date: 2002/03/14

HITACHI SEMICONDUCTOR TECHNICAL UPDATE

Classification of Production	Development Environment				No	TN-CSX-034A/E	
THEME	Release of patch against SuperH RISC engine C/C++ compiler Package Ver.7.0B		Classification of Information	[✓ 1:] Spec change 2. Supplement of Documents 3. Limitation of Use 4. Change of Mask 5. Change of Production Line			
PRODUCT NAME	SH-1,SH-2,SH-2E, SH2-DSP,SH-3, SH3-DSP,SH-4	Lot No.	Reference Documents	—		Rev.	Effective Date
		All				1	Eternity

The patch to fix problems in SuperH RISC engine C/C++ compiler Package Ver.7.0B is released.
A user who has the following product should be notified.

SuperH RISC engine C/C++ compiler Package Ver.7.0B for Windows (P0700CAS7-MWR)
SuperH RISC engine C/C++ compiler Package Ver.7.0B for SPARC (P0700CAS7-SLR)
SuperH RISC engine C/C++ compiler Package Ver.7.0B for HP9000 (P0700CAS7-H7R)

In this release of the C/C++ compiler, 9 bugs including generation of illegal objects have been fixed.
We will continue to improve the reliability of the compiler until the beginning of the MARCH.
We will announce the new release then.

Attached: P0700CAS7-020220E
SuperH RISC engine C/C++ compiler Package Ver. 7.0.01 (PC version)
SuperH RISC engine C/C++ compiler Package Ver. 7.0.02 (UNIX version)
Updates

SuperH RISC engine C/C++ compiler Package Ver. 7.0.01 (PC version) SuperH RISC engine C/C++ compiler Package Ver. 7.0.02 (UNIX version) Updates

The contents of updates in this package are shown below.
The item 1 and 2 hold true only for PC version and the item 5 hols true only for UNIX version.

1. Hitachi Embedded Workshop

1.1 Modification of the generation data for project generator

The following CPU's I/O definition file (iodefine.h) was modified:

SH7046, SH7047, SH7049, SH7148: st_dtc DTCSR definition
SH7055F: st_bsc RAMER, st_aut10 TIOR definition
SH7065: st_pg DR, st_pfc PFIORL definition
SH7604: st_frt OCR definition
SH7702, SH7707, SH7708, SH7708R, SH7709: st_bsc PCR definition
SH7750: Address definition of PWM1,2

In addition, the VBR setting descriptions of the reset program (resetprg.c) of the following CPU series were modified.

SH-1, SH-2, SH2-DSP, and SH-2E

1.2 Countermeasure against illegal deletion of the compiler Define option

A countermeasure was taken against the problem that a definition not to set 'Value' could not be deleted among the compiler Define options that can be set in [Options-"toolchain name"].

1.3 Countermeasure against illegal succeeding of an option when an assembly source is output in the compiler

A countermeasure was taken against the problem that an assembly option is not correctly set at build even if an assembly source is output with [Options- "toolchain name"] and an option is set for the output assembly source.

2. SuperH RISC engine simulator/debugger

2.1 Watch and Local Window

The following failure was modified:

If the size of the load module (.ABS) is large (the file size is about 64k bytes or more), the variable of the Watch Window may be displayed as 'Not available now'.

In addition, the variable (locale variable) defined in the function may not be displayed in the Locals window.

3. Compiler

3.1 Incorrect sign/zero extension

The problem in which the compiler might incorrectly sign/zero-extend the lower 1 or 2 byte in the result of a common subexpression is fixed.

<Example>

```
short sub(long);
short a,b,c,d;
void f() {
    if (d) {
        b = sub((long)a * (long)b * (long)c / 0x4000);
            /* ^^^^^^^^^^^^^^^^^ */
    } else {
        b = sub((long)a * (long)b * / 0x80);
            /* ^^^^^^^^^^^^^^^^^ */
    }
}
```

[Conditions]

The problem occurs if both of the following conditions are satisfied.

(1) A function has a common subexpression (marked with ^^^^ above) which produces the same result.

(2) It is guaranteed that the ranges of the operands in the common subexpression are no larger than 1 or 2 byte integer. (In the above example, it is obvious that the ranges of the operands in the common subexpression are no larger than 2 byte integer because the types of the variables before type-conversion are short.)

3.2 Illegal destruction of register

The problem in which the live contents of a register might be illegally destroyed with optimize=1 option is fixed.

[Conditions]

The problem occurs if all of the following conditions are satisfied.

(1) optimize=1 option is specified.

(2) Two or more basic blocks in a function have the same branch target including the loop entrance and return statements, where a basic block means a sequence of instrcutions without branches and labels inside.

(3) Each of the basic blocks mentioned in (2) above has assignments to the same variable.

3.3 Illegal assignment of 1-bit bitfield member

The following problem is fixed.

An internal error may occur or incorrect object code may be generated if a structure has one or more 1-bit bitfields, if they are assigned constant values sequentially and if more than one assignment to the same bitfield appears in sequence.

<Example>

```
struct {
    unsigned char b1:1;
    unsigned char b2:1;
} ST;

/* Internal error */
void f() {
    ST.b1=1;
    ST.b2=0;
    ST.b2=0;
}

/* Illegal object */
void g() {
    ST.b1=1;
    ST.b2=1;
    ST.b2=0;
}
```

[Conditions]

The problem occurs if both of the following conditions are satisfied.

(1) A structure has one or more 1-bit bitfields and a sequence of simple assignments (=) of constant values to several 1-bit bitfields is written.

(2) More than one assignment to the same 1-bit bitfield in the sequence assign the same constant value (then an internal error occurs), or an assignment of a value other than 0 to a 1-bit bitfield is followed by an assignment of 0 to the same 1-bit bitfield in the sequence (then incorrect object code is generated).

3.4 Saveing/restoring registers illegally

The following problem is fixed.

An interrupt function might not save or restore R0, R1 or R3 even though they are used in the function.

[Conditions]

The problem occurs if all of the following conditions are satisfied.

(1) The function is an interrupt function.

(2) The function does not have any function calls including runtime library calls.

(3) The function has one of the following.

(a) a switch statement

(b) a call of a function specified by #pragma inline_asm

(the size of this function is no less than 255 or unspecified.)

(c) an intra-function branch that does not reach using the 8 bit displacement

3.5 Internal error

The bug in which an internal error occurs with one of the following conditions is fixed.

(1) A file path contains a blank.

(2) A Chinense/Japanese character is specified in the file path of the map option.

4. Optimizing Linkage Editor

4.1 Incorrect debug information caused by the rename option

The problem in which the debug information of the symbols in the renamed section is deleted with the -form=relocate option is fixed.

[Conditions]

If both of the following, (1) and (2), are satisfied, the problem occurs.

(1) the -form=relocate option is specified.

(2) the -rename option is specified.

4.2 Invalid optimization of constant or literal data

The problem in which symbols are incorrectly unified with the -optimize=string_unify is fixed.

[Conditions]

If both of the following, (1) and (2), are satisfied, the problem occurs.

(1) A C source file is compiled with the -goptimize option.

(2) The -optimize=string_unify option is specified to the optimizing linkage editor.

4.3 Internal error caused by cache optimization

The problem in which an internal error occurs when both -optimize and -cache options are specified is fixed.

5. SuperH RISC engine simulator/debugger

5.1 Supporting the SH-4R Simulator

This simulator/debugger supports SH-4R(SH7750R).

The simulator supports 16KB/2WAY cache.

5.2 Supporting the Cache with the SH-4BSC Simulator

The following failure was modified:

If the P1 area (H'80000000 to H'9FFFFFFF) is used in the copy-back mode (write-through mode by default) with the SH-4BSC simulator, the cache operation will be illegal.

When the P1 area is used in the copy-back mode, use the SH-4 simulator.