RENESAS Tool News

RENESAS TOOL NEWS on September 1, 2009: 090901/tn2

Notes on Using the C/C++ Compiler Package for the H8SX, H8S, and H8 MCU Families V.6

Please take note of the following problems in using the C/C++ compiler package for the H8SX, H8S, and H8 MCU families V.6:

- With volatile-qualified local variables and non-volatile-qualified global variables (H8C-0084)
- With member values of structure-type or union-type variables (H8C-0085)

1. Problem with Volatile-Qualified Local Variables and Non-Volatile-Qualified Global Variables (H8C-0084)

1.1 Product and Versions Concerned

C/C++ compiler package for the H8SX, H8S, and H8 families
V.6.00 Release 00 through V.6.02 Release 01

1.2 Description

When optimization is effective, the access codes to members of structure-type or union-type local variables and the assignment codes to non-volatile-qualified global variables cannot be generated.

1.3 Conditions

This problem arises if the following conditions are all satisfied:
(1) Any one of the following CPU type is selected:
H8SXX, H8SXA, H8SXM, H8SXN, AE5, 2600A, 2600N, 2000A, 2000N, and RS4.
That is, the cpu option corresponding to the CPU type is selected out of the following:
-cpu=h8sxx, =h8sxa, =h8sxm, =h8sxn, =ae5, =2600a, =2600n, =2000a, =2000n, and =rs4
(2) When 2000N, 2000A, 2600N, or 2600A is selected as the CPU type, the Compatibility of output code option (-legacy=v4) is not selected.

(3) The optimizing option -optimize=1 (optimization is effective) is selected.
(4) Either (4-1) or (4-2) below is met.
(4-1) The following four conditions are all satisfied:
(4-1-1) A structure or union having a volatile-qualified member is declared. Here, the size of the structure is 4 bytes or less.
(4-1-2) A local variable of the structure or union type in (4-1-1) is defined.
(4-1-3) A member of the variable in (4-1-2) is referenced in an expression.
(4-1-4) No pointer (operator "->") is used in the expression in (4-1-3).
(4-2) The following six conditions are all satisfied:
(4-2-1) Option -volatile (external variables not optimized) is not selected.
(4-2-2) Option -noscope (optimized range not split) is not selected.
(4-2-3) An assignment is made to a non-volatile-qualified global variable.
(4-2-4) The optimized range of a function is split into two or more. (To examine the splitting, select the message option to output the C0101 message.)
(4-2-5) In the function in (4-2-4) exists the infinite loop of a for statement that has no initialization expression, loop exit condition, and loop control expression.
(4-2-6) An if or goto statement is placed before the infinite loop in (4-2-5).

Example 1: (Condition (4-1) satisfied)

```
------------------------------------------------------------------
// -cpu=h8sxa, -optimize=1
struct st {
  volatile unsigned char  c;       // Condition (4-1-1)
};
void main( void )
{
  struct st sti;                   // Condition (4-1-2)
  sti.c = 'a';                     // Conditions (4-1-3) and (4-1-4)
}
------------------------------------------------------------------
```

Result of compilation 1:

```
------------------------------------------------------------------
_main:
      ; Access code to volatile-qualified member (sti.c) lost.
   RTS
   .END
------------------------------------------------------------------
```

Example 2: (Condition (4-2) satisfied)

```
------------------------------------------------------------------
//-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;    // Condition (4-2-3)
  if (y) {  // Condition (4-2-6)
    y = 2;  // Condition (4-2-3)
  }
  INC_N1T;  // Condition (4-2-4)
  for (;;); // Condition (4-2-5)
}
------------------------------------------------------------------
```

Result of compilation 2:

```
------------------------------------------------------------------
_main:
   .STACK    _main=4
             ; Code for making assignment to external variable x lost.
   MOV.W     @_y:32,R0
             ; Code for making assignment to external variable y lost.
   ADD.W     #H'03E8:16,@_n:32
L21:
   BRA       L21:8
------------------------------------------------------------------
```

1.4 Workarounds

To avoid this problem, perform either of the following procedures:

Procedure 1. (Condition (4-1) is satisfied)
Avoid this problem in either of the following ways:
(1) Select the optimizing option -optimize=0 (optimization not effective),
or issue #pragma option nooptimize in order not to optimize the function involved.
(2) When defining a local variable of the structure or union type, qualify it to be volatile.

Example:

```
struct st { volatile unsigned char  c; };
void main( void )
{
  volatile struct st sti ; // Variable qualified to be volatile
                           // when defined.
  sti.c = 'a';
}
```

Procedure 2. (Condition (4-2) satisfied)
Avoid this problem in either of the following ways:
(1) Select the optimizing option -optimize=0 (optimization not effective).
(2) When 2000N, 2000A, 2600N, or 2600A is selected as the CPU type, select the Compatibility of output code option (-legacy=v4).
(3) Select option -volatile (external variables not optimized).
(4) Select option -noscope (optimized range not split).
(5) Use a for or while statement having the loop exit condition to make an infinite loop.
Example: for (;1;) or while (1)

2. Problem with Member Values of Structure-Type or Union-Type Variables (H8C-0085)

2.1 Product and Versions Concerned

C/C++ compiler package for the H8SX, H8S, and H8 families
V.6.01 Release 03 through V 6.02 Release 01

2.2 Description

Member values of structure-type variable stored in register may be rewritten.

2.3 Conditions

This problem may arise if the following conditions are all satisfied:

(1) Any one of the following CPU type is selected:
H8SXX, H8SXA, H8SXM, H8SXN, AE5, 2600A, 2600N, 2000A, 2000N, and RS4.
That is, the cpu option corresponding to the CPU type is selected out of the following:
-cpu=h8sxx, =h8sxa, =h8sxm, =h8sxn, =ae5, =2600a, =2600n, =2000a, =2000n, and =rs4

(2) The optimizing option -optimize=1 (optimization is effective) is selected.

(3) In the program exists a structure-type or union-type variable that satisfies the following three conditions:
(3-1) The variable has 2 or more members.
(3-2) The size of the variable is 4 bytes or less.
(3-3) The variable is a local variable or an argument passed to a function via a register.

(4) Assignments are made to at any of the members of the variable in (3) twice or more.

(5) Between the assignments in (4), a function call is made.

(6) A program flow that does not access the variable in (3) is placed after the function call in (5).

Example:

```
----------------------------------------------------------------------
//-cpu=h8sxa, -optimize=1
char x, y, z;
struct ST{ char a; char b; }; // Conditions (3-1) and (3-2)
sub(){}
sub2(){}

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

```
    if (y) {
        if(m.a == 'C') {
            sub();  // Condition (5)
            return; // Condition (6)
        }
    }

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

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

Result of compilation:

```
; Value of m.a has been stored to R0H, and that of m.b to R0L.
; if (!z) {
    L38:
        MOV.B     @_z:32,R1H
        BNE       L42:8
; m.b = 'B';
        MOV.W     @SP,R0 ; Value of m.a stored in R0H rewritten.
        BRA/S     L43:8
        MOV.B     #H'42:8,R0L
; } else {
```

2.4 Workarounds

To avoid this problem, use any of the following ways:

(1) Make the size of the structure-type variable greater than 4 bytes.

(2) When defining the structure- or union-type variable, qualify it to be volatile.

(3) Select option -optimize=0 (optimization not effective), or issue #pragma option nooptimize to suppress optimization, function by function.

3. Schedule of Fixing the Problems

The above problems have already been fixed in the C/C++ compiler package for the H8SX, H8S, and H8 MCU families V.7.00 Release 00, which was released on September 1, 2009.
For details of the V.7.00 Release 00 product, see RENESAS TOOL NEWS Document No. 090901/tn3 at:
http://tool-support.renesas.com/eng/toolnews/090901/tn3.htm
This Web page will be opened from September 7 on.

And also, we plan to fix these problems in the C/C++ compiler package for the H8SX, H8S, and H8 MCU families V.6.02 Release 02.
