RENESAS Tool News

RENESAS TOOL NEWS on January 16, 2008: 080116/tn1

A Note on Using the C/C++ Compiler Package for the M32R Family of MCUs

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

- With using an assignment expression whose left term is of a shorter data type than the right term and after which is placed an expression that is equivalent to the right term of the assignment expression

1. Product and Versions Concerned

C/C++ compiler package--M3T-CC32R--for the M32R family
V.3.20 Release 1 through V.5.00 Release 00

2. Description

Consider the case where an assignment expression whose left term is of a shorter data type than the right term exists, and after it is placed another expression that contains an expression equivalent to the right term of the assignment expression.
When the second expression is evaluated, such a value is used that is obtained by casting the operation result of the right term of the assignment expression to the shorter data type; not the above operation result itself.

3. Conditions

This problem may occur if the following conditions are all satisfied:

(1) The optimizing option used in compilation meets either (a) or (b) shown below.
(a) Option -O, -O2, -O3, -O6, or -O7 is used.
(b) Option -Ospace or -Otime is used with -O0, -O1, -O4, and -O5 not used.

(2) An expression exists which contains more than one variable of an integral type and whose operation result is also of an integral type.

The expression can contain only one variable of an integral type.[h1]

(3) An assignment expression exists in which the operation result of the expression in (2) is assigned to an auto variable of a shorter integral type than the operation result in (2).

(4) An expression equivalent to the expression in (2) is used as a part of another expression evaluated after the assignment expression in (3).

(5) Between the assignment in (3) and the use of an expression in (4) exists no possibility of changing the values of any variables contained in the expression in (2).

(6) The auto variable in (3) is saved on the stack; not on a register.

(7) The operation result of the expression in (2) is saved on a register. This is valid when the expression contains only one variable.

NOTICE:

Whether the objects in Conditions (6) and (7) are saved on the stack or a register depends on your compiler and is not determined by the source code. It should be checked using the code generated after compilation.

Example:

In the operation of expression val_ulong >> 8 below is used the value of val_ulong after assigned to val_uchar. So the operation result of this expression becomes incorrect. However, this problem may not arise depending on the descriptions of the program lines that are omitted below.

```
----------------------------------------------------------------------
unsigned long func_ulong(unsigned short);
void func(void)
{
   unsigned long  val_ulong;
   unsigned char  val_uchar;
   unsigned int   val_another;
   unsigned short val_ushort;
   /*
 . . . . . . . . . . . . . . . . . . . . . . .
    */
   val_ulong   = func_ulong(val_ushort);
   val_uchar   = val_ulong;       /* Conditions (2) and (3) */
   val_another = val_ulong >> 8;  /* Conditions (4) and (5) */
   /*
 . . . . . . . . . . . . . . . . . . . . . . .
```

```
    */
}
-----------------------------------------------------------------------
```

4. Workarounds

Avoid this problem in either of the following ways:

(1) Suppress the optimization in Level 2.
If you are using option -O, -O2, -O3, -O6, or -O7, replace it with -O0, -O1, -O4, or -O5.
If you are using -Ospace or -Otime, use -O0, -O1, -O4, or -O5 at the same time.

(2) Exchange the expressions in Conditions (3) and (4) in their order.

Example modified:

```
---------------------------------------------------------------------
unsigned long func_ulong(unsigned short);
void func(void)
{
 . . . . . . . . . . . . . . . . . . . . . .
    val_ulong   = func_ulong(val_ushort);
    val_another = val_ulong >> 8;  /* Conditions (4) and (5) */
    val_uchar   = val_ulong;       /* Conditions (2) and (3) */
 . . . . . . . . . . . . . . . . . . . . . .
}
---------------------------------------------------------------------
```

5. Schedule of Fixing the Problem

This problem has already been fixed in the V.5.01 Release 00 package. So use the latest version of the package. Note that the package V.4.30 Release 00 or earlier cannot be updated to the latest version. If this is the case, please purchase the latest version.

[Disclaimer]

The past news contents have been based on information at the time of publication. Now changed or invalid information may be included. The URLs in the Tool News also may be subject to change or become invalid without prior notice.