RENESAS Tool News

RENESAS TOOL NEWS on June 1, 2006: RSO-H8C_1-060601D

The C/C++ Compiler Package for the H8, H8S, and H8SX Families of MCUs Revised to V.6.01 Release 02

We have revised the C/C++ compiler package for the H8, H8S, and H8SX families of MCUs from V.6.01 Release 01 to V.6.01 Release 02.

---

1. **Revised Product**

   The C/C++ compiler package V.6 for the H8, H8S, and H8SX families
   (Products for Windows, Solaris, and HP-UX concerned.)

2. **Descriptions of Revision**

   2.1 **Functions Introduced**

   2.1.1 In the Optimizing Linkage Editor (Linker)

   (1) The SEction_forbid option is newly introduced. This option prevents optimizations section by section.

   (2) The capability of the STARt option is enlarged to introduce the notation of "()".

   For details of the capabilities of the options in (1) and (2) above, see the following document included with the compiler package:
   "The new feature of Optimizing Linkage Editor V.9.01"

   2.2 **Items Improved and Problems Fixed**

   2.2.1 In the High-performance Embedded Workshop (Windows Edition Only)
   The High-performance Embedded Workshop included in the

package has been updated to V.4.00.03.

For details, see RENESAS TOOL NEWS Doc. No. RSO-HEW-051001D, published on October 1, 2005.

2.2.2 In the Compiler

(1) The 16 problems described in RENESAS TOOL NEWS Doc. No. RSO-H8C-060522D, published on May 22, 2006, have been fixed.

(2) The following three problems have been fixed (they are concerned with internal errors):

(a) If source files on a network are compiled, an internal error may arise (Error No. C5000).

(b) If source files are compiled using the structreg option, an internal error may arise (Error No. C4974).

(c) If source files that contain a character string of 80 or more characters consisting of only ASCII control characters 0x20--0x7F, an internal error may arise (Error No. C4709).

(3) The expression below has been made conformed to the ANSI standard:

(a) An expression that casts the type of another expression to void when the latter makes a call to a function that returns a value of type void. Example: (void)fun();.

(4) The problem reported in RENESAS TOOL NEWS ("On assignment expressions before and after an iteration or conditional branch block (H8C-0027)") has been fixed. For details see:

http://tool-support.renesas.com/eng/toolnews/n051101/tn9.htm

(5) The problem reported in RENESAS TOOL NEWS ("On accessing an incorrect addresses if a structure nested in another has members of a structure-type array (H8C-

0026)")has been fixed. For details see:

http://tool-support.renesas.com/eng/toolnews/n051001/tn5.htm

2.2.3 In the Optimizing Linkage Editor

The problems listed below have been fixed.

(1) On generating incorrect object code by optimizing the saving and restoring of registers (optimize=register)

Conditions:
This problem may occur if the following conditions are all satisfied:

(a) Any one of the MCU types H8SXN, H8SXM, H8SXA, and H8SXX is selected in compilation.

(b) The goptimize option is selected in compilation.

(c) The optimization of saving and restoring registers (optimize=register) is performed at linking.

(d) A MOV instruction that has any one of the following addressings is generated during compilation.*
- @(<value>:2,SP)
- @(<value>:16,SP)
- @(<value>:32,SP)

NOTE: SP (stack pointer) is the same register as ER7.

(e) The MOV instruction in (d) has another addressing that references a variable name at the same time.
Example:
MOV.L @(<value>:2,SP),@_<variable name>:32

* You can see the instructions generated during compilation on the compile list (.lst file). To do so, use the show=object option in compilation to create the compile list.

(2) On moving the label of a jump address in a goto statement to an incorrect place by optimizing the saving and restoring of registers (optimize=register)

Conditions:
This problem may occur if the following conditions are all satisfied:

(a) Any one of the MCU types H8SXN, H8SXM, H8SXA, and H8SXX is selected in compilation.

(b) The goptimize option is selected in compilation.

(c) The optimization of saving and restoring registers (optimize=register) is performed at linking.

(d) In the program exists a function containing the label of a jump address in a goto statement.

(e) The label of a jump address in (d) is moved to the beginning of the function during compilation.

(3) On incorrect run-time routines being created by optimizing the saving and restoring of registers (optimize=register)

Conditions:
This problem may occur if the following conditions are all satisfied:

(a) Any one of the MCU types 300, 300L, 300reg, 300HN, and 300HA is selected in compilation.

(b) The goptimize option is selected in compilation.

(c) The optimization of saving and restoring registers (optimize=register) is performed at linking.

(d) The optimization of subroutinizing common code (optimize=same_code) is performed at linking.

(e) The optimization in (c) creates the run-time routines with the following names:*
- "_opt_regsvpat<numeral>"

- "_opt_regldpat<numeral>"

* You can make the linker display a message of No. L0002 when a run-time routine is created. To do so, use the message option of the linker.

(4) On setting a const-qualified variable of 1 byte wide to 0 at linking

Conditions:
This problem may occur if the following conditions are all satisfied:

(a) A linker whose version is V.9.00.00 or later is used.
To check to see the version number of your linker, see Section 3 later.

(b) The goptimize option is selected in compilation.

(c) The optimization of the deletion of unreferenced symbols (optimize=symbol_delete) is performed at linking.

(d) The Optimization of the unifying constants and literals (optimize=string_unify) is performed at linking.

(e) The optimization in (c) deletes functions.

(f) In the program exists a const-qualified variable of 1 byte wide that can only be accessed by the function to be deleted in (e).

* You can make the linker display a message of No. L0004 when a variable or function is deleted. To do so, use the message option of the linker.

(5) On displaying incorrect stack usage specified by the .stack directive command when Call Walker, the stack analyzing tool, is used

Conditions:
This problem may occur if the following conditions are all satisfied:

(a) Symbols (functions and data items) are defined in an assembly source program.

(b) Symbols in (a) are referenced from another assembly source file.

(c) The .stack directive command is not used for any symbols in the assembly source program in (a) and is used for them in the assembly source program in (b).

(6) On displaying incorrect symbol names (_$ind_opt<numeral>) when the optimization of using indirect addressing mode information (the number of references of symbols) of the linkage list file (.map) is provided.

(7) The problems causing the following two errors

(a) If the optimization of the unifying constants and literals (optimize=string_unify) and optimization of the deletion of unreferenced symbols (optimize=symbol_delete) are performed at the same time, and the ELF/DWARF format converter (HELFCNV) is used, an error arises to provide the message below.

G2003 (E)Illegal file format "filename"

(b) If any address is specified by using CPU option SBR, and optimization of the using short absolute addressing mode (optimize=variable_access) is performed, an error arises to provide the message below.

L2330 (E)Relocation size overflow

(8) The problems causing the following three internal errors

(a) If .EQU labels are used in an assembly source program, and optimization is performed at linking, an internal error (Error No. L4001) may arise.

(b) If an output file is split by specifying address ranges using the output option, an internal error (Error No. L4000-5560) may arise.

(c) If any one of the MCU types 300, 300L, 300reg, 300HN, and 300HA is selected with the linker of version V.9.00.03 used, an internal error (Error No. L4001) may arise.
To check to see the version number of your linker, see Section 3 later.

(d) If the stack option is selected, an internal error (Error No. L4000) may arise.

3. How to See the Version Numbers of the Included Tools

Perform the following procedures:

(1) In the High-performance Embedded Workshop, open the Tools menu and select the Administration command. The Tool Administration dialog box appears.

(2) Select the compiler package you are using among Toolchains in the Registered Components list of the above dialog box; then click the Properties button. The Properties dialog box opens.

(3) In the Information tab of this dialog box, the version numbers of your tools will be shown.
Example of a linker: Optimizing Linkage Editor (V.9.00.02)

4. How to Update Your Product and Purchase the Revised One

4.1 Free-of-Charge Update

Free-of-charge update is available if you are using any one of the editions concerned.

(1) For Windows edition
To update yours online, download the update program from the Download site. and execute it.

(2) For Solaris and HP-UX editions

Supply the following items of information to your local Renesas Technology sales office or distributor. We will send you the latest version of the product package by return:

Solaris or HP-UX edition
Version No.: V.6.01
Release No.: Release 02

4.2 Ordering Information

If you place an order for any of the editions, please supply the following items of information to your local Renesas Technology sales office or distributor:

Windows, Solaris or HP-UX edition
Version No.: V.6.01
Release No.: Release 02

Host OS:
- Windows XP, Windows Me, Windows 98, Windows 2000, or Windows NT 4.0 (Windows edition)
- Solaris 2.5, or Solaris 8 (Solaris edition)
- HP-UX 10.2 (HP-UX edition)
