Replacement Guide for R8C Family to RL78 Family (CcnvNC30)

R01AN3508EC0100
Rev.1.00
Dec. 22, 2016

Introduction

This application note describes how to replace the programs for R8C with the programs for RL78.

Target Device

R8C Family

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. How to Replace Programs from R8C Family to RL78 Family

This section explains how to replace the programs for R8C family with the programs for RL78 family.

First, use the C source converter CcnvNC30 to convert the extended functions for the C compiler NC30 to the extended functions for the C compiler CC-RL.

Next, use the integrated development environment CS+ or e2studio to create a project. Because the R8C family and the RL78 family have different peripheral functions, use the code generator for the RL78 family to generate programs for peripheral functions of the RL78 family instead of using the programs for peripheral functions of the R8C family.

Combine the programs converted with CcnvNC30 and the above programs for peripheral functions to replace programs.

2. Program Conversion Using CcnvNC30

2.1 About CcnvNC30

CcnvNC30 converts extended language specifications (such as macro names, reserved words, #pragma directives, and extended functions) in C source programs for NC30 into extended language specifications for CC-RL.

CcnvNC30 is the software that supports the porting of the programs for NC30 to the programs for CC-RL. Since we do not guarantee the correct operation of the programs converted by CcnvNC30, be sure to check the operation of the program after conversion.

In addition, the device-dependent codes such as location addresses, access to an SFR, and assembly-language codes cannot be converted. Convert these codes manually into the code for the RL78 family as required.

For details, see "CcnvNC30 C Source Converter User's Manual (R20UT3685E)".

2.2 How to Use CcnvNC30

The method of converting a program with CcnvNC30 is shown below.

(1) Place CcnvNC30 (CcnvNC30.exe) and a program for NC30 in the same folder of your choice.
(2) Launch Command Prompt in Windows.
(3) Change the current directory to the folder where CcnvNC30 is stored.

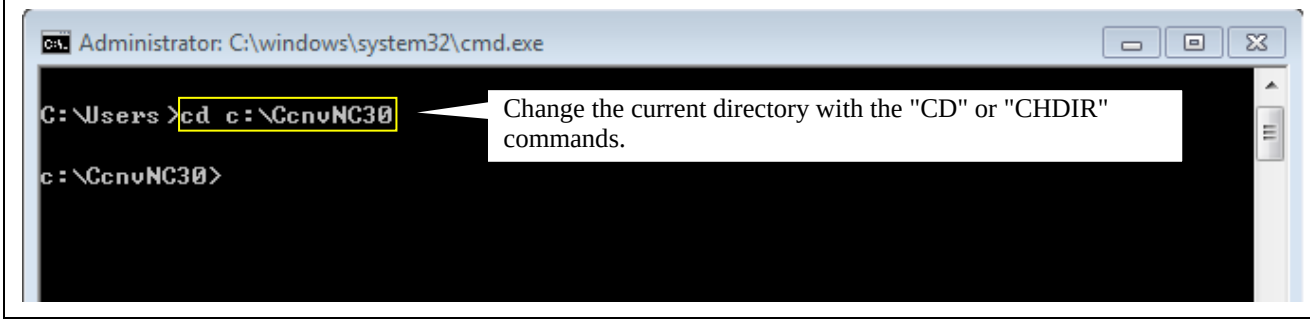

Figure 2.1 Command Prompt Window

(4) Specify an output file name with the -o option before execution. After the execution, a program for CC-RL is output. In addition, when outputting messages in a specified file, use the -r option.

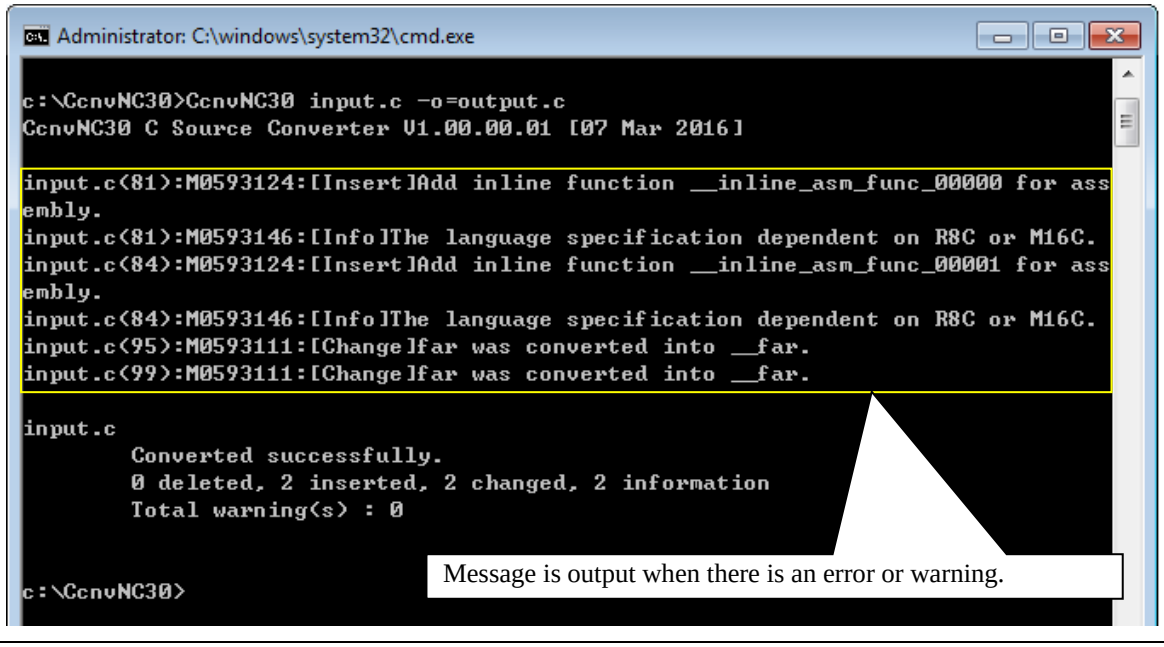

Figure 2.2 CcnvNC30 Execution Window

(5) When converting multiple files at the same time, create a list file and execute conversion with the -l option specified. After the execution, programs for CC-RL are output to the specified folder.

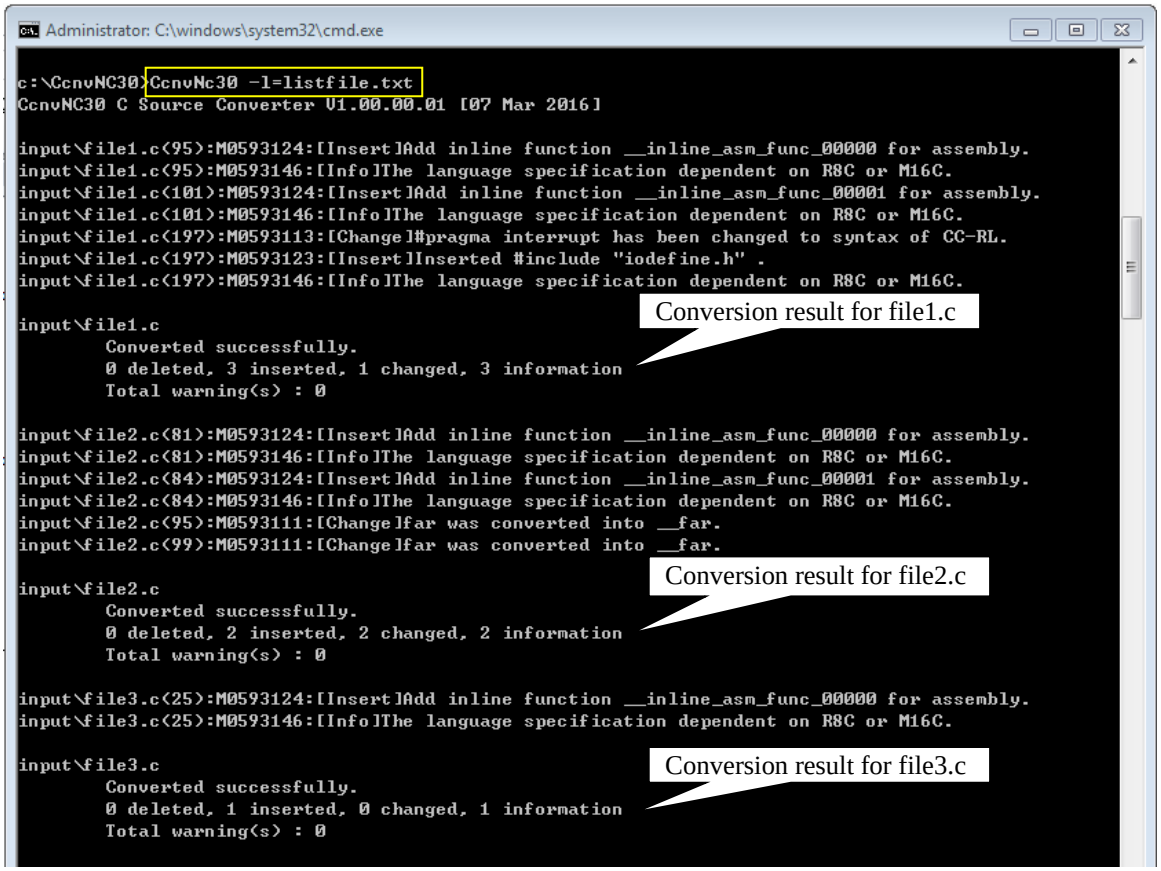

Figure 2.3 CcnvNC30 Execution Window (Multiple Files)

The example below shows the description in a list file.

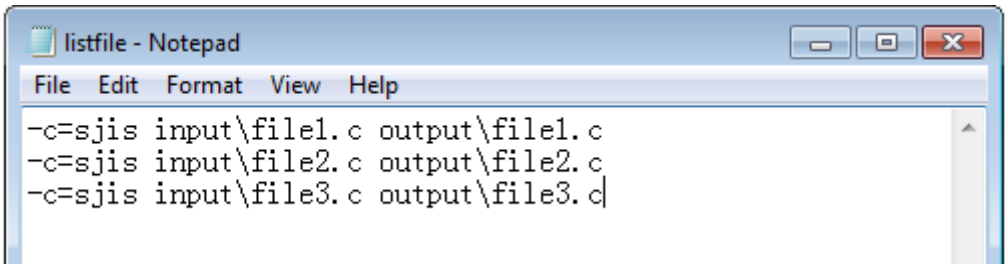

Figure 2.4 Example of Description in List File

(6) Correct the parts that are not converted by CcnvNC30. For the parts that require corrections, refer to "CONVERSION SPECIFICATIONS" of "CcnvNC30 C source converter User's Manual (R20UT3685E)".

2.3 When CcnvNC30 is Not Used

When CcnvNC30 is not used, extended functions of NC30 need to be converted manually into extended functions of CC-RL. For the extended language specifications supported by CC-RL, see "CC-RL Compiler User's Manual (R20UT3123E)".

3. Converting Programs for Peripheral Functions

3.1 Generating Programs Automatically

Programs are automatically generated for the RL78 family peripheral functions equivalent to the peripheral functions that were used by the R8C family by using the code generator for the RL78 family provided in the integrated development environment CS+ or e2studio. For how to use the code generator, see "CS+ Code Generator Integrated Development Environment User's Manual: Peripheral Function Operation [CS+ for CC][CS+ for CA,CX] (R20UT3104E)".

(1) Under [Project Tree], click [Clock Generator] in [Code Generator (Design Tool)] and perform “pin assignment”.When the pin assignment setting is decided once, it is not possible to be changed it later.

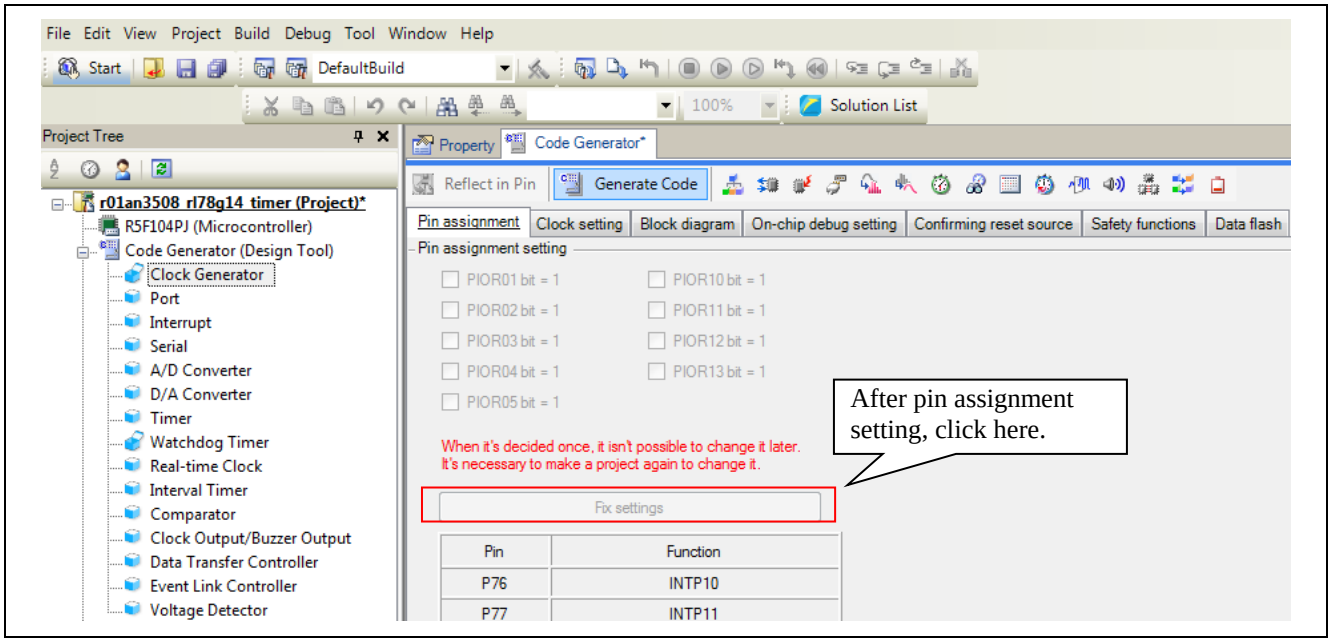

Figure 3.1 Code Generator Setting Window (1)

(2) Refer to the program for the R8C family and set each function.

(3) On completion of all the peripheral function settings, click the [Generate Code] button at the top of the window to generate codes (automatic program generation). Use the automatically generated functions for peripheral functions to replace programs.

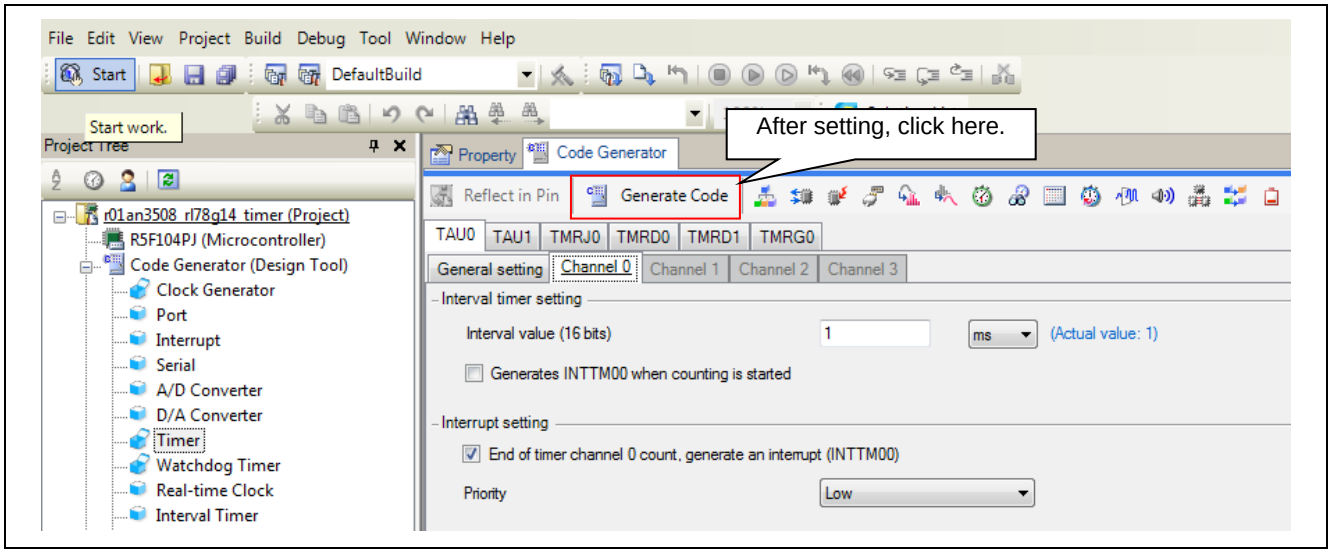

Figure 3.2 Code Generator Setting Window (2)

3.2 Adding Programs

Add the programs that cannot be automatically generated by the code generator (such as main function, interrupt function process, and variables).

Add a program between "/* Start user code for adding. Do not edit comment generated here */" and "/* End user code. Do not edit comment generated here */" in each file that was automatically generated. A program needs to be added manually. Note that any program added outside this range is automatically deleted during automatic generation of a program.

Be sure to confirm the operation of the system using the added programs.

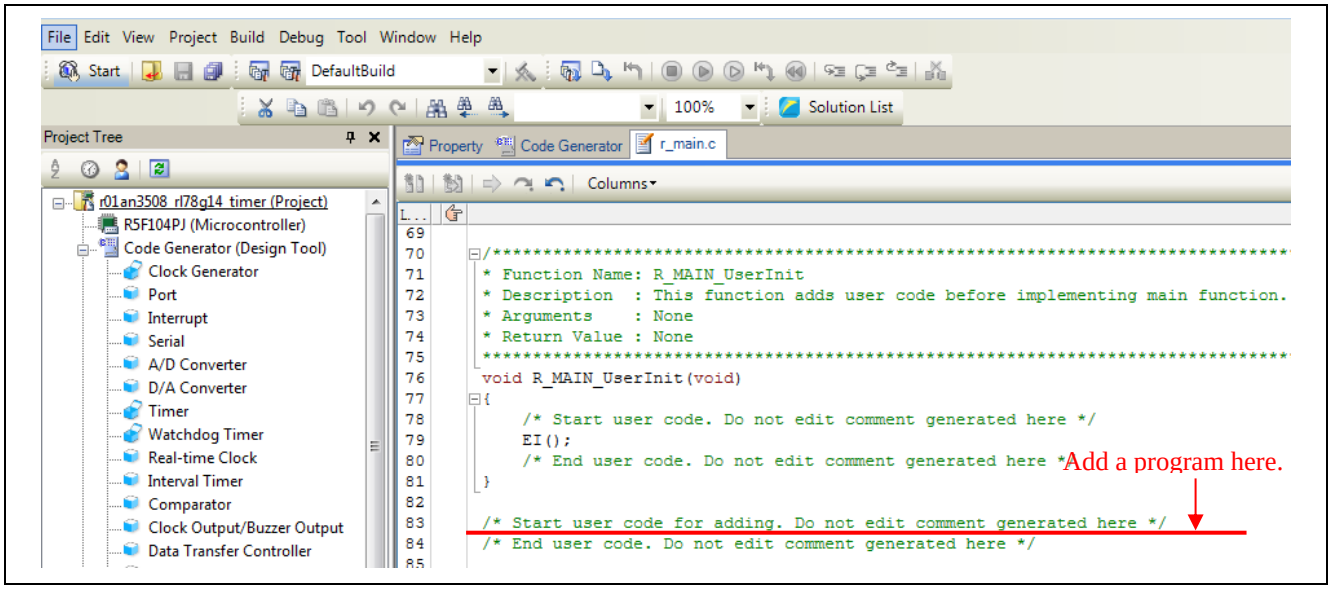

Figure 3.3 Adding Existing Program

3.3 When Code Generator is Not Used

When the code generator is not used, you need to create a new project first with the integrated development environment CS+ or e2studio and then manually create a program for a peripheral function. For details of peripheral functions, see the user's manual for the RL78 family.

4. Replacement Examples

4.1 R8C Sample Program (Clock Operation Using RTC)

The program of "R8C/35A Group Clock Operation Using RTC (R01AN0079E)" is replaced with the program for RL78/G14. The project file after replacement is "r01an3508_rl78g14_rtc".

This program uses timer RE (Real-Time Clock Mode) to operate a clock. Use the 20 MHz XIN clock for the CPU clock. Use fC4 (XCIN clock (32.768 kHz) divided by 4) for the timer RE count source.

4.1.1 Porting Source to CC-RL with CcnvNC30

(1) Create a list file to specify a C source file to be converted.

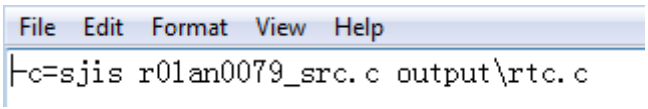

Figure 4.1 Example of Description in List File (Clock Operation Using RTC)

(2) Launch Command Prompt to convert the C source file specified with the list file. In addition, the output conversion result file indicates changes.

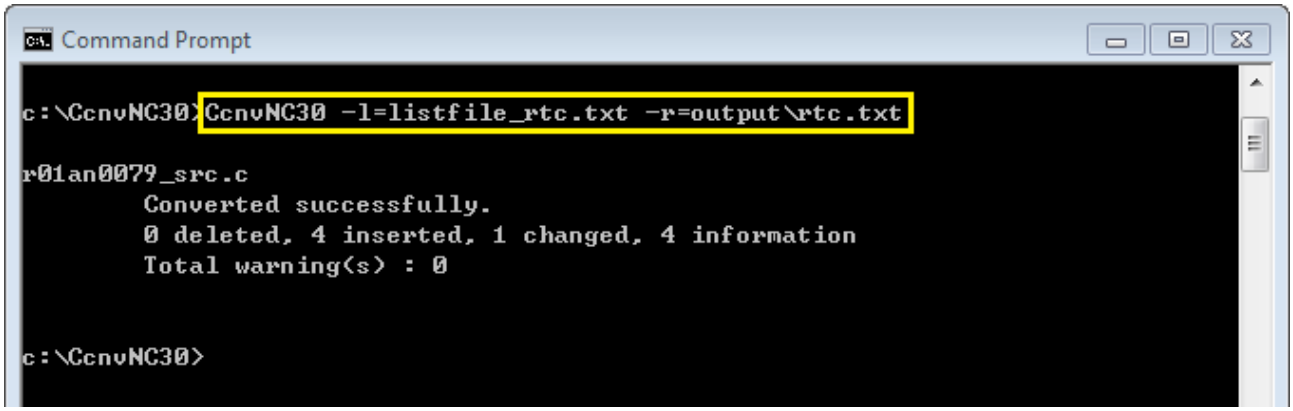

Figure 4.2 CcnvNC30 Execution Window (Clock Operation Using RTC)

The conversion result file indicates the conversion result as shown below. For details of the conversion result, see "CcnvNC30 C source converter User's Manual (R20UT3685E)".

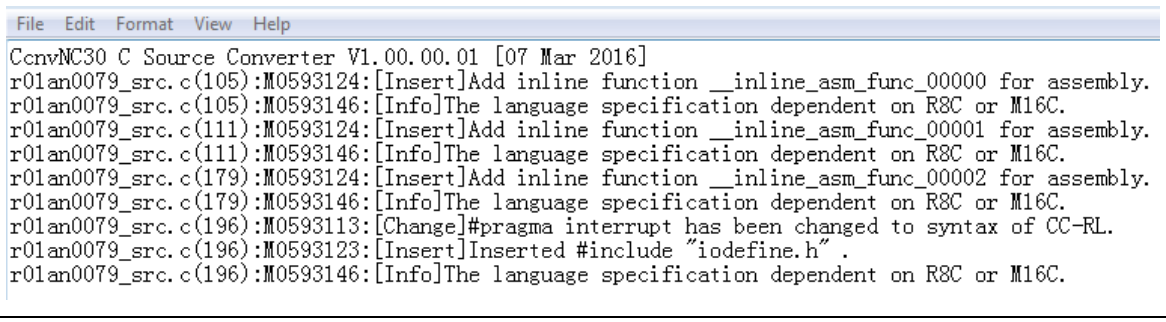

Figure 4.3 Details of Conversion Result (Clock Operation Using RTC)

(3) Correct the converted C source file.

There may be redundant interrupt function declarations. As CC-RL produces an error in this case, delete the converted #pragma directive.

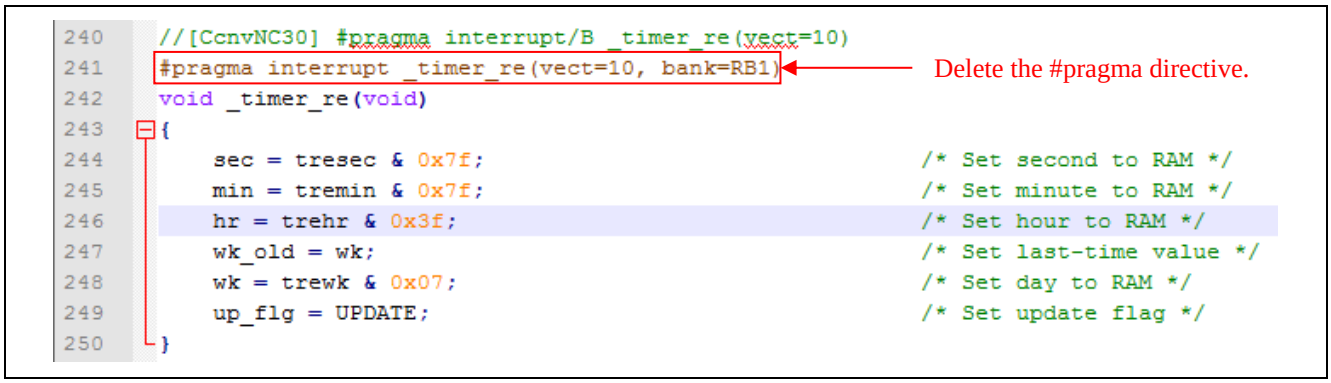

Figure 4.4 Changing Interrupt Function Declaration

4.1.2 Generating Programs Automatically

(1) Create a new project with the integrated development environment CS+ or e2studio.
(2) Set each function with the code generator.

Set the CPU clock to 16MHz high-speed OCO clock.
Set RTC and interval timer operation clock is 32.768kHz.

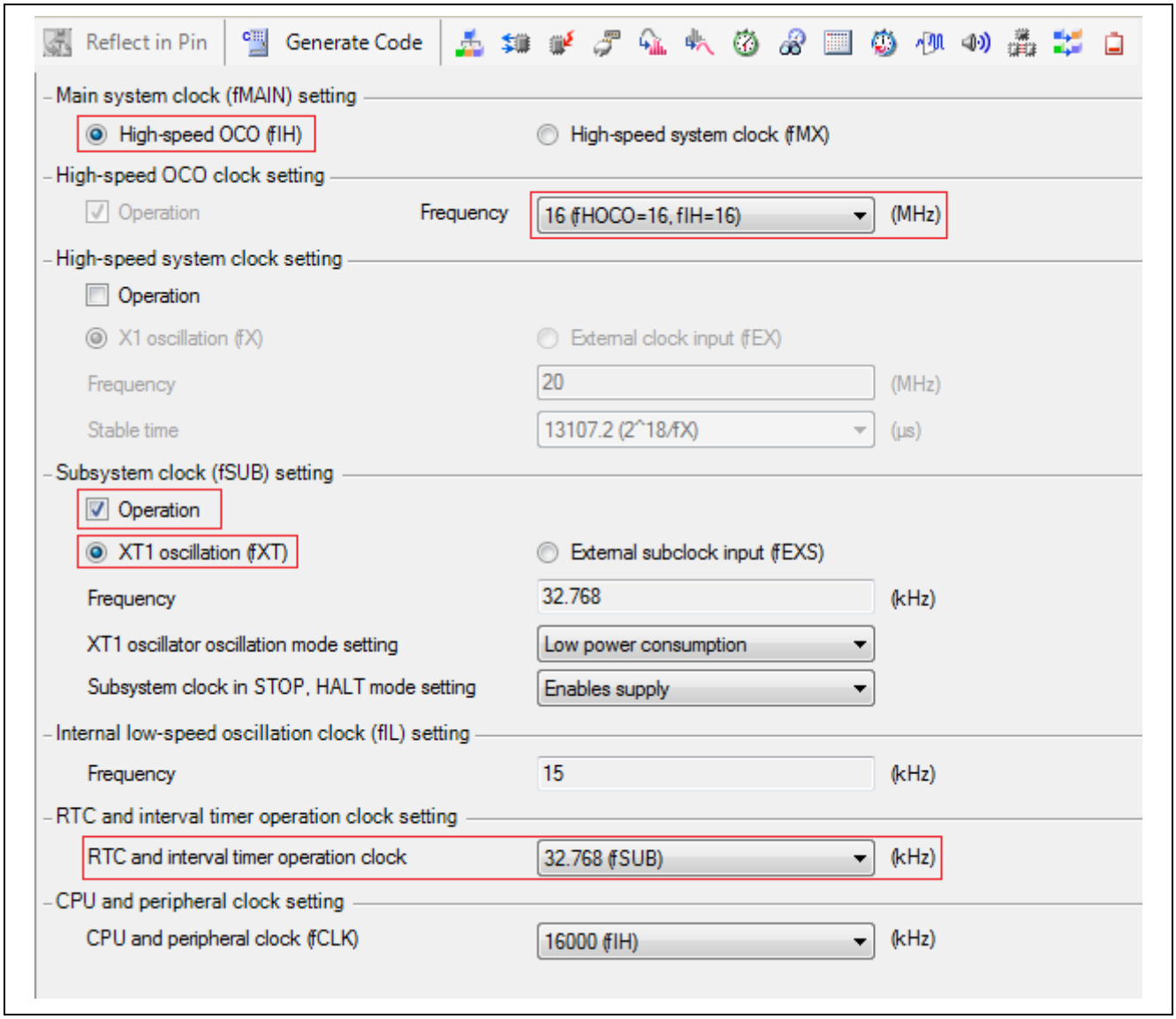

Figure 4.5 Setting Window in Code Generator (Clock)

Set Real-time Clock (RTC) which is the equivalent function to Timer RE of the R8C family.

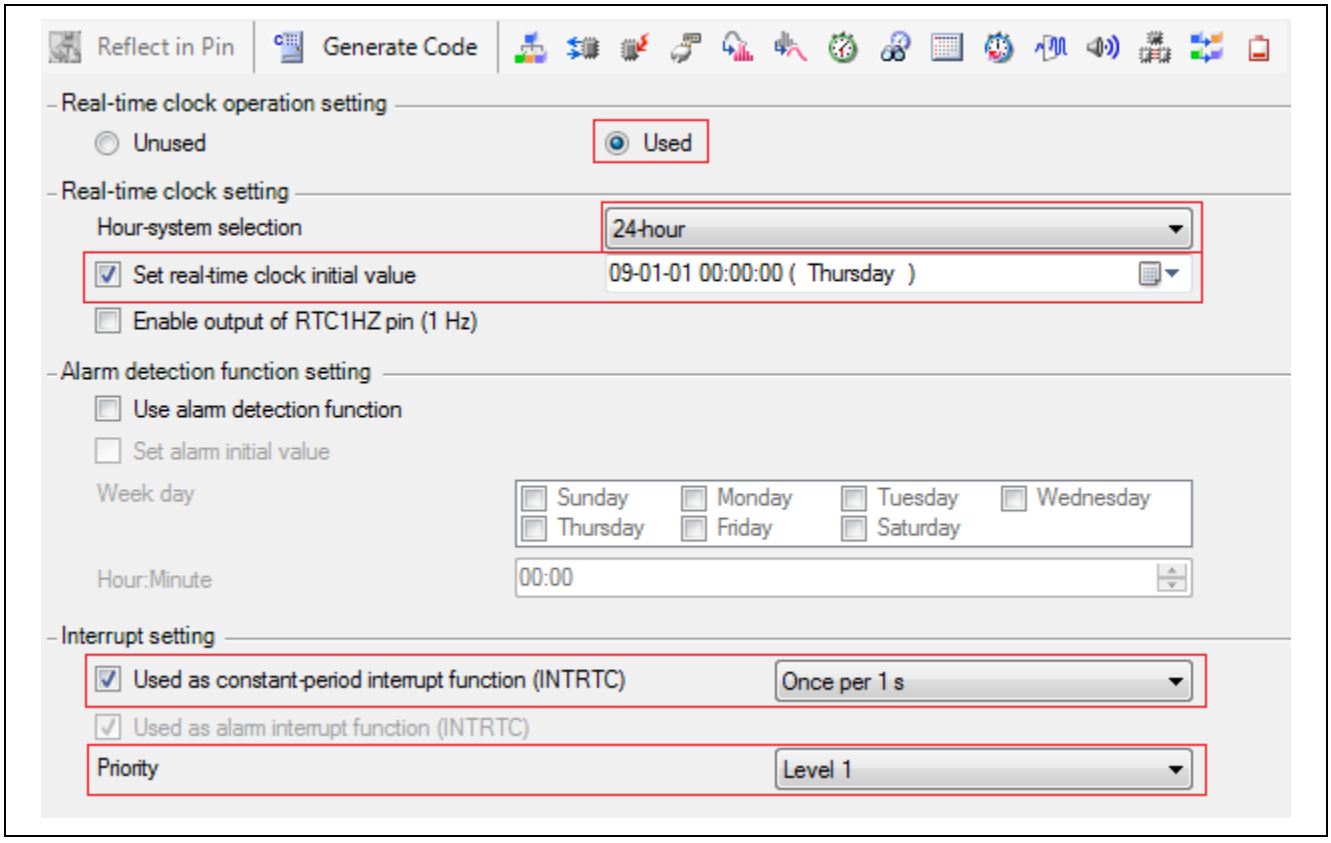

Figure 4.6 Setting Window in Code Generator (Real-time Clock)

(3) Set ports, watchdog timer, voltage detection circuit, etc. based on your environment.
(4) Click [Generate Code] to generate a file.

4.1.3 Adding Programs

Add the processes of symbol definition and the main function to the program with generated code. Use the programs with generated code for other programs (such as clock setting and RTC function setting).

- Symbol definition

Add symbol definition to r_cg_rtc.h.

Program for R8C

```
94  /************************************************************************************/
95  /* DEFINE                                                                           */
96  /************************************************************************************/
97  #define NO_UPDATE       0                       /* No update */
98  #define UPDATE          1                       /* Update */
99  #define COMM            0                       /* Common year */
100 #define LEAP            1                       /* Leap year */
101 #define DEC             0x12                    /* December */
102 #define WEEK            0x04                    /* A day of the week(thursday) */
```

Add contents corresponding to the red box manually.

r_cg_rtc.h file for RL78/G14

```
146 /* Start user code for function. Do not edit comment generated here */
147 #define NO_UPDATE       0                       /* No update */
148 #define UPDATE          1                       /* Update */
149 #define COMM            0                       /* Common year */
150 #define LEAP            1                       /* Leap year */
151 #define DEC             0x12                    /* December */
```

Figure 4.7 Replacement of Symbol Definition

- main function

When the code generator for RL78/G14 is used, the R_Systeminit function is executed before the main function is executed. The R_Systeminit function performs the initial setting of the clock and RTC. Thus, only the process indicated in the red box is added manually. The R_RTC_Start function starts the operation of RTC.

In R8C/35A, customers can use Timer RE for real-time clock. In RL78/G14, customer can use RTC for real-time clock. In the main function of "R8C/35A Group Clock Operation Using RTC (R01AN0079E)", the initialization function and the start command of Timer RE are executed. So in main function of "r01an3508_rl78g14_rtc", the initialization function (RTC generate the interrupt per one second) and the start function of RTC are executed.

Program for R8C

```
103  void main(void)
104  {
105      asm("FCLR I");                                  /* Interrupt disabled */
106
107      mcu_init();                                     /* MCU initialize */
108
109      timer_re_init();                                /* TimerRE initialize */
110
111      asm("FSET I");                                  /* Interrupt enabled */
112
113      leap_flg = leap_chk();                          /* Leap year check */
114
115      tstart_trecr1 = 1;                              /* TRE1 count start */
116      while(tcstf_trecr1== 0);                        /* If TRE1 count start? else wait */
117
118      while(1){
119          if(up_flg == UPDATE){
120              update();                               /* Update processing */
121          }
122      }
123  }
```

Add contents corresponding to the red box manually.

r_main.c file for RL78/G14

```
61   void main(void)
62   {
63       R_MAIN_UserInit();
64       /* Start user code. Do not edit comment generated here */
65       leap_flg = leap_chk();                          /* Leap year check */
66
67       R_RTC_Set_ConstPeriodInterruptOn(SEC);          /* Interrupt turned on */
68       R_RTC_Start();                                  /* Enabled */
69
70       while (1U)
71       {
72           if(up_flg == UPDATE){
73               update();                               /* Update processing */
74           }
75       }
76       /* End user code. Do not edit comment generated here */
77   }
```

Figure 4.8 Replacement of main Function

Additionally, "update" and "leap_chk" functions, which are used by "main" function, should be added into r_main.c file. And the variables that are used by these two functions are also needs to be added in r_main.c file.

Variables for R8C

```
63  /***********************************************************************/
64  /* RAM                                                                 */
65  /***********************************************************************/
66  unsigned short year = 0x2009;                   /* Year */
67  unsigned char month = 0x01;                     /* Month */
68  unsigned char day = 0x01;                       /* Day */
69  unsigned char wk = WEEK;                        /* Sun,Mon,Tue,Wed,Thu,Fri,Sat */
70  unsigned char hr = 0x00;                        /* Hour */
71  unsigned char min = 0x00;                       /* Minute */
72  unsigned char sec = 0x00;                       /* Second */
73
74  /* Work area */
75  unsigned char wk_old = 0x00;                    /* Last-time value of 'wk' */
76
77  /* Flags */
78  unsigned char up_flg = NO_UPDATE;               /* Update flag */
79  unsigned char leap_flg = COMM;                  /* Leap flag */
80
81  /***********************************************************************/
82  /* ROM                                                                 */
83  /***********************************************************************/
84  unsigned char const MONTH_DAYS[13] = {          /* This table contains the number of days in different months */
85      0x00,0x31,0x28,0x31,0x30,0x31,0x30,0x31,0x31,0x30,0x31,0x30,0x31
86  };
87
88  unsigned char const MONTH_DAYS_L[13] = {        /* This table contains the number of days in different months (leap year) */
89      0x00,0x31,0x29,0x31,0x30,0x31,0x30,0x31,0x31,0x30,0x31,0x30,0x31
90  };
```

Add contents corresponding to the red box manually.

r_main.c file for RL78/G14

```
46  /***********************************************************************************************************************
47  Global variables and functions
48  ***********************************************************************************************************************/
49  /* Start user code for global. Do not edit comment generated here */
50  unsigned short year = 0x2009;                   /* Year */
51  unsigned char month = 0x01;                     /* Month */
52  unsigned char day = 0x01;                       /* Day */
53  unsigned char wk = 0x04;                        /* Sun,Mon,Tue,Wed,Thu,Fri,Sat */
54  unsigned char hr = 0x00;                        /* Hour */
55  unsigned char min = 0x00;                       /* Minute */
56  unsigned char sec = 0x00;                       /* Second */
57
58  unsigned char wk_old = 0x00;                    /* Last-time value of 'wk' */
59
60  unsigned char up_flg = NO_UPDATE;               /* Update flag */
61  unsigned char leap_flg = COMM;                  /* Leap flag */
62
63  unsigned char const MONTH_DAYS[13] = {          /* This table contains the number of days in different months */
64      0x00,0x31,0x28,0x31,0x30,0x31,0x30,0x31,0x31,0x30,0x31,0x30,0x31
65  };
66
67  unsigned char const MONTH_DAYS_L[13] = {        /* This table contains the number of days in different months (leap year) */
68      0x00,0x31,0x29,0x31,0x30,0x31,0x30,0x31,0x31,0x30,0x31,0x30,0x31
69  };
```

Figure 4.9 Add Variables

Program for R8C

```
/*""FUNC COMMENT""************************************************
* Outline                  : Update processing
* Declaration              : void update(void)
* Description              : Update check
* Argument                 : none
* Return Value             : none
*""FUNC COMMENT END""*******************************************/
void update(void)
{
    unsigned char   num = 0;

    up_flg = NO_UPDATE;                                 /* Update flag clear */

    if(wk != wk_old){                                   /* Change day ? */
}

/*""FUNC COMMENT""************************************************
* Outline                  : Leap year check processing
* Declaration              : unsigned char leap_chk(void)
* Description              : Leap year check
* Argument                 : none
* Return Value             : none
*""FUNC COMMENT END""*******************************************/
unsigned char leap_chk(void)
{
    unsigned char   chk = 0;                            /* Result of check */
    unsigned short  dec = 0;
    unsigned short  work = 0;

    work = year;                                        /* Copy year */

    dec  = (((work & 0xF000) >> 12) * 1000);            /* Conversion to decimal */
    dec += (((work & 0x0F00) >>  8) *  100);
    dec += (((work & 0x00F0) >>  4) *   10);
    dec += (((work & 0x000F) >>  0) *    1);

    chk = COMM;                                         /* Set Common year */

    if((dec % 4) == 0){
    return(chk);
}
```

Add contents corresponding to the red box manually.

r_main.c file for RL78/G14

```
/* Start user code for adding. Do not edit comment generated here */
/***********************************************************************************************************************
* Function Name: update
* Description  : Update check
* Arguments    : None
* Return Value : None
***********************************************************************************************************************/
void update(void)
{ ... }

/***********************************************************************************************************************
* Function Name: leap_chk
* Description  : Leap year check
* Arguments    : None
* Return Value : None
***********************************************************************************************************************/
unsigned char leap_chk(void)
{ ... }
/* End user code. Do not edit comment generated here */
```

Figure 4.10 Add Functions

4.1.4 Other Items to be Corrected

If RTC is set with the code generator, interrupt processes are automatically generated. Add the user program to interrupt routine.

Program for R8C

```
/*""FUNC COMMENT""************************************************
* Outline                 : Timer RE interrupt processing
* Declaration             : void _timer_re(void)
* Description             : Timer RE interrupt
* Argument                : none
* Return Value            : none
*""FUNC COMMENT END""*******************************************/
#pragma interrupt/B _timer_re(vect=10)
void _timer_re(void)
{
    sec = tresec & 0x7f;                                /* Set second to RAM */
    min = tremin & 0x7f;                                /* Set minute to RAM */
    hr = trehr & 0x3f;                                  /* Set hour to RAM */
    wk_old = wk;                                        /* Set last-time value */
    wk = trewk & 0x07;                                  /* Set day to RAM */
    up_flg = UPDATE;                                    /* Set update flag */
}
```

Add contents corresponding to the red box manually.

r_cg_rtc_user.c file for RL78/G14

```
/***********************************************************************************************************************
* Function Name: r_rtc_interrupt
* Description  : This function is INTRTC interrupt service routine.
* Arguments    : None
* Return Value : None
***********************************************************************************************************************/
static void __near r_rtc_interrupt(void)
{
    if (1U == RIFG)
    {
        RTCC1 &= (uint8_t)~_08_RTC_INTC_GENERATE_FLAG;    /* clear RIFG */
        r_rtc_callback_constperiod();
    }
}

/***********************************************************************************************************************
* Function Name: r_rtc_callback_constperiod
* Description  : This function is real-time clock constant-period interrupt service handler.
* Arguments    : None
* Return Value : None
***********************************************************************************************************************/
static void r_rtc_callback_constperiod(void)
{
    /* Start user code. Do not edit comment generated here */
    sec = SEC & 0x7f;                                   /* Set second to RAM */
    min = MIN & 0x7f;                                   /* Set minute to RAM */
    hr = HOUR & 0x3f;                                   /* Set hour to RAM */
    wk_old = wk;                                        /* Set last-time value */
    wk = WEEK & 0x07;                                   /* Set day to RAM */
    up_flg = UPDATE;                                    /* Set update flag */
    /* End user code. Do not edit comment generated here */
}
```

Figure 4.11 Add Interrupt Program

Additionally, these variables need external declaration in r_cg_rtc_user.c file.

```
/* Start user code for global. Do not edit comment generated here */
extern unsigned short year;                             /* Year */
extern unsigned char month;                             /* Month */
extern unsigned char day;                               /* Day */
extern unsigned char wk;                                /* Sun,Mon,Tue,Wed,Thu,Fri,Sat */
extern unsigned char hr;                                /* Hour */
extern unsigned char min;                               /* Minute */
extern unsigned char sec;                               /* Second */

extern unsigned char wk_old;                            /* Last-time value of 'wk' */

extern unsigned char up_flg;
/* End user code. Do not edit comment generated here */
```

Figure 4.12 Add External Declaration

4.1.5 Sample Code After Replacement

Obtain the sample code "an-r01an3508ec0100-rl78-migrate.zip" from the Renesas Electronics Website. "rl78g14_migrate_rtc" in the "workspace" folder is the sample code that replaces the program of "R8C/35A Group Clock Operation Using RTC (R01AN0079E)".

4.2 R8C Sample Program (DTC Operation)

The program of "R8C/35C Group DTC Operation in Chain Transfers (R01AN0372E)" is replaced with the program for RL78/G14. The project file after replacement is "r01an3508_rl78g14_dtc".

This program uses DTC operation in chain transfers.

4.2.1 Porting Source to CC-RL with CcnvNC30

(1) Create a list file to specify a C source file to be converted.

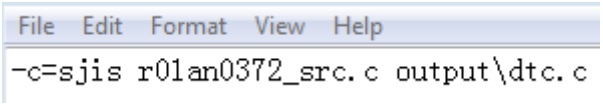

Figure 4.13 Example of Description in List File (DTC Operation in Chain Transfers)

(2) Launch Command Prompt to convert the C source file specified with the list file. In addition, the output conversion result file indicates changes.

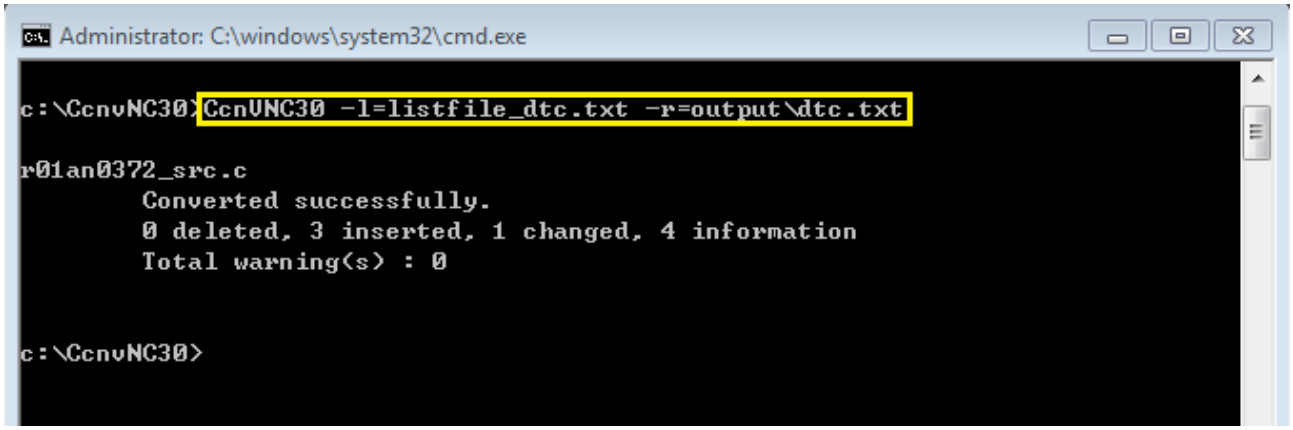

Figure 4.14 CcnvNC30 Execution Window (DTC Operation in Chain Transfers)

The conversion result file indicates the conversion result as shown below. For details of the conversion result, see "CcnvNC30 C source converter User's Manual (R20UT3685E)".

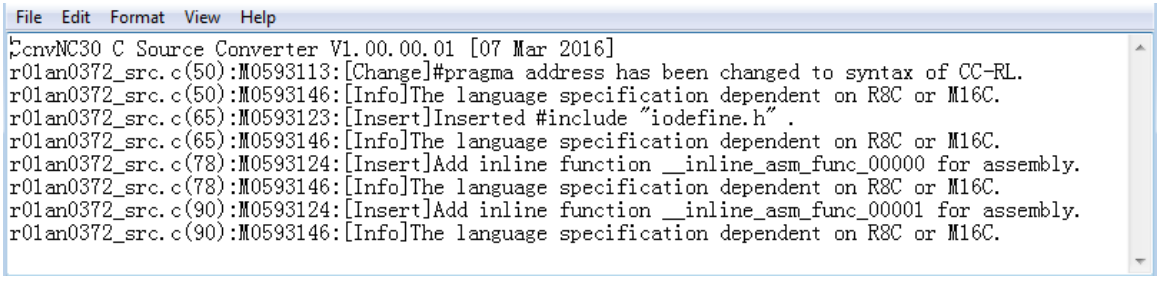

Figure 4.15 Details of Conversion Result (DTC Operation in Chain Transfers)

(3) Correct the converted C source file.

There may be redundant interrupt function declarations. As CC-RL produces an error in this case, delete the converted #pragma directive.

```
86  //[CcnvNC30] #pragma ADDRESS ad_value    00600H      /* Destination address of A/D conversion data */
87  #pragma address ad_value=0x00600
```

Delete the #pragma directive.

Figure 4.16 Changing Interrupt Function Declaration

4.2.2 Generating Programs Automatically

(1) Create a new project with the integrated development environment CS+ or e2studio.
(2) Set each function with the code generator.
Set the CPU clock to 16MHz high-speed OCO clock.

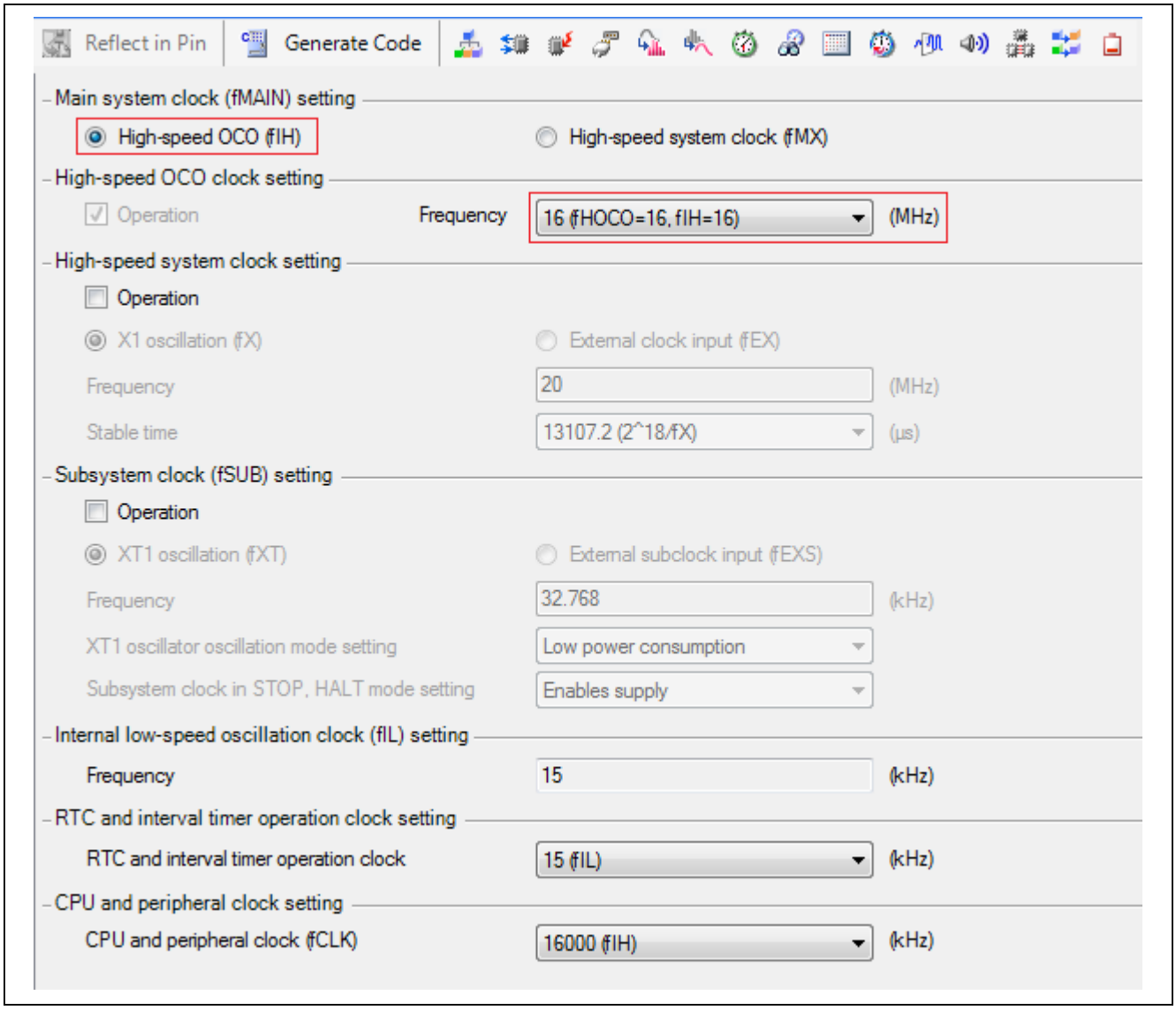

Figure 4.17 Setting Window in Code Generator (Clock)

Set A/D Converter (ADC) which is the equivalent function to ADC of the R8C family.
Use A/D converter in software trigger mode, scan mode, and one-shot conversion mode.

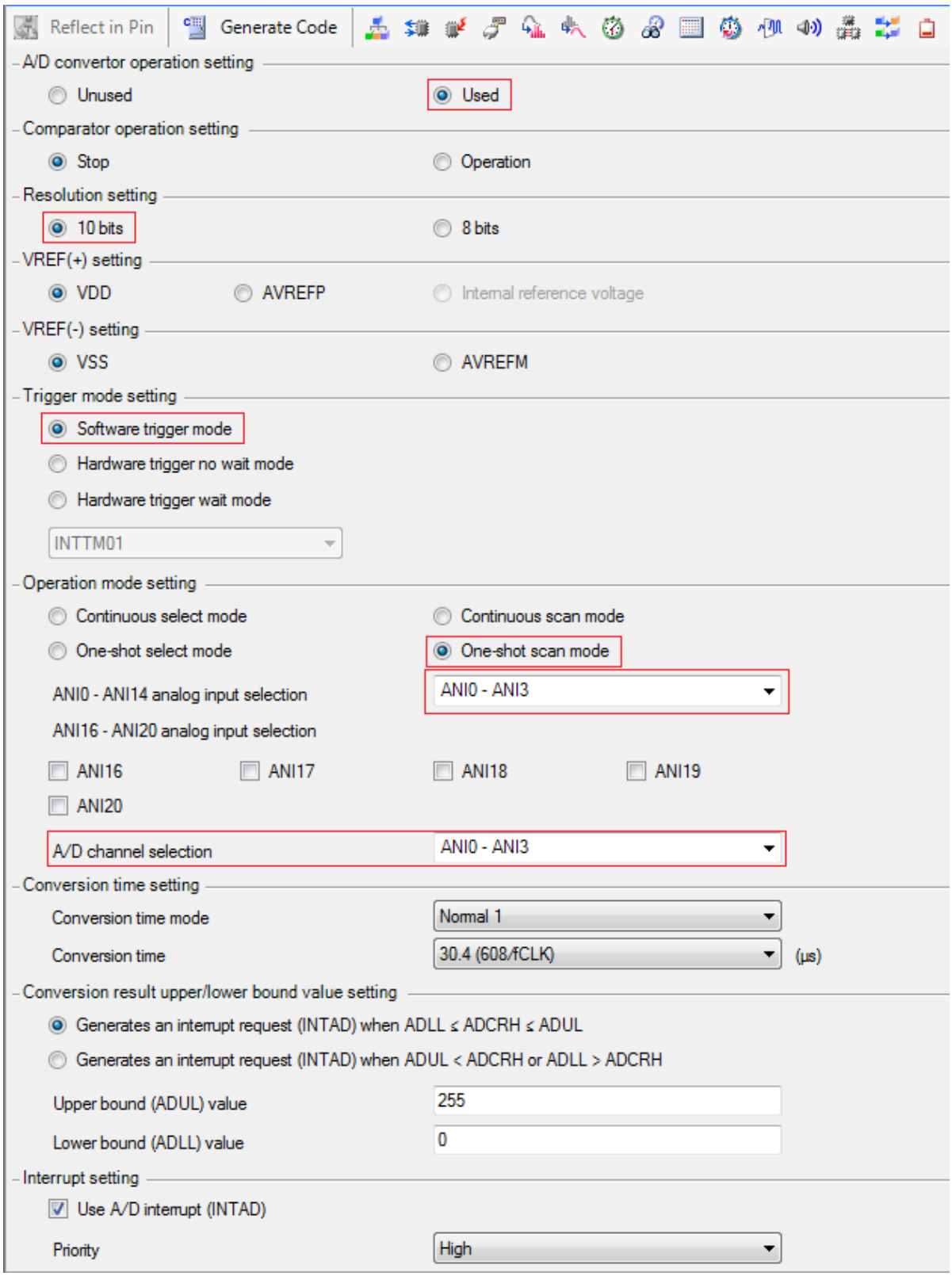

Figure 4.18 Setting Window in Code Generator (ADC)

Set Data Transfer Controller (DTC) which is the equivalent function to DTC of the R8C family.

In the sample program of "R8C/35C Group DTC Operation in Chain Transfers (R01AN0372E)", A/D converted values that have been stored to A/D registers (AD0 to AD3) are transferred to the internal RAM using the DTC chain transfer. But, in RL78/G14 Group, there is only one A/D conversion result register, so use DTC (normal mode, not use chain transfer). Perform A/D conversion on analog input voltage input to pins ANI0 to ANI3 while in scan mode, and use DTC transfer to store the A/D converted value to the RAM. Perform A/D conversion for individual pins successively. Every time A/D conversion for a pin is completed, store the converted result to the 10-bit A/D conversion result register (ADCR), activate the DTC, and transfer the A/D converted result from the ADCR register to the RAM. When A/D conversion and DTC transfer for all of the above pins are completed, an A/D conversion end interrupt request is generated.

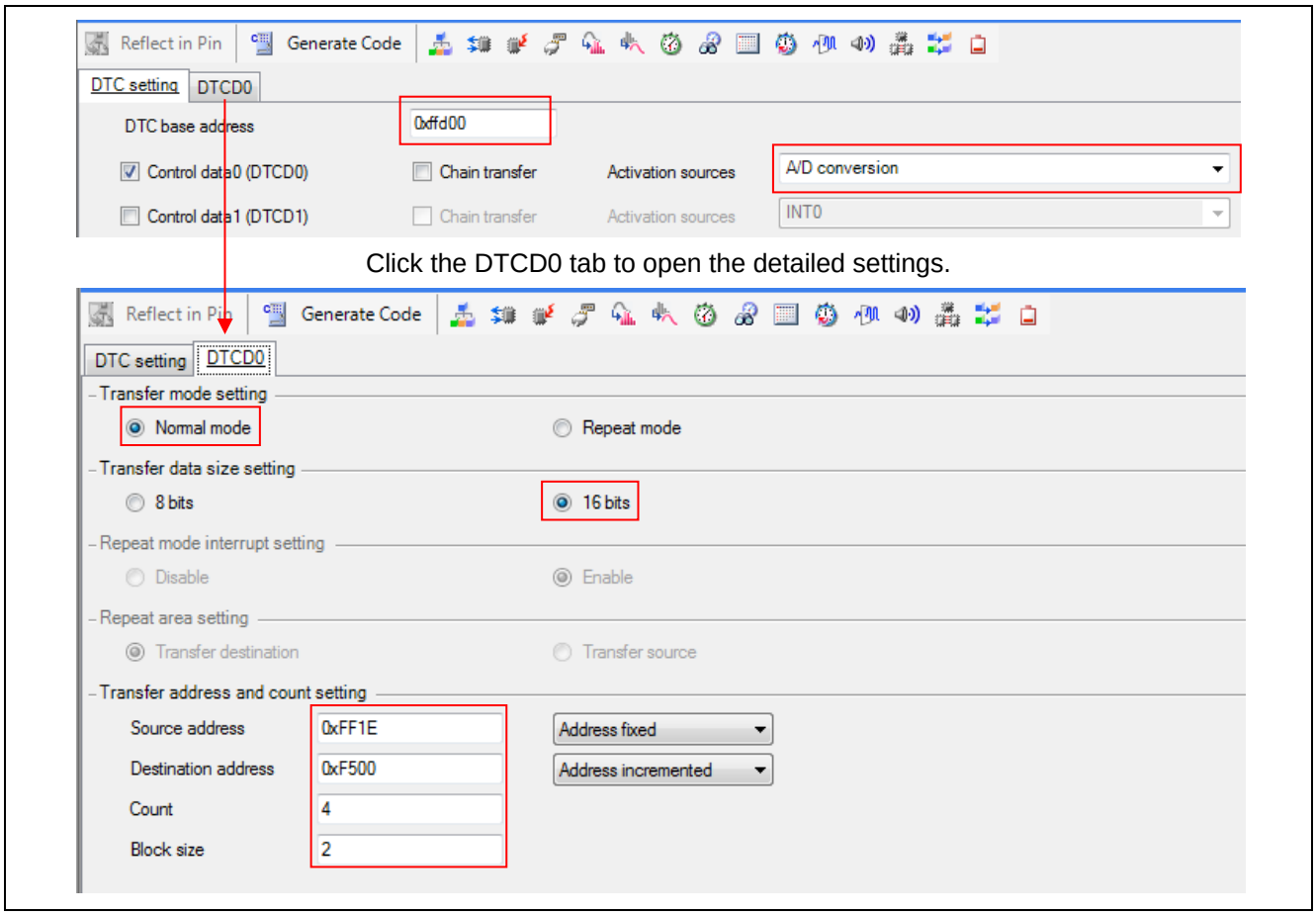

Figure 4.19 Setting Window in Code Generator (DTC)

(3) Set ports, watchdog timer, voltage detection circuit, etc. based on your environment.
(4) Click [Generate Code] to generate a file.

4.2.3 Adding Programs

Add the processes of symbol definition and the main function to the program with generated code. Use the programs with generated code for other programs (such as clock setting, ADC function and DTC function setting).

- main function

When the code generator for RL78/G14 is used, the R_Systeminit function is executed before the main function is executed. The R_Systeminit function performs the initial setting of the clock, ADC and DTC. Thus, only the process indicated in the red box is added manually. The R_DTCD0_Start function starts the operation of DTC. The R_ADC_Start function starts the operation of ADC.

Program for R8C

```
75  void main(void)
76  {
77      /* ==== Interrupt disabled ==== */
78      asm("FCLR I");
79
80      /* ==== Set High-speed on-chip oscillator clock to System clock ==== */
81      mcu_init();
82
83      /* ==== A/D converter initialize ==== */
84      ad_converter_enable();
85
86      /* ==== DTC initialize ==== */
87      dtc_enable();
88
89      /* ==== Interrupt enabled ==== */
90      asm("FSET I");
91
92      dtcen16 = 1;                        /* Activation of A/D interruption enable */
93
94      adst = 1;                           /* A/D conversion starts */
95
96      /* ==== Main loop ==== */
97      while(1){
98      }
99  }
```

Add contents corresponding to the red box manually.

r_main.c file for RL78/G14

```
59  void main(void)
60  {
61      R_MAIN_UserInit();
62      /* Start user code. Do not edit comment generated here */
63      R_DTCD0_Start();                    /* DTC enabled */
64      R_ADC_Set_OperationOn();
65      R_ADC_Start();                      /* A/D converter enabled */
66
67      while (1U)
68      {
69          NOP();
70      }
71      /* End user code. Do not edit comment generated here */
72  }
```

Figure 4.20 Replacement of main Function

Even if a DTC transfer request is generated, DTC transfer is held pending immediately when there are no commands in the instruction "while (1U)", so add the command "NOP ()" into the "while (1U)".

4.2.4 Other Modifications

If ADC is set with the code generator, interrupt processes are automatically generated. Add the user program to interrupt routine.

Variables for R8C

```
49  /* **** Global Variables **** */
50  #pragma ADDRESS ad_value    00600H      /* Destination address of A/D conversion data */
51  unsigned short ad_value[4];             /* A/D data from AN8 to AN11 addressed from */
52                                          /* 0x0600 to 0x0607 */
53
54  unsigned short an8_value;               /* AN8 value */
55  unsigned short an9_value;               /* AN9 value */
56  unsigned short an10_value;              /* AN10 value */
57  unsigned short an11_value;              /* AN11 value */
```

Add contents corresponding to the red box manually.

r_cg_adc_user.c file for RL78/G14

```
37  #pragma address ad_value=0xFF500
38  unsigned short ad_value[4];             /* A/D data from ANI0 to AN13 addressed from */
39                                          /* 0xFF500 to 0xFF507 */
40
41  unsigned short an0_value;               /* ANI0 value */
42  unsigned short an1_value;               /* ANI1 value */
43  unsigned short an2_value;               /* ANI2 value */
44  unsigned short an3_value;               /* ANI3 value */
```

Figure 4.21 Add Variables for Interrupt Program

Program for R8C

```
301  void ad_converter_int(void)
302  {
303      /* ==== Setting A/D conversion data ==== */
304      an8_value = (ad_value[0]&0x03FF);   /* Setting AN8 value */
305      an9_value = (ad_value[1]&0x03FF);   /* Setting AN9 value */
306      an10_value = (ad_value[2]&0x03FF);  /* Setting AN10 value */
307      an11_value = (ad_value[3]&0x03FF);  /* Setting AN11 value */
308
309      /* ==== A/D converstion starts ==== */
310      dtcct0 = 1;                         /* One-time is set to transfer */
311      dtcct1 = 1;                         /* One-time is set to transfer */
312      dtcen16 = 1;                        /* Activation of A/D interruption enable */
313      adst = 1;                           /* A/D conversion starts */
314  }
```

Add contents corresponding to the red box manually.

r_cg_adc_user.c file for RL78/G14

```
66   static void __near r_adc_interrupt(void)
67   {
68       /* Start user code. Do not edit comment generated here */
69       an0_value = (ad_value[0U] & 0xFFC0U) >> 6U;
70       an1_value = (ad_value[1U] & 0xFFC0U) >> 6U;
71       an2_value = (ad_value[2U] & 0xFFC0U) >> 6U;
72       an3_value = (ad_value[3U] & 0xFFC0U) >> 6U;
73       /* ==== A/D converstion starts ==== */
74       R_DTC_Create();
75       R_DTCD0_Start();                /* DTC enabled */
76       R_ADC_Start();                  /* A/D converter enabled */
77       /* End user code. Do not edit comment generated here */
78   }
```

Figure 4.22 Add Interrupt Program

4.2.5 Sample Code After Replacement

Obtain the sample code "an-r01an3508ec0100-rl78-migrate.zip" from the Renesas Electronics Website. "rl78g14_migrate_dtc" in the "workspace" folder is the sample code that replaces the program of "R8C/35C Group DTC Operation in Chain Transfers (R01AN0372E)".

4.3 Conditions for Confirming Operations of Sample Programs

The operations of the sample codes after replacement are confirmed under the following conditions.

Table 4.1 Conditions for Confirming Operations

Item	Description
Microcontroller used	RL78/G14 (R5F104PJ)
Integrated development environment (CS+)	CS+ for CC V4.01.00 from Renesas Electronics Corp.
C compiler (CS+)	CC-RL V1.03.00 from Renesas Electronics Corp.
Integrated development environment (e2 studio)	e2 studio V5.2.0.020 from Renesas Electronics Corp.
C compiler (e2 studio)	CC-RL V1.03.00 from Renesas Electronics Corp.
Board used	Renesas original

5. Sample Code

The sample code is available on the Renesas Electronics website.

6. Reference Documents

User's Manual

RL78 Family User's Manual: Software (R01US0015E)
CC-RL Compiler User's Manual (R20UT3123E)
CS+ Code Generator Tool Integrated Development Environment User's Manual: Peripheral Function Operation[CS+ for CC][CS+ for CA,CX] (R20UT3104E)
CcnvNC30 C Source Converter User's Manual (R20UT3685E)

Application Note

R8C/35A Group Clock Operation Using RTC (R01AN0079E)
R8C/35C Group DTC Operation in Chain Transfers (R01AN0372E)

(The latest versions can be downloaded from the Renesas Electronics website.)

Website and Support

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

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

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Dec. 22, 2016	-	First edition issued

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

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

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

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

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

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

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

Notice

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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

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

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0