Date: Jul.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-080A/EA	Rev.	1.0
Title	SuperH RISC engine C/C++ Compiler ver.7 Known Bug Report(13)		Information Category	Usage Limitation		
Applicable Product	P0700CAS7-MWR P0700CAS7-SLR P0700CAS7-H7R	Lot No. Ver.7.x	Reference Document	SuperH RISC engine C/C++ Compiler Assembler Optimizing Linkage Editor User's Manual REJ10B0047-0100H Rev.1.00		

Attached is the description of the known bug in Ver. 7 series of the SuperH RISC engine C/C++ compiler.

The bug will affect the package version in the table below.

	Package Version	Compiler Version
P0700CAS7-MWR	7.0B	7.0B
	7.0.01	7.0.03
	7.0.02	7.0.04
	7.0.03	7.0.06
	7.1.00	7.1.00
	7.1.01 7.1.02	7.1.01
	7.1.03	7.1.02
	7.1.04	7.1.03
P0700CAS7-SLR	7.0B	7.0B
	7.0.02	7.0.03
	7.0.03	7.0.04
	7.0.04	7.0.06
	7.1.00	7.1.00
	7.1.01 7.1.02	7.1.01
	7.1.03	7.1.02
	7.1.04	7.1.03
P0700CAS7-H7R	7.0B	7.0B
	7.0.02	7.0.03
	7.0.03	7.0.04
	7.0.04	7.0.06
	7.1.00	7.1.00
	7.1.01 7.1.02	7.1.01
	7.1.03	7.1.02
	7.1.04	7.1.03

The check tool can be downloaded from the following URL.

http://www.renesas.com/eng/products/mpumcu/tool/index.html

Attached: P0700CAS7-040722E

SuperH RISC engine C/C++ Compiler Ver. 7 Known Bug Report (13)

SuperH RISC engine C/C++ Compiler ver.7
Known Bug Report(13)

The bug detected in the ver.7 of the SuperH RISC engine C/C++ Compiler is shown below.
The check tool can be downloaded from the following URL:
http://www.renesas.com/eng/products/mpumcu/tool/index.html

1. Incorrect replacement of loop induction variable (SHC-0003)
[Description]
When loop induction variables existed and their type differs others in a loop, they might be commonized incorrectly.

[Example]
```
extern void g();
void func(unsigned int x) {
    unsigned long i=3;
    signed long k=3;

    while (i<x) {
        if (k<-3) {   /* variable k was replaced illegally by variable i. */
            break;
        }
        g();
        --i;
        --k;
    }
}
```

[Conditions]
This problem might occur when all of the following conditions were fulfilled.
(1) The optimize=1 option was specified.
(2) A loop existed.
(3) The loop of (2) had a signed int type or signed long type loop induction variable and an unsigned int type or unsigned long type one.
(4) Initial values of the loop induction variables of (3) were constant value.
(5) Updating values of the loop induction variables of (3) were the same value.

[Solution]
If a relevant failure exists, prevent the problem by one of the following methods.
(1) Specify optimize=0.
(2) Declare either of the loop induction variables of (3) as volatile.
(3) Declare either of the loop induction variables of (3) as char/unsigned char/short/ unsigned short type variable.
(4) Declare the loop induction variables of (3) as the same type variables.