ご使用上のお願いーSuperH RISC engine C/C++コンパイラ Ver.7 不具合内容(11)

SuperH RISC engine C/C++コンパイラパッケージ V.7 の使用上の注意事項 4 件

該当製品

	パッケージバージョン	コンパイラバージョン
P0700CAS7-MWR	7.0B、7.0.01～7.1.04	7.0B、7.0.03～7.1.03
P0700CAS7-SLR	7.0B、7.0.02～7.1.04	7.0B、7.0.03～7.1.03
P0700CAS7-H7R	7.0B、7.0.02～7.1.04	7.0B、7.0.03～7.1.03

内容

1. コピー伝播不正
2. 不要式削除不正
3. GBR 相対論理演算不正
4. 符号拡張削除不正

恒久対策

本内容は、SuperH RISC engine ファミリ C/C++コンパイラ Ver.7.1.04 以降（コンパイラパッケージ Ver.7.1.05 以降）では、全て修正されています。

チェックツール

本不具合のチェックツールをルネサス エレクトロニクス株式会社のホームページよりダウンロードいただけます。
http://tool-support.renesas.com/jpn/toolnews/shc/shcv7/dr_shcv7_7.html

1. コピー伝播不正

現象

複数の分岐元を持つブロックにコピー命令が存在した場合、不正にコピー命令を削除する場合があります。

発生例

```
int func(int *x) {
  int ret=0;
  while(*x++){
    if(*x==1){
      ret+=2;
    }
```

```
    }
    return (ret+2);
  }

  _func:
      MOV     #0,R5      ; 不正にコピーを削除した結果 R7 -> R5 へ変換
  L11:
      MOV.L   @R4,R2
      ADD     #4,R4
                         ; *1 MOV R7,R5 を不正に削除
      TST     R2,R2
      ADD     #2,R5
      BT      L13
      MOV.L   @R4,R0
      CMP/EQ  #1,R0
      BT      L11        ; *2 *3 により BF L11 を変換
      BRA     L11
      NOP                ; *3 MOV R5,R7 を不正に削除
  L13:
      RTS
      MOV     R5,R0
```

発生条件

以下のすべての条件を満たした場合に発生することがあります。

(1) optimize=1 を指定している。
(2) 条件文を記述している。
(3) 複数の分岐元を持つブロックにコピー命令が存在する(例では*1)。
(4) (3)の分岐元のブロックにコピー命令のコピー元レジスタ(例では R7)の定義がないパスがある
(例では*2 から L11 へ分岐するパス)。

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ください。

・該当ファイルを optimize=0 オプション指定する。

2. 不要式削除不正

現象

条件文の then 節、あるいは else 節に代入式があり、その直後に同じ変数同士の代入式を記述した場合、不正に条件文を削除する場合があります。

発生例

```
  int x;
```

```
void f(int y){
  if (y>=256){        /* 不正に削除 */
     x=0;             /* *1 */
  }
  x=x;                /* *2  同じ変数同士の代入文削除 */
  x++;
}
```

↓

```
void f(int y){
  x=0;
  x++;                /* x=0 を伝播 */
}
```

↓

```
void f(int y){
  x=1;
}
```

発生条件

以下のすべての条件を満たした場合に発生することがあります。

(1) optimize=1 を指定している。
(2) 条件文を記述している。
(3) (2)の条件文の then 節もしくは else 節に代入式がある(例では*1)。
(4) (2)の条件文の後に(3)の代入先変数同士の代入式がある(例では*2)。

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ください。

(1) 該当ファイルを optimize=0 オプション指定する。
(2) 該当ファイルを opt_range=noblock オプション指定する(Ver.7.0.06 以降)。

3. GBR 相対論理演算不正

現象

#pragma gbr_base/gbr_base1 を指定した 1byte の配列、もしくはビットフィールドメンバに対し論理演算を行った場合、論理演算の結果を不正な領域に書き込む場合があります。

発生例

```
#pragma gbr_base a,b
char a[2],b[2];
```

```
void f() {
  a[0] = b[0] & 1;
}

 MOV      #_b-(STARTOF $G0),R0
 RTS
 AND.B    #1,@(R0,GBR) ; 演算結果をb[0]に書き込み
```

発生条件

以下のすべての条件を満たした場合に発生することがあります。

(1) gbr=user を指定している。
(2) #pragma gbr_base/gbr_base1 で以下の変数を指定している。
  (a) (unsigned)char 型の配列
  (b) (unsigned)char 型のメンバを持つ構造体配列
  (c) (unsigned)char 型の配列メンバを持つ構造体
  (d) 8bit 以下のビットフィールドメンバを持つ構造体
(3) (2)の変数(例では b[0])に対して定数との論理演算(&, |, ^)を行っている。
(4) (3)の演算の代入先変数(例では a[0])が(2)の条件を満たす。
(5) (3),(4)の変数は別変数、同じ配列で別要素、または同じ構造体で別メンバである。

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ください。

(1) #pragma gbr_base/gbr_base1 指定をやめる。
(2) gbr=auto を指定する(Warning を出力し、#pragma gbr_base/gbr_base1 指定を無効にする)。
(3) 論理演算の結果をいったん volatile 指定したテンポラリ変数に代入する。

例：
```
void f() {
    volatile char temp;
    temp = b[0] & 1;
    a[0] = temp;
}
```

4. 符号拡張削除不正

現象

変数、定数アドレスや配列のインデックスを明示的に 1,2byte にキャストした後に、その値を用いてメモリアクセスを行った場合、キャストが削除され不正なメモリ領域をアクセスする場合があります。

発生例

```
unsigned short x;
```

```
char a[1000];

void f() {
  a[(char)x] = 0;
}

 MOV.L    L11+2,R2 ; _x
 MOV.L    L11+6,R6 ; _a
 MOV.W    @R2,R5
 EXTU.B   R5,R0
                              ; EXTS.B R0,R0 を削除
 MOV      #0,R5               ; H'00000000
 RTS
 MOV.B    R5,@(R0,R6)         ; x が 0～127 の範囲外の時、
                              ; 不正アドレスを参照する場合あり
```

発生条件

以下のすべての条件を満たした場合に発生することがあります。

(1) optimize=1 を指定している。

(2) 変数アドレス、定数アドレス、配列のインデックスを明示的に 1,2byte にキャストしている、もしくは当該関数が char/short の仮引数を持ち、その引数を配列のインデックスのみで使用している。

(3) (2)の値を用いてメモリアクセスを行う。

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ください。

・該当ファイルを optimize=0 オプション指定する。

株式会社 日立超LSIシステムズ

(c) Hitachi ULSI Systems Co., Ltd. 1995,2014. All rights reserved.