RENESAS TOOL NEWS on January 16, 2004: RSO-M3T-NC308WA_2-040116D

A Note on Using C Compiler M3T-NC308WA

Please take note of the following problem in using the M3T-NC308WA C-compiler (with an assembler and integrated development environment) for the M32C/80 and M16C/80 series MCUs:

- On casting an integer-type value to a pointer-type one

1. **Versions Concerned**
   M3T-NC308WA V.2.00 Release 1 through V.5.10 Release 1

2. **Description**
   When an integer-type value representing an address outside of the address space is cast to a pointer-type one, the comparison of it with another pointer-type value may not be performed correctly. The reason is as follows:

   In the M3T-NC308WA, a pointer-type value in the far area is 32-bit wide. However, it is assigned to a 24-bit address register in some versions of the products because the whole address space can be represented in 24 bits.

   If an integer-type value indicating the outside of the address space is cast to a pointer-type one in the far area, all the uppermost 8 bits of 32-bit value must be changed to zeros since the real size of the storage area that holds far pointers is a minimum of 24 bits.

   We are sorry that in the above case the uppermost 8 bits are not changed to zeros in the product concerned. Then, casting an integer-type value indicating the outside of the address space to a pointer-type value in the far area and making the comparison of it with another pointer-type value may not bring a correct result depending on whether the cast value to compare is assigned to a 24-bit address register or not.

   Note that in pointer-type values pointing to variables or functions on memory, their uppermost 8 bits definitely become zeros, so that no problems arise.

3. Conditions

This problem occurs if the following four conditions are satisfied:

(1) An integer-type variable or constant is cast to a pointer-type one.

(2) The comparison of the cast value in (1) with another pointer-type value is made.

(3) The original value in (1) is not within a range from 0x00000000 to 0x00ffffff.

(4) As a result of compilation, either of the two values in (2) is assigned to a 24-bit address register before or at the comparison.

4. Example

The following is an example under the condition that optimizing option -O is used in the M3T-NC308WA V.5.10 Release 1.
Since whether a pointer-type value is assigned to a 24-bit address register or not depends on the product's version, this problem does not occur if your product's version is not involved.

```
-----------------------------------------------------------------------
void func(char *p)
{
    if (p == 0) return;     /* Equality testing with 0: no problem */

    if (p == (int *)(-1)) { /* Condition (1): integer-type constant
                                             (-1) is cast */
                        /* Condition (2): cast value is tested */
                        /* Condition (3): -1 is not within a range
                                         from 0 to 0x00ffffff */
        return;
    }
    *p = 0;
}

void caller(void)
{
     func((int *)(-1));     /* Condition (1): integer-type constant
                                             (-1) is cast */
}
-----------------------------------------------------------------------
```

5. Workaround

When casting a value outside of a range from 0 to 0x00ffffff to a pointer-type value, perform an AND operation between the value before casting and a constant of 0x00ffffff to mask the uppermost 8 bits. The following is a case where a constant of type int is handled; this method is also applicable to a variable of type int.

```
---------------------------------------------------------------------------
  void func(char *p)
  {
     if (p == 0) return;

     if (p == (int *)(-1 & 0x00ffffff)) {   /* Uppermost 8 bits masked */
        return;
     }
     *p = 0;
  }

  void caller(void)
  {
      func((int *)(-1 & 0x00ffffff));       /* Uppermost 8 bits masked */

  }
  ---------------------------------------------------------------------------
```

6. Schedule of Fixing the Problem

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

[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.