CubeSuite+版 RX ファミリ用 C/C++コンパイラパッケージ ご使用上のお願い

CubeSuite+版 RX ファミリ用 C/C++コンパイラパッケージの使用上の注意事項をお知らせします。

- **アドレス定数式の初期値を伴う関数内の static な集成体と共用体に関する注意事項 (RXC#035)**
- **変数に一致と大小比較判定を両方行う場合の注意事項 (RXC#036)**

注: 各注意事項の後ろの番号は、注意事項の識別番号です。

---

1. アドレス定数式の初期値を伴う関数内の static な集成体と共用体に関する注意事項 (RXC#035)

1.1 該当製品およびバージョン

CubeSuite+版 RX ファミリ用 C/C++コンパイラパッケージ V2
型名: PRX00CSP2-MWR
CC-RX コンパイラ V2.02.00

1.2 内容

static 宣言 または extern 付きではない const 宣言された変数または配列の初期化が正しく行われない場合があります。

1.3 発生条件

以下を全て満たす場合に発生することがあります。

(1) C++言語としてコンパイルしている。
(2) 内部に自動変数を持つ static 宣言された関数がある。
(3) static 宣言 または extern 付きではない const 宣言された変数または配列があり、さらに、それらの初期化式に(2)の関数のアドレスを含むものがある。
(4) 内部に自動変数を持たない static 宣言された関数があり、その関数内で(3)の 変数 または 配列 を参照している。
(5) (2)の自動変数を使用する(注 1)処理が、(2)の関数内に 1 つ以上ある。
　注 1: "自動変数の使用"には、自動変数の読み出し、書き込み、アドレス取得に加え、該当する自動変数を "宣言と同時に初期化する場合" や "他の変数の初期化式に用いる場合" も含みます。
(6) (2)および(4)の関数は共に、extern 付きの関数および変数のいずれからも参照されていない。

発生条件例: 以下を C++言語としてコンパイルする場合 発生条件(1)

```
------------------------------------------------------------------------
static void OtherFunc(void);
static int LocalFunc(void);
int (*const sf_Table[] )(void) = {    // 発生条件(3)
      LocalFunc, 0,                   // 発生条件(3)
```

```
};
extern int (*const *pFunc)(void);
static void OtherFunc(void)
{
     pFunc = sf_Table;                  // 発生条件(4)
}
static int LocalFunc(void)              // 発生条件(2)
{
     int ret;                           // 発生条件(2)
     return ret;                        // 発生条件(5)
}
```

1.4 回避策

以下のいずれかの方法で回避可能です。

(1) C++言語特有の機能を使用していない場合は、C 言語でコンパイルする。

(2) 発生条件(2),(4)の関数のいずれかを呼び出す、static でない関数を追加する。

(3) 発生条件(2),(4)の関数のいずれかを削除する。
なお、発生条件(2)の関数を削除する場合は、合わせて発生条件(3)の初期化式から、発生条件(2)の関数のアドレスを含まないようにする必要があります。

(4) 発生条件(2),(4)の関数のいずれかを static 宣言から extern 宣言に変更する。

(5) 発生条件(2)の自動変数を使用しない。

2. 変数に一致と大小比較判定を両方行う場合の注意事項 (RXC#036)

2.1 該当製品およびバージョン

CubeSuite+版 RX ファミリ用 C/C++コンパイラパッケージ V2
型名: PRX00CSP2-MWR
CC-RX コンパイラ V2.00.00 ～ V2.02.00

2.2 内容

同一関数内で、if 文 または ループの制御式として、"!=" または "==" による比較式と、大小比較判定式を組み合わせて記述している場合に、その if 文 または ループ の判定を誤る場合があります。

2.3 発生条件

以下を全て満たす場合に発生することがあります。

(1) -optimize=1, 2 または max のいずれかを指定している。

(2) 定数(注 1)と変数(注 2)を"!=" または "==" により比較する式が存在する。
注 1: 静的に定数と分かる式を含む。
注 2: 配列変数、構造体メンバ、および共用体メンバを含む。

(3) (2)の比較式を含む関数内に、定数と(2)の変数を "<"、">"、"<="、または ">=" により比較する式が存在する。

(4) (2)の変数は volatile 修飾されていない。

(5) (2)および(3)の比較式は以下のいずれかを満たす。

(5-1) (2)と(3)の比較式が "||" または "&&" で結合されている

(5-2) (2)と(3)の比較式が含まれる "if 文" または "ループ"が連続して実行される

(6) (2)と(3)の比較式間に別の式および文がない。

発生条件例:

```
------------------------------------------------------------------------
int ZZZ[3] = {0x7FFF,0x7FFF,0x7FFF};            // 発生条件(4)
void test(void){
      if(( ( ZZZ[0] <= 180  ) &&                // 発生条件(3)、
                                                // 発生条件(5-1)
           ( (unsigned int)ZZZ[0] != (unsigned int)0x8000U ) ) ){
                                                // 発生条件(2) (6)
      }
       ・・・・
}
------------------------------------------------------------------------
```

2.4 回避策

以下のいずれかで回避可能です。

(1) -optimize=0 を指定する。

(2) 発生条件(2)の変数を volatile 修飾する。

(3) 発生条件(2)および(3)の比較のうち後に実行される方の直前に、
ダミーの volatile 変数を参照する。

回避策の適用例: 回避策(3)の場合

```
------------------------------------------------------------------------
volatile int dummy;                          //ダミーの volatile 変数の宣言
int ZZZ[3] = {0x7FFF,0x7FFF,0x7FFF};
void test(void){
      if(( ( ZZZ[0] <= 180  ) &&
            (dummy,((unsigned int)ZZZ[0] != (unsigned int)0x8000U ) )) )
      {                                      // 回避策(3)
      }
       ・・・・
}
------------------------------------------------------------------------
```

3. 恒久対策

CubeSuite+版 RX ファミリ用 C/C++コンパイラパッケージ CC-RX V2.03.00 で改修しました。

株式会社 日立超LSIシステムズ