

【注意事項】

R20TS1082JJ0100

Rev.1.00

2024.12.05

RX ファミリ用 C/C++コンパイラパッケージ

(注意事項 No.68)

概要

RX ファミリ用 C/C++コンパイラ CC-RX の使用上の注意事項を連絡します。

1. 多重ループ構造における、内側ループの周回数に関する注意事項 (No.68)

注: 注意事項の後ろの番号は、注意事項の識別番号です。

1. 多重ループ構造における、内側ループの周回数に関する注意事項 (No.68)

1.1 該当製品

Windows 用 CC-RX V2.00.00 ~ V3.06.00

Linux 用 CC-RX V3.06.01 ~ V3.06.02

1.2 内容

多重ループ構造における、内側ループの周回数がソースコードの記述と異なる実行結果になる場合があります。

1.3 発生条件

[発生条件]

次の(1)から(6)のすべてを満たす場合に発生する可能性があります。

- (1) 最適化レベルオプションの指定が`-optimize=2` または `-optimize=max` である。
- (2) プログラム中に多重ループ構造がある。
- (3) 最外側ループのループ制御変数は整数型である。かつ、`volatile` 修飾されていない。
- (4) 内側ループのループ制御変数は 1 バイトまたは 2 バイト長の整数型である。かつ、`volatile` 修飾されていない。
- (5) (3)のループも(4)のループも、当該ループ内で値が変化しない式(例: 定数)とループ制御変数との比較がループ終了条件式である。
- (6) (4)のループはループ内には一度しか入らないループである。(繰り返すことのないループ)

[発生例]

この例の場合は、一度しかループ内に入るはずのない内側ループの周回数を誤り、二度入ってしまいます。よって、2 となるはずの関数 `test()` の返却値は、6 となってしまいます。

```
volatile int vi = 0;
int test(void) {
    int result = 0;
    int i;                /* (3) */
```

```

short j;                                /* (4) */

for (i = 0; i < 2; i++) { /* (2)(3)(5) */

    j = 1;
    do {

        result += j;
        if (vi != 0) {
            break;
        }

        j++;
    } while (j == 1); /* (2)(4)(5)(6)*/

}
return result;
}

```

1.4 回避策

以下のいずれかを行うことで回避できます。

- 最適化レベルを`-optimize=0` または `-optimize=1` にする。
- 発生条件(4)の内側ループの制御変数の型を発生条件(4)で示す型以外の型にする(long 型など)。
- 発生条件(4)の内側ループの制御変数を `volatile` 修飾する。
- 発生条件(3)の最外側のループの制御変数を `volatile` 修飾する。

1.5 恒久対策

CC-RX V3.07.00 で改修する予定です。リリース時期は未定です。

以上

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Dec.05.24	-	新規発行

本資料に記載されている情報は、正確を期すため慎重に作成したのですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。

過去のニュース内容は発行当時の情報をもとにしており、現時点では変更された情報や無効な情報が含まれている場合があります。

ニュース本文中の URL を予告なしに変更または中止することがありますので、あらかじめご承知ください。

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。