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_1.html

2002 年 4 月 15 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-036A		
題名	SuperH RISC engine C/C++コンパイラ Ver.7.0.04 不具合のご連絡		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	SH-1,SH-2,SH-2E,SH2-DSP, SH-3,SH3-DSP,SH-4	対象ロット等 全ロット	関連資料	―	Rev. 第１版	有効期限 永年

SuperH RISC engine C/C++コンパイラ Ｖｅｒ.7.0.04 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
P0700CAS7-SLR	7.0B	7.0B
	7.0.02	7.0.03
	7.0.03	7.0.04
P0700CAS7-H7R	7.0B	7.0B
	7.0.02	7.0.03
	7.0.03	7.0.04

なお、プログラムが本不具合に該当しているかを検出するチェックツールを以下よりダウンロードできます。

http://www.hitachisemiconductor.com/sic/jsp/japan/jpn/PRODUCTS/MPUMCU/TOOL/download/crosstool/release/rest_shcv7002.html

添付 ： P0700CAS7-020405J
SuperH RISC engine C/C++コンパイラ Ver.7.0.04 不具合内容

SuperH RISC engine C/C++ コンパイラ Ver.7.0.04
不具合内容

本コンパイラの不具合内容を以下に示します。

以下不具合はチェックツールを使用することにより、プログラムに当該ケースが存在するか確認することができます。チェックツールは以下 URL より入手できます。

http://www.hitachisemiconductor.com/sic/jsp/japan/jpn/PRODUCTS/MPUMCU/TOOL/download/crosstool/release/rest_shcv7002.html

1. R0 レジスタの不正破壊

【内容】

スタック渡しのパラメタがあるとき、R0 を不正に書き換える場合がある。

＜例＞

```
 short func1(short a0, int *a1, int a2, short a3, short a4, short a5, short a6,
             int a7, int a8, int a9);
 void func0(short a0, int *a1, int a2, short a3, short a4, short a5, short a6)
 {
                      :
    r1=func1(0,a1,0,0,0,0,0,0,0,0);
    if((r1>0)&&(r1!=1)) {
       func1(a0,a1,0,a3,a4,a5,a6,0,0,0);
    }
                      :
 }

                      :
    MOV.L      R0,@(32,R15)  ; -> R0 を@(32,R15)に退避
    MOV        R8,R5
    MOV        #66,R0        ; -> R0 を破壊
    MOV.W      @(R0,R15),R3
    MOV        R9,R6
    MOV        #70,R0        ; -> R0 を破壊
    MOV.W      @(R0,R15),R1
    MOV        R0,R4         ; ->MOV.L @(32,R15),R4をMOV R0,R4に置換えコード不
正
    MOV.L      R3,@(4,R15)
    MOV.L      R1,@(8,R15)
    MOV.L      R9,@(12,R15)
    MOV.L      R9,@(16,R15)
    BSR        _func1
    MOV.L      R9,@(20,R15)
                      :
```

【発生条件】

以下の条件をすべて満たす場合、発生することがあります。

(1) optimize=1 を指定している。

(2) 当該関数にスタック渡しのパラメタが存在する。

【回避方法】

以下のいずれかの方法で当該ケース回避することができます。

(1) チェックツールによって当該ケースが見つかったプログラムを optimize=0 でコンパイルする。

(2) スタック渡しパラメタを関数先頭でローカル変数にコピーし、直後に nop()を入れる。関数内ではコピーしたローカル変数のみ参照する。

＜例＞

```
#include <machine.h> /* for nop() */

void func0(short a0,int *a1,int a2,short a3,short a4,short a5,short a6)
{
    short r1;

    /* 関数内処理の先頭でスタック渡しパラメタをローカル変数へコピー */
    short tmp4=a4,tmp5=a5,tmp6=a6;

    /* その直後に nop()を挿入 */
    nop();

    /* これ以降ではスタック渡しパラメタを参照せずコピーしたものを参照する */
                    :
   r1=func1(0,a1,0,0,0,0,0,0,0,0);
   if((r1>0)&&(r1!=1)) {
       /* a4-a6をtmp4-tmp6へ変更 */
       func1(a0,a1,0,a3,temp4,temp5,temp6,0,0,0);
   }
                    :
}
```

2. BRA 命令飛び先不正

【内容】

無条件分岐を含むプログラムにおいて、分岐を BRA 命令で行いかつ飛び先までの距離が 4094 バイトの時、飛び先が不正になるコードを生成する場合がある。

【発生条件】

以下の条件をすべて満たす場合、発生することがあります。

(1) code=machinecode を指定している。または code オプションを指定していない。

(2) BRA 命令から飛び先までの距離が 4094 バイトである。

【回避方法】

以下のいずれかの方法で当該ケースを回避することができます。

(1) チェックツールによって当該ケースが見つかったプログラムを code=asmcode 指定してコンパイルする。

(2) チェックツールによって当該ケースが見つかった関数に nop()を挿入することにより回避できる場合があります。

＜例＞

```
void func(int a) {
    if (a) {
        :      /*  判定が偽の時、4094 バイト先へ飛ぶ BRA 命令になる  */
    }
}
```

↓

```
#include <machine.h>    /* for nop() */
void func(int a) {
    if (a) {
        :
        nop();    /* nop()を挿入する  */
    }
}
```