【注意事項】
RL78 ファミリ用 C コンパイラパッケージ

R20TS0105JJ0100
Rev.1.00
2016.12.16 号

概要

RL78 ファミリ用 C コンパイラパッケージ CC-RL の使用上の注意事項を連絡します。

1. 配列要素、構造体メンバまたは共用体メンバへの値の更新が反映されない注意事項(CCRL#013)

注: 注意事項の後ろの番号は、注意事項の識別番号です。

1. 配列要素、構造体メンバまたは共用体メンバへの値の更新が反映されない注意事項 (CCRL#013)

1.1 該当製品

CC-RL V1.00.00 ～ V1.03.00

1.2 内容

配列要素、構造体メンバまたは共用体メンバの値の更新が反映されない場合があります。

1.3 発生条件

下記の(1)～(10)の条件を全て満たす場合に、(4)で参照される配列要素、構造体メンバまたは共用体メンバの値が、(5)で更新される前の値となる場合があります。

(1) -Onothing 以外の最適化レベルを指定しているか、最適化レベルを指定していない。

(2) 配列要素、構造体メンバまたは共用体メンバを参照している。

(3) (2)の構造体、共用体は下記のいずれかを満たす。

・構造体の場合、最初のメンバが配列である。

・共用体の場合、配列のメンバが含まれる。

(4) (2)の参照後に(2)と同じ配列要素、構造体メンバまたは共用体メンバを参照する。なお、(2)の参照がループ中にある場合、ループによって繰り返し実行される(2)の参照を含む。また、呼び出した関数の中にある参照も含む。

(5) (2)の参照と(4)の参照の間で、(2)(4)で参照している配列要素、構造体メンバまたは共用体メンバの値を更新している。

(6) (5)の更新は下記のいずれかを満たす。

・構造体の場合、構造体アドレスもしくは構造体に含まれるいずれかのメンバのアドレスを指すポインタを使用して更新している。

・共用体の場合、共用体アドレスもしくは共用体に含まれるいずれかのメンバのアドレスを指すポインタを使用して更新している。

・配列の場合、配列アドレスもしくは配列に含まれるいずれかの要素のアドレスを指すポインタを使用して更新している。

(7) (5)の更新は、(2)(4)で参照している配列要素、構造体メンバまたは共用体メンバとは異なる配列要素、構造体メンバまたは共用体メンバから開始している。

(8) (5)の更新は、更新を開始する配列要素、構造体メンバまたは共用体メンバのサイズ(構造体メンバまたは共用体メンバが配列型である場合、その要素型のサイズも含む)単位で行われている。

(9) (2)(4)で参照している配列要素、構造体メンバまたは共用体メンバと(5)において更新を開始する配列要素、構造体メンバまたは共用体メンバでは、アドレスの小さい方に配置されている要素またはメンバの終端と、アドレスの大きい方に配置されている要素またはメンバの先端の差が、(4)で参照しているサイズまたは(5)で一度に更新される領域のサイズのいずれかよりも大きい。

(10) (5)の更新以降、(4)の参照までに配列要素、構造体メンバまたは共用体メンバの更新が全くないか、または(6)(7)(8)(9)の条件を満たす更新のみがある。

1.4 発生例

【C ソース】

```
 1:  #include <string.h>
 2:  typedef struct test {
 3:      unsigned char  param01[8];  // 発生条件(3)：更新を開始する構造体メンバ
 4:      unsigned char  param02;
 5:      unsigned short param03;
 6:      unsigned int   param04;     // 発生条件(2)(4)：参照する構造体メンバ
 7:  } test_t;
 8:
 9:  int test_func(void) {
10:    test_t        t1;
11:    test_t        t2;
12:    int           ret = 1;
13:    unsigned char  *src;
14:    unsigned char  *dst;
15:    int           size;
16:    memset(&t1, 0, sizeof(test_t));
17:    if (t1.param04 == 0) {        //  発生条件(2)
18:         t2.param04 = 10;
19:         src = (unsigned char *)&t2;
20:         dst = (unsigned char *)&t1;  //  発生条件(6)(7)
21:         size = sizeof(test_t);
22:         while (size-- > 0){
23:             *dst++ = *src++;     // 発生条件(5)(6)(7)(8)(9)：構造体変数 t1 更新
24:         }
25:    }
26:    if (t1.param04 < 5) {         // 発生条件(4)
27:         ret = -1;
28:    }
29:    return(ret);
30:  }
```

(2)(4)で参照している t1.param04 と、(5)で更新を開始している t1.param01 の差は 4byte です。(5)で更新しているサイズ unsigned char 型の 1 バイトよりも大きいため発生条件(9)に該当します。

また、t1.param04 に対する 17 行目の参照から 26 行目の参照までに(6) (7) (8) (9)の条件を満たす更新が(5)以外に存在しないため発生条件(10)にも該当します。

この結果、26 行目で参照する t1.param04 の値には、22～24 行目の更新が反映されません。

1.5 回避策

以下のいずれかにより回避可能です。

(1) 最適化レベル-Onothing を指定する。

(2) 発生条件(5)における更新を以下のいずれかに変更する。

・構造体の場合、構造体代入に変更する。

【変更前】

```
19:         src = (unsigned char *)&t2;
20:         dst = (unsigned char *)&t1;
21:         size = sizeof(test_t);
22:         while (size-- > 0){
23:             *dst++ = *src++;
24:         }
```

【変更後】

```
19:         t1 = t2;
```

・memcpy ライブラリ関数の呼び出しに変更する。

【変更前】

```
22:         while (size-- > 0){
23:             *dst++ = *src++;
24:         }
```

【変更後】

```
22:         memcpy(dst, src, size);
```

1.6 恒久対策

次期バージョンで改修予定です。

以上

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2016.12.16	-	新規発行

ルネサスエレクトロニクス株式会社

〒135-0061　東京都江東区豊洲 3-2-24（豊洲フォレシア）

■総合お問い合わせ先

http://japan.renesas.com/contact/