Tool News

RENESAS TOOL NEWS on June 1, 2005: RSO-M3T-NC308WA_2-050601D

A Note on Using C-Compiler Package M3T-NC308WA

--On Referencing Array-Type Variables Qualified with "const"--

Please take note of the following problem in using the M3T-NC308WA C-compiler package (this C-compiler package is used for the M32C/90, M32C/80 and M16C/80 series of MCUs):

- On referencing array-type variables qualified with "const"

1. **Versions Concerned**
   M3T-NC308WA V.5.00 Release 1 through V.5.20 Release 02

2. **Description**
   Referencing an array-type variable declared with "typedef" and qualified with "const" delivers a warning message at linking, and the executable file created will not run properly.

   2.1 Conditions
   This problem occurs if the following conditions are all satisfied:
   (1) The type of an array-type variable is defined using the "typedef" specifier.
   (2) The array-type variable in (1) is not qualified to be "const", "near", or "far".
   (3) Then the array-type variable in (1) is declared outside a function, or declared inside a function using the "static" storage-class specifier.
   (4) The array-type variable declared in (3) is qualified using the "const" qualifier; not "near", or "far".
   (5) An element of the array-type variable in (3) is directly referenced using the array name.
   (6) None of those compile options, -fconst_not_ROM(-fCNR), -ffar_RAM(-fFRAM), and -fnear_ROM(-fNROM), is selected.

2.2 Example

```
-----------------------------------------------------------------
typedef char    arr_t[2];              /* Conditions (1) and (2) */
const arr_t     arr = { 0x11, 0x22 };  /* Conditions (3) and (4) */
char c;

void func(void)
{
    c = arr[1];                        /* Condition (5) */
}
-----------------------------------------------------------------
```

Warning message delivered at linking:

```
-----------------------------------------------------------------
a.c 7 Warning (ln308): a.r30 : 16-bits unsigned value is out of
     range 0 -- 65535. address='fe0124'
-----------------------------------------------------------------
```

3. **Workaround**

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

(1) Don't declare the array-type variable qualified to be "const" using the "typedef" specifier.

Example

```
-----------------------------------------------------------
const char      arr = { 0x11, 0x22 };
-----------------------------------------------------------
```

(2) Declare the array-type variable qualified to be "const" by explicitly qualifying it as "far".

Example

```
-----------------------------------------------------------
const arr_t far arr = { 0x11, 0x22 };
-----------------------------------------------------------
```

(3) Make the typedef declaration and the const type qualification of the array-type variable at the same time.

Example

```
---------------------------------------------------------------
typedef const char  c_arr_t[2];
c_arr_t           arr = { 0x11. 0x22 };
---------------------------------------------------------------
```

4. **Schedule of Fixing the Problem**

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