H8S,H8/300 シリーズ C/C++コンパイラ V.6.01.00 で改修した問題

1. int 型のオーバフローを伴う演算(H8C-0004)

int 型の変数においてオーバフローを伴う演算において、演算結果が異なる場合がある。
該当バージョン：V.6.00 Release 00 ～ 03

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。
(1) CPU オプションに、H8SXA または H8SXX(-cpu=h8sxa または h8sxx)を選択している。
(2) 最適化オプション(-optimize=1)を選択している。
(3) int または short 型の変数を加算し、その結果に対して unsigned long 型で乗算している式がある。
(4) (3)の加算は、オーバフローを伴う演算である。

[例]
```
------------------------------------------------------------
signed int sub(){
    return 32767;
}
unsigned long ul1,ul2;
main(){
    signed int i1,i2;
    i1 = sub();
    ul1 = 2;
    i2 = 1;
    ul2 = ul1 * (i1 + i2); // ul2=2*(unsigned long)(32767+1)
}
------------------------------------------------------------
```

2. 構造体配列サイズが 32767 バイトを超えたときのメンバアクセス(H8C-0005)

32767 バイトを超える構造体型の配列をアクセスした時、不正なデータをアクセスする場合がある。
該当バージョン：V.6.00 Release 00 ～ 03

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。
(1) CPU オプションに、H8SXN, H8SXM, H8SXA または H8SXX(-cpu=h8sxn, h8sxm, h8sxa, または h8sxx)
を選択している。
(2) 構造体のサイズが、2 バイトまたは 4 バイトである。
(3) (2)の構造体の配列のサイズが、32767 バイトを超えている。

[例]
```
------------------------------------------------------------
struct st2{
    char a;
    char b;
}st_2[32767];
void main(void){
    char i;
    for(i = 0 ; i < 10 ; i++){
        st_2[i].b = i;
```

```
            // 構造体内の指定領域とは異なる領域に値を設定
    }
}
```

---

3. ループ文内での配列アクセス(H8C-0006)

ループ文内で配列をアクセスすると、不正な要素をアクセスする場合がある。
該当バージョン: V.4.0 ～ V.5.0.06

発生条件

以下の(1), (2), (4), および(5) または (1), (3), (4), および(6)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) CPU オプションに、300HA, 2000A, 2600A, H8SXX または H8SXA(-cpu=300ha, 2000a, 2600a, h8sxx, または h8sxa)を選択している。
(2) 300HA, 2000A, または 2600A の場合:最適化(-optimize=1)オプション、スピードオプション(-speed, -speed=loop[1|2])を選択している。
(3) H8SXX または H8SXA の場合 : ポインタサイズ指定オプション(-ptr16)を選択している。
(4) unsigned short または unsigned int 型の変数を宣言している。
(5) 300HA, 2000A, または 2600A の場合 : (4)の変数を以下のように使用している。
    ・ループ変数の初期化時、定数で初期化している。
    ・ループ内のループ変数として使用している。
    ・配列の添え字として（定数-変数）の形で使用している。
(6) H8SXX または H8SXA の場合 : (4)の変数を配列の添え字として(定数-変数)の形で使用している。または、(定数+<式>)の<式>が負数である。

例 1:

```
extern unsigned char a[20];
void sub(void){
    unsigned short j;
    for(j=0; j<10; j++){
        a[20-j] = 10; //配列領域外を参照
    }
}
```

例 2:

```
unsigned short a[20];
unsigned short j;
void sub3(void){
    a[20-j] = 10;
}
```

---

4. switch 文のテーブル展開コード(H8C-0007)

テーブル方式で展開された switch 文の中と外で、同じ変数に同一値を設定すると、その変数の設定値が不定値になる場合がある。

該当バージョン：V.6.00 Release 00 ～ 03

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。
(1) CPU オプションに、H8SXA または H8SXX(-cpu=h8sxa,h8sxx)を選択している。
(2) switch 文展開方式オプションで、自動(-case=auto)でテーブル方式に展開されているコードが存在する。またはテーブル展開方式(-case=table)を選択している。
(3) 設定する変数が char,unsigned char,short,unsigned short,int または unsigned int の型の配列である。
(4) 同じ値の定数が複数箇所で使用されており、switch 文の前と switch 文の中で使用されている。

5. 可変個の引数を持つ関数の引数割り付け (H8C-0008)

構造体パラメタのレジスタ割り付けオプション(-structreg)を指定し、4 バイト以下の構造体を引数に指定すると、スタックで渡されるべき引数がレジスタ渡しとなる場合がある。
該当バージョン：V.6.00 Release 00 ～ 03

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。
(1) CPU オプションに、H8SXN, H8SXM, H8SXA または H8SXX(-cpu=h8sxn,h8sxm, h8sxa, または h8sxx)を選択している。
(2) 構造体パラメタのレジスタ割り付けオプション(-structreg)が選択されている。
(3) 可変個の引数を持つ関数が存在し、4 バイト以下の構造体変数が引数にある。
(4) (3)の構造体はスタックを介して関数呼び出しが実行される。
(5) (3)の構造体を格納できる引数割り付け用のレジスタに空きがある。

[例]
```
------------------------------------------------------------------
struct A{
    int is_keyword ;
} flags;
void sub(const char *,...);
void main(void){
    sub("test" flags); // flags の内容をレジスタにコピー
}
------------------------------------------------------------------
```

6. switch 文の table 展開時のセクション名(H8C-0009)

短絶対アドレスオプションが選択され、table 方式に展開される switch 文が存在していると、セクション名が不正になる場合がある。
該当バージョン: V.4.0 ～ V.5.0.06

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。

(1) CPU オプションに、300HN, 2000N または 2600N(-cpu=300hn, 2000n, または 2600n)を選択している。
(2) 短絶対アドレスオプション(-abs16)を選択している。
(3) switch 文展開方式オプションで、自動(-case=auto)でテーブル方式に展開されているコードが存在する。またはテーブル展開方式(-case=table)を選択している。
(4) オブジェクト形式オプションで、機械語出力(-code=machinecode)を選択している。

[例]

```
------------------------------------------------------------
char c;
void func(void){
    switch (c) {
        case 0:
            c-=2;
            break;
        case 1:
            c--;
            break;
        case 2:
            c++;
            break;
        case 3:
            c+=2;
            break;
        case 4:
            c+=3;
            break;
        case 5:
            c+=4;
            break;
    }
}
------------------------------------------------------------
```

7. speed 優先時の乗算の演算(H8C-0010)

long 型変数と int 型変数との乗算を行うと、不正なオブジェクトが生成される、または内部エラーが出力される場合がある。
該当バージョン: V.4.0 ～ V.5.0.06

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。
(1) CPU オプションに、300HN, 300HA, 2000N, 2000A, 2600N, または 2600A(-cpu=300hn, 300ha, 2000n, 2000a, 2600n, または 2600a)を選択している。
(2) スピード優先最適化オプション(-speed)またはスピード優先最適化の四則演算、比較、代入式の高速化オプション(-speed=expression)を選択している。
(3) long 型変数と short 型または int 型との乗算を行なっている。
(4) 乗算の 1 項および 2 項が、いずれも関数呼び出しまたは演算式である。

[例]

```
------------------------------------------------------------
long  l;
short func_s(void);
```

```
long  func_l(void);
void func(void){
  l = func_s() * func_l();
      // short 型と long 型の乗算
}
```
---

8. __asm{}内での DATA 制御命令(H8C-0011)

__asm{}内で.DATA.B/.DATA.W で変数を定義する際、記述した、DATA 制御命令のサイズよりも大きい値を記述すると、設定される値が不正となる場合がある。

該当バージョン：V.6.00 Release 00 ～ 03

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。

(1) 埋め込みアセンブリ機能(__asm)内で、.DATA 制御命令を記述している。

(2) (1)の.DATA 制御命令は、.DATA.B/.DATA.W で宣言されている。

(3) (1)の.DATA 制御命令の整数データが、宣言した型のサイズを超えている。

[例]

```
void func(void){
    __asm{
        L1: .data.w "0x12345678"
            //  実際には 0xffff までしか表現できません。
        L2: .data.b "0x1234"
            //  実際には 0xff までしか表現できません。
    }
}
```
---

9. オーバフロー判定の組み込み関数使用時のアクセス(H8C-0012)

組み込み関数 ovfadd または ovfsub を使用すると、不当にオーバフローと出力される場合がある。

該当バージョン：V.6.00 Release 00 ～ 03

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。

(1) 組み込み関数 ovfadd または ovfsub を使用している。

(2) 以下の演算を実施している。

- ovfadd ： 正の最大値 + 負の最大値
- ovfsub ： 正の最大値 - 負の最大値

注: 正の最大値 = 2147483647(0x7FFFFFFF)

負の最大値 = -2147483648(0x80000000)

[例]

```
#include<stdio.h>
#include<machine.h>
```

```
void main(void){
    if (!ovfaddl(0x7fffffff,0x80000000,0)){
        printf("OK¥n");
    } else {
        printf("NG¥n");
    }
}
```

---

10. 構造体型へのポインタ代入(H8C-0013)

メンバの属する構造体型へのポインタをメンバに代入すると、不正なデータをアクセスする場合がある。
該当バージョン：V.6.00 Release 00 ～ 03

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。
(1) CPU オプションに、H8SXN,H8SXM,H8SXA または H8SXX(-cpu=h8sxn,h8sxm,h8sxa,h8sxx)を選択している。
(2) 最適化オプション(-optimize=1)を選択している。
(3) 構造体のメンバの型が、そのメンバの属している構造体型へのポインタである。
(4) (3)のメンバに、そのメンバが属している構造体のアドレスを代入している使用している。

[例]

```
struct data {
   struct data *p ;
};
struct data *P1, *P2 ;
void fnc( void ){
   P1->p = P2->p = P2 ;
}
```

---

11. cpuexpand 指定時の乗算結果(H8C-0014)

cpuexpand 指定時に、乗算結果が不正になる場合がある。
該当バージョン：V6.00 Release 00 ～ 03

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。
(1) CPU オプションに、H8SXN,H8SXM,H8SXA または H8SXX(-cpu=h8sxn,h8sxm,h8sxa,h8sxx)を選択している。
(2) 乗除算の拡張解釈オプション(-cpuexpand)を選択している。
(3) 乗算式がある

[例]

```
void  f (void){
   int a,b;
   long c,d;
```

```
    a=b=d=0;
    ++d;
    ++a;
    b+=2;
    c=(volatile long)(a*b);// a*a を計算
    ++d;
    if ((d==(volatile long)(a*b)) && (c==2)){
        printf("t026_04 OK¥n");
    } else {
        printf("t026_04 NG¥n");
    }
}
```

---

12. abs16 指定時のセクション名不一致(H8C-0015)

テーブル方式で展開される switch 文が含まれるソースを、H8SXN または H8SXM の CPU 用にコンパイルすると、ABS16 の接頭語の付いたセクションが出力される場合がある。
該当バージョン：V.6.00 Release 00 ～ 03

発生条件

以下の条件をすべて満たした場合、発生することがあります。
(1) CPU オプションに、H8SXN または H8SXM(-cpu=h8sxn,h8sxm)を選択している。
(2) 短絶対アドレスオプション(-abs16)を選択している。
(3) switch 文展開方式オプションで、自動(-case=auto)でテーブル方式に展開されているコードが存在する。またはテーブル展開方式(-case=table)を選択している。

[例]

```
void main(void){
switch(l){
    case -3:
    case -1:
        l = 10;
        break;
    }
}
```

---

13. 構造体代入(H8C-0016)

構造体メンバの代入において不正な値が代入される場合がある。
該当バージョン：V.6.00 Release 00 ～ 03

発生条件

以下の条件をすべて満たした場合、発生することがあります。
(1) CPU オプションに、H8SXX,H8SXA,H8SXN または H8SXM(-cpu=h8sxx,h8sxa,h8sxn,h8sxm)を選択している。
(2) 最適化オプション(-optimize=1)を選択している。

(3) 構造体サイズが 4 バイト以下(各メンバのサイズは 4 バイト未満)の変数がレジスタに割り付けられている。
(4) (3)の変数のメンバを使用している。
(5) 変数を割り付けるために、レジスタの内容をスタック退避する。
(6) (5)で退避されるレジスタには、(3)の変数が割り付けられている。
(7) (6)で退避された変数にアクセスする式がある。

[例]

```
struct _ST{
   char c;
   int i;
}st1;
void f(){
   struct _ST lst1;
   . . . . . .
   lst1.i=1;
   . . . . . .
   switch(x){
       . . . . . .
       switch(y){
           case 3:
               . . . . . .
               lst1.i =2;
               . . . . . .
               break;
       }
       . . . . . .
   }
   . . . . . .
   st1=lst1;
   . . . . . .
}
```

14. その他

(1) try-catch を使用したプログラムに対してコンパイルまたはリンク時にエラーを出力する場合があります。
(2) コンパイル時に(C)4098 を出力する場合があります。
(3) デバッガの Watch または Locals ウィンドウに不正な値を表示する場合があります。
(4) スタック解析ツールで"ドレス参照未解決関数"と表示する場合があります。

株式会社 日立超LSIシステムズ