RENESAS Tool News

RENESAS TOOL NEWS on January 16, 2004: RSO-M3T-CC32R-040116D

A Note on Using Cross-Tool Kit M3T-CC32R

Please take note of the following problem in using the M3T-CC32R cross-tool kit for the M32R family MCUs:

- On displaying a value less than 0.5 by using a formatted output function such as printf

---

1. **Versions Concerned**
   M3T-CC32R V.1.00 Release 1 through V.4.20 Release 1

2. **Description**
   An absolute value less than 0.5 may be displayed as "0." instead of the correct value "0" (where the decimal point and fractions are not displayed).

   2.1 Conditions

   This problem occurs if the following four conditions are satisfied:

   (1) A floating constant is displayed using the %f conversion specifier in a formatted output function (printf, vprintf, fprintf, vfprintf, sprintf, or vsprintf).

   (2) The precision in the output conversion specification is zero.

   (3) The # flag (alternate form) is not used in the output conversion specification.

   (4) The absolute value of the constant to display is less than 0.5 and greater than 0.0.

   2.2 Examples

   Source file:

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

  char bf1[32], bf2[32], bf3[32], bf4[32], bf5[32];
```

```
int main()
{
   double x;
   int prec;

/* Example Condition (4) satisfied */
   x = 0.1;                    /* Condition (4) */
   sprintf(bf1, "%4.0f¥n", x);      /* Conditions (1)--(3) */
   /* Processing handling bf1 */

   x = -0.4999;                  /* Condition (4) */
   sprintf(bf2, "%3.0f¥n", x);      /* Conditions (1)--(3) */
   /* Processing handling bf2 */

   x = 0.064;                   /* Condition (4) */
   prec = 0;                    /* Condition (2) */
   sprintf(bf3, "%1.*f¥n", prec, x); /* Conditions (1), and (3) */
   /* Processing handling bf3 */

/* Example Condition (4) unsatisfied (no problem) */
   x = 0.0;                    /* Condition (4) unsatisfied */
   sprintf(bf4, "%5.0f¥n", x);    /* Conditions (1)--(3) */
   /* Processing handling bf4 */

   x = -0.5;                   /* Condition (4) unsatisfied */
   sprintf(bf5, "%5.0f¥n", x);    /* Conditions (1)--(3) */
   /* Processing handling bf5 */

   return 0;
}
```

Outputs of bf1, bf2, bf3, bf4, and bf5 in the above source file:

	Display	Correct Display
bf1	" 0."	" 0"
bf2	"-0."	" -0"
bf3	"0."	"0"
bf4	" 0"	Same as left
bf5	" -1"	Same as left

3. Workaround

This problem can be circumvented in either of the following methods:

(1) Use a macro to round down an absolute value less than 0.5 to 0.0.

Modified example of the source file in 2.2:

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

 #define TRUNC00(x) ((-0.5<(x)&&(x)<+0.5)?0.0*(x):(x))
        /* A macro to round down an absolute value
                        less than 0.5 to 0.0 defined */


 char bf1[32], bf2[32], bf3[32], bf4[32], bf5[32];

 int main()
 {
    double x;
    int prec;

 /* Example Condition (4) satisfied */
    x = 0.1;
    sprintf(bf1, "%4.0f¥n", TRUNC00(x));      /* Macro used */
    /* Processing handling bf1 */

    x = -0.4999;
    sprintf(bf2, "%3.0f¥n", TRUNC00(x));      /* Macro used */
    /* Processing handling bf2 */

    x = 0.064;
    prec = 0;
    sprintf(bf3, "%1.*f¥n", prec, TRUNC00(x)); /* Macro used */
    /* Processing handling bf3 */

  /* Example Condition (4) unsatisfied (no problem)
          but the macro used for coordinative descriptions */
    x = 0.0;
    sprintf(bf4, "%5.0f¥n", TRUNC00(x));      /* Macro used */
    /* Processing handling bf4 */
```

```
    x = -0.5;
    sprintf(bf5, "%5.0f¥n", TRUNC00(x));       /* Macro used */
    /* Processing handling bf5 */

    return 0;
 }
```

Outputs of bf1, bf2, bf3, bf4, and bf5 in the above source file before and after modification by Method (1):

	Before	After
bf1	" 0."	" 0"
bf2	"-0."	" -0"
bf3	"0."	"0"
bf4	" 0"	" 0"
bf5	" -1"	" -1"

(2) Use the # flag to display a floating value.
Using the # flag displays an absolute value greater than 0.5 as a decimal number, so the same type of representation is available for displaying both absolute values greater and less than 0.5.

Modified example of the source file in 2.2:

```
 #include <stdio.h>

 char bf1[32], bf2[32], bf3[32], bf4[32], bf5[32];

 int main()
 {
    double x;
    int prec;

   /* Example Condition (4) satisfied */
    x = 0.1;
    sprintf(bf1, "%#4.0f¥n", x);       /* # flag used */
    /* Processing handling bf1 */

    x = -0.4999;
    sprintf(bf2, "%#3.0f¥n", x);       /* # flag used */
```

```
    /* Processing handling bf2 */

    x = 0.064;
    prec = 0;
    sprintf(bf3, "%#1.*f¥n", prec, x); /* # flag used */
    /* Processing handling bf3 */

 /* Example Condition (4) unsatisfied (no problem)
         but the # flag used for coordinative descriptions */
    x = 0.0;
    sprintf(bf4, "%#5.0f¥n", x);      /* # flag used */
    /* Processing handling bf4 */

    x = -0.5;
    sprintf(bf5, "%#5.0f¥n", x);      /* # flag used */
    /* Processing handling bf5 */

    return 0;
 }
```

Outputs of bf1, bf2, bf3, bf4, and bf5 in the above source file before and after modification by Method (2):

	Before	After
bf1	" 0."	" 0."
bf2	"-0."	"-0."
bf3	"0."	"0."
bf4	" 0"	" 0."
bf5	" -1"	" -1."

4. Schedule of Fixing the Problem

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