お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

2003 年　10 月　8 日

－ ルネサス 技術情報 －

〒１００－０００４
東京都千代田区大手町 2 丁目 6 番 2 号
(日本ビル)
TEL　(03)5201-5081（ダイヤルイン）
株式会社ルネサスソリューションズ
ツール技術部

製品分類	開発環境	発行番号	TN-CSX-054A	Rev.	第１版
題名	SuperH RISC engine C/C++コンパイラ Ver.7 不具合のご連絡(9)	情報分類	１．仕様変更 ２．ドキュメント訂正追加等 ③．使用上の注意事項 ４．マスク変更 ５．ライン変更		

適用製品	対象ロット等	関連資料	有効期限
P0700CAS7-MWR P0700CAS7-SLR P0700CAS7-H7R	Ver.7.x 台	SuperH RISC engine(SH シリーズ)C/C++ コンパイラ、アセンブラ、最適化リンケージエディタ ユーザーズマニュアル ADJ-702-444A 第 2 版	永年

SuperH RISC engine C/C++コンパイラ Ver.7 に別紙に示す不具合があります。

次に示す製品を御使用のお客様につきましては周知願います。

型名	パッケージバージョン	コンパイラバージョン
P0700CAS7-MWR	7.0B	7.0B
	7.0.01	7.0.03
	7.0.02	7.0.04
	7.0.03	7.0.06
	7.1.00	7.1.00
	7.1.01 7.1.02	7.1.01
	7.1.03	7.1.02
P0700CAS7-SLR	7.0B	7.0B
	7.0.02	7.0.03
	7.0.03	7.0.04
	7.0.04	7.0.06
	7.1.00	7.1.00
	7.1.01 7.1.02	7.1.01
	7.1.03	7.1.02
P0700CAS7-H7R	7.0B	7.0B
	7.0.02	7.0.03
	7.0.03	7.0.04
	7.0.04	7.0.06
	7.1.00	7.1.00
	7.1.01 7.1.02	7.1.01
	7.1.03	7.1.02

なお、チェックツールを以下の URL より入手できます。

http://www.renesas.com/jpn/products/mpumcu/tool/index.html

添付：P0700CAS7-030912J

SuperH RISC engine C/C++コンパイラ Ver.7 不具合内容(9)

SuperH RISC engine C/C++コンパイラ Ver.7
不具合内容(9)

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。
本不具合のチェックツールを以下 URL より入手できます。
http://www.renesas.com/jpn/products/mpumcu/tool/index.html

1. NEG 命令後の不当拡張命令削除

【現象】

unsigned char/short 型変数を含む式が同一関数内で A－B、B－A(A、B：当該変数を含む式)と同時に存在する場合、共通式削除の最適化により不当に拡張命令を削除する場合がある。

【例】

```
unsigned short var_a, var_b, var_c;
long result;

void f() {
   unsigned short x;
   if (var_a >= var_b) {
      x = var_a-var_b;
      result = x * var_c;
   } else {
      x = var_b - var_a;
      result = x * var_c;
   }
}

_f:
       MOV.L      L14,R2     ; _var_a
       MOV.L      L14+4,R4   ; _var_b
       MOV.W      @R2,R5
       MOV.W      @R4,R2
       MOV.L      L14+8,R4   ; _var_c
       MOV        R5,R6
       SUB        R2,R6      ; temp <- var_a-var_b
       MOV.W      @R4,R7
       EXTU.W     R5,R5
       EXTU.W     R2,R2
       CMP/GE     R2,R5
       BF/S       L12
       EXTU.W     R7,R4
       EXTU.W     R6,R2      ; x <- (unsigned short)temp
       MOV.L      L14+12,R5  ; _result
       MUL.L      R2,R4
       STS        MACL,R2
       RTS
       MOV.L      R2,@R5
L12:
       EXTU.W     R6,R6
       MOV.L      L14+12,R5  ; _result
       NEG        R6,R2      ; x <- (long)(-temp)
                             ; EXTU.W R2,R2 を不当に削除
       MUL.L      R2,R4
       STS        MACL,R2
       RTS
       MOV.L      R2,@R5
```

【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 オプションを指定している。
(2) unsigned char/short 型変数を使用している。
(3) (2)の変数を含む式が同一関数内で以下の演算で使用されている。
A－B
B－A
A、B は(2)の変数を含む式
上の例では、A：var_a、B：var_b
(4) 共通式削除の最適化により、(3)の演算が共通化される。
上の例では、以下の通り。

```
void f() {
    unsigned short x;
    long temp = var_a – var_b;
    if (var_a >= var_b) {
        result = (unsigned short)temp * var_c;
                                          /* var_a-var_b を temp に置換 */
    } else {
        result = (unsigned short)(-temp) * var_c;
                                          /* var_b-var_a を-temp に置換 */
    }
}
```

【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 オプション指定する。
(2) 当該変数もしくは当該変数を含む式の代入先変数を volatile 宣言する。
例：

```
void f() {
    volatile unsigned short x;  /* volatile を追加 */
    if (var_a >= var_b) {
        x = var_a-var_b;
        result = x * var_c;
    } else {
        x = var_b - var_a;
        result = x * var_c;
    }
}
```

2. ポインタ変数ロード後の不当拡張命令削除

【現象】

unsigned char/short 型ポインタ変数のポインタ先と 0 との加減算もしくは 1 との乗算を行う式を記述した場合、不当に変数のロード後のゼロ拡張命令を削除する場合がある。

【例】

```
unsigned char *p;
int a;
void func(){
   a = 0;
   a += *p;
}

_func:
       MOV.L      L11,R5     ; _a
       MOV        #0,R2      ; H'00000000
       MOV.L      R2,@R5
       MOV.L      L11+4,R2   ; _p
       MOV.L      @R2,R6
       MOV.B      @R6,R2     ; ロード時に符号拡張
       RTS
       MOV.L      R2,@R5     ; ゼロ拡張せずに 4byte 領域にストア
```

【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 オプションを指定している。
(2) unsigned char/short 型変数をポインタを介して使用している。
(3) (2)の変数と 0 の加減算もしくは 1 との乗算を行っている(最適化により 0 との加減算もしくは 1 との乗算になる場合もあり)。

【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) 明示的に 0 との加減算を行っている場合は、代入式にする。
    例：
```
void func() {
   a = *p;
}
```
(2) optimize=0 を指定する。
(3) 当該変数を volatile 宣言した変数に代入して使用する。
    例：
```
void func() {
   volatile unsigned char temp = *p;
   a = 0;
   a += temp;
}
```

以上