RENESAS ツールニュース

RENESAS TOOL NEWS [2004年01月16日 : RSO-M3T-CC32R-040116D]

M3T-CC32R
ご使用上のお願い

M32Rファミリ用 クロスツールキット M3T-CC32Rの使用上の注意事項を連絡します。

- printfなどの書式付き出力関数を使用して0.5未満の値を表示する場合の注意事項

---

1. **該当製品**
   M32Rファミリ用クロスツールキット
   M3T-CC32R V.1.00 Release 1 ～ V.4.20 Release 1

2. **内容**
   発生条件に該当する場合、絶対値が0.5未満である数値が「0.」と表示されます。 正しくは、「0」と表示されます(小数点と小数点以下の桁を表示しません)。

   2.1 発生条件
   次の(1)～(4)全ての条件を満たす場合に発生します。

   (1) 書式付き出力関数 (printf, vprintf, fprintf, vfprintf, sprintf, vsprintf) の %f変換指定子を使用して浮動小数点型の数値を表示している。

   (2) (1)の%f変換指定子の精度には0が指定されている。

   (3) (1)の%f変換指定子には#フラグ(代替形式)が指定されていない。

   (4) 表示する数値の絶対値が、0.5未満でかつ0.0より大きい。

   2.2 発生例

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

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

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

  /* 数値が条件(4)に該当する例 */
   x = 0.1;                    /* 条件(4) */
   sprintf(bf1, "%4.0f¥n", x);      /* 条件(1),(2),(3) */
   /* bf1を使った処理 */

   x = -0.4999;                  /* 条件(4) */
   sprintf(bf2, "%3.0f¥n", x);      /* 条件(1),(2),(3) */
   /* bf2を使った処理 */

   x = 0.064;                    /* 条件(4) */
   prec = 0;                     /* 条件(2) */
   sprintf(bf3, "%1.*f¥n", prec, x); /* 条件(1),(3) */
   /* bf3を使った処理 */

  /* 数値が条件(4)に該当しないので問題ではない例 */
   x = 0.0;                    /* 条件(4)に非該当 */
   sprintf(bf4, "%5.0f¥n", x);      /* 条件(1),(2),(3) */
   /* bf4を使った処理 */

   x = -0.5;                   /* 条件(4)に非該当 */
   sprintf(bf5, "%5.0f¥n", x);      /* 条件(1),(2),(3) */
   /* bf5を使った処理 */

   return 0;
}
```

[ソースファイル例 の bf1,bf2,bf3,bf4およびbf5の出力]

	出力内容	正しい出力
bf1	" 0."	" 0"
bf2	"-0."	" -0"
bf3	"0."	"0"
bf4	" 0"	同左
bf5	" -1"	同左

3. **回避策**

次のいずれかの方法で抑止してください。

(1) マクロを使用して、絶対値が0.5未満の値を、0.0に切り捨てる。

[ソースファイル例の変更例]

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

  #define TRUNC00(x) ((-0.5<(x)&&(x)<+0.5)?0.0*(x):(x))
        /* 絶対値が0.5未満の値を 0.0 に切り捨てるマクロを定義 */

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

  int main()
  {
     double x;
     int prec;

   /* 数値が条件(4)に該当する例 */
     x = 0.1;
     sprintf(bf1, "%4.0f¥n", TRUNC00(x));      /* マクロを使用 */
     /* bf1を使った処理 */

     x = -0.4999;
     sprintf(bf2, "%3.0f¥n", TRUNC00(x));      /* マクロを使用 */
     /* bf2を使った処理 */

     x = 0.064;
     prec = 0;
     sprintf(bf3, "%1.*f¥n", prec, TRUNC00(x)); /* マクロを使用 */
     /* bf3を使った処理 */

   /* 数値が条件(4)に該当しないので問題ではないが、同様の記述で統一した例 */
     x = 0.0;
     sprintf(bf4, "%5.0f¥n", TRUNC00(x));      /* マクロを使用 */
     /* bf4を使った処理 */

     x = -0.5;
     sprintf(bf5, "%5.0f¥n", TRUNC00(x));      /* マクロを使用 */
     /* bf5を使った処理 */
```

```
    return 0;
}
```

[回避策(1)適用前後のソースファイル例の bf1,bf2,bf3,bf4およびbf5の出力]

	変更前	変更後
bf1	" 0."	" 0"
bf2	"-0."	" -0"
bf3	"0."	"0"
bf4	" 0"	" 0"
bf5	" -1"	" -1"

(2) #フラグを指定して、常に浮動小数点を表示する。
#フラグを使うと、絶対値が0.5以上の値にも小数点が付くため、0.5未満の 値と表示形式を合わせることができます。

[ソースファイル例の変更例]

```
#include <stdio.h>

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

int main()
{
    double x;
    int prec;

  /* 数値が条件(4)に該当する例 */
    x = 0.1;
    sprintf(bf1, "%#4.0f¥n", x);      /* #フラグを追加 */
    /* bf1を使った処理 */

    x = -0.4999;
    sprintf(bf2, "%#3.0f¥n", x);      /* #フラグを追加 */
    /* bf2を使った処理 */

    x = 0.064;
    prec = 0;
    sprintf(bf3, "%#1.*f¥n", prec, x); /* #フラグを追加 */
```

```
    /* bf3を使った処理 */

   /* 数値が条件(4)に該当しないので問題ではないが、同様の記述で統一した例 */
    x = 0.0;
    sprintf(bf4, "%#5.0f¥n", x);      /* #フラグを追加 */
    /* bf4を使った処理 */

    x = -0.5;
    sprintf(bf5, "%#5.0f¥n", x);      /* #フラグを追加 */
    /* bf5を使った処理 */

    return 0;
  }
```

[回避策(2)適用前後のソースファイル例の bf1,bf2,bf3,bf4およびbf5の出力]

	変更前	変更後
bf1	" 0."	" 0."
bf2	"-0."	"-0."
bf3	"0."	"0."
bf4	" 0"	" 0."
bf5	" -1"	" -1."

4. 恒久対策

本問題は、次期バージョンで改修する予定です。
