To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

(Note)
These problems were fixed by SuperH RISC engine C/C++ compiler package Ver.6.0C.
Please download the update program from the following URL.
http://www.renesas.com/fmwk.jsp?cnt=/upgrades_shv60c.jsp&fp=/support/downloads/download_results/C2000501-C2000600

HITACHI SEMICONDUCTOR TECHNICAL UPDATE

DATE	5 February 2002	No.	TN-CSX-031A/E
THEME	Release of patch against SuperH RISC engine C/C++ compiler Package Ver.6.0 and Ver.6.0A		
CLASSIFICATION	☒ Spec. change ☐ Limitation on Use ☐ Supplement of documents		
PRODUCT NAME	P0700CAS6-MWR P0700CAS6-SLR P0700CAS6-H7R	Lot No. etc.	Ver6.0, Ver.6.0R1,Ver.6.0A, Ver.6.0AR1,Ver.6.0AR2
REFERENCE DOCUMENT	P0700CAS6-020117E (attached)	Effective Date	Eternity
		From	

The patch to fix problems in SuperH RISC engine C/C++ compiler Package Ver. 6.0 and Ver.6.0A is released.
Make contact with sales office if you have a UNIX version.
If you have a PC version, you can download the patch from the URL below.

http://www.hitachisemiconductor.com/sic/jsp/japan/eng/pdf/mpumcu/tool/cross/updater/v6bpatch.exe

A user who has the following product should be notified.
SuperH RISC engine C/C++ compiler Package Ver. 6.0, Ver. 6.0R1, Ver.6.0A, Ver.6.0AR1, or Ver.6.0AR2.

Attached:

(1) “SuperH RISC engine C/C++ compiler Ver.6.0B Updates”
(P0700CAS6-020117E), 3 page

SuperH RISC engine C/C++ compiler Ver. 6.0B
Updates

1. Compiler (Ver.6.0A -> Ver.6.0B)

1.1 Illegal stack information
Fixed the problem in which register save area size is not contained in the stack amount displayed by the stack analysis tool.

1.2 Internal Error 4970
Fixed the internal error 4970 which occurs when listfile option is specified.

1.3 Illegal optimization at linkage
Fixed the problem in which the register save/restore code is illegally optimized at linkage when both FPU registers and PR are saved and restored.

[Conditions]
(1) Both FPU registers and PR are saved and restored in the same function.
(2) goptimize option is specified at compilation.
(3) optimize=register option is specified at linkage.

1.4 Illegal source listing
Fixed the problem in which source listing outputs illegally when show=source,object option is specified.

[Conditions]
(1) show=source,object option is specified.
(2) width=<value> option is specified.
<value>:0 or more than 132
(3) Length of one line is more than 132 without CR code.

1.5 Illegal deletion of variable at linkage
Fixed the problem in which a variable specified by #pragma gbr_base/gbr_base1 is illegally deleted at linkage.

[Conditions]
(1) A variable is specified by #pragma gbr_base/gbr_base1.
(2) goptimize option is specified at compilation.
(3) optimize=symbol option is specified at linkage.

1.6 Illegal error with #line
Fixed the problem in which an error occurs when string literal exists following the #line.

1.7 Illegal debugging information of structure
Fixed the problem in which a debugging information of structure which has a size of more than 255 is illegal.

[Conditions]
(1) code=machinecode option is specified.
(2) debug option is specified.
(3) A structure which has a size of more than 255 exists.

1.8 Illegal debugging information of include file
Fixed the problem in which a include file information in debugging information is illegal when more than 255 files are included.

[Conditions]
(1) code=machinecode option is specified.
(2) debug option is specified.
(3) More than 255 files are included.

1.9 Error L2332 at linkage
Fixed the problem in which the error "L3310 (E) Relocation value is odd number" occur when a variable specified by #pragma gbr_base1 is declared as char type or unsigned char type.

[Conditions]
(1) #pragma gbr_base1 is specified.
(2) A variable specified by #pragma gbr_base1 is declared as char type or unsigned char type.
(3) This variable is located in odd address.

2. Optimizing Linkage Editor (Ver.7.0A -> Ver.7.1.02)

2.1 Error at creating a library

Fixed the following problem.

The error "L3310 (F) Cannot open temporary file" occurs and no library is generated, when the standard library generator for UNIX is used and when HLNK_DIR is specified at generating a standard library.

2.2 Illegal linking of absolute address sections

Fixed the problem in which an object is illegal when an absolute address section of size 0 is linked with the same name after the absolute address section.

2.3 Illegal execution with goptimize option

Fixed the problem in which the execution is illegal when there is a relationship that calls a function with the goptimize option from functions without (including an assembly function) and with the goptimize option.

2.4 Illegal branch width for a symbol

Fixed the problem in which the branch width for a symbol created by optimizing the common code is illegal when the following conditions are satisfied at the same time.

[Conditions]

(1) The goptimize option is specified in an input object.
(2) A code section is specified as the operand of rom option.
(3) Optimize is specified at linking.

2.5 Deleting an illegal symbol

Fixed the problem in which a symbol is illegally deleted when the following conditions are satisfied at the same time.

[Conditions]

(1) A code exists to refer to an object without goptimize option from an object with gopttimize option.
(2) ENTRY is specified.
(3) Deleting unreferred variables and functions is optimized (at specifying optimize=symbol_delete).

2.6 Illegal save/retrieve register

Fixed the problem in which the register save/restore code is illegally optimized when a function that is defined next from a function in the same C source.

2.7 Illegal literal pool value

Fixed the bug in which a literal pool value becomes invalid when the following conditions are satisfied at the same time with the optimization of register save/restore code.

[Conditions]

(1) The register save/restore code of the corresponding function is optimized.
(2) The corresponding function has a stack access code across the register save/restore area.
(3) The corresponding function recieves a parameter passed via the stack and its offset from SP is 2-byte.

2.8 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]

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

2.9 Invalid optimization of constant or literal data

[Conditions]

Fixed the problem in which symbols are incorrectly unified with the optimize=string_unify.

[Conditions]

(1) A C source file is compiled with the -goptimize option.
(2) The -optimize=string_unify option is specified to the optimizing linkage editor.

2.10 Internal error caused by cache optimization

Fixed the problem in which an internal error occurs when both -optimize and -cache options are specified.