RENESAS Tool News

RENESAS TOOL NEWS on June 16, 2004: RSO-M3T-CC32R-040616D

A Note on Using C-Compiler Packages M3T-CC32R V.3.10 Release 1 and V.3.20 Release 1

Please take note of the following problem in using the M3T-CC32R C-compiler packages for the M32R family MCUs:

- On using the result of an AND, OR, or XOR operation between two objects of type unsigned char after casting its type to unsigned char

---

1. **Versions Concerned**
   M3T-CC32R V.3.10 Release 1 and V.3.20 Release 1

2. **Description**
   Level 1 optimization may make the result of an AND, OR, or XOR operation incorrect.

   2.1 Conditions
   This problem occurs if the following five conditions are satisfied. However, depending on the register to which the processing data of function that contains the statements satisfying all the conditions is assigned, this problem may not arise.

   (1) Any of the optimizing options -O7, -O5, -O3, and -O1, or -Ospace only or -Otime only is selected.

   (2) Two static objects of type unsigned char exist.

   (3) An AND, OR, or XOR operation between the two objects in (2) is performed.

   (4) The result of the operation in (3) is cast to unsigned char in its type and then used as any of the following:

   (a) The right operand of an assignment expression to an auto variable

   (b) An object in an operational expression or a logical

expression

(c) An argument to a function

(d) A return value

(e) The subscript of an array

(5) The result of the operation in (3) or its value whose type was cast to unsigned char is not assigned to a static variable of type unsigned char between the operation in (3) and the conversion in (4) above.

2.2 Example

Note that in this example, the problem may not arise depending on the register to which the function containing the statements shown in this example is assigned, and which register to be used is affected by the program statements before and after the exemplified ones.

```
-----------------------------------------------------------------
struct strtag {
    unsigned char a;
    unsigned char b;          /* Condition (2) */
    unsigned char c;          /* Condition (2) */
} *ptr;

int func(void)
{
   unsigned char flag;
   unsigned long result = 0;

. . . . . .

   flag =                    /* Condition (4)(a) */
      (ptr->b | ptr->c);     /* Conditions (3) and (5) */

   if (flag) {
      result = 1;
   }

. . . . . .

   return result;
}
```

---

3. **Workaround**

This problem can be circumvented in any of the following ways:

(1) Upgrade your M3T-CC32R to the latest version, which can be downloaded from **HERE**.

(2) Don't use any of the level 1 optimizations
If you are using -O7, -O5, -O3, or -O1, change it to any of these, -O6, -O4, -O2, and -O0.
If you want to use -Ospace or -Otime, use -O6, -O4, -O2, or -O0 at the same time.

(3) Change to a static variable the variable of type unsigned char to which the result of an AND, OR, or XOR operation is assigned (only when Condition (4)-(a) satisfied).

Modification of the example in 2.2

```
------------------------------------------------------------------
  static unsigned char flag;     /* Declared to be static */
------------------------------------------------------------------
```

(4) Don't cast the type of the result of an AND, OR, or XOR operation to unsigned char.

Modification of the example in 2.2

```
------------------------------------------------------------------
struct strtag {
    unsigned char a;
    unsigned char c;
    unsigned char b;
} *ptr;

int func(void)
{
  int flag;                /* Declared to be int */
  unsigned long result = 0;

. . . . . .

  flag =                   /* Result of OR operation not
    (ptr->b | ptr->c);         cast to unsigned char since
                             flag is of type int */
  if (flag) {
    result = 1;
  }

. . . . . .
```

```
    return result;
}
------------------------------------------------------------------
```

4. Schedule of Fixing the Problem

This problem has already been fixed in the M3T-CC32R V.4.00 and later.
