从 78K0 转至 RL78 的迁移指南（CcnvCA78K0）

R01AN3471CC0100
Rev.1.00
2016.12.31

要点

本篇应用说明介绍了将程序从 78K0 转至 RL78 的方法。

对象 MCU

78K0 族

将本篇应用说明应用到其他 MCU 时，请根据 MCU 的规格进行详细的评价。

目录

1. 将程序从 78K0 转至 RL78 的方法

本节介绍将程序从 78K0 转至 RL78 的方法。

首先通过 C 源代码转换器 CcnvCA78K0 将 C 编译器 CA78K0（或者 CC78K0）的扩展功能转换为 C 编译器 CC-RL 的扩展功能。

然后通过集成开发环境 CS+或者 e2studio 创建工程。由于 78K0 与 RL78 的外围功能不同，所以不使用 78K0 的外围功能程序而选择 RL78 的代码生成工具来生成 RL78 的外围功能程序。

将通过 CcnvCA78K0 转换的程序与上述外围功能程序组合后替换程序。

2. 通过 CcnvCA78K0 进行的程序转换

2.1 CcnvCA78K0

CcnvCA78K0 利用 CA78K0（或 CC78K0）的 C 源程序，将扩展语言说明（如宏名称、保留字、#pragma 指令、扩展功能）转换为 CC-RL 的扩展语言说明。

CcnvCA78K0 是一种将用于 CA78K0 的程序转为 CC-RL 上可用的程序的软件。但因无法保证转换后的程序一定可以运行，因此务必要在系统上对转换后的程序进行动作确认。

另外，配置地址 、SFR 的访问以及汇编语言编码等依存于设备的编码是无法转换的，因此需要根据具体情况进行手动修改以适用于 RL78 族。

详情请参照 CcnvCA78K0 C Source Converter User's Manual (R20UT3684EJ)。

2.2 CcnvCA78K0 的用法

通过 CcnvCA78K0 转换程序的方法如下所示。

（1） 将 CcnvCA78K0（CcnvCA78K0.exe）与用于 CA78K0 的程序置于同一个任意的文件夹中。
（2） 在 Windows 中启动命令提示符（Command Prompt）。
（3） 将当前目录变更为存有 CcnvCA78K0 的文件夹。

图 1.1 命令提示符窗口

（4） 执行前在-o 选项中指定输出文件名。执行后即输出用于 CC-RL 的程序。另外，如要将信息输出至指定文件中，则使用-r 选项。

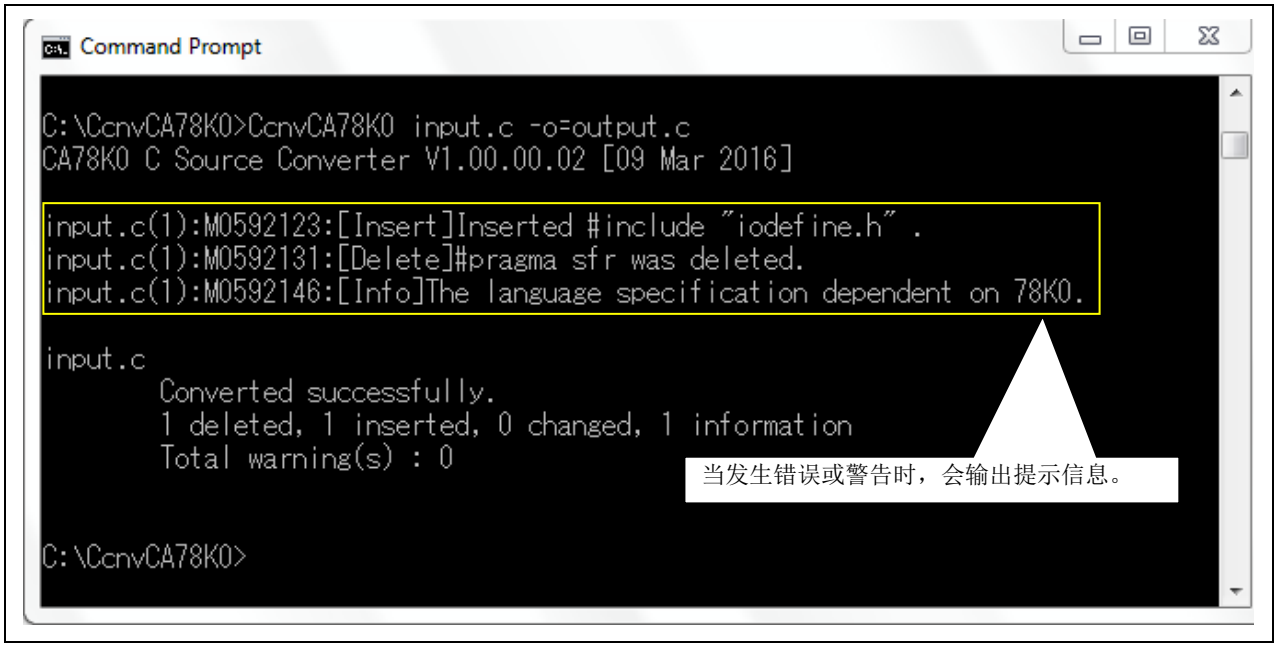


图 1.2 CcnvCA78K0 执行界面

（5） 若要同时转换多个文件，需创建编目文件后执行-l 选项。执行后，CC-RL 程序会被输出至指定的文件夹中。

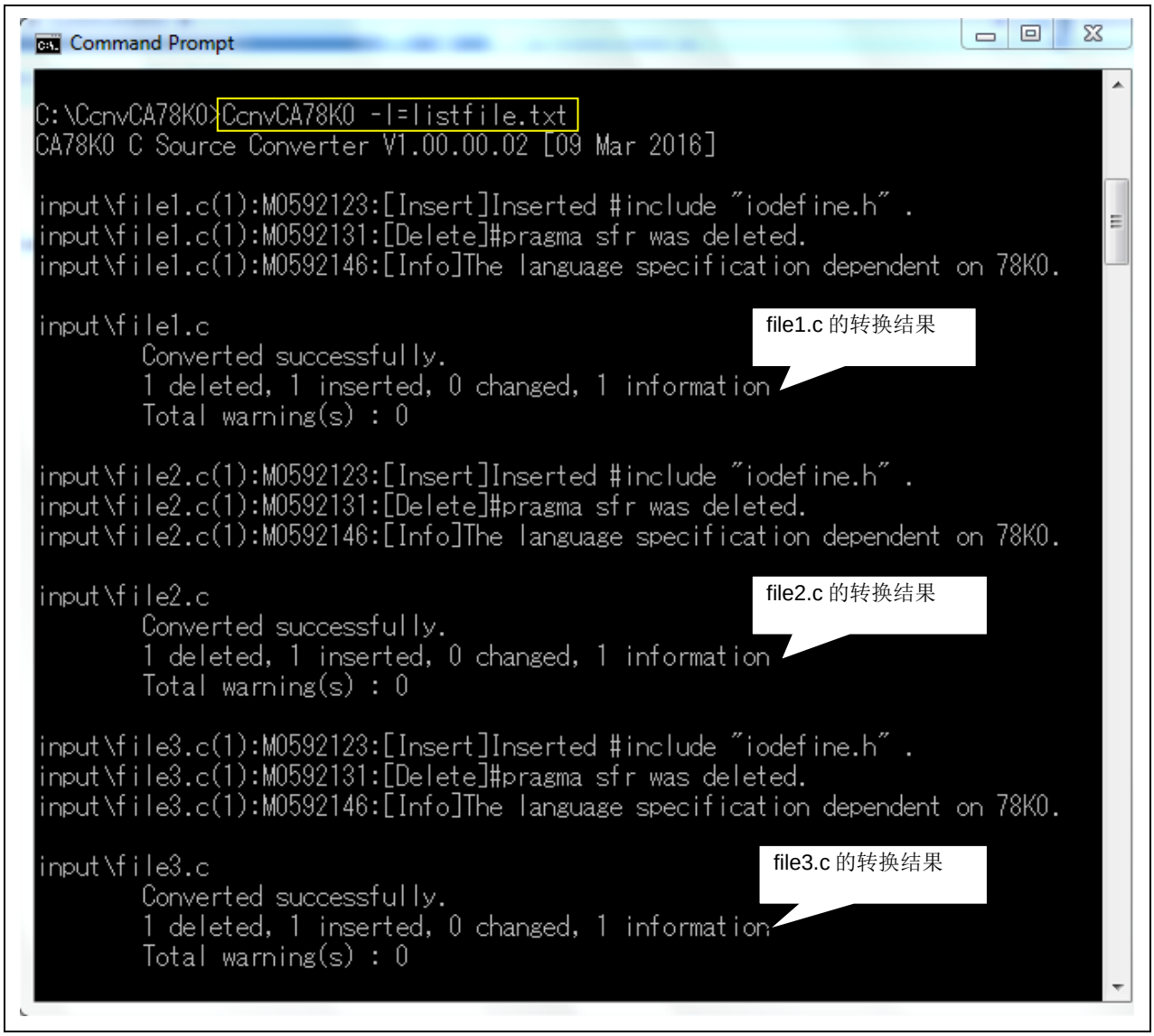


图 1.3 CcnvCA78K0 执行界面（多个文件）

以下为编目文件记述例。

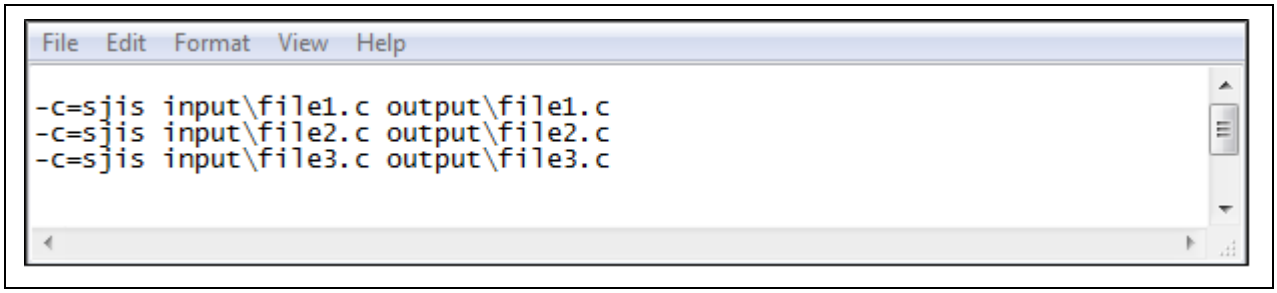


图 1.4 编目文件记述例

（6） 对 CcnvCA78K0 未转换的地方进行修改。具体修改处请参照“CcnvCA78K0 C Source Converter User's Manual (R20UT3684EJ)”中的“CONVERSION SPECIFICATIONS”。

2.3 不使用 CcnvCA78K0 时

不使用 CcnvCA78K0 时，必须手动将 CA78K0（或 CC78K0）的扩展功能修改为 CC-RL 的扩展功能。关于 CC-RL 的扩展语言说明，请参照“CC-RL Compiler User’s Manual (R20UT3123EJ)”。

3. 外围功能的程序转换

3.1 程序的自动生成

通过集成开发环境 CS+或 e2studio 中用于 RL78 族的代码生成器，为 RL78 族的外围功能自动生成对应于 78K0 族所用外围功能的程序。关于代码生成工具的操作方法，请参照“CS+ Code Generator Integrated Development Environment User’s Manual: Peripheral Function Operation (R20UT3104EJ)”。

（1） 在“Project Tree”中点击“Code Generator (Design Tool) ”下的“Clock Generator”，并进行“Pin assignment”的设置。

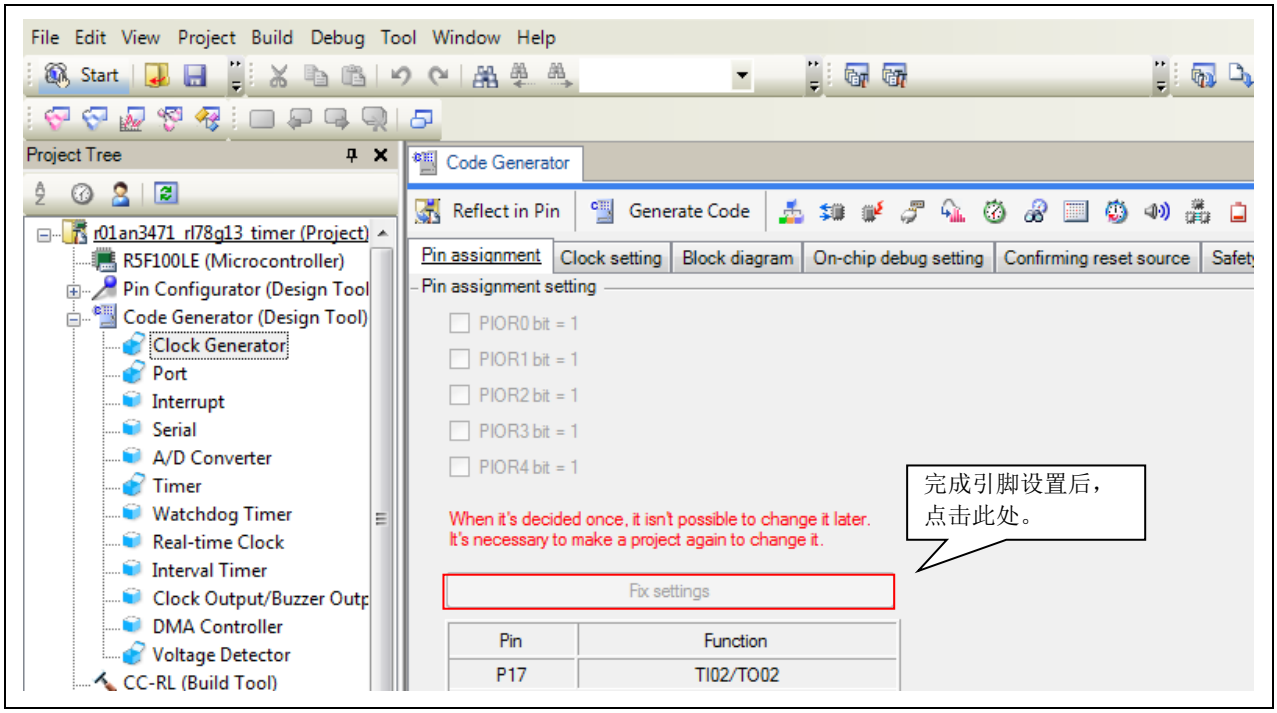


图 3.1 代码生成器窗口（1）

（2） 参照 78K0 族的程序设置每个功能。

（3） 所有外围功能设置完成后，请点击位于界面上方的“Generate Code”生成代码（即程序的自动生成）。然后用自动生成的外围功能函数置换程序。

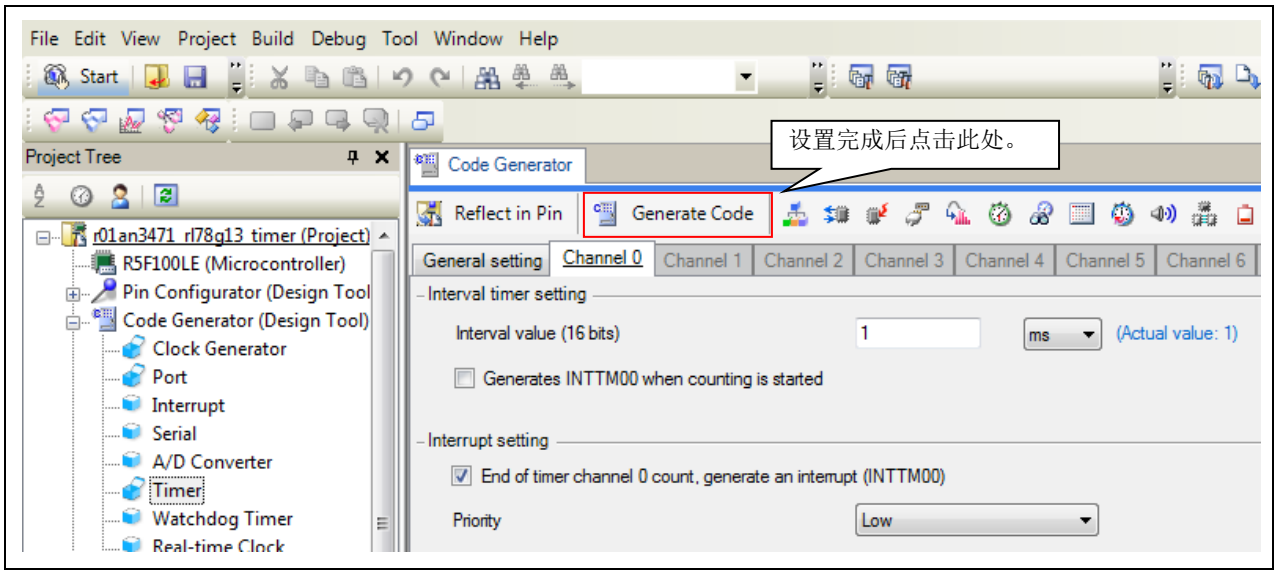


图 3.2 代码生成器窗口（2）

3.2 程序的添加

无法通过代码生成工具自动生成的程序（如 main 函数、中断函数处理以及变量等），需手动添加。

在自动生成的各文件中的“/* Start user code for adding. Do not edit comment generated here */” 和 “/* End user code. Do not edit comment generated here */”之间添加程序。程序的添加需要手动进行。另外请注意，如在上述范围之外添加程序，在自动生成程序时，会被自动删除。

请务必使用添加好的程序对系统进行动作确认。

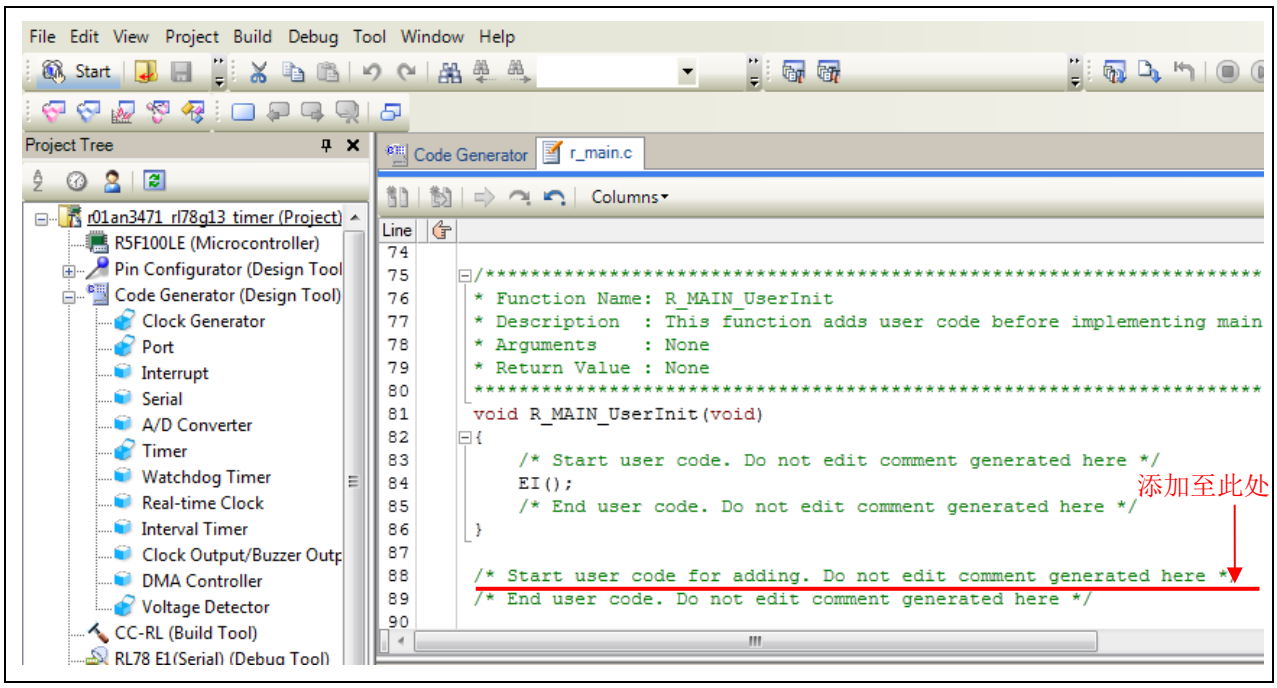


图 3.3 添加现有程序

3.3 不使用代码生成工具时

不使用代码生成工具时，通过集成开发环境 CS+或 e2studio 创建了一个新程序后，需手动创建外围功能程序。关于外围功能的详细情况，请参照 RL78 族用户手册。

4. 置换实例

4.1 78K0/Kx1 参考例程（串行接口 UART0）

“78K0/Kx1,78K0/Kx1+シリアル通信プログラム集”中的串行接口 UART0 程序被替换为 RL78/G13 的程序。替换后的程序文件为“r01an3471_rl78g13_serial”。

该程序使用 UART0 模块每隔 2ms 以 9600bps 的传送速度发送一个 0x55 的数据。CPU 时钟为 10MHz 的高速系统时钟。

4.1.1 使用 CcnvCA78K0 为 CC-RL 导入源文件

（1） 创建编目文件以指定一个要转换的 C 源文件。

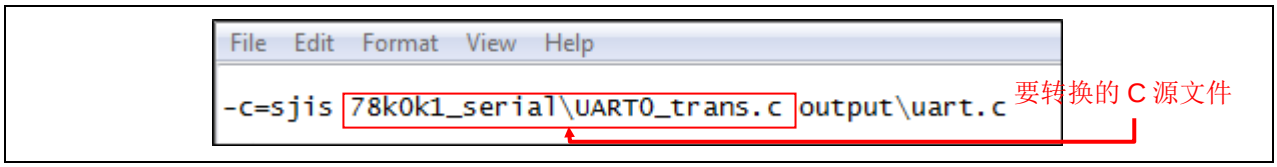


图 4.1 编目文件记述例（串行接口 UART0）

（2） 启动命令提示符（Command Prompt），对编目文件指定的 C 源文件进行转换。输出的转换结果文件中标有变更处。

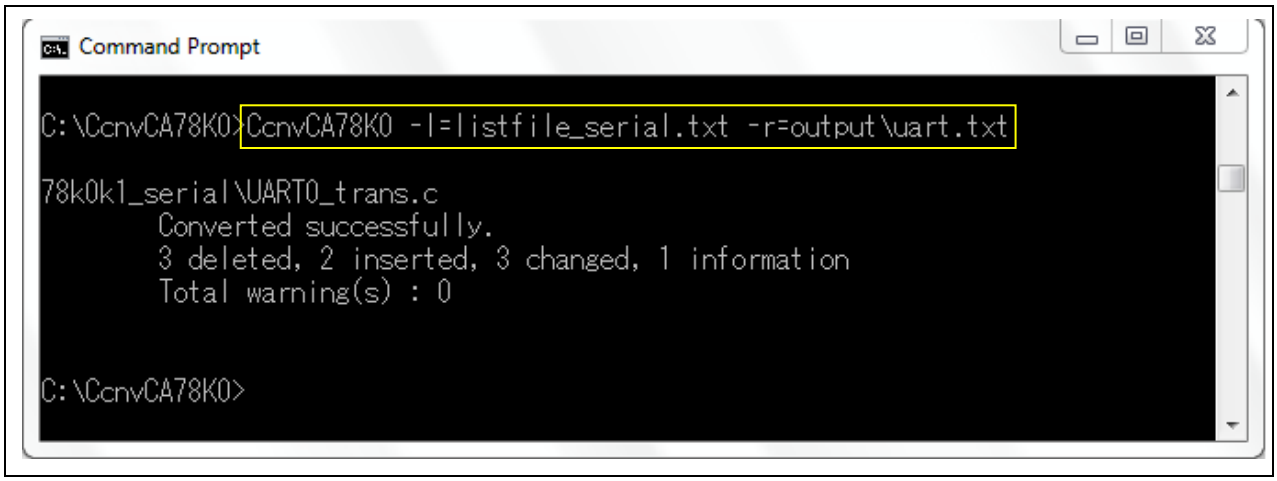


图 4.2 CcnvCA78K0 执行界面（串行接口 UART0）

转换结果文件中记载着转换结果（如下图所示）。关于具体的转换结果，请参照“CcnvCA78K0 C Source Converter User's Manual (R20UT3684EJ)”。

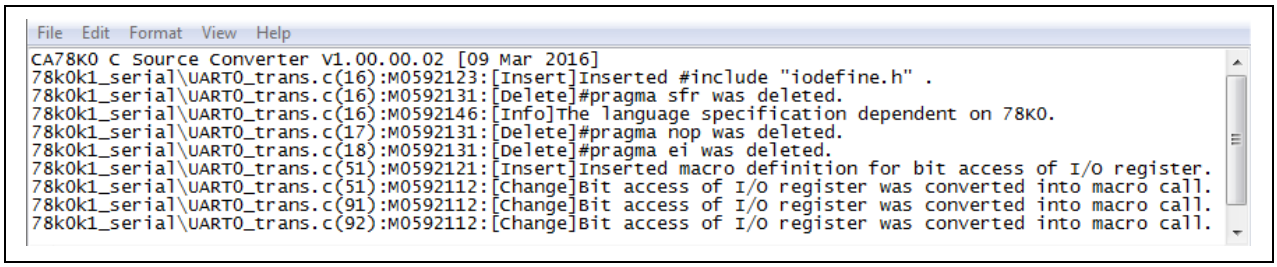
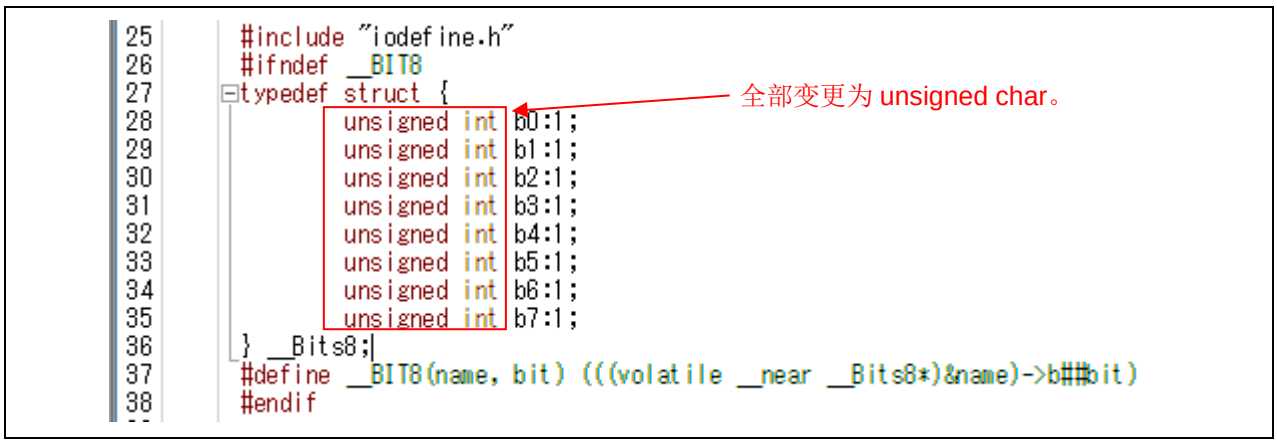


图 4.3 具体的转换结果（串行接口 UART0）

（3） 修改转换后的 C 源文件。

对 SFR 及 saddr 变量进行的位访问被置换为位字段和宏的声明（如下图所示）。当对 8 位的 SFR 进行位访问时，要将 unsigned int 改为 unsigned char。




```
25      #include "iodefine.h"
26      #ifndef __BIT8
27      typedef struct {
28              unsigned int b0:1;
29              unsigned int b1:1;
30              unsigned int b2:1;
31              unsigned int b3:1;
32              unsigned int b4:1;
33              unsigned int b5:1;
34              unsigned int b6:1;
35              unsigned int b7:1;
36      } __Bits8;
37      #define __BIT8(name, bit) (((volatile __near __Bits8*)&name)->b##bit)
38      #endif
```

图 4.4 更改位访问的声明

4.1.2 程序的自动生成

（1） 通过集成开发环境 CS+或者 e2studio 创建一个新的工程。
（2） 利用代码生成工具设置各个功能。
CPU 时钟设为 10MHz 的高速系统时钟。

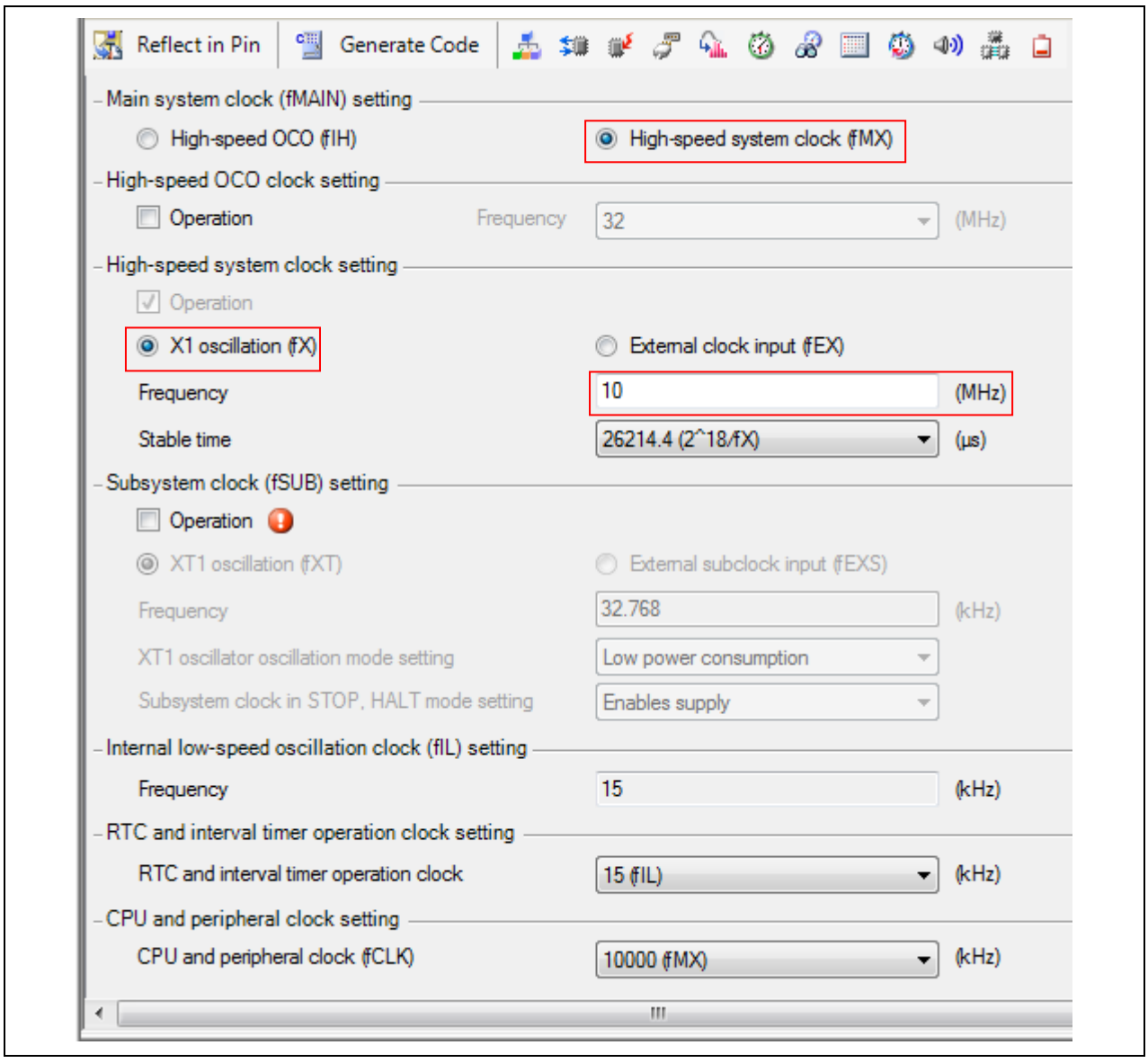


图 4.5 代码生成工具的设置界面（时钟）

对等同于 78K0 族的串行接口 UART0 的串行阵列单元的 UART0 进行设置。

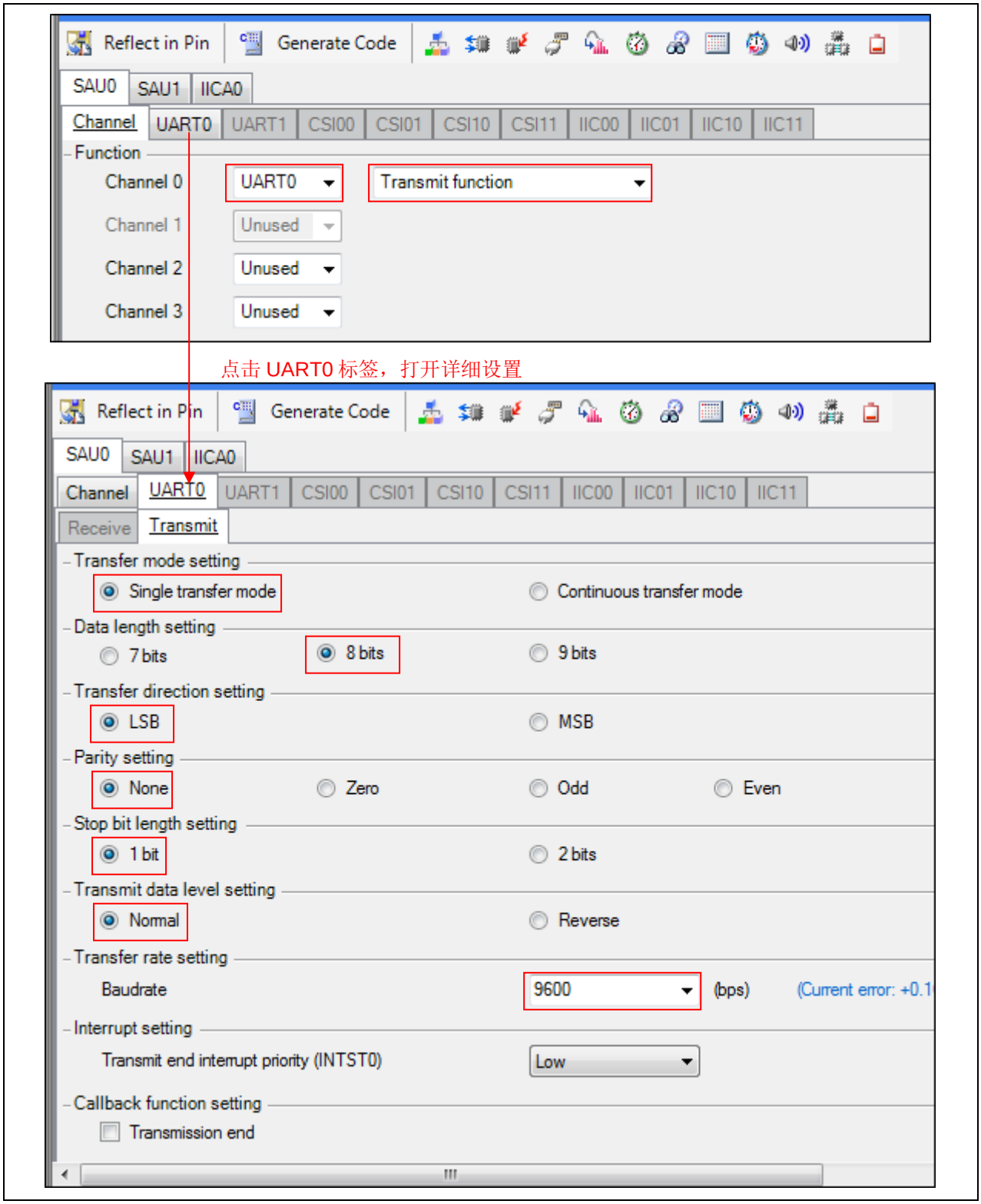


图 4.6 代码生成工具的设置界面（串行阵列单元）

（3） 设置“端口”、“看门狗定时器”以及“电压检测电路”。
（4） 点击“Generate Code”以生成文件。

4.1.3 程序的添加

本节介绍在代码生成的程序上添加符号定义和主函数处理的内容。关于其他程序（如时钟设置、UART 功能设置等），请使用由代码生成的程序。

- 符号定义
  在 r_cg_userdefine.h 上添加符号定义。

78K0 的程序

```
22  ------------------------------------------------------------------------
23      Constants/Variables
24  ------------------------------------------------------------------------
25  */
26
27  #define UART_BAUDRATE_M0     0x3
28  #define UART_BAUDRATE_K0     0x10
29
30  /*status list definition*/
31  #define TRUE                 1
32  #define FALSE                0
```

```
32  /************************************************************************
33  User definitions
34  ************************************************************************/
35
36  /* Start user code for function. Do not edit comment generated here */
37  #define TRUE                 1
38  #define FALSE                0
39  /* End user code. Do not edit comment generated here */
```

图 4.7 符号定义的置换

• 主函数

使用了 RL78/G13 的代码生成工具后，在执行 main 函数之前会先执行 R_Systeminit 函数。通过 R_Systeminit 函数对时钟和 UART0 进行初始设置。因此只需对红框内的处理进行程序的手动添加。通过 R_UART_Start 函数启动 UART0 的运行。

78K0 的程序

```
43      void main( void )
44      {
45          unsigned short i;
46
47          PCC  = 0x00;                        /* CPU clock: fx */
48          WDTM = 0x77;                        /* Watchdog Timer Stop */
49
50          /* Waiting for oscillation stable time */
51          while( OSTC.0 == 0);
52          MCM0 = 1;                           /* supply clock: X1 */
53          /* Waiting for X1 clock change */
54          while( MCS == 0 );
55
56          UART0_Init();                       /* UART0 initialization function */
57          UART0_Enable();                     /* UART0 enable function */
58
59          while(TRUE)                         /* main loop */
60          {
61              TXS0 = 0x55;
62              /* Waiting for the completion of transmitting */
63              while( DUALIF0 == 0 );
64              DUALIF0 = 0;
65
66              for( i=0 ; i<1000 ; i++ );      /* wait 2 ms */
67          }
68      }
```

RL78/G13 的 r_main.c 文件

```
59      void main(void)
60      {
61          R_MAIN_UserInit();
62          /* Start user code. Do not edit comment generated here */
63          {
64              unsigned short i = 0;
65
66              R_UART0_Start();
67
68              while(TRUE)                             /* main loop */
69              {
70                  TXD0 = 0x55;
71                  /* Waiting for the completion of transmitting */
72                  while( STIF0 == 0 );
73                  STIF0 = 0;
74
75                  for( i=0 ; i<4000 ; i++ );      /* wait 2 ms */
76              }
77          }
78          /* End user code. Do not edit comment generated here */
79      }
```

图 4.8 主函数的置换

4.1.4 其他变更事项

（1） 通过代码生成工具设置 UART0 之后，自动生成中断处理。此次不使用中断，将其设为禁止。
（2） 由于不使用中断功能，因此将 R_MAIN_UserInit 的“EI();”改为“DI();”。
（3） 再次调整软件定时器的处理时间。因编译器不同，处理时间也可能发生变化。

4.1.5 置换后的参考例程

请从瑞萨电子官网下载“an-r01an3471cc0100-rl78-migrate.zip”参考例程。

“workspace ”文件夹中的“rl78g13_migrate_serial”为“ 78K0/Kx1,78K0/Kx1+シリアル通信プログラム集”中置换串行接口 UART0 程序的参考例程。

4.2 78K0/Kx2 参考例程（间隔定时器）

“78K0/Kx2 サンプル・プログラム インターバル・タイマ編（U19031JJ2V0AN00）”中包含的程序被替换为 RL78/G13 的程序。替换后的程序文件为“r01an3471_rl78g13_timer”。

该程序使用 16 位定时器/事件计时器 00 每 1ms 生成一个间隔中断。在间隔中断处理内翻转 P10。每发生 100 次间隔中断，翻转一次 P11。

4.2.1 使用 CcnvCA78K0 为 CC-RL 导入源文件

（1） 创建编目文件以指定一个要转换的 C 源文件。

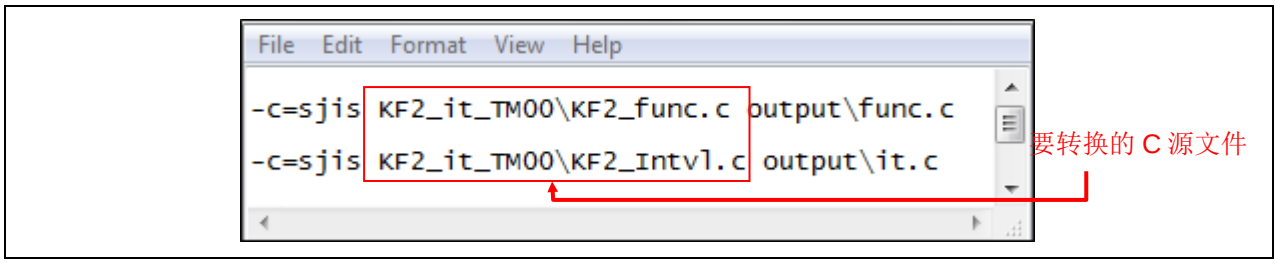


图 4.9 编目文件记述例（间隔定时器）

（2） 启动命令提示符（Command Prompt），对编目文件指定的 C 源文件进行转换。输出的转换结果文件中标有变更处。

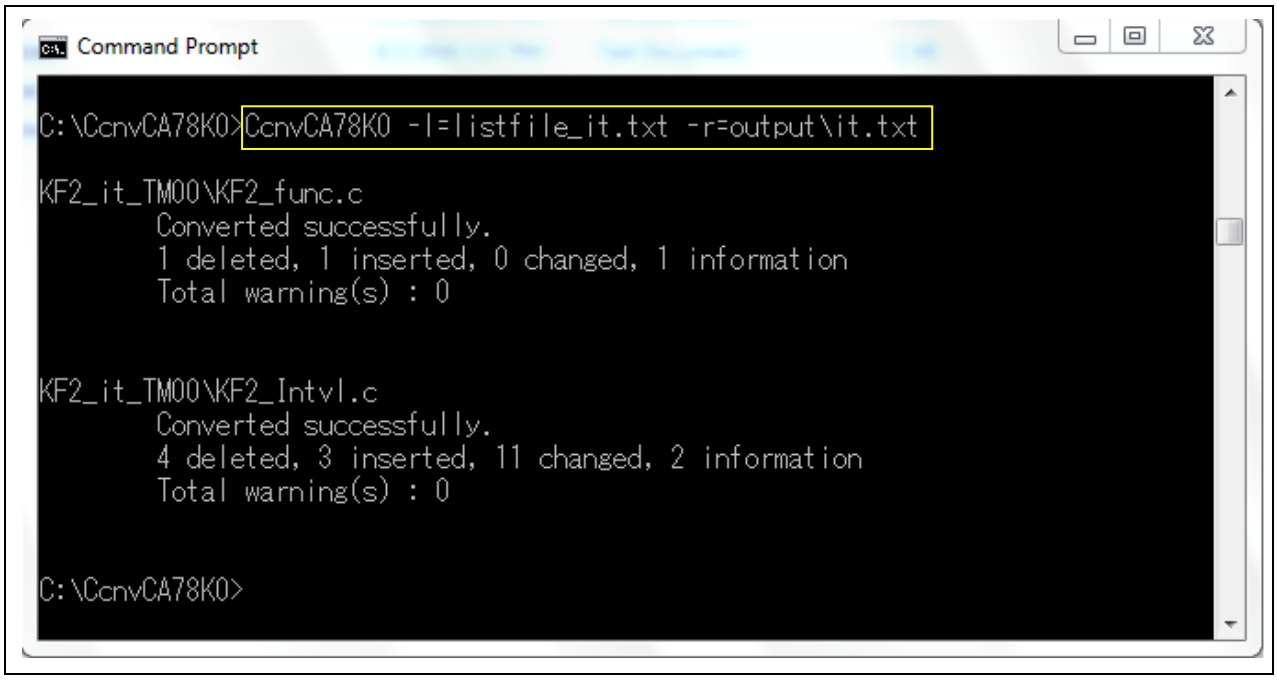


图 4.10 CcnvCA78K0 执行界面（间隔定时器）

转换结果文件中记载着转换结果（如下图所示）。关于具体的转换结果，请参照“CcnvCA78K0 C Source Converter User's Manual (R20UT3684EJ)”。

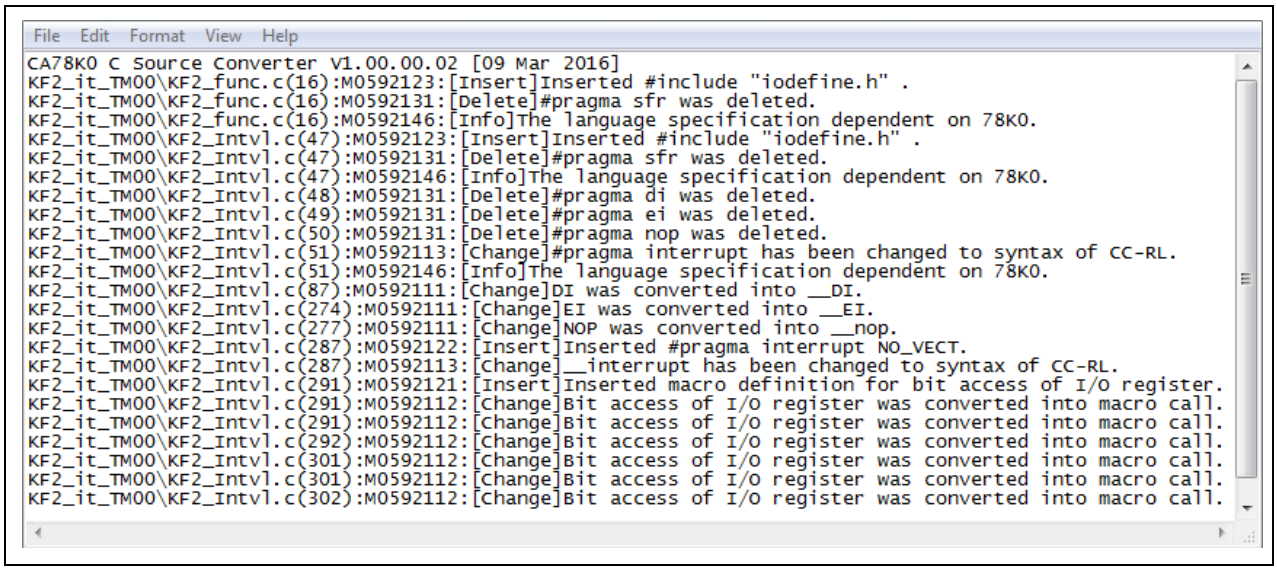


图 4.11 具体的转换结果（间隔定时器）

（3） 修改转换后的 C 源文件。

对 SFR 及 saddr 变量进行的位访问被置换为位字段和宏的声明（如下图所示）。当对 8 位的 SFR 进行位访问时，要将 unsigned int 改为 unsigned char。

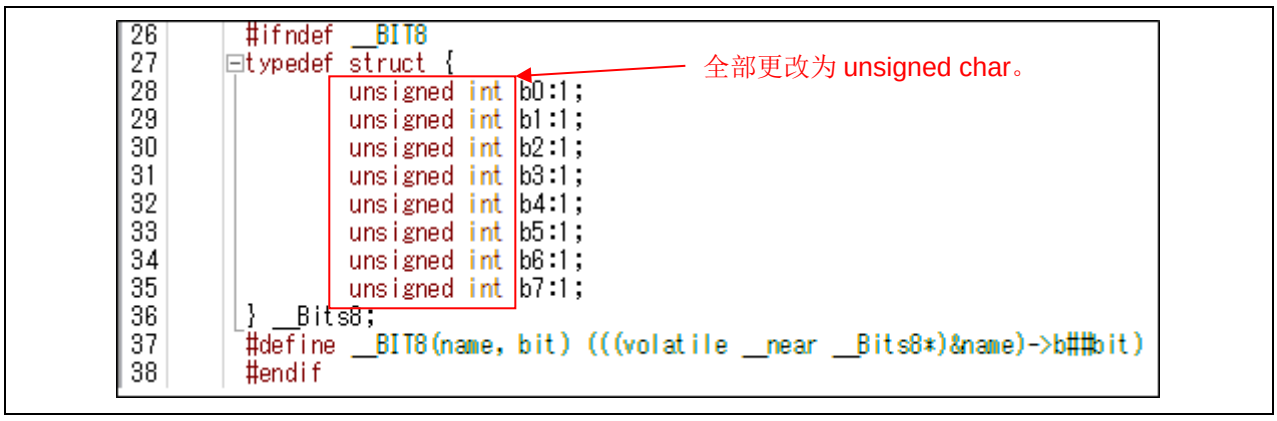


图 4.12 更改位访问的声明

有时会出现中断函数声明重复的情况。这种情况下 CC-RL 会发生错误，因此要删掉转换来的#pragma 指令。

```
329     //[CcnvCA78K0] __interrupt void fn_intTimerInterval(void)
330     //{
331     #pragma interrupt fn_intTimerInterval
332     void fn_intTimerInterval(void)
333     {
334             g_ucIntCnt++;                                       /* Incremented
335
336
337     //[CcnvCA78K0]  if( P1.0 )       P1.0    =       0;       /* To invert th
338             if( __BIT8( P1, 0 ) )    __BIT8( P1, 0 ) =       0;       /* To i
339     //[CcnvCA78K0]  else             P1.0    =       1;
340             else             __BIT8( P1, 0 ) =       1;
341
342             if(g_ucIntCnt == 100){
343
344     //[CcnvCA78K0]          if( P1.1 )       P1.1    =       0;       /* Inve
345                     if( __BIT8( P1, 1 ) )    __BIT8( P1, 1 ) =       0;
346     //[CcnvCA78K0]          else             P1.1    =       1;
347                     else             __BIT8( P1, 1 ) =       1;
348
349                     g_ucIntCnt               =       0;       /* Initializes
350             }
351     }
```

删除#pragma 指令。

图 4.13 更改中断函数的声明

4.2.2 程序的自动生成

（1） 通过集成开发环境 CS+或者 e2studio 创建一个新的工程。
（2） 利用代码生成工具设置各个功能。
CPU 时钟设为 8MHz 的高速内部振荡器时钟。

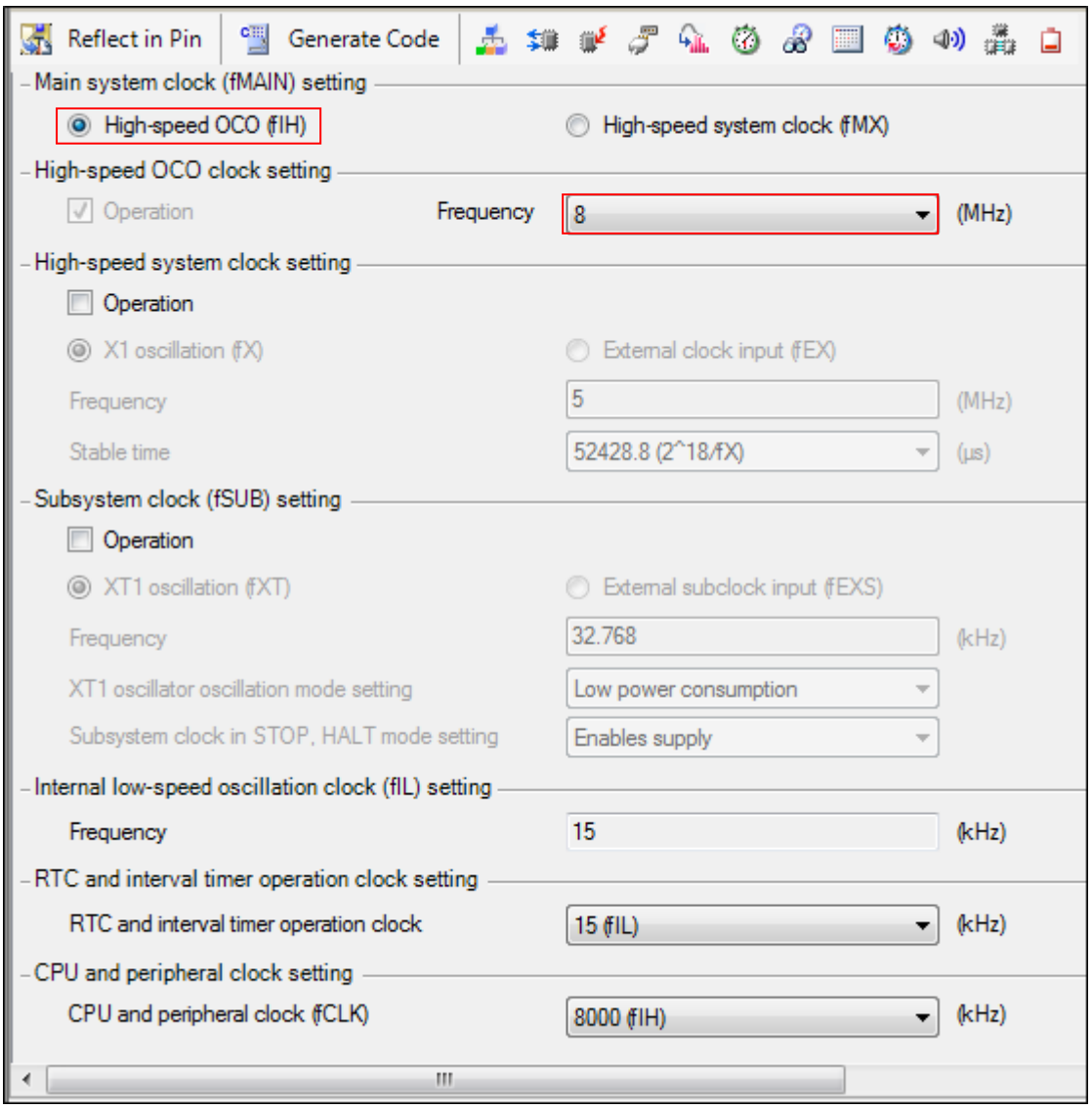


图 4.14 代码生成工具的设置界面（时钟）

对等同于 78K0 族的 16 位定时器/事件计时器 00（TM00）的定时器阵列单元的间隔定时器功能进行设置。

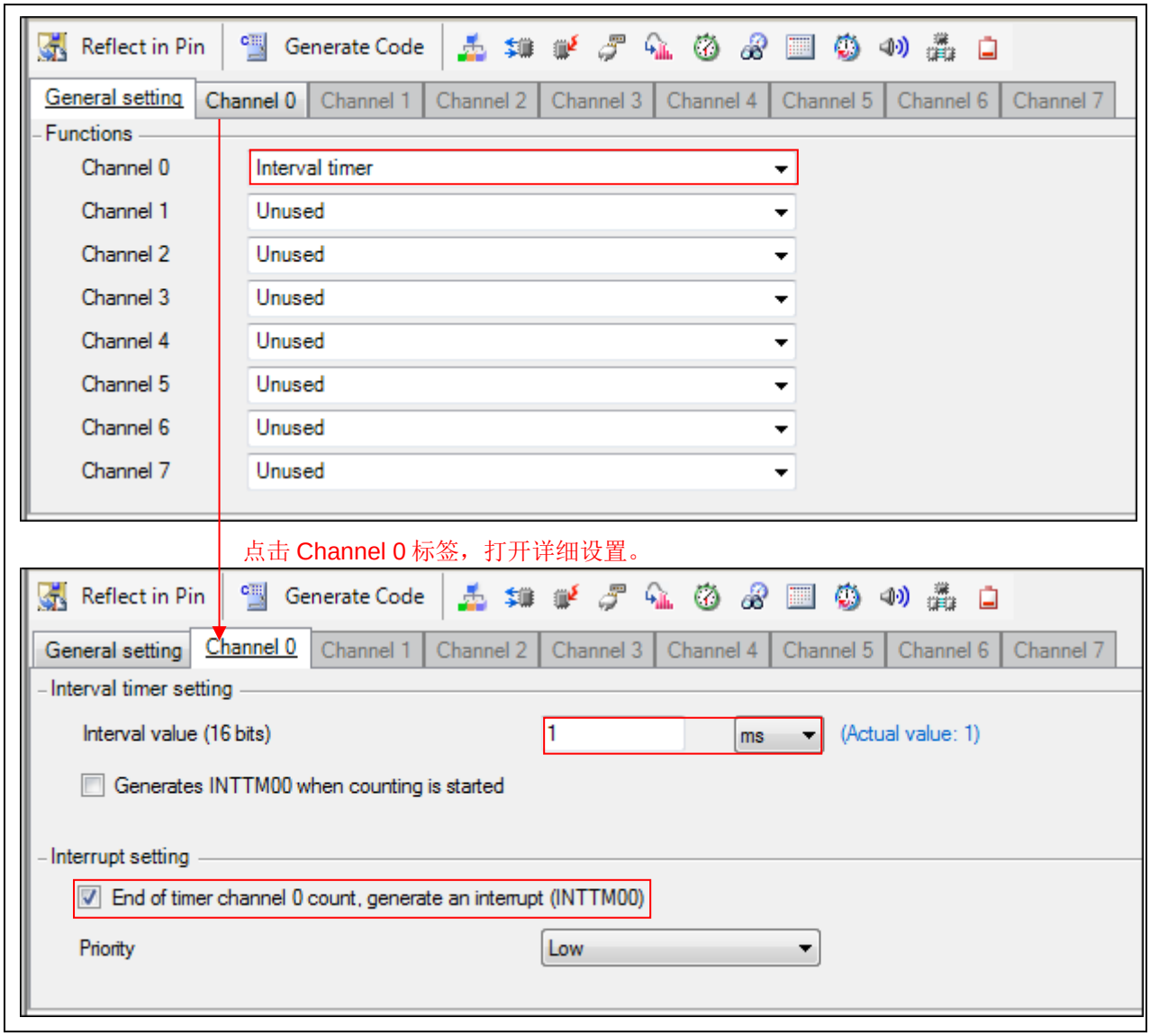


图 4.15 代码生成工具的设置界面（定时器阵列单元）

（3） 设置“端口”、“看门狗定时器”以及“电压检测电路”。
（4） 点击“Generate Code”以生成文件。

4.2.3 程序的添加

本节介绍在代码生成的程序上添加符号定义、主函数和中断函数处理的内容。关于其他程序（如时钟设置、定时器阵列单元的设置等），请使用由代码生成的程序。

- 变量
  在 r_main.c 和 r_cg_timer_user.c 上添加变量。并在 r_cg_timer_user.c 上添加位字段的声明。

78K0 的程序

```
112  /*************************************************************************
113  ;       Global variables and functions
114  ;*************************************************************************/
115  static  unsigned        char    g_ucIntCnt;
```

RL78/G13 的 r_main.c 文件

```
46   /*************************************************************************
47   Global variables and functions
48   *************************************************************************/
49    /* Start user code for global. Do not edit comment generated here */
50     unsigned    char    g_ucIntCnt = 0;     /* Count of the interrupt gen
51    /* End user code. Do not edit comment generated here */
```

RL78/G13 的 r_cg_timer_user.c 文件

```
45   /*************************************************************************
46   Global variables and functions
47   *************************************************************************/
48    /* Start user code for global. Do not edit comment generated here */
49   #ifndef __BIT8
50   typedef struct {
51       unsigned char b0:1;
52       unsigned char b1:1;
53       unsigned char b2:1;
54       unsigned char b3:1;
55       unsigned char b4:1;
56       unsigned char b5:1;
57       unsigned char b6:1;
58       unsigned char b7:1;
59   } __Bits8;
60   #define __BIT8(name, bit) (((volatile __near __Bits8*)&name)->b##bit)
61    #endif
62
63   extern      unsigned    char    g_ucIntCnt;
64
65    /* End user code. Do not edit comment generated here */
```

输入位字段的声明

图 4.16 变量的置换

• 主函数

将 78K0 所用程序中的主函数处理添加至 RL78/G13 的 r_main.c 主函数中。关于定时器阵列单元的运行开始，请更改为 RL78/G13 的代码生成工具自动生成的 R_TAU0_Channel_Start()。

78K0 的程序

```
311     void main(void){
312
313             g_ucIntCnt      =       0;              /* Initializes the counter
314             fn_InitTimer();                 /* Initializes the interval timer
315     //[CcnvCA78K0]  EI();                           /* Vector interrupt
316             __EI();                         /* Vector interrupt enable
317
318             while (1){
319     //[CcnvCA78K0]          NOP();
320                     __nop();
321             }
322     }
```

RL78/G13 的 r_main.c 文件

```
60      void main(void)
61      {
62          R_MAIN_UserInit();
63          /* Start user code. Do not edit comment generated here */
64          g_ucIntCnt = 0;             /* Initializes the counter */
65          R_TAU0_Channel0_Start();        /* Timer array unit operation start */
66          __EI();                         /* Vector interrupt enable */
67
68          while (1)
69          {
70              __nop();
71          }
72          /* End user code. Do not edit comment generated here */
73      }
```

图 4.17 主函数的置换

• 中断函数
将中断处理添加到 r_cg_timer_user.c 中的 r_tau0_channel0_interrupt() 。

78K0 的程序

```
331     #pragma interrupt fn_intTimerInterval
332     void fn_intTimerInterval(void)
333     {
334             g_ucIntCnt++;                                   /* Incremented each time inter
335
336
337     //[CcnvCA78K0]  if( P1.0 )      P1.0    =       0;      /* To invert the P10 each time
338             if( __BIT8( P1, 0 ) )   __BIT8( P1, 0 ) =       0;      /* To invert the P10 e
339     //[CcnvCA78K0]  else            P1.0    =       1;
340             else            __BIT8( P1, 0 ) =       1;
341
342             if(g_ucIntCnt == 100){
343
344     //[CcnvCA78K0]          if( P1.1 )      P1.1    =       0;      /* Inverted each time
345                     if( __BIT8( P1, 1 ) )   __BIT8( P1, 1 ) =       0;      /* Inverted ea
346     //[CcnvCA78K0]          else            P1.1    =       1;
347                     else            __BIT8( P1, 1 ) =       1;
348
349                     g_ucIntCnt              =       0;      /* Initializes the counter */
350             }
351     }
```

RL78/G13 的 r_cg_timer_user.c 文件

```
73      static void __near r_tau0_channel0_interrupt(void)
74      {
75          /* Start user code. Do not edit comment generated here */
76          g_ucIntCnt++;                   /* Incremented each time interrupt generation
77
78          if( __BIT8( P1, 0 ) ) __BIT8( P1, 0 ) = 0;      /* To invert the P10 each tim
79          else __BIT8( P1, 0 ) = 1;
80
81          if(g_ucIntCnt == 100){
82              if( __BIT8( P1, 1 ) ) __BIT8( P1, 1 ) = 0;  /* Inverted each time an inte
83              else __BIT8( P1, 1 ) = 1;
84
85              g_ucIntCnt = 0;         /* Initializes the counter */
86          }
87          /* End user code. Do not edit comment generated here */
88      }
```

图 4.18 中断函数的置换

4.2.4 置换后的参考例程

请从瑞萨电子官网下载“an-r01an3471cc0100-rl78-migrate.zip”参考例程。

“workspace”文件夹中的“rl78g13_migrate_timer”为“78K0/Kx1,78K0/Kx1+サンプル・プログラムインターバル・タイマ編（U19031JJ2V0AN00）”中置换了程序的参考例程。

4.3 78K0/Kx2 参考例程（A/D 转换器）

“78K0/Kx2 サンプル・プログラム A/D コンバータ（ZUD-CC-10-0016）”中包含的程序被替换为 RL78/G13 的程序。替换后的程序文件为“r01an3471_rl78g13_ad”。

该程序使用 4 个模拟输入通道，每 1ms 更换一次通道并进行 A/D 转换。以 32ms 为 1 个周期，在进行了 4 通道×8 次的采样之后，将平均值保存在各个变量中。根据平均值，使对应于模拟输入通道的 LED 灯点亮/熄灭。

4.3.1 使用 CcnvCA78K0 为 CC-RL 导入源文件

（1） 创建编目文件以指定一个要转换的 C 源文件。

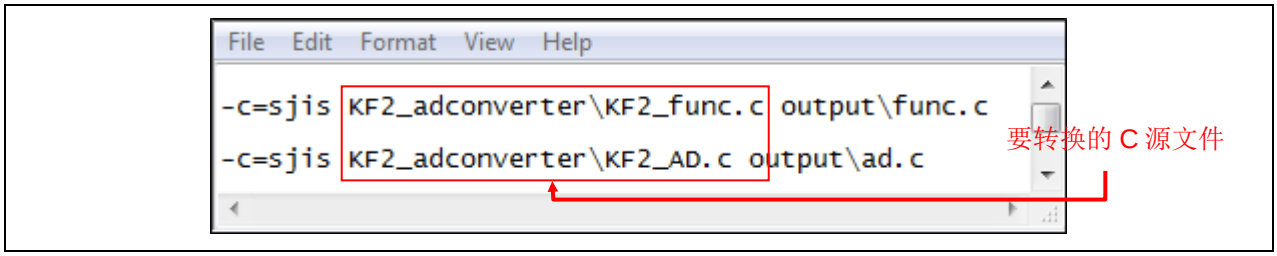


图 4.19 编目文件记述例（A/D 转换器）

（2） 启动命令提示符（Command Prompt），对编目文件指定的 C 源文件进行转换。输出的转换结果文件中标有变更处。

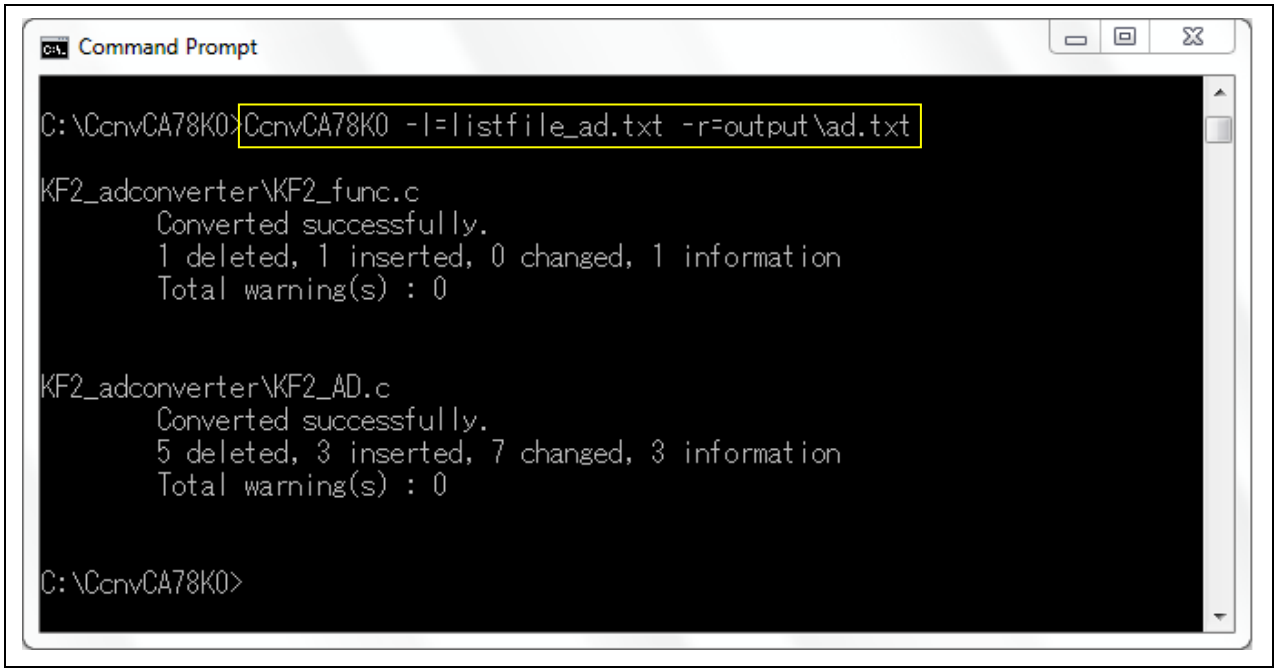


图 4.20 CcnvCA78K0 执行界面（A/D 转换器）

转换结果文件中记载着转换结果（如下图所示）。关于具体的转换结果，请参照“CcnvCA78K0 C Source Converter User's Manual (R20UT3684EJ)”。

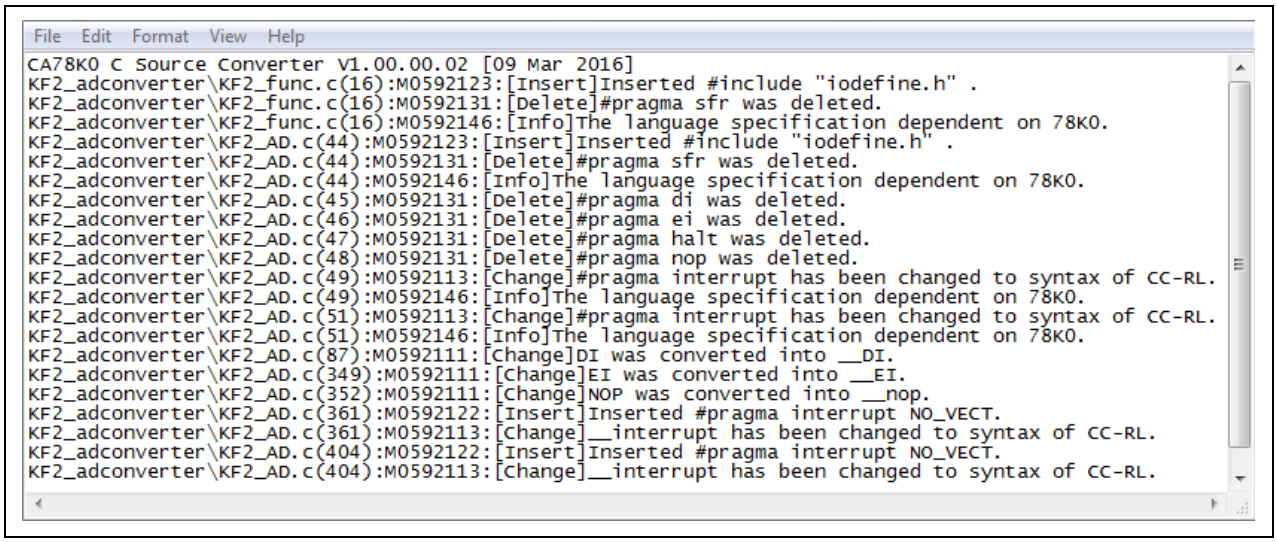


图 4.21 具体的转换结果（A/D 转换器）

（3） 修改转换后的 C 源文件。

有时会出现中断函数声明重复的情况。这种情况下 CC-RL 会发生错误，因此要删掉转换来的#pragma 指令。

```
392     //{
393     #pragma interrupt fn_intAdConverter        <-- 删除#pragma 指令。
394     void fn_intAdConverter(void)
395     {
396             if(g_ucAdCnt == 0){
397                     for(g_ucAdCh = 0; g_ucAdCh < 4; g_ucAdCh++){
398                             g_ucAdData[g_ucAdCh] = 0;
399                                                             /* Clear the A / D conversion result buff
400                     }
401                     g_ucAdCh = 0;
402             }
403
404             g_ucAdData[g_ucAdCh] += ADCR >> 6;
405                                                             /* Remove the lower 6 bits of the A / D c
406             g_ucAdCh++;                                     /* Increments the A/D conversion channel
407
408             g_ucAdCh &= 0b00000011;
409             ADS = g_ucAdCh;
410                                                             /* Change the analog input channel*/
411
412             g_ucAdCnt++;                                    /* Increments the A/D conversion counter
413
414             if(g_ucAdCnt >= 32){
415                     for(g_ucAdCh = 0; g_ucAdCh < 4; g_ucAdCh++){
416
417                             g_ucAdCnt = g_ucAdCnt << 1;
418                             g_ucAdData[g_ucAdCh] = g_ucAdData[g_ucAdCh] >> 3;       /* Average the A
419                             if(g_ucAdData[g_ucAdCh] >= 612){                        /* ANI pin more t
420                                     g_ucAdCnt &= 0b11111110;
421                             }
422                             else{                                                   /* ANI pin is les
423                                     g_ucAdCnt |= 0b00000001;
424                             }
425                     }
426                     g_ucAdCnt &= 0b11111111;
427                     P1 = g_ucAdCnt;
428                     g_ucAdCnt = 0;                          /* Clear the A / D conversion counter */
429             }
430             ADCS = 0;                                       /* A/D conversion is stopped */
431             ADIF = 0;
432     }
433     /************************************************************************
434     ;
435     ;       interval timer interrupt processing(INTTM000)
436     ;
437     ;************************************************************************/
438     //[CcnvCA78K0] __interrupt void fn_intTimerInterval(void)
439     //{
440     #pragma interrupt fn_intTimerInterval       <-- 删除#pragma 指令。
441     void fn_intTimerInterval(void)
442     {
443             ADCS    =       1;                              /* A/D conversion start */
444     }
```

图 4.22 更改中断函数的声明

4.3.2 程序的自动生成

（1） 通过集成开发环境 CS+或者 e2studio 创建一个新的工程。
（2） 利用代码生成工具设置各个功能。
CPU 时钟设为 8MHz 的高速内部振荡器时钟。

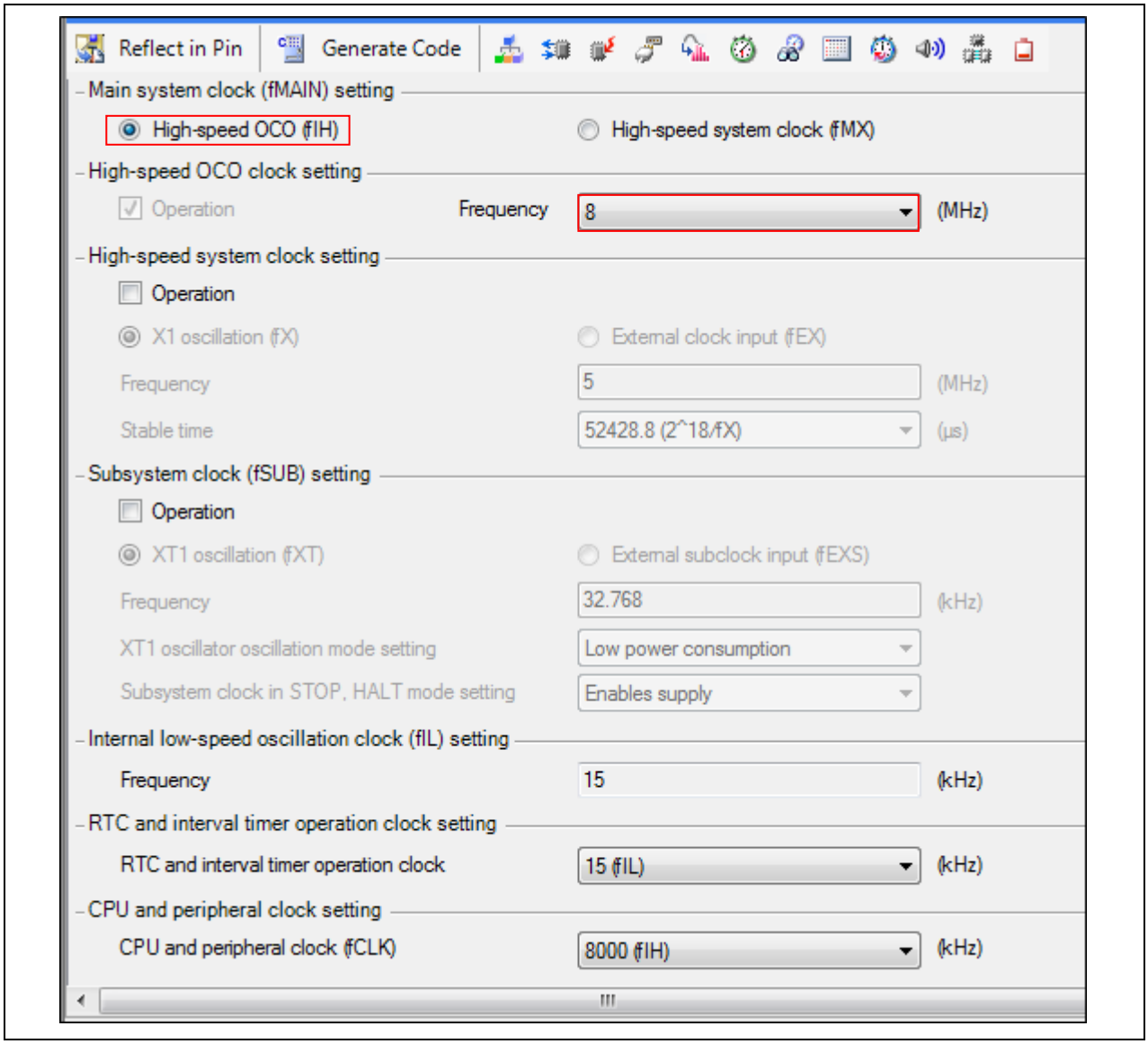


图 4.23 代码生成工具的设置界面（时钟）

设置 A/D 转换器。

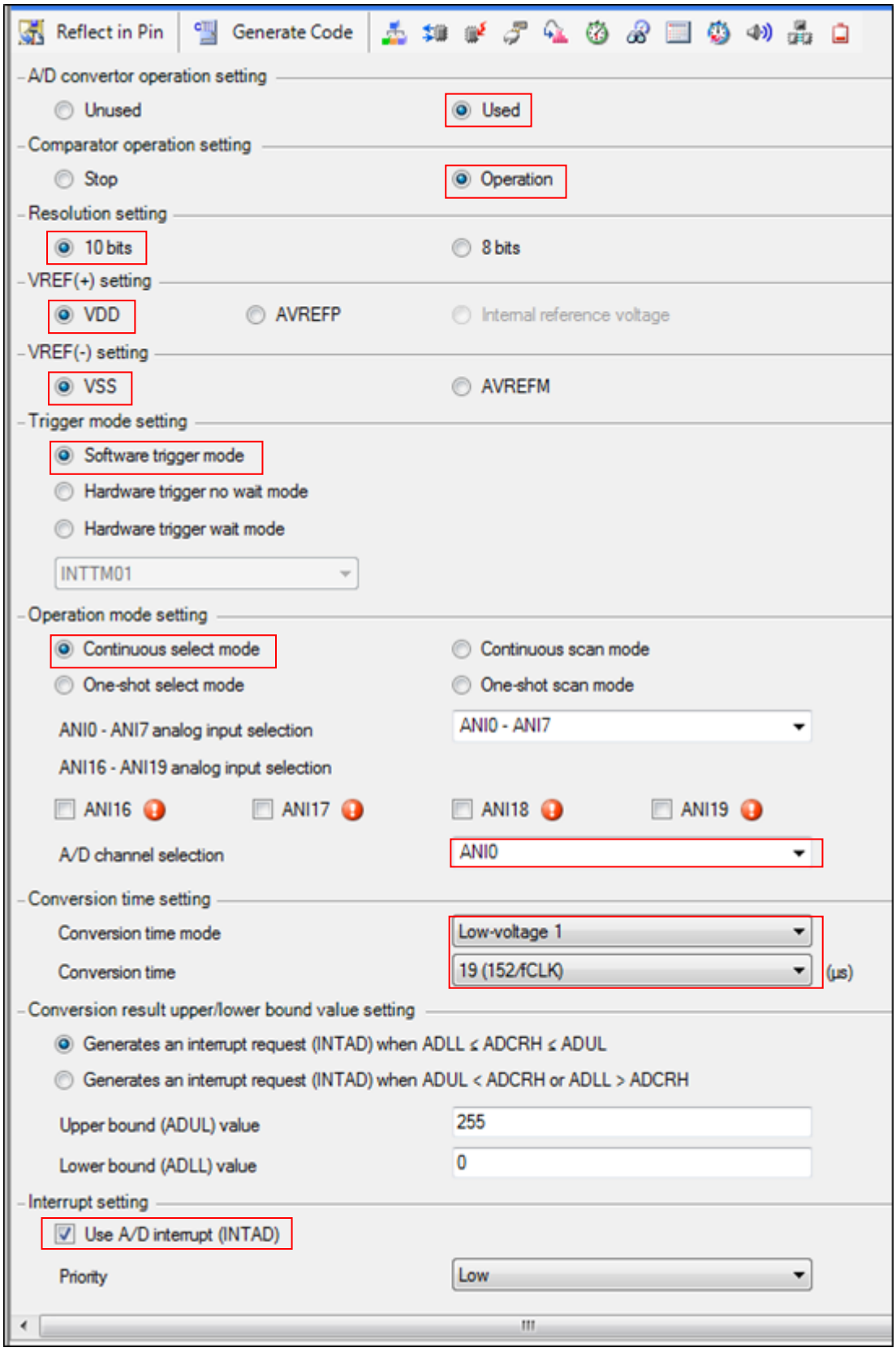


图 4.24 代码生成工具的设置界面（A/D 转换器）

对等同于 78K0 族的 16 位定时器/事件计时器 00（TM00）的定时器阵列单元的间隔定时器功能进行设置。

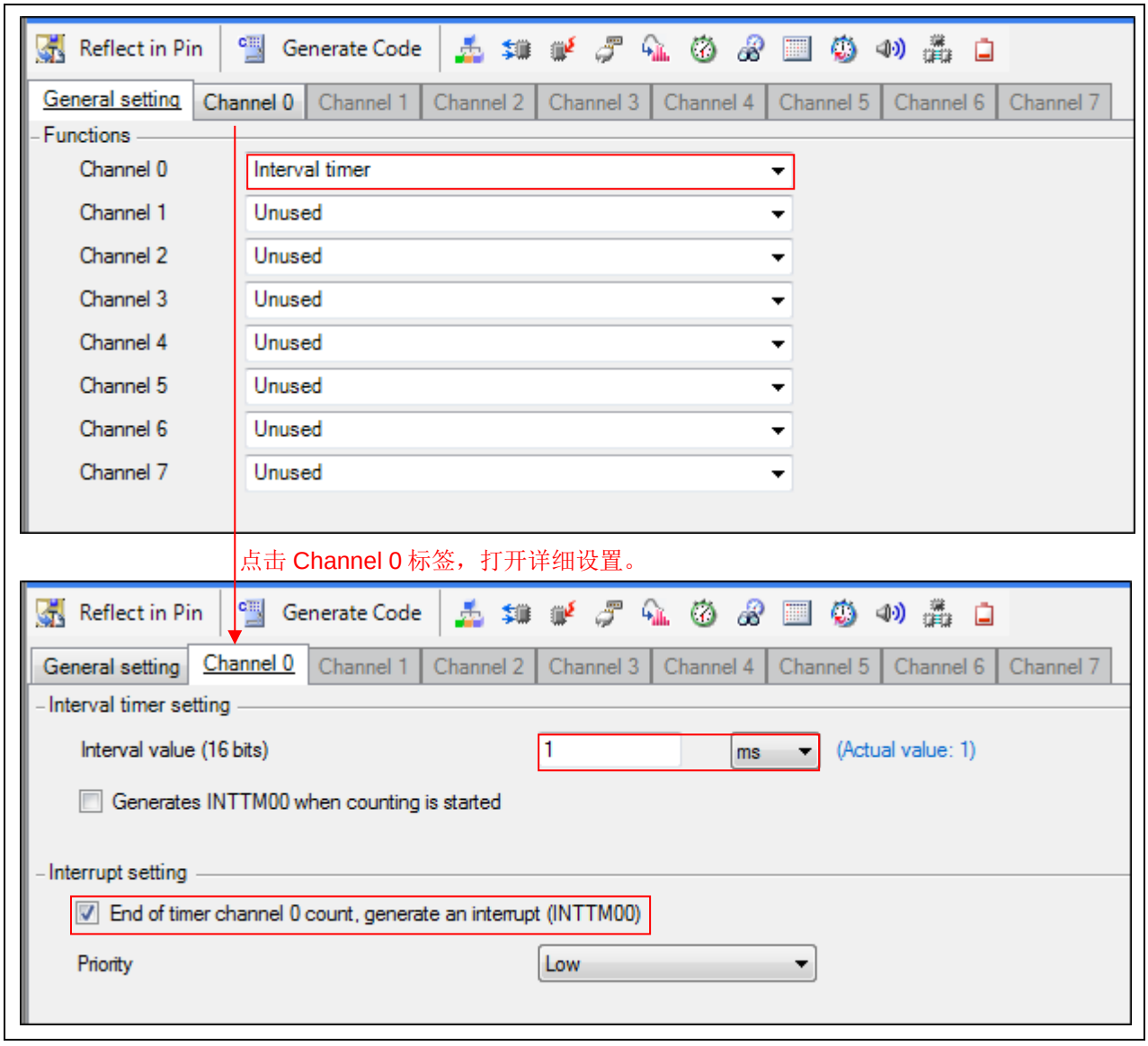


图 4.25 代码生成工具的设置界面（定时器阵列单元）

（3） 设置“端口”、“看门狗定时器”以及“电压检测电路”。
（4） 点击“Generate Code”以生成文件。

4.3.3 程序的添加

本节介绍在代码生成的程序上添加符号定义、主函数和 A/D 转换器处理的内容。关于其他程序（如时钟设置、A/D 转换器的设置等），请使用由代码生成的程序。

- 变量
  在 r_main.c 和 r_cg_adc_user.c 上添加变量。

78K0 的程序

```
100  /*=============================================================================
101  ;       Global variables and functions
102  ;=============================================================================*/
103  static  unsigned        char    g_ucAdCnt;
104  static  unsigned        char    g_ucAdCh;
105  static  unsigned        short   g_ucAdData[4];
```

RL78/G13 的 r_main.c 文件

```
47  /***********************************************************************
48  Global variables and functions
49  ***********************************************************************/
50  /* Start user code for global. Do not edit comment generated here */
51  extern unsigned    char    g_ucAdCnt;
52
53  /* End user code. Do not edit comment generated here */
```

RL78/G13 的 r_cg_adc_user.c 文件

```
45  /***********************************************************************
46  Global variables and functions
47  ***********************************************************************/
48  /* Start user code for global. Do not edit comment generated here */
49  unsigned char g_ucAdCnt = 0;
50  unsigned char g_ucAdCh = 0;
51  unsigned short g_ucAdData[4] = {0,0,0,0};
52  /* End user code. Do not edit comment generated here */
```

图 4.26 变量的置换

• 主函数

使用了 RL78/G13 的代码生成工具后，在执行 main 函数之前会先执行 R_Systeminit 函数。通过 R_Systeminit 函数对定时器和 A/D 转换器进行初始设置。通过 R_TAU0_Channel0_Start 函数启动定时器阵列单元的计数运行。

78K0 的程序

```
373     void main(void){
374             fn_InitAd();
375             fn_InitTimer();
376             g_ucAdCnt       =       0;      /* Initialization of variables */
377
378     //[CcnvCA78K0]  EI();
379             __EI();
380
381             while (1){
382     //[CcnvCA78K0]          NOP();
383                     __nop();
384             }
385     }
```

RL78/G13 的 r_main.c 文件　　变更

```
62      void main(void)
63      {
64          R_MAIN_UserInit();
65          /* Start user code. Do not edit comment generated here */
66          R_TAU0_Channel0_Start();
67          g_ucAdCnt = 0;                  /* Initialization of variables */
68
69          while (1)
70          {
71              __nop();
72          }
73          /* End user code. Do not edit comment generated here */
74      }
```

图 4.27 主函数的置换

• 中断函数（定时器阵列单元）
将中断处理添加到 r_cg_timer_user.c 中的 r_tau0_channel0_interrupt() 。

78K0 的程序

```
439     void fn_intTimerInterval(void)
440     {
441             ADCS    =       1;                      /* A/D conversion start */
442     }
```

RL78/G13 的 r_cg_timer_user.c 文件

```
57      static void __near r_tau0_channel0_interrupt(void)
58      {
59          /* Start user code. Do not edit comment generated here */
60          ADCS = 1;                           /* A/D conversion start */
61          /* End user code. Do not edit comment generated here */
62      }
```

图 4.28 中断函数（定时器阵列单元）的置换

• 中断函数（A/D 转换器）
将中断处理添加到 r_cg_adc_user.c 中的 r_adc_interrupt() 。

78K0 的程序

```
393     #pragma interrupt fn_intAdConverter
394     void fn_intAdConverter(void)
395     {
396             if(g_ucAdCnt == 0){
397                     for(g_ucAdCh = 0; g_ucAdCh < 4; g_ucAdCh++){
398                             g_ucAdData[g_ucAdCh] = 0;
399                                                             /* Clear the A / D conve
400                     }
401                     g_ucAdCh = 0;
402             }
403
404             g_ucAdData[g_ucAdCh] += ADCR >> 6;
405                                                             /* Remove the lower 6 bi
406             g_ucAdCh++;                                     /* Increments the A/D co
407
408             g_ucAdCh &= 0b00000011;
409             ADS = g_ucAdCh;
410                                                             /* Change the analog inp
411
412             g_ucAdCnt++;                                    /* Increments the A/D co
413
414             if(g_ucAdCnt >= 32){
415                     for(g_ucAdCh = 0; g_ucAdCh < 4; g_ucAdCh++){
416
417                             g_ucAdCnt = g_ucAdCnt << 1;
418                             g_ucAdData[g_ucAdCh] = g_ucAdData[g_ucAdCh] >> 3;
419                             if(g_ucAdData[g_ucAdCh] >= 612){
420                                     g_ucAdCnt &= 0b11111110;
421                             }
422                             else{                           /* ANI pin is less than
423                                     g_ucAdCnt |= 0b00000001;
424                             }
425                     }
426                     g_ucAdCnt &= 0b11111111;
427                     P1 = g_ucAdCnt;
428                     g_ucAdCnt = 0;                  /* Clear the A / D conversion co
429             }
430             ADCS = 0;                                       /* A/D conversion is sto
431             ADIF = 0;
432     }
```

图 4.29 中断函数（A/D 转换器）的置换（1/2）

RL78/G13 的 r_adc_user.c 文件

```
 60      static void __near r_adc_interrupt(void)
 61      {
 62          /* Start user code. Do not edit comment generated here */
 63          if(g_ucAdCnt == 0){
 64              for(g_ucAdCh = 0; g_ucAdCh < 4; g_ucAdCh++){
 65                  g_ucAdData[g_ucAdCh] = 0;
 66                                          /* Clear the A / D conversion result buffe
 67              }
 68              g_ucAdCh = 0;
 69          }
 70
 71          g_ucAdData[g_ucAdCh] += ADCR >> 6;
 72                                          /* Remove the lower 6 bits of the A / D cc
 73          g_ucAdCh++;                     /* Increments the A/D conversion channel *
 74
 75          g_ucAdCh &= 0b00000011;
 76          ADS = g_ucAdCh;
 77                                          /* Change the analog input channel*/
 78
 79          g_ucAdCnt++;                    /* Increments the A/D conversion counter *
 80
 81          if(g_ucAdCnt >= 32){
 82              for(g_ucAdCh = 0; g_ucAdCh < 4; g_ucAdCh++){
 83
 84                  g_ucAdCnt = g_ucAdCnt << 1;
 85                  g_ucAdData[g_ucAdCh] = g_ucAdData[g_ucAdCh] >> 3;    /* Average the
 86                  if(g_ucAdData[g_ucAdCh]    >= 612){                  /* ANI pin mor
 87                      g_ucAdCnt &= 0b11111110;
 88                  }
 89                  else{                   /* ANI pin is less than 3V */
 90                      g_ucAdCnt |= 0b00000001;
 91                  }
 92              }
 93              g_ucAdCnt &= 0b11111111;
 94              P1 = g_ucAdCnt;
 95              g_ucAdCnt = 0;              /* Clear the A / D conversion counter */
 96          }
 97          ADCS = 0;                       /* A/D conversion is stopped */
 98          ADIF = 0;
 99          /* End user code. Do not edit comment generated here */
100      }
```

图 4.30 中断函数（A/D 转换器）的置换（2/2）

4.3.4 置换后的参考例程

请从瑞萨电子官网下载“an-r01an3471cc0100-rl78-migrate.zip”参考例程。

“workspace”文件夹中的“rl78g13_migrate_ad”为“78K0/Kx1,78K0/Kx1+サンプル・プログラム A/D コンバータ（ZUD-CC-10-0016）”中置换了程序的参考例程。

4.4 参考例程的动作确认条件

置换后的参考例程是在下面的条件下进行动作确认的。

表 4.1 动作确认条件

项目	内容
所用微控制器	RL78/G13 （R5F100LEA）
集成开发环境（CS+）	CS+ for CC V4.00.00 （瑞萨电子开发）
C 编译器（CS+）	CC-RL V1.02.00 （瑞萨电子开发）
集成开发环境（e^{2}studio）	e^{2}studio V4.3.1.001 （瑞萨电子开发）
C 编译器（e^{2}studio）	CC-RL V1.02.00 （瑞萨电子开发）
所用电路板	RL78/G13 目标板（QB-R5F100LE-TB）（瑞萨电子开发）

5. 参考例程

参考例程请从瑞萨电子网页上取得。

6. 参考文献

RL78 family User’ s Manual: Software (R01US0015E）
RL78 CC-RL Compiler User’s Manual (R20UT3123E)
CS+ Code Generator Integrated Development Environment User’s Manual: Peripheral Function Operation (R20UT3104E)
CcnvCA78K0 C Source Converter User's Manual (R20UT3684E)

78K0/Kx1,78K0/Kx1+ シリアル通信プログラム集
78K0/Kx2 サンプル・プログラム インターバル・タイマ編（U19031JJ2V0AN00）
78K0/Kx2 サンプル・プログラム A/D コンバータ（ZUD-CC-10-0016）

（最新版本请从瑞萨电子网页上取得）

公司主页和咨询窗口

瑞萨电子主页

- http://www.renesas.com/zh-cn/

咨询

- https://www.renesas.com/zh-cn/support/contact.html

修订记录

Rev.	发行日	修订内容	
		页	要点
1.00	2016.12	—	初版发行

产品使用时的注意事项

本文对适用于单片机所有产品的 “使用时的注意事项”进行说明。有关个别的使用时的注意事项请参照正文。此外，如果在记载上有与本手册的正文有差异之处，请以正文为准。

1. 未使用的引脚的处理
   【注意】将未使用的引脚按照正文的 “未使用引脚的处理”进行处理。
   CMOS产品的输入引脚的阻抗一般为高阻抗。如果在开路的状态下运行未使用的引脚，由于感应现象，外加LSI周围的噪声，在LSI内部产生穿透电流，有可能被误认为是输入信号而引起误动作。
   未使用的引脚，请按照正文的 “未使用引脚的处理”中的指示进行处理。

2. 通电时的处理
   【注意】通电时产品处于不定状态。
   通电时， LSI内部电路处于不确定状态，寄存器的设定和各引脚的状态不定。通过外部复位引脚对产品进行复位时，从通电到复位有效之前的期间，不能保证引脚的状态。
   同样，使用内部上电复位功能对产品进行复位时，从通电到达到复位产生的一定电压的期间，不能保证引脚的状态。

3. 禁止存取保留地址 （保留区）
   【注意】禁止存取保留地址 （保留区）
   在地址区域中，有被分配将来用作功能扩展的保留地址 （保留区）。因为无法保证存取这些地址时的运行，所以不能对保留地址 （保留区）进行存取。

4. 关于时钟
   【注意】复位时，请在时钟稳定后解除复位。
   在程序运行中切换时钟时，请在要切换成的时钟稳定之后进行。复位时，在通过使用外部振荡器（或者外部振荡电路）的时钟开始运行的系统中，必须在时钟充分稳定后解除复位。另外，在程序运行中，切换成使用外部振荡器 （或者外部振荡电路）的时钟时，在要切换成的时钟充分稳定后再进行切换。

5. 关于产品间的差异
   【注意】在变更不同型号的产品时，请对每一个产品型号进行系统评价测试。
   即使是同一个群的单片机，如果产品型号不同，由于内部ROM、版本模式等不同，在电特性范围内有时特性值、动作容限、噪声耐量、噪声辐射量等也不同。因此，在变更不认同型号的产品时，请对每一个型号的产品进行系统评价测试。

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.

以下"注意事项"为从英语原稿翻译的中文译文，仅作为参考译文，英文版的"Notice"具有正式效力。

注意事项

1. 本文档中所记载的关于电路、软件和其他相关信息仅用于说明半导体产品的操作和应用实例。用户如在设备设计中应用本文档中的电路、软件和相关信息，请自行负责。对于用户或第三方因使用上述电路、软件或信息而遭受的任何损失，瑞萨电子不承担任何责任。
2. 在准备本文档所记载的信息的过程中，瑞萨电子已尽量做到合理注意，但是，瑞萨电子并不保证这些信息都是准确无误的。用户因本文档中所记载的信息的错误或遗漏而遭受的任何损失，瑞萨电子不承担任何责任。
3. 对于因使用本文档中的瑞萨电子产品或技术信息而造成的侵权行为或因此而侵犯第三方的专利、版权或其他知识产权的行为，瑞萨电子不承担任何责任。本文档所记载的内容不应视为对瑞萨电子或其他人所有的专利、版权或其他知识产权作出任何明示、默示或其它方式的许可及授权。
4. 用户不得更改、修改、复制或者以其他方式部分或全部地非法使用瑞萨电子的任何产品。对于用户或第三方因上述更改、修改、复制或以其他方式非法使用瑞萨电子产品的行为而遭受的任何损失，瑞萨电子不承担任何责任。
5. 瑞萨电子产品根据其质量等级分为两个等级："标准等级"和"高质量等级"。每种瑞萨电子产品的推荐用途均取决于产品的质量等级，如下所示：
   标准等级： 计算机、办公设备、通讯设备、测试和测量设备、视听设备、家用电器、机械工具、个人电子设备以及工业机器人等。
   高质量等级： 运输设备（汽车、火车、轮船等）、交通控制系统、防灾系统、预防犯罪系统以及安全设备等。
   瑞萨电子产品无意用于且未被授权用于可能对人类生命造成直接威胁的产品或系统及可能造成人身伤害的产品或系统（人工生命维持装置或系统、植埋于体内的装置等）中，或者可能造成重大财产损失的产品或系统（核反应堆控制系统、军用设备等）中。在将每种瑞萨电子产品用于某种特定应用之前，用户应先确认其质量等级。不得将瑞萨电子产品用于超出其设计用途之外的任何应用。对于用户或第三方因将瑞萨电子产品用于其设计用途之外而遭受的任何损害或损失，瑞萨电子不承担任何责任。
6 使用本文档中记载的瑞萨电子产品时，应在瑞萨电子指定的范围内，特别是在最大额定值、电源工作电压范围、移动电源电压范围、热辐射特性、安装条件以及其他产品特性的范围内使用。对于在上述指定范围之外使用瑞萨电子产品而产生的故障或损失，瑞萨电子不承担任何责任。
7. 虽然瑞萨电子一直致力于提高瑞萨电子产品的质量和可靠性，但是，半导体产品有其自身的具体特性，如一定的故障发生率以及在某些使用条件下会发生故障等。此外，瑞萨电子产品均未进行防辐射设计。所以请采取安全保护措施，以避免当瑞萨电子产品在发生故障而造成火灾时导致人身事故、伤害或损害的事故。例如进行软硬件安全设计（包括但不限于冗余设计、防火控制以及故障预防等）、适当的老化处理或其他适当的措施等。由于难于对微机软件单独进行评估，所以请用户自行对最终产品或系统进行安全评估。
8. 关于环境保护方面的详细内容，例如每种瑞萨电子产品的环境兼容性等，请与瑞萨电子的营业部门联系。使用瑞萨电子产品时，请遵守对管制物质的使用或含量进行管理的所有相应法律法规（包括但不限于《欧盟RoHS指令》）。对于因用户未遵守相应法律法规而导致的损害或损失，瑞萨电子不承担任何责任。
9. 不可将瑞萨电子产品和技术用于或者嵌入日本国内或海外相应的法律法规所禁止生产、使用及销售的任何产品或系统中。也不可将本文档中记载的瑞萨电子产品或技术用于与军事应用或者军事用途有关的任何目的（如大规模杀伤性武器的开发等）。在将本文档中记载的瑞萨电子产品或技术进行出口时，应当遵守相应的出口管制法律法规，并按照上述法律法规所规定的程序进行。
10. 向第三方分销或处分产品或者以其他方式将产品置于第三方控制之下的瑞萨电子产品买方或分销商，有责任事先向上述第三方通知本文档规定的内容和条件；对于用户或第三方因非法使用瑞萨电子产品而遭受的任何损失，瑞萨电子不承担任何责任。
11. 在事先未得到瑞萨电子书面认可的情况下，不得以任何形式部分或全部转载或复制本文档。
12. 如果对本文档所记载的信息或瑞萨电子产品有任何疑问，或者用户有任何其他疑问，请向瑞萨电子的营业部门咨询。

(注1) 瑞萨电子：在本文档中指瑞萨电子株式会社及其控股子公司。
(注2) 瑞萨电子产品：指瑞萨电子开发或生产的任何产品。

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, HAL Ⅱ 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