

H8SX, H8S およびH8ファミリ用C/C++ コンパイラパッケージ V.6 ご使用上のお願い

H8SX, H8S およびH8ファミリ用C/C++コンパイラパッケージ V.6の使用上の注意事項 2件を連絡します。

- volatile修飾のある局所変数およびvolatile修飾のない大域変数に関する注意事項 (H8C-0084)
- 構造体および共用体型変数のメンバ値に関する注意事項 (H8C-0085)

1. volatile修飾のある局所変数およびvolatile修飾のない 大域変数に関する注意事項 (H8C-0084)

1.1 該当製品

H8SX, H8SおよびH8ファミリ用C/C++コンパイラパッケージ
V.6.00 Release 00 ~ V 6.02 Release 01

1.2 内容

最適化が有効の場合に、volatile修飾した構造体型または共用体型局所変数のメンバへのアクセスコードおよびvolatile修飾のない大域変数への代入コードが生成されません。

1.3 発生条件

以下の条件をすべて満たす場合に発生します。

(1) 以下のいずれかのCPU種別を選択している。

H8SXX, H8SXA, H8SXM, H8SXXN, AE5, 2600A, 2600N, 2000A, 2000NおよびRS4

(つまり、CPU種別に対応したcpuオプションが、-cpu=h8sxx, =h8sxa, =h8sxm, =h8sxn, =ae5, =2600a, =2600n, =2000a, =2000nおよび =rs4から選択されている場合)

(2) CPU種別が2000N, 2000A, 2600N, 2600Aの場合に、出力オブジェクト互換オプション(-legacy=v4)を選択していない。

(3) 最適化オプションに最適化あり(-optimize=1)を選択している。

(4) (4-1)または(4-2)のいずれかの条件を満たしている。

- (4-1) 以下の(4-1-1)から(4-1-4)のすべての条件を満たしている。
(4-1-1) volatile修飾したメンバをもつサイズが4バイト以下の構造体または共用体を宣言している。
(4-1-2) (4-1-1)の構造体または共用体型局所変数を定義している。
(4-1-3) (4-1-2)の変数のメンバが式中で参照されている。
(4-1-4) (4-1-3)の式内でポインタ(演算子"->")を使用していない。

- (4-2) 以下の(4-2-1)から(4-2-6)のすべての条件を満たしている。
(4-2-1) 外部変数の最適化なし(-volatile)を選択していない。
(4-2-2) 最適化範囲の分割なし(-noscope)を選択していない。
(4-2-3) volatile修飾していない大域変数への代入式がある。
(4-2-4) 関数の最適化範囲が複数に分割される。
(messageオプションを選択し、C0101メッセージ出力で確認可能)
(4-2-5) (4-2-4)の関数内に初期化式、終了条件式および制御式が無いfor文の無限ループがある。
(4-2-6) (4-2-5)の無限ループの前に、if文またはgoto文がある。

発生例1:

```
-----  
// -cpu=h8sxa, -optimize=1  
struct st {  
    volatile unsigned char c;          // 発生条件(4-1-1)  
};  
void main( void )  
{  
    struct st sti;                    // 発生条件(4-1-2)  
    sti.c = 'a';                      // 発生条件(4-1-3)および(4-1-4)  
}
```

コンパイル結果1:

```
-----  
_main:  
    ;volatileで修飾したメンバ(sti.c)へのアクセスコードがない。  
    RTS  
    .END  
-----
```

発生例2:

```
-----  
// -cpu=h8sxa -optimize=1  
int x, y, n;
```

```

#define INC_N n++;
#define INC_N10 INC_N;INC_N;INC_N;INC_N;INC_N;¥
INC_N;INC_N;INC_N;INC_N;INC_N;
#define INC_N1H INC_N10;INC_N10;INC_N10;INC_N10;INC_N10;¥
INC_N10;INC_N10;INC_N10;INC_N10;INC_N10;
#define INC_N1T INC_N1H;INC_N1H;INC_N1H;INC_N1H;INC_N1H;¥
INC_N1H;INC_N1H;INC_N1H;INC_N1H;INC_N1H;

void main(void)
{
    x = 1; //発生条件(4-2-3)
    if (y) { //発生条件(4-2-6)
        y = 2; //発生条件(4-2-3)
    }
    INC_N1T; //発生条件(4-2-4)
    for (;;) //発生条件(4-2-5)
}

```

コンパイル結果2:

```

_main:
    .STACK    _main=4
             ;外部変数xへの代入処理コードがない。
    MOV.W    @_y:32,R0
             ;外部変数yへの代入処理コードがない。
    ADD.W    #H'03E8:16,@_n:32
L21:
    BRA     L21:8

```

1.4 回避策

以下のいずれかの方法で回避してください。

発生条件(4-1)を満たす場合:

以下のいずれかの方法で回避してください。

- (1) 最適化オプションに最適化なし(-optimize=0)を選択するか、当該関数に対して#pragma option nooptimizeを指定してください。
- (2) 構造体および共用体型の変数定義時にvolatile修飾する。

例:

```

struct st { volatile unsigned char c; };

```

```
void main( void )
{
    volatile struct st sti;//変数定義時にvolatile修飾する
    sti.c = 'a';
}
```

発生条件(4-2)を満たす場合:

以下のいずれかの方法で回避してください。

- (1) 最適化なし(-optimize=0)を選択する。
- (2) cpu種別に2000N, 2000A, 2600Nまたは2600Aを指定時に、出力オブジェクト互換オプション(-legacy=v4)を選択する。
- (3) 外部変数の最適化なし(-volatile)を選択する。
- (4) 最適化範囲の分割なし(-noscope)を選択する。
- (5) 終了条件式を記載したfor文またはwhile文で無限ループを記述する。

例: for (;1;) または while(1)

2. 構造体および共用体型変数のメンバ値に関する注意事項 (H8C-0085)

2.1 該当製品

H8SX, H8SおよびH8ファミリ用C/C++コンパイラパッケージ
V.6.01 Release 03 ~ V 6.02 Release 01

2.2 内容

レジスタに格納された構造体型変数のメンバの値が変更されることがあります。

2.3 発生条件

以下の条件をすべて満たす場合に発生する可能性があります。

- (1) 以下のいずれかのCPU種別を選択している。
H8SXX, H8SXA, H8SXM, H8SXN, AE5, 2600A, 2600N, 2000A, 2000NおよびRS4
(つまり、CPU種別に対応したcpuオプションが、-cpu=h8sxx, =h8sxa, =h8sxm, =h8sxn, =ae5,2600a, =2600n, =2000a, =2000nまたは =rs4から選択されている場合)
- (2) 最適化あり(-optimize=1)を選択している。
- (3) 以下のすべての条件を満たす構造体または共用体型変数がある。
 - (3-1) 2つ以上のメンバがある。
 - (3-2) サイズが4バイト以下である。
 - (3-3) 局所変数または関数の引数(レジスタ渡し)である。
- (4) (3)の変数の任意のメンバへの代入が2回以上ある。
- (5) (4)の代入間に関数呼び出し式がある。
- (6) (5)の関数呼び出し式の後に、(3)の変数にアクセスしないフローがある。

発生例:

```
//-cpu=h8sxa, -optimize=1
char x, y, z;
struct ST{ char a; char b; }; //発生条件(3-1)および(3-2)
sub(){}
sub2(){}

void main(void){
    struct ST m; //発生条件(3-3)
    if (x == 1) {
        m.a = 'C'; //発生条件(4)
    } else {
        m.a = 'A'; //発生条件(4)
    }

    if (y) {
        if(m.a == 'C') {
            sub(); //発生条件(5)
            return; //発生条件(6)
        }
    }

    if (!z) {
        m.b = 'B'; //発生条件(4)
    } else {
        m.b = 1; //発生条件(4)
    }

    if (y == m.b) {
        sub2();
    }
}
```

コンパイル結果:

```
; m.aの値はR0H, m.bの値はR0Lに格納されている。
; if (!z) {
L38:
    MOV.B    @_z:32,R1H
    BNE     L42:8
```

```
; m.b = 'B';
    MOV.W    @SP,R0 ; R0Hに格納されたm.aの値が書き換えられる。
    BRA/S    L43:8
    MOV.B    #H'42:8,R0L
; } else {
-----
```

2.4 回避策

以下のいずれかの方法で回避してください。

(1) 構造体型変数のサイズを4バイトより大きくする。

例： `struct ST{ char a; char b; char dummy[3]; };`

(2) 構造体または共用体型の定義の際に変数をvolatile修飾する。

(3) 最適化なし(-optimize=0)を選択するか、#pragma option nooptimizeを使用して関数単位で最適化を抑止する。

3. 恒久対策

本内容は、2009年9月1日にリリースしたV.7.00 Release 00 で改修しています。

V.7.00 Release 00の詳細は、RENASAS TOOL NEWS 資料番号090901/tn3 を参照ください。以下のURLでも参照できます。(9月7日から公開予定)

<http://tool-support.renesas.com/jpn/toolnews/090901/tn3.htm>

なお、V.6台では次期リビジョンで改修する予定です。

[免責事項]

過去のニュース内容は発行当時の情報をもとにしており、現時点では変更された情報や無効な情報が含まれている場合があります。ニュース本文中のURLを予告なしに変更または中止することがありますので、あらかじめご承知ください。