SuperH RISC engine C/C++コンパイラ Ver.7 不具合内容

- 過去のお知らせ

SuperH RISC engine C/C++コンパイラ Ver.7 台における不具合内容を以下に示します。
1.～3.のチェックツールおよび、4.の修正ファイルをルネサス エレクトロニクス株式会社のホームページより入手できます。
http://tool-support.renesas.com/jpn/toolnews/shc/shcv7/dr_shcv7_3.html

1. ゼロ拡張命令の不当削除

現象：
ループ内で符号なし変数を複数回参照した時、必要なゼロ拡張命令が削除される場合がある。
[例]

```
  extern unsigned char X;
  int sub(int a, int b, int n) {
      int i, sum=0;
      for (i = 0; i < n; i++) {
          if (X == (unsigned char)0xff) {    // X の参照
              sum += a;
          }
          if (X == (unsigned char)0xf0) {    // X の参照
              sum += b;
          }
          X = X + 1;                         // X の定義
      }
      return (sum);
  }
```

[出力コード]

```
  _sub:
    MOV.L   R12,@-R15
    MOV.L   R13,@-R15
    MOV.L   R14,@-R15
    MOV     R5,R13
    MOV     #0,R5       ; H'00000000
    MOV.L   L19,R1      ; _X
    MOV     R6,R7
    MOV     R4,R12
    MOV     R5,R6
    MOV     #-1,R4      ; H'FFFFFFFF
    MOV.B   @R1,R2
                        ; <- EXTU.B R2,R2 が削除
    MOV     #-16,R14    ; H'FFFFFFF0
    EXTU.B  R4,R4
    BRA     L11
    EXTU.B  R14,R14
  L12:
    CMP/EQ  R4,R2       ; 比較結果が正しくない場合あり(X>127 の時)
    BF      L14
    ADD     R12,R5
  L14:
    CMP/EQ  R14,R2      ; 比較結果が正しくない場合あり(X>127 の時)
    BF      L16
    ADD     R13,R5
  L16:
    ADD     #1,R2
```

```
    EXTU.B  R2,R2
    ADD     #1,R6
  L11:
    CMP/GE  R7,R6
    BF      L12
    MOV.B   R2,@R1
    MOV     R5,R0
    MOV.L   @R15+,R14
    MOV.L   @R15+,R13
    RTS
    MOV.L   @R15+,R12
```

発生条件：

次の(1)から(4)の条件、もしくは(5)から(8)の条件を全て満たす時、発生する可能性ことがあります。

該当するかどうかはチェックツールを使用することにより確認することができます。

(1) optimize=1 を指定している。

(2) unsigned char/unsigned short 型変数を使用している。

(3) (2)の変数がループ内で 2 回以上参照した後、定義されている(例参照)。

(4) (2)の変数が最初の参照前にメモリからロードされる。

(5) optimize=1 を指定している。

(6) unsigned char/unsigned short 型ローカル変数を使用している。

(7) (6)の変数がループ内で右シフトのシフト元変数として使用されている。

(8) (7)のシフト数が 2 以上である(SHLR2/SHLR8/SHLR16 に展開される)。

回避方法：

該当箇所が存在した場合、以下のいずれかの方法で回避していただきますようお願いします。

なお、optimize=0 の場合でもチェッカがメッセージを出力する場合がありますが、optimize=0 の場合には、本不具合には該当しません。

(1) 当該変数を(unsigned)long で宣言する。

(2) 該当ファイルを optimize=0 指定する。

2. 浮動小数点定数ロードの不当削除

現象：

同じ値の浮動小数点定数を複数回使用した場合、不当に浮動小数点定数ロード式を削除する場合がある。

発生条件：

以下の条件をすべて満たした場合に発生することがあります。

該当するかどうかはチェックツールを使用することにより確認することができます。

(1) optimize=1 を指定している。

(2) 同じ値の浮動小数点定数をループ内で 2 回以上使用している。

(3) (2)と同じ値の定数をループ外で使用している。

回避方法：

該当箇所が存在した場合、以下の方法で回避していただきますようお願いします。

該当ファイルを optimize=0 指定する。

3. 配列アクセス不正

現象：

配列アクセス a[c-exp]において、c が定数、exp の型が unsigned char/unsigned short のとき、配列を正しくアクセスできない場合がある。

[例]

```
    unsigned char dd[2];
    void func(unsigned char a, unsigned char b) {
        dd[15-a] = b;
    }
```

[出力コード]

```
  _func:
    MOV.L   L11,R2      ; _dd
    NEG     R4,R6       ; R6 <- -a
    EXTU.B  R6,R0       ; -a をゼロ拡張
    ADD     #15,R2
    RTS
    MOV.B   R5,@(R0,R2) ; 異なるアドレスにアクセス
```

[例]

```
    unsigned char dd[2];
    void func(unsigned char a, unsigned char b) {
        dd[15-a] = b;
    }
```

[出力コード]

```
  _func:
    MOV.L   L11,R2      ; _dd
    NEG     R4,R6       ; R6 <- -a
    EXTU.B  R6,R0       ; -a をゼロ拡張
    ADD     #15,R2
    RTS
    MOV.B   R5,@(R0,R2) ; 異なるアドレスにアクセス
```

発生条件：

次の条件を全て満たす時、発生することがあります。

該当するかどうかはチェックツールを使用することにより確認することができます。

(1) optimize=1 を指定している。

(2) 配列の要素を"定数-式(unsigned char/unsigned short)"でアクセスしている(例では dd[15-a])。

回避方法：

該当箇所が存在した場合、以下のいずれかの方法で回避していただきますようお願いします。

(1) 該当ファイルを optimize=0 を指定してコンパイルする。

(2) 該当配列要素を "-式(unsigned char/unsigned short)+定数"でアクセスする(例では dd[-a+15])。

4. memmove ライブラリ不正

現象：

memmove ライブラリ関数でメモリ領域を上位アドレスのメモリ領域に移動すると、指定した文字数以上の移動を行ってしまう。

発生条件：

次の条件を全て満たす時、発生します。

(1) 移動サイズが 31byte 以上である。

(2) 移動先アドレスが移動元アドレスより大きい(メモリ上位へ移動)。

(3) (移動元アドレス＋移動サイズ)が移動先領域内にある(移動領域に重なりがある)。

(4) 移動先アドレス、移動元アドレスのどちらかが 4 の倍数でない。

回避方法：

該当箇所が存在した場合、修正ファイルを使用して回避していただきますようお願いします。

ソースでの回避方法は以下の通りです。

移動サイズを 30byte 以下で区切って memmove を行う。

株式会社 日立超LSIシステムズ