RENESAS Tool News

RENESAS TOOL NEWS on October 1, 2012: 121001/tn3

Note on Using C Compiler Packages for R8C and M16C Families

When using C compiler packages for R8C and M16C families, take note of the following problem:

- With using single-precision floating-point libraries

---

1. Products and Versions Concerned

- C compiler package for R32C series
  V.1.01 Release 00 through V.1.02 Release 01
- C compiler package for M32C series (M3T-NC308WA)
  V.5.10 Release 1 through V.5.42 Release 00
- C compiler package for M16C series and R8C family (M3T-NC30WA)
  V.5.10 Release 1 through V.6.00 Release 00

2. Description

If a call is made to the single-precision floating-point library function modff which takes an argument, its return value may be incorrect.
In addition, the single-precision floating-point library function ceilf, floorf, or fmodf, which is made a call to modff, may return an incorrect value.

Note that if any of the compile options such as -fdouble_32(-fD32), -OR_MAX(-ORM), and -OS_MAX(-OSM) is selected, double-precision floating-point libraries are interpreted as single-precision ones. So if any of the above options is used, the double-precision floating-point library functions modf, ceil, floor, and fmod may return incorrect values.

3. Conditions

Here we explains the conditions under which the problem arises when calls are made to modff, ceilf, floorf, and fmodf.
In the case of the double-precision floating-point library functions,

the above-mentioned modff, ceilf, floorf, and fmodf must be modf, ceil, floor, and fmod.

3.1 Call Made to Function modff

This problem arises if the following conditions are all satisfied:
(1) A call is made to modff.
(2) The argument passed to modff is equal to or greater than 1.
(3) The fractional part of the argument in (2) is any of the following:
0.125, 0.250, 0.375, 0.500, 0.625, 0.750, and 0.875

Example:

```
/* nc30 -c sample.c */
#include <math.h>
main(){
   float x, y, iptr;

    x = 3.625;
    y = modff(x, &iptr);   // Variable y takes a value of 0.0,
                              but it must be 0.625.
}
```

3.2 Call Made to Function ceilf

This problem arises if the following conditions are all satisfied:
(1) A call is made to ceilf.
(2) The argument passed to ceilf is equal to or greater than 1.
(3) The fractional part of the argument in (2) is any of the following:
0.125, 0.250, 0.375, 0.500, 0.625, 0.750, and 0.875

3.3 Call Made to Function floorf

This problem arises if the following conditions are all satisfied:
(1) A call is made to floorf.
(2) The argument passed to floorf is equal to or less than -1.
(3) The fractional part of the argument in (2) is any of the following:
0.125, 0.250, 0.375, 0.500, 0.625, 0.750, and 0.875

3.4 Call Made to Function fmodf

This problem arises if the following conditions are all satisfied:
(1) A call is made to fmodf.
(2) The dividend of the argument passed to fmodf is equal to or less than -1, or equal to or greater than 1.
(3) The divisor of the argument in (2) is equal to or less than -1, or equal to or greater than 1.
(4) The fractional part of the dividend of the argument in (2) is any

of the following:
0.250, 0.500, and 0.750

4. Workaround

In the source file modff.c of function modff(), modify as follows:

Before modification: if (m_mant & 0xfffff) {

After modification: if (m_mant & 0x7fffff) {

Then use it in the project.
