Date: Jan.27.2004

RENESAS TECHNICAL UPDATE

Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan
RenesasTechnology Corp.

Product Category	User Development Environment		Document No.	TN-CSX-061A/EA	Rev.	1.0
Title	SuperH RISC engine C/C++ Compiler Ver.8 bug information		Information Category	Usage Limitation		
Applicable Product	P0700CAS8-MWR P0700CAS8-SLR P0700CAS8-H7R	Lot No. Ver.8.0.00	Reference Document	SuperH RISC engine C/C++ Compiler Assembler Optimizing Linkage Editor User's Manual REJ10B0047-0100H		

Attached is the description of the detected bug information in Ver. 8 series of the SuperH RISC engine C/C++ Compiler.

The bug will affect this package version.

Attached: P0700CAS8-040120E

SuperH RISC engine C/C++ Compiler Ver. 8.0.00 The details of the detected bug information

SuperH RISC engine C/C++ Compiler Ver. 8.0.00
The details of the detected bug information

The bug detected in the ver. 8.0.00 of the SuperH RISC engine C/C++ Compiler is shown below.

1. Illegal bit field member comparison

[Description]

If compared to a 1-bit bit field member of the volatile-qualified structure referenced via a pointer with a constant, the compared result may be incorrect.

[Example]

```
typedef volatile struct {  /* The type of the structure is volatile-qualified */
    unsigned int a:19,
                 b:1;      /* The member is not volatile-qualified */
} ST;

int f(ST *p) {
    if (p->b) {            /* Expression p->b is used for an indirect reference via a pointer */
        return 1;
    }
    return 0;
}
```

```
         MOV.L    @R4,R0
         TST      #16,R0   ; Illegal mask value
         MOVT     R2
         MOV      R2,R0
         RTS
         XOR      #1,R0
```

[Conditions]

This problem may occur when all of the following conditions are fulfilled.

(1) optimize=1 is specified.
(2) A conditional expression in an if statement includes an expression to compare the bit field to a constant.
(3) The bit field referred in expression(2) is an indirection reference via a pointer.
(4) Comparison between a constant 0 and an unsigned-type 1-bit bit field member by "==" or "!=" expression, or comparison between a constant 1 and an unsigned-type 1-bit bit field member by "==" expression.
(5) The type of the structure is volatile-qualified but the member is not.
(6) The type of pointer is not volatile-qualified.
(7) The pointer is a scalar variable.
(8) The address of the pointer is not referred to.
(9) The type of the bit field is (unsigned) (short | int | long).

[Solutions]

This problem can be prevented by taking any of the following ways.

(1) volatile-qualify the type of the bit field member.
(2) Place an expression to refer to the address of the pointer within the same file.
(3) Use expression (*&p)->b, not p->b, for an indirection reference via a pointer.
(4) Specify optimize=0.