RENESAS Tool News

RENESAS TOOL NEWS on March 16, 2007: 070316/tn5

A Note on Using the C Compiler Packages --M3T-NC308WA and M3T-NC30WA-- for the M16C MCU Family

Please take note of the problem described below in using the C compiler packages--M3T-NC308WA and M3T-NC30WA--for the M16C MCU family.

1. Products and Versions Concerned

(1) The C compiler package--M3T-NC308WA--for the M32C series*1
V.5.40 Release 00 through V.5.41 Release 01
(2) The C compiler package--M3T-NC30WA--for the M16C series**
V.5.40 Release 00 through V.5.43 Release 00

*Generic name of the M32C/80, and M16C/80 series.
**Generic name of the M16C/60, /30, /20, /10, /Tiny and R8C/Tiny series.

2. Problem

When arguments are passed to functions via the stack, they may incorrectly be referenced, or the programs containing such functions may run away.

3. Conditions

This problem occurs if the following conditions are all satisfied:
(1) Compile option -OR or -OR_MAX (-ORM) is selected.
(2) Compile option -Ono_asmopt (-ONA) is not selected.
(3) In a file, a function to which its argument is passed via the stack (function A) is called from other two or more functions (for example, functions B and C below).
(4) At least either function B or C satisfies any of the following conditions (under any of these conditions, the "exitd" instruction is generated, which terminates the function):
- The function takes an argument passed via the stack.

- The function is using an auto variable (corresponding
  the temporary area generated by the compiler).
- The -genter option is used at compilation.

(5) After the program has returned from function A, no code for adjusting the stack exists in it until an exitd instruction is executed.

(6) After the argument of function A has been saved, the same code exists in functions B and C until function A is called by a jsr instruction.

(7) Function A takes an argument passed via the stack.

Example:

```
-------------------------------------------------
void  func1( void )  //  Function B
{
   . . . . . .
   comm_func ( int i, int *p, int *q );    // Function A called
   . . . . . .
}
void  func2 ( void )  // Function C
{
   . . . . . .
   comm_func ( int i, int *p, int *q );    // Function A called
   . . . . . .
}
-------------------------------------------------
```

Explanations:

In the above example, the code shown below is generated at the places where function A is called from functions B and C.

```
-------------------------------------------------
   . . . . . .
        push.w    _q
        push.w    _p
        mov.w     _i,R1          // D
        jsr       $comm_func     // E
        exitd
   . . . . . .
-------------------------------------------------
```

Since compile option -OR or -OR_MAX is selected under Condition (1), it converts the lines D and E above to the common function _aopt_xxx

as shown below.

As a result, the arguments in the lines F and G below that are saved on the stack will be stored at an address to which the addresses consumed by the stack to call _aopt_xxx are accumulated.
On the other hand, because the code generated from $comm_func remains the same as before the lines D and E are converted to _aopt_xxx, the code will reference an address different from the one where the arguments of function A has been saved.

```
-------------------------------------------------
  . . . . . . . . . . .
       push.w   _q       // F
       push.w   _p       // G
       jsr      _aopt_xxx
       exitd
  . . . . . . . . . . .

    _aopt_xxx:
       mov.w    _i,R1
       jsr      $comm_func
       exitd
-------------------------------------------------
```

4. Workarounds

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

(1) If function A is called indirectly, select the -ONA compile option.

(2) If function A is a special page function, select the -ONA option or place a dummy asm function immediately after a call to function A, which is shown in Example below.

(3) If function A does not meet the assumptions in (1) and (2) above, select the -ONA or -OSA option, or place a dummy asm function the same manner as described in (2)

Example:

```
-------------------------------------------------
void  func1(  void  )
{
   . . . . . . . . . . .
   comm_func ( int i, int *p, int *q );
   asm();     // asm function placed
   . . . . . . . . . . .
}
```

---

5. Schedule of Fixing the Problem

We plan to fix this problem in the next release of the products.
