-FEA5 bits for RL78 MCUs depend on the actual Code flash size of the individual RL78 device used.

When the RL78 On-Chip-Debug (OCD) is used, there is additional debug code inserted at code flash address locations 00002H-00003H (2 bytes), 000CEH-000D7H (10bytes), and monitor code into the last 256 to 768bytes of code flash. Since this debug code is inserted by the debugger after the compiler linker has calculated the CRC/checksum, running the High Speed CRC Hardware function during a debug session will never yield matching results compared to the linker-generated CRC. Therefore when testing the High Speed CRC function, it should be done with the RL78 MCU in stand-alone mode with the debugger turned off. Stand-alone RL78 MCU mode can be achieved by using Renesas Flash Programmer (RFP) to flash a release code image.

3.2 Operating the RL78 General Purpose (“Low Speed”) HW CRC function

It is possible to run the RL78 General Purpose HW CRC function on a portion of code flash image during an RL78 On-Chip-Debug (OCD) session provided:

- (1) The debug code areas are not included, addresses 00002H-00003H (2 bytes), 000CEH-000D7H (10bytes), and debug monitor code (last 256-768 bytes of code flash)
- (2) There are no breakpoints inserted in the code space being CRC checked

Then, the compiler/linker-generated 16bit CRC value can correctly match the General Purpose Hardware CRC run-time value.

4. Windows-based CRC calculation Software utility (TestCRC)

This application note includes a Windows-based CRC calculation Software utility written in Microsoft Visual Studio. This utility is a stand-alone executable that calculates 3 different CRC functions corresponding to the RL78 CRC

functions implemented by the application code Sample project. It provides an independent means of verifying a 16bit CRC result obtained in the run-time RL78 Sample Software project

4.1 Operating Windows based TestCRC :

The included Windows based TestCRC utility can be used to verify the results of the RL78 compiler, linker-generated CRC/checksum value or to verify the RL78 Hardware CRC function result. This utility operates on .bin, .hex, or .mot files, but only on code flash memory space and must be contiguous memory space. To run TestCRC, just launch the TestCRC.exe application. The Windows based TestCRC has 3 different CRC generation settings:

- (1) Low Speed CRC (LSB) mode, which duplicates the CRC/checksum result of the RL78 General Purpose CRC Hardware function
- (2) High Speed CRC (MSB) mode, which duplicates the CRC/checksum result of the RL78 HIGH SPEED CRC Hardware function
- (3) Low Speed As High Speed mode which internally inputs single bytes, bit-reversing each byte, and byte reversing each 4byte input to emulate the RL78 High Speed Hardware function.

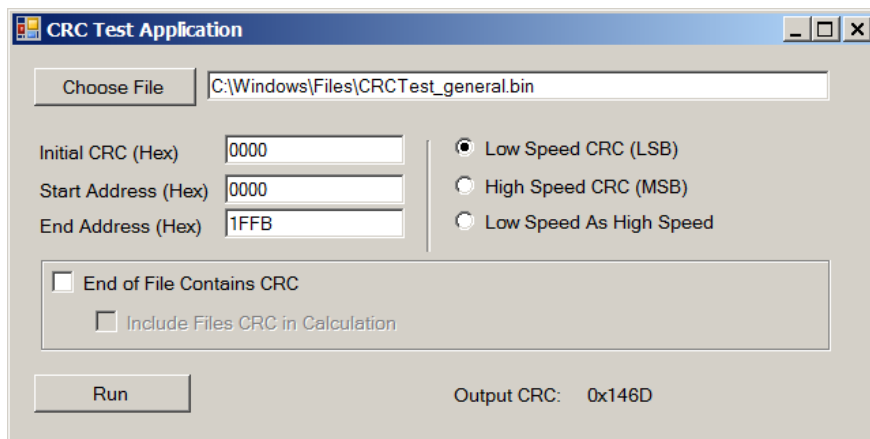


Figure 1: TestCRC GUI Interface in Windows

4.2 Running “TestCRC Low Speed CRC (LSB)” Mode

Normally, the initial CRC value would be set to 0x0000. The Start and End address values can be any valid address and must be within the code flash address space found in the target .bin, .hex or .mot file. Also, the End address must be greater or equal to the Start address. Normally, the “End of File Contains CRC” setting should remain unchecked.

4.3 Running “TestCRC High Speed CRC (MSB)” Mode

Normally, the initial CRC value would be set to 0x0000. To match the RL78 High Speed Hardware CRC function, the Start address must always be 0x00000 and the End address must match the RL78 HS Hardware CRC CRC0CTL register setting to have a valid match. Of course, the Start/End address range must be within the code flash address space found in the target .bin, .hex or .mot file. The “End of File Contains CRC” setting must remain unchecked.

4.4 Running “Low Speed As High Speed” Mode:

This is an optional mode where the setting requirements would normally be the same as the TestCRC High Speed CRC (MSB) mode. It would be used to verify a combination of SW used with General Purpose CRC function to emulate High Speed Hardware CRC function.

5. Operating the RL78 Sample Projects

This application note sample code implements the High Speed and General Purpose RL78 CRC Hardware functions outlined in Section 1 and Section 3, Table 1, with 3 different Software sample projects listed in Section 2, Environment. To run the sample projects as a demonstration on RSKRL78G13 (with R5F100LE device), select either the RL78 High Speed Hardware CRC result or General purpose Hardware CRC result. The sample software is set up to simply

calculate CRC on these memory ranges (00000H-03FFBH for High Speed CRC function or 000D8-03FFBH for General Purpose CRC function), but the User can modify for their own memory range requirements:

5.1 Running the application note sample code projects as-is

Select either High Speed or General Purpose CRC function call

- (1) High Speed CRC: address range 00000H to 03FFBH, in stand-alone MCU operation only, with linker-generated 16-bit CRC stored at address 03FFEh

To set main.c for High Speed Hardware CRC test:

```
Include the line for #ifdef:
#define HI_SPEED_CRC_TEST
```

- (2) General Purpose CRC: address range 000D8H to 03FFB, in stand-alone MCU operation or during a debug session (provided target address range is not being modified by debug code), with linker-generated 16-bit CRC stored at address 03FFEh

To set main.c for General Purpose Hardware CRC test

```
Comment out the line for #ifdef:
//#define HI_SPEED_CRC_TEST
```

Application Note Sections 6, 7, and 8 show how to modify the linker settings for High Speed CRC or General Purpose CRC, and how to set the desired CRC memory range being checked.

5.2 Implementing the CRC code example in User Code projects

To utilize the sample CRC code in another User Software project, the following must be done:

- Include any Hardware specific header files (similar to BSP_RSKRL78G13.h) that specify memory “RANGE” (if needed for High Speed CRC) and any port pin identification for LED annunciators (if needed). If the target RL78 MCU has a memory range larger than 64KB, the RANGE options will need to be expanded.
- Include fast_crc.c or general_crc.c file, depending on which CRC type is needed
- Include crc.h file
- Portions of main.c will be needed to call either r_crc_fast_hardware() or r_crc_general_hardware(), and to access the linker-generated fast or general CRC value stored in code flash, these lines are needed:

```
uint16_t __far *g_linker_crc_pointer = (uint16_t __far *) CRC_ADDRESS_LINKER;
uint16_t linker_crc;
linker_crc = (*g_linker_crc_pointer);
```

Application Note Sections 6, 7, and 8 show how to modify the linker settings for High Speed CRC or General Purpose CRC, and how to set the desired CRC memory range being checked.

6. Using IAR EWRL78 sample software project

6.1 Linker settings for High Speed Hardware CRC use (EWRL78)

Set EWRL78 linker GUI (go to Project >> Options >> Linker >> Checksum)

- Fill unused code memory, fill pattern = 0xFF
- Start address = 0x0000, End address = 0x3FFB or other valid High Speed CRC ending address setting
- “Generate checksum” box checked, Checksum size = 2 bytes, Alignment = 1
- Algorithm = CRC16, Complement = As is, Initial value = 0x00
- Bit Order = MSB first, “Use as input” box checked

- “Reverse byte order within word” box un-checked, Checksum unit size = 32-bit

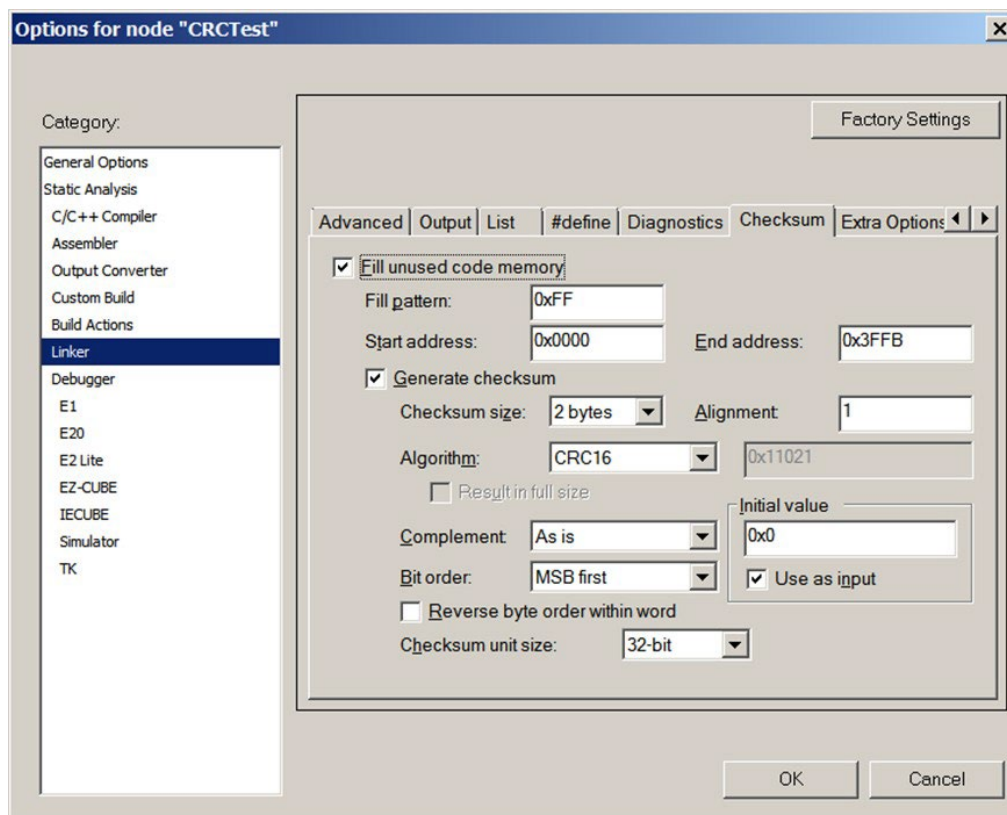


Figure 2: Linker CRC/Checksum Options for High Speed CRC in EWRL78

6.2 Linker settings for General Purpose Hardware CRC use (EWRL78)

Set EWRL78 linker GUI - go to Project >> Options >> Linker >> Checksum (See Figure 3.)

- Fill unused code memory, fill pattern = 0xFF
- Start address = 0x00D8, End address = 0x3FFB or other valid ending address setting
- “Generate checksum” box checked, Checksum size = 2 bytes, Alignment = 1
- Algorithm = CRC16, Complement = As is, Initial value = 0x00
- Bit Order = LSB first, “Use as input” box checked
- “Reverse byte order within word” box **checked**, Checksum unit size = 8-bit

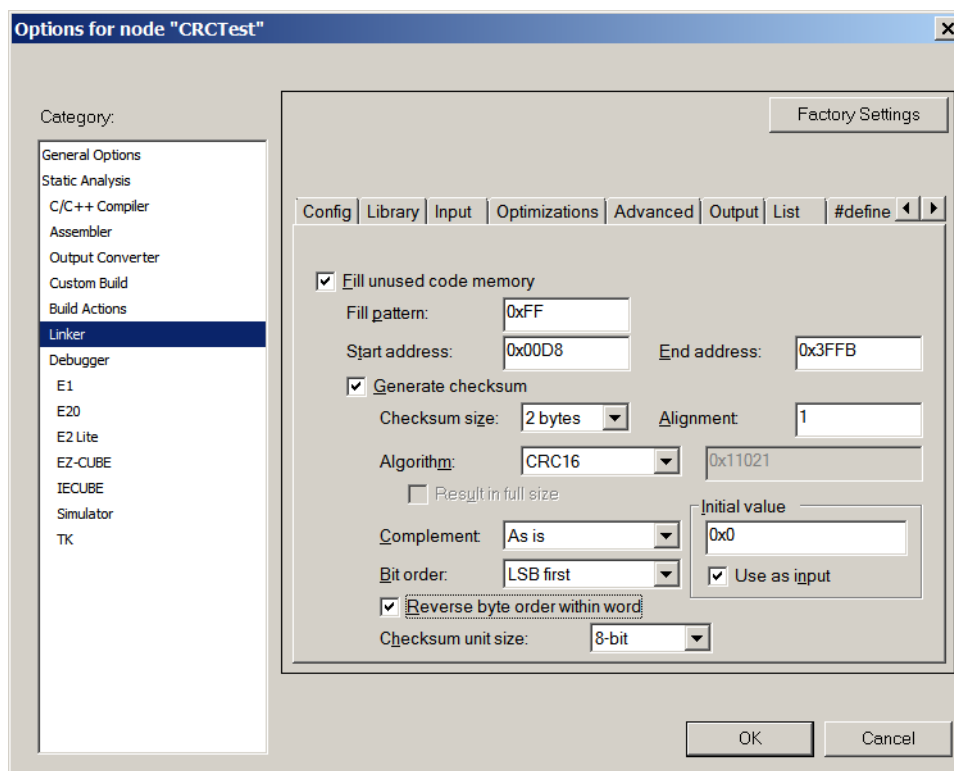


Figure 3: Linker CRC/Checksum Options for General Purpose CRC in EWRL78

6.3 Settings for both High Speed CRC and General Purpose CRC use (EWRL78)

The sample SW project uses a modified copy of the .icf file (ILINK Configuration File), r5f100le.icf in IAR EWRL sample project source folder.

- Verify this line in .icf file if using 16KB code flash setting in High Speed CRC,
place at address mem:0x03FFE { ro section .checksum };
- Modify CRC result placement address if different CRC calculation range is used, but the result must be placed outside the calculated CRC range. For example, when using High Speed CRC function, modify the placement address to accommodate the CRC0CTL register setting, using a CRC result address in the last 2 bytes of last 16KB section (example; 0x0FFFE for CRC0CTL, FEA5-FE0 = 0x03, 64KB code flash setting)

7. Using e2studio with IAR compiler sample software project

7.1 Linker settings for High Speed Hardware CRC (e2studio/IAR)

Set e2studio linker GUI - go to Project >> Properties >> Settings >> IAR RL78 ILINK Linker >> Checksum (see Figure 4.).

- Fill unused code memory, fill pattern = 0xFF
- Start address = 0x0000, End address = 0x3FFB or other valid ending address setting
- “Generate checksum” box checked, Checksum size = 2 bytes, Alignment = 1
- Algorithm = CRC16, Complement = As is, Initial value = 0x0000
- Bit Order = MSB first, “Use as input” box checked
- “Reverse byte order within word” box **unchecked**, Checksum unit size = 32-bit

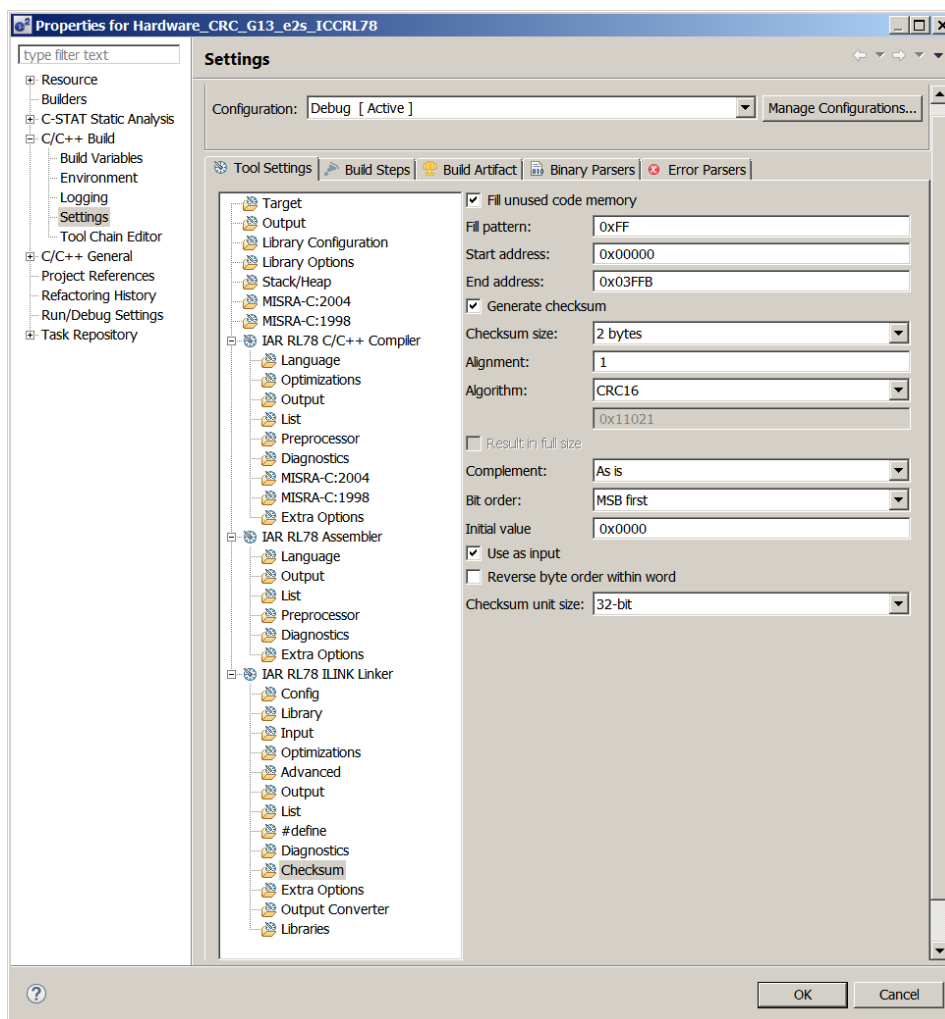


Figure 4: Linker CRC/Checksum Options for High Speed CRC in e2studio/IAR settings

7.2 Linker settings for General Purpose Hardware CRC (e2studio/IAR)

Set e2studio linker GUI (go to Project >> Properties >> Settings >> IAR RL78 ILINK Linker >> Checksum)

- Fill unused code memory, fill pattern = 0xFF
- Start address = 0x00D8, End address = 0x3FFB or other valid ending address setting
- “Generate checksum” box checked, Checksum size = 2 bytes, Alignment = 1
- Algorithm = CRC16, Complement = As is, Initial value = 0x0000
- Bit Order = LSB first, “Use as input” box checked
- “Reverse byte order within word” box **checked**, Checksum unit size = 8-bit

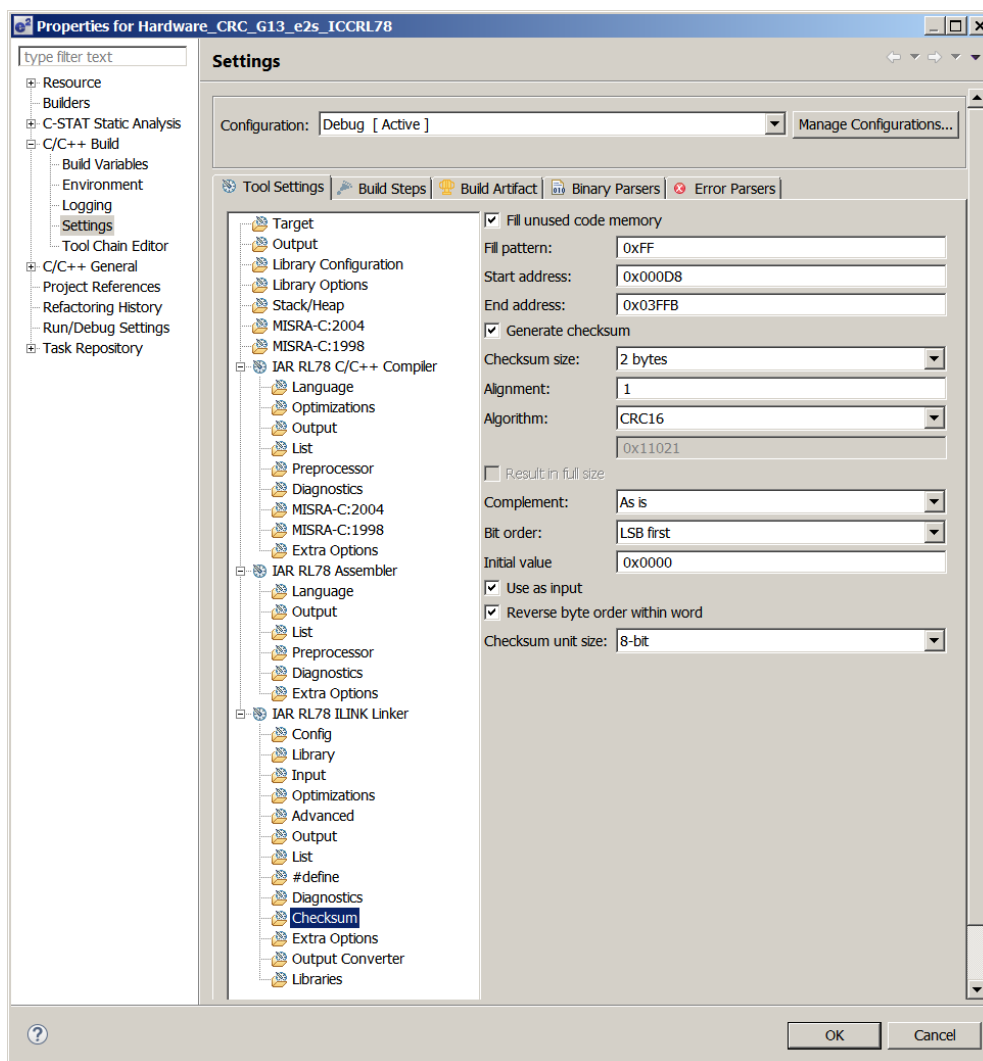


Figure 5: Linker CRC/Checksum Options for General Purpose CRC in e2studio/IAR settings

7.3 Settings for both High Speed CRC and General Purpose CRC use (e2studio/IAR)

- Override the default .icf (Linker) file in e2studio Project >> Properties >> Settings >> IAR RL78 ILINK Linker >> Config (See Figure 6.)
- Check the “Override default” box and use entry:
`${workspace_loc}/${ProjName}}\lnkr5f100le.icf` (See Figure 6.)
- Set the CRC/Checksum placeholder location, by entering “_checksum” in e2studio Project >> Properties >> Settings >> IAR RL78 ILINK Linker >> Input, “Keep symbols” box (See Figure 7.)
- Set an additional hex output file in e2studio Project >> Properties >> Settings >> IAR RL78 ILINK Linker >> Output Converter (See Figure 8.)
- Use the same directions as in section 5.3 for EWRL78, IAR compiler sample project to set the address location for linker CRC/Checksum result location in .icf file

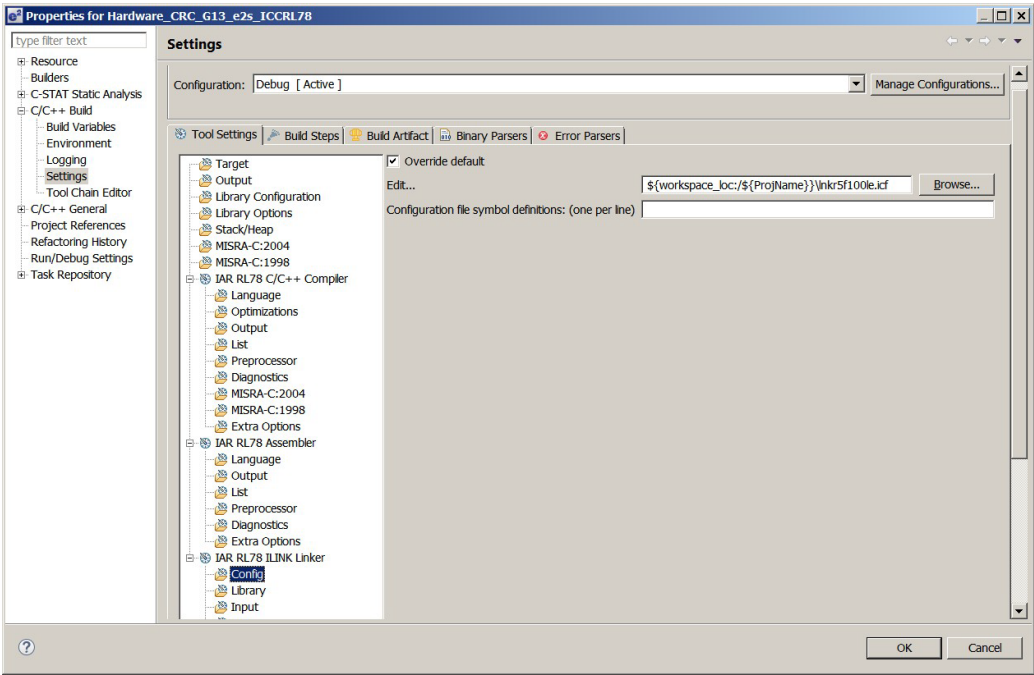


Figure 6: Linker Config Options in e2studio/IAR settings

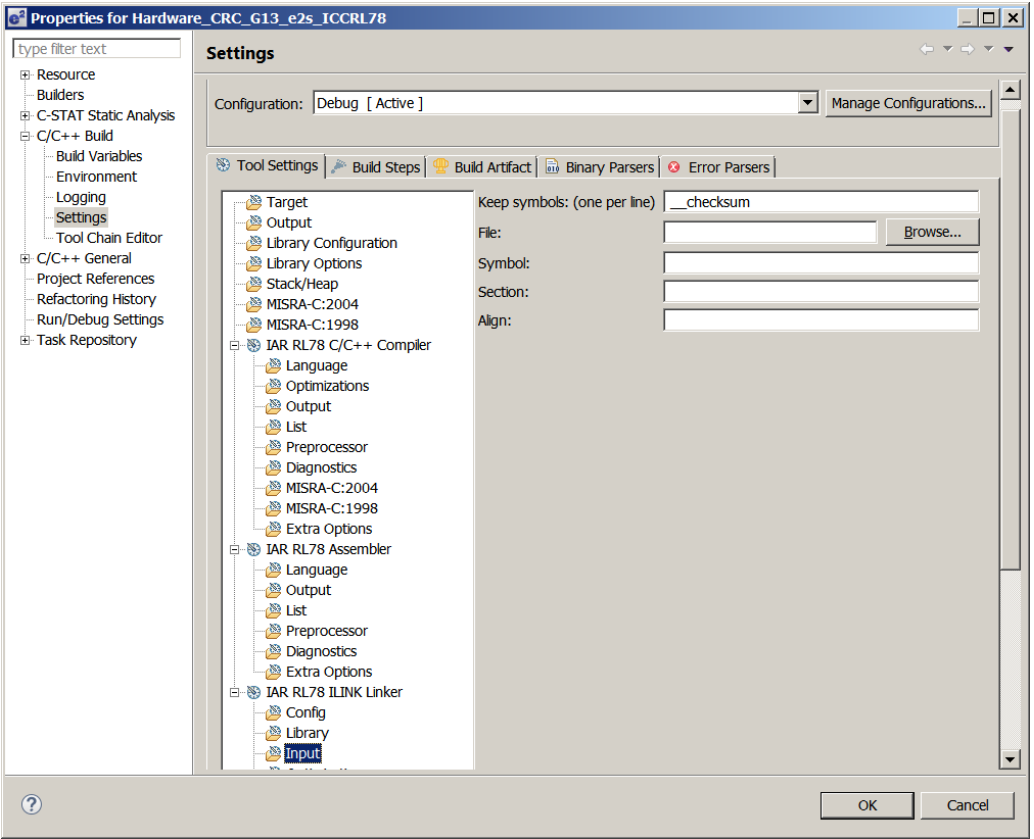


Figure 7: Linker Input Options in e2studio/IAR settings

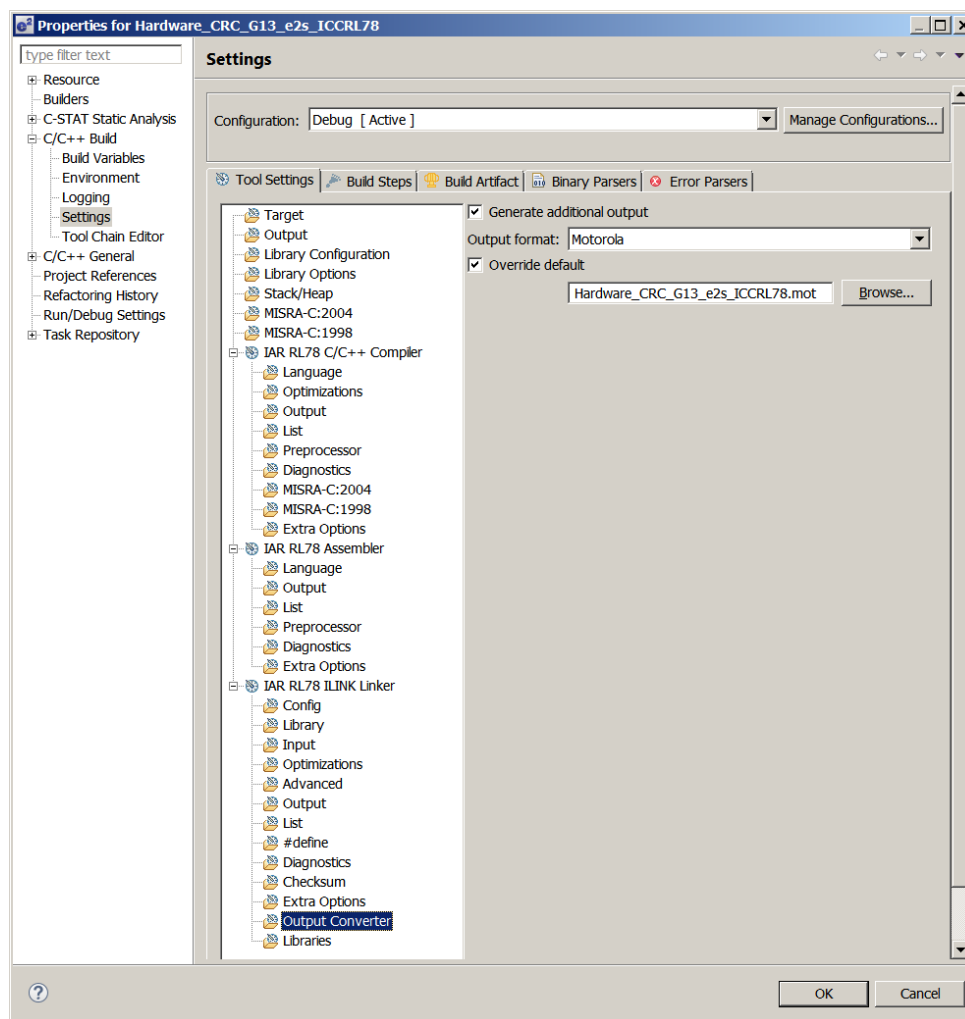


Figure 8: Linker Output Converter Options in e2studio/IAR settings

8. Using e2studio with CCRL compiler sample software project

8.1 Settings for both High Speed Hardware and General Purpose CRC (e2studio/CCRL)

Go to Project >> Properties >> Settings >> Converter >> Output (see Figure 9.)

- Check the “Output hex file” checkbox
- Set the Hex file format
- Set the Output file path
- Enter the Division output file (example: output .mot= 0000-FFFF)

Go to Project >> Properties >> Settings >> Converter >> Hex format (see Figure 10.)

- Set “Fill unused areas in the output ranges with the value:” to “Yes(Specification value)”
- Set “Output padding data” to “FF”.
- Check the “Output S9 record at the end” checkbox

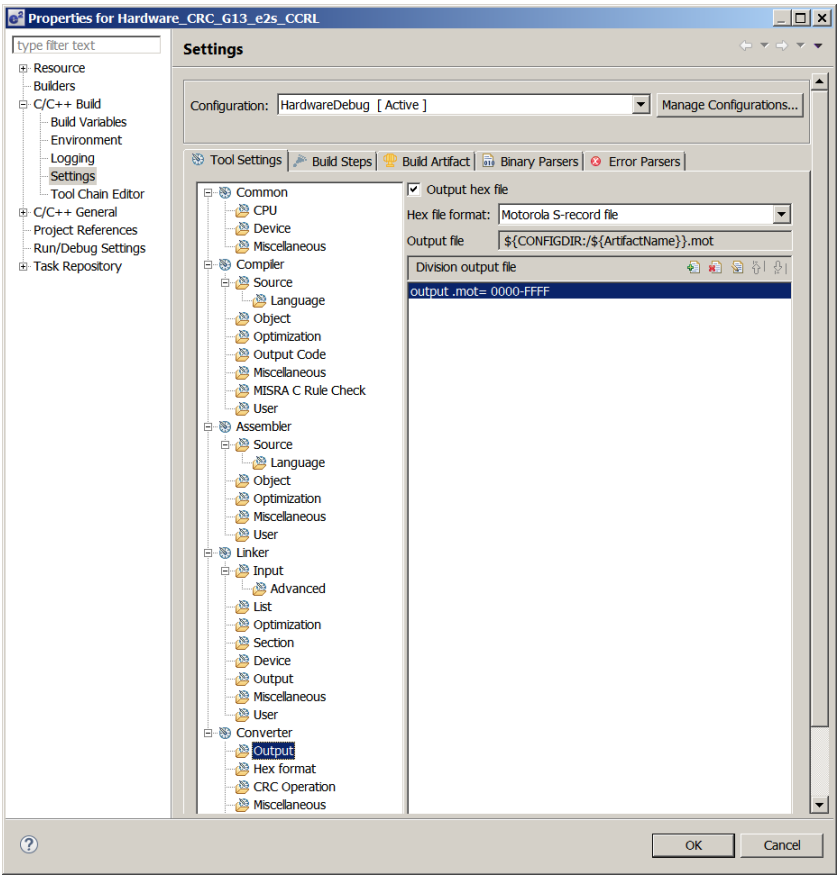


Figure 9: Converter Output Options in e2studio/CCRL settings

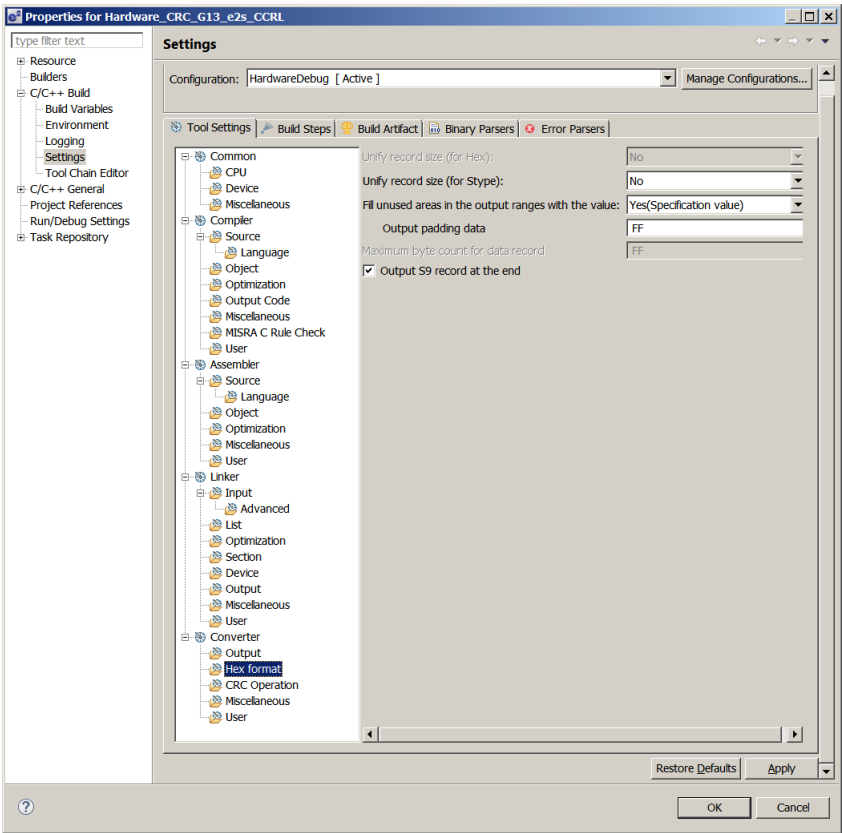


Figure 10: Converter Hex format Options in e2studio/CCRL settings

8.2 Converter CRC Operation settings for High Speed Hardware CRC (e2studio/CCRL)

Set e2studio Converter, CRC Operation settings in GUI - go to Project >> Properties >> Settings >> Converter >> CRC Operation (see Figure 11.)

- Check the “Outputs the calculation result of CRC” checkbox
- Set the “Output address” (compatible with High Speed CRC register CRC0CTL setting). Example: 03FFE for CRC0CTL, bits EA5-EA0 = 0x00 (High Speed CRC calculation on 0x00000 to 0x03FFB)
- Set the “Type of CRC:” to “CRC-CCITT(MSB,LITTLE,4 bytes) type”
- Set “Initial value:” to 0000
- Set “Endian and Output size:” to “LITTLE-2-0”
- Set Target range to “0000-3FFB (or other range to match CRC0CTL settings)”

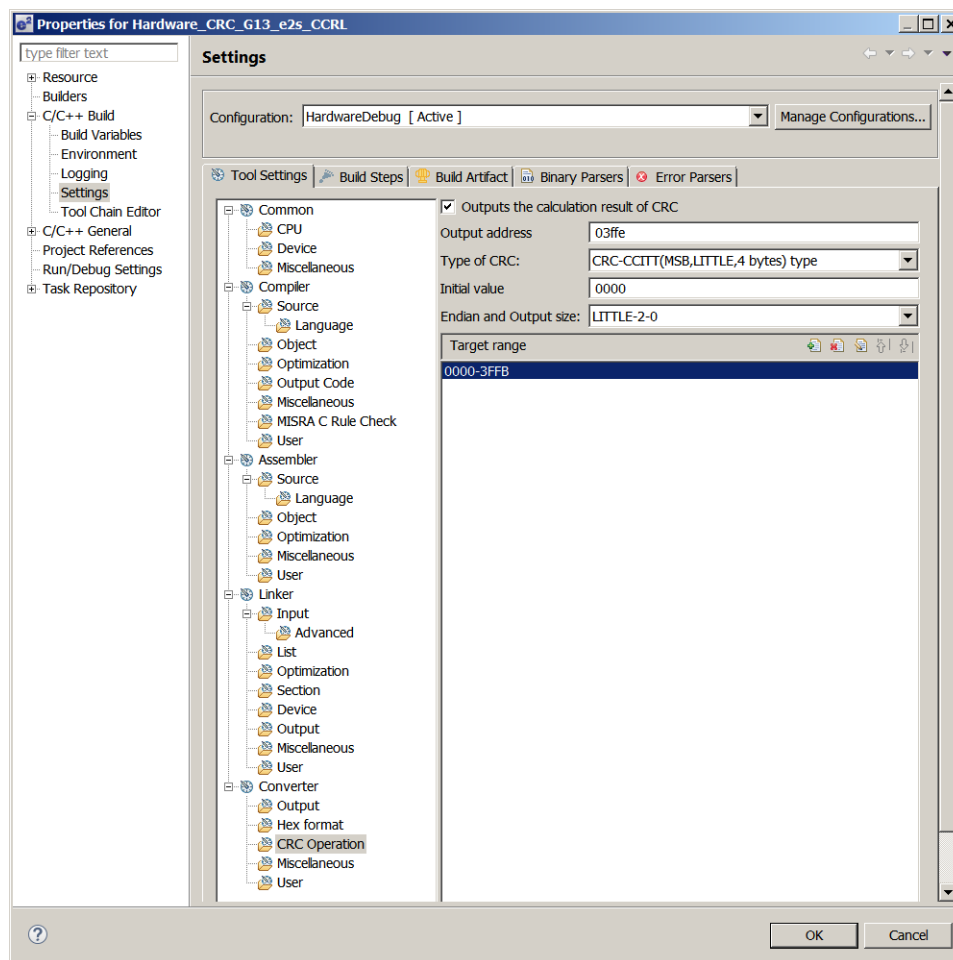


Figure 11: Converter CRC Operations Options in e2studio/CCRL settings for High Speed CRC

8.3 Converter CRC Operation settings for General Purpose Hardware CRC (e2studio/CCRL)

Set e2studio Converter, CRC Operation settings in GUI - go to Project >> Properties >> Settings >> Converter >> CRC Operation (see Figure 12.)

- Check the “Outputs the calculation result of CRC” checkbox
- Set the “Output address” to 3FFE
- Set the “Type of CRC” to “CRC-CCITT(LSB) type”

- Set “Initial value:” to 0000
- Set “Endian and Output size:” to “LITTLE-2-0”
- Set “Target range” to “00D8-3FFB (or other)”

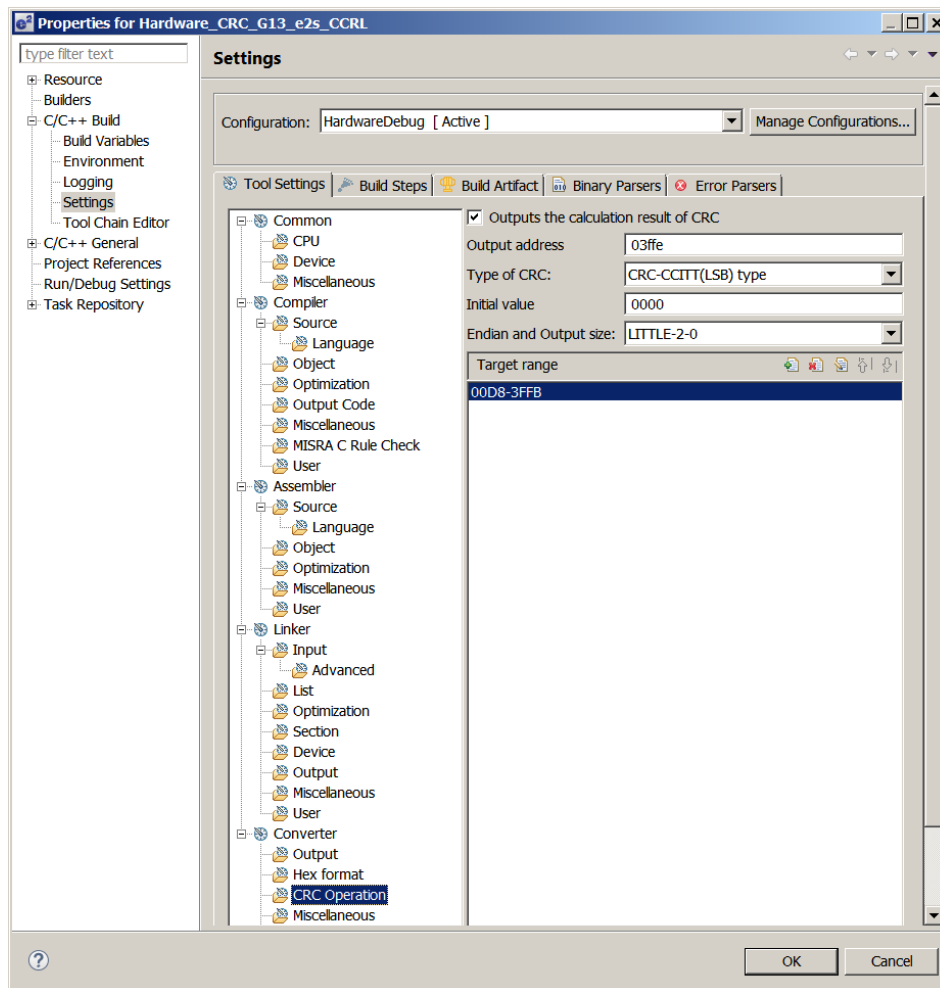


Figure 12: Converter CRC Operations Options in e2studio/CCRL settings for General Purpose CRC

9. Hardware platform used for RL78 CRC sample Software program

RSKRL78G13 board with R5F100LE (64pin RL78/G13 with 64KB code flash, 4KB RAM, 4KB Data Flash) is used for the application note sample Software testing.

The 4 LEDs on RSKRL78G13 board are used to indicate match or non-match of RL78 Hardware CRC versus the compiler/linker-generated CRC. After either the High Speed CRC or General Purpose CRC comparison is made, the match or non-match LED will light. Only one LED will activate on the RSKRL78/G13 board at a time after running the sample software CRC Tests:

LED3	LED2	LED1	LED0
			
(red)	(red)	(orange)	(green)
General Purpose CRC non- Match	High Speed CRC non- Match	General Purpose CRC Match	High Speed CRC Match

Figure 13: LED indicators RSKRL78G13 board

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jan 13, 2017	-	Initial Release
1.10	June 24, 2022	4	Update Software Environment

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/.