RENESAS Tool News

RENESAS TOOL NEWS on February 16, 2005: RSO-M3T-CC32R-050216D

A Note on Using C-Compiler Package M3T-CC32R V.4.30 Release 00

Please take note of the following problem in using C-compiler package M3T-CC32R V.4.30 Release 00, which is used for the M32R family of MCUs:

- On setting the address value of an object containing a structure, array or union using the #pragma ADDRESS directive

---

1. **Product Concerned**
   M3T-CC32R V.4.30 Release 00

2. **Description**
   Consider that a #pragma ADDRESS directive is used for defining the address of an object that is an array or structure; or a union that has members of type array or structure.
   If a member or element placed at any address except the beginning address of the object is accessed (read or written) or referenced by an address operator (&), incorrect code is generated using the address of the member or element placed at the beginning of the object.

   2.1 Conditions
   This problem occurs if the following conditions are satisfied:
   (1) Any of the following three objects is declared outside of a function:
   (a) A structure
   (b) An array
   (c) A union having members of type array or structure
   (2) A #pragma ADDRESS directive is used for defining the object in (1).
   (3) For a member or element placed at any address except the beginning address of the object, either of the following is performed:
   (a) Accessing (reading or writing)
   (b) Referencing by an address operator (&)

2.2 Examples

Source file: sample1.c

```
------------------------------------------------------------------------
#pragma ADDRESS data1 0x10000       /* Condition (2) */
struct stg1 {
   int a;
   int b;
} data1;                        /* Condition (1)-(a) */

void func1(void)
{
   data1.b = 3;                 /* Condition (3)-(a) */
}
------------------------------------------------------------------------
```

Source file: sample2.c

```
------------------------------------------------------------------------
#pragma ADDRESS data2 0x20000       /* Condition (2) */
short data2[10];                  /* Condition (1)-(b) */
short *d2;

void func2(void)
{
   d2 = &data2[3];               /* Condition (3)-(b) */
}
------------------------------------------------------------------------
```

Source file: sample3.c

```
------------------------------------------------------------------------
union utg3 {
   double a[10];
   struct {
      int b;
      double c[5];
   } d;
} data3;                        /* Condition (1)-(c) */

#pragma ADDRESS data3 0x30000       /* Condition (2) */

void func3(int i)
{
```

```
    data3.d.c[i] =              /* Condition (3)-(a) */
      data3.a[4];               /* Condition (3)-(a) */
}
-----------------------------------------------------------------------
```

3. **Workaround**

   Access the object using the address cast to a pointer. It is convenient to use macros shown in the examples below.

   Modification of sample1.c

```
-----------------------------------------------------------------------
#define data1 ( *(struct stg1*) 0x10000 )   /* Converted to macro */

struct stg1 {
   int a;
   int b;
} /* data1 */ ;       /* Only stg1 declared; data1 commented out */
void func1(void)
{
   data1.b = 3;      /* Replaced by macro */
}
-----------------------------------------------------------------------
```

   Modification of sample2.c

```
-----------------------------------------------------------------------
#define data2 ( (short *) 0x20000)    /* Converted to macro */

/* short data2[10]; */       /* data2 commented out */
short *d2;

void func2(void)



{


   d2 = &data2[3];          /* Replaced by macro */
}
-----------------------------------------------------------------------
```

   Modification of sample3.c

```
-----------------------------------------------------------------------
```

```
union utg3 {
   double a[10];
   struct {
      int b;
      double c[5];
   } d;
} /* data3 */ ;       /* Only utg3 declared; data3 commented out */

#define data3 ( *(union utg3*) 0x30000 )   /* Converted to macro */

void func3(int i)
{
   data3.d.c[i] =                /* Replaced by macro */
      data3.a[4];                /* Replaced by macro */
}
```
---

4. **Schedule of Fixing the Problem**
   We plan to fix this problem in our next release of the product.
