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

（注）本文中のURLは、以下の様に変更になりました。
http://tool-support.renesas.com/jpn/toolnews/shc/shcv7/dr_shcv7_3.html

2003 年　3 月　25 日

― 日立半導体技術情報 ―

〒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)5201-5022（ダイヤルイン）
株式会社 日立製作所 半導体グループ

製品分類	開発環境		発行番号	TN-CSX-049A	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(6)		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	P0700CAS7-MWR P0700CAS7-SLR P0700CAS7-H7R	対象ロット等 全ロット	関連資料	SuperH RISC engine 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-702-304A 第 1 版	有効期限 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	7.1.01
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	7.1.01
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	7.1.01

なお、1,2,3 のチェックツールおよび、4 の修正ファイルを以下の URL より入手できます。

http://www.hitachisemiconductor.com/sic/jsp/japan/jpn/PRODUCTS/MPUMCU/TOOL/

添付：P0700CAS7-030225J

SuperH RISC engine C/C++コンパイラ Ver.7 不具合内容(6)

SuperH RISC engine C/C++コンパイラ Ver.7
不具合内容(6)

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.、2.、３のチェックツールおよび、4.の修正ファイルを以下 URL より入手できます。
http://www.hitachisemiconductor.com/sic/jsp/japan/jpn/PRODUCTS/MPUMCU/TOOL/

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);
}
```

```
_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】
以下の(1)から(4)の条件、もしくは(5)から(8)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) 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。
(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。
(1) 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;
 }

_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】
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) 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】
以下の条件をすべて満たした場合に発生します。
(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。
(1) 移動サイズを 30byte 以下で区切って memmove を行う。

以上