[Notes]

C Compiler Package for RL78 Family

R20TS0072EJ0100
Rev.1.00
Sep. 01, 2016

Outline

When using the CC-RL C Compiler Package for the RL78 Family, take note of the problem described in this note regarding the following point.

1. Programs which include loops that should be iterated more than once (CCRL#012)

   Note: The number which follows the description of the precautionary note is an identifying number for the precaution.

1. Programs which Include Loops that should be Iterated More than Once (CCRL#012)

1.1 Applicable Product

CC-RL V1.00.00 to V1.03.00

1.2 Details

When certain optimization options are specified, code might be output which only loops once when the program code in the C source file is intended to loop two or more times.

1.3 Conditions

This problem may arise if the following conditions are all met:

(1) The optimization option -Onothing is not specified.

(2) The conditional expression for the loop has the form: *formula including loop control variable* > *formula including the decisive value*.
"Formula" above also includes expressions such as the loop control variable or decisive value alone.

(3) The loop control variable in condition (2) is a variable of a signed type, and does not have the volatile attribute.

(4) The decisive value in condition (2) is neither a constant nor has the volatile attribute.

(5) The value in condition (4) is the minimum value of the variable in condition (3), and will not change during the loop.

```
1:  unsigned long count;
2:  signed char var1, var2;                    /* Conditions (3),(4) */
3:  void func(void)
4:  {
5:    count = 0x00000000;
6:    var2 = -128;                                    /* Condition (5) */
7:    for( var1 = 127; var1 > var2; var1--){count++;} /* Condition (2) */
8:    if(0x000000FF != count){funcsub();}
9:    …
10: }
```

If the 7th line following this code is for looping 255 times, and the variable “count” is set to 255, optimization will instead cause it to become a single loop.

That is, the variable “count” will become 1 on the 8th line.

1.4 Workarounds

To avoid this problem, take any of the following steps.

(1) Change the optimization option so as to not meet the condition for failure.

(2) Change the form of the loop control expression so that it does not meet the condition.

```
for( var1 = 127; var2 < var1; var1--){count++;}
```

```
for( var1 = 127; var1 >= var2+1; var1--){count++;}
```

(3) Change the loop control variable to an unsigned type of the same size or give it the volatile attribute.
In the case of changing the loop control variable to an unsigned type, both the initial value and the decisive value should be changed.

```
volatile signed char var1;
```

(4) Change the decisive value to a constant or a volatile variable.

```
volatile signed char var2;
```

1.5 Schedule for Fixing the Problem

This problem will be fixed in the next version.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 01, 2016	-	First edition issued

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

Renesas Electronics Corporation

■Inquiry

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