RENESAS Tool News

A Note on Using C-Compiler Package M3T-NC30WA V.5.30 Release 02

Please take note of the following problem in using the M3T-NC30WA C-compiler package, which is used for the M16C/60, M16C/30, M16C/Tiny, M16C/20, M16C/10, and R8C/Tiny series of MCUs:

- On passing the address of an object qualified as const as an argument to a function

1. **Product and Version Concerned**
   M3T-NC30WA V.5.30 Release 02

2. **Description**
   Consider that a function takes the address of an object qualified as const as an argument. When such a function is called, an error message appears even if a correct type of argument is passed. However, code is properly generated in this case.

   2.1 Conditions

   This problem occurs if the following conditions are all satisfied:

   (1) A parameter to the function to be called is a pointer pointing to an object qualified as const.

   (2) The argument corresponding to the parameter in (1) is either of the following:
   - The result of operation where an address operator is applied to an object qualified as const
   - The name of an array qualified as const

   (3) The object and the array in (2) above are of the same type as the object pointed to by the pointer in (1).

   2.2 Examples

```
-----------------------------------------------------------------
    const int   ten = 10;         /* Condition (3) */
    const int   arr[2] = { 1, 2 };  /* Condition (3) */

    void subr(const int *);       /* Conditions (1) and (3) */

    void mainr(void)
    {
      subr(&ten);              /* Condition (2) */
      subr(arr);               /* Condition (2) */
    )
-----------------------------------------------------------------
```

Examples of messages

```
-----------------------------------------------------------------
[Warning(ccom):example.c,line 8] assignment from const pointer to
non-const pointer
  ===>    subr(&ten);
[Warning(ccom):example.c,line 9] assignment from const pointer to
non-const pointer
  ===>    subr(arr);
-----------------------------------------------------------------
```

3. **Workaround**

   Operate a cast operator on the argument to which a warning message is sent.

```
-----------------------------------------------------------------
void mainr(void)
{
  subr((const int *)&ten);     /* Cast operator operated */
  subr((const int *)arr);      /* Cast operator operated */
)
-----------------------------------------------------------------
```

4. **Schedule of Fixing the Problem**

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