RENESAS Tool News

RENESAS TOOL NEWS on January 16, 2009: 090116/tn3

A Note on Using the C Compiler Package for the M16C Series, R8C Family of MCUs

Please take note of the following problem in using the C compiler package for the M16C series, R8C family of MCUs:

- With division operations between single-precision floating-point numbers

1. Product and Versions Concerned

The C compiler package for the M16C series, R8C family (M3T-NC30WA)
V.5.42 Release 00 through V.5.44 Release 00

2. Description

A division operation between single-precision floating-point numbers may result in the lower 16 bits of the quotient's significant part being incorrect. So the significant part will have a precision of about 3 digits in decimal numbers.
Usually, the significant parts of the quotients of such division operations have a precision of about 6 digits in decimal numbers.

2.1 Conditions

This problem occurs if the following conditions are all satisfied:
(1) The division operation is of type double or float.
(2) In a double-type division, option -fdouble_32(-fD32), -OR_MAX, or -OS_MAX is selected; in a float-type division, however, the problem is independent of selection of options.
(3) The code for calling the _f4div runtime routine is created after compilation.
(4) In the _f4div runtime routine, division instruction divu is executed twice for the upper and lower parts of the dividend to obtain the quotient, where the remainder in the first division is taken as the upper 16 bits of the dividend in the second division. At this time, the remainder in the first

division and the upper 16 bits of the divisor in the second division have the same value.

2.2 Example

C-language source file

```
---------------------------------------------------------------------
#include <stdio.h>

int main(void)
{
   float   a, b, c;

   a = 60.0F;
   b =  1.2F;
   c = a / b;
   printf("%f¥n", c);

   a = 411999436800.0F;
   b = 3483491.0F;
   c = a / b;
   printf("%f¥n", c);
   return 0;
}
---------------------------------------------------------------------
```

Results of execution (incorrect)

```
---------------------
   49.948486
   118130.000000
---------------------
```

Results that would be (correct)

```
---------------------
   49.999996
   118272.000000
---------------------
```

3. Workaround

Convert the type of the dividend or divisor to long double.

```
---------------------------------------------------------------------
#include <stdio.h>

int main(void)
```

```
{
   float   a, b, c;

   a = 60.0F;
   b =  1.2F;
   c = (long double)a / b;
   printf("%f¥n", c);

   a = 411999436800.0F;
   b = 3483491.0F;
   c = (long double)a / b;
   printf("%f¥n", c);
   return 0;
}
```

---

4. Schedule of Fixing the Problem

We plan to fix this problem in the release of V.5.45 Release 00.
