R20TS0128EJ0100
Rev.1.00
Jan. 16, 2017

[Notes]
C Compiler Package for RL78 Family

Outline

When using the CC-RL C Compiler Package for the RL78 family, note the following point.

1. The loop that has the operation expression of which result is decremented by one (CCRL#014)

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

1. The Loop That Has the Operation Expression of Which Result is Decremented by One (CCRL#014)

1.1 Applicable Product

CC-RL V1.03.00

1.2 Details

When an operation expression of which result is decremented by one every loop repetition is described in a loop, the repeat count of the loop might be improperly short by one.

1.3 Conditions

This problem may arise in the loop of (2) if all of the following conditions are met.

(1) The optimization level other than -Onothing is specified, or the optimization level is not specified.

(2) A loop which satisfies all the conditions from (2-1) to (2-5) exists.

(2-1) None of break statements, goto statements, and return statements are contained.

(2-2) The loop control variable(*1) satisfies both of the (2-2-a) and (2-2-b) conditions.

(2-2-a) The initial value and end decisive value are a constant(*2).

(2-2-b) The incremental value is 1(*3).

(2-3) The loop contains a loop induction variable(*4) and an addition or subtraction expression of which the operation result is decremented by one every loop repetition.

(2-4) The initial operation result of (2-3) results in "(loop repeat count) -1".

(2-5) The loop induction variable of (2-3) satisfies both of the (2-5-a) and (2-5-b) conditions.

(2-5-a) The initial value is a constant(*2).

(2-5-b) The incremental value is 1 or -1(*5).

*1: This is a variable to control the end condition of a loop.

*2: Includes cases in which such a value is statically known to be a constant.

*3: Includes cases in which such a value is statically known to be 1.

*4: This is a variable which increments or decrements by a fixed value every loop.

A loop control variable is absolutely a loop induction variable.

*5: Includes cases in which such a value is statically known to be 1 or -1.

1.4 Example

An occurrence example of 1.3 (2) is shown below. Characters in red are the parts that correspond to the conditions.

```
1:  int result;
2:  int sub(void)
3:  {
4:    volatile int v1, v2;
5:    int i;                    // Variable i is a loop control variable and
                                //also a loop induction variable
6:    for (i=1; i<=8; i++) {    // Conditions (2-2) and (2-5)
7:      if (i != 8) {
8:        v1 = 8-i;             // Conditions (2-3) and (2-4)
9:      }
10:     v2 = i;
11:   }
12:   return v2;
13: }
14:
15: void main(void)
16: {
17:   result = sub();
18: }
```

- Lines 6 to 11 correspond to Condition (2-1) because these lines contain none of break statements, goto statements, and return statements in the for loop statement in which a loop occurs eight times.
- Line 6 corresponds to Condition (2-2) because the loop control variable i has an initial value 1, end decisive value a constant of 8, and incremental value 1.
- The loop control variable i is also a loop induction variable because the loop control variable increments by 1 every loop. Additionally, line 8 in the loop contains the loop induction variable i and an operation expression "v1 = 8-i;" of which the result is decremented by 1 every loop repetition, and consequently Condition (2-3) is applied.
- Line 8 corresponds to Condition (2-4) because the initial operation result of the operation expression is 7.
- The initial value of the loop induction variable i is 1 and the incremental value is 1, and consequently Condition (2-5) is applied.

As described above, the for loop statement in lines 6 to 11 corresponds to 1.3 (2). Therefore, if the for loop statement also corresponds to 1.3 (1), it should loop eight times, but actually loops only seven times. The value of the result variable in line 17 also becomes 7 rather than 8.

1.5 Workarounds

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

(1) Specify the optimization level as -Onothing.

(2) Replace the end decisive value with a variable qualified by volatile.

```
5:   int i;
6:   volatile int j = 8
7:   for (i=1; i<=j; i++) {
```

1.6 Schedule for Fixing the Problem

This problem will be fixed in V1.04.00. (This revision will be available from January 20.)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jan. 16, 2017	-	First edition issued

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

Renesas Electronics Corporation

■Inquiry

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