RENESAS Tool News

RENESAS TOOL NEWS on October 16, 2012: 121016/tn2

A Note on Using C/C++ Compiler Package for SuperH RISC engine Family V.9

When using the C/C++ compiler package for SuperH RISC engine family of MCUs V.9, take note of the following problem:

- With using the optimization of register save/restore in the optimizing linkage editor (LNK-0011)

Here the number at the end of the above item is a consecutive number for indexing the problems in the optimizing linkage editor included in this compiler package.

---

1. Product and Versions Concerned

C/C++ compiler package for SuperH RISC engine family
V.9.00 Release 00 through V.9.04 Release 01

2. Description

In the optimizing linkage editor included in the compiler package, if the optimization of register save/restore is used the program may not operate correctly.

3. Conditions

This problem may arise if the following conditions are all satisfied:
(1) The compile option Inter-module optimization information (-goptimize) is selected.
(2) The optimization of register save/restore in the optimizing linkage editor is used.
The optimization is effective when either of the following options is set:
(a) The -optimize option with no sub-options is used, or the -nooptimize option is not used.
(b) The -optimize option is used together with any one of its sub-options: register, speed, or safe

(3) The assembly code generated by the compiler satisfies either of the following:

(3-1) A literal is placed at the address 4 bytes less than the one where the RTS instruction resides. (See NOTE.)
Example:

```
-----------------------------------------
 .DATA.L    H'1000B200   ; Literal placed at address 4 bytes
                         ; less than address of RTS.
 LDS.L      @R15+,PR
 RTS
-----------------------------------------
```

(3-2) Immediately before the RTS instruction, instructions for restoring general registers exist, and a literal is placed at the address 4 bytes less than the one where the first Restore instruction exists. (See NOTE.)
Example:

```
-----------------------------------------
 .DATA.L  H'1000B200  ; Literal placed at address 4 bytes
                      ; less than address of first Restore
                      ; instruction.
 LDS.L    @R15,PR
 MOV.L    @R15+,R10   ; First Restore instruction
 MOV.L    @R15+,R9
 RTS
-----------------------------------------
```

NOTE: Literals are constant values expressing the addresses assigned to variables or functions.

(4) The literal value is the same as any of the following operation codes of delayed jump instructions:
BF/S (8Fxx)*, BT/S (8Dxx)*, BRA (Axxx)*, BRAF (0x23)**, BSR (Bxxx)*, BSRF (0x03)**, JMP (4x2B)**, JSR (4x0B)**, RTS (000B), and RTE (002B)

*: xx or xxx is an address
**: x is a register number

4. Example

Result of compilation:

```
---------------------------------------------------------------
 .DATA.L    H'1000B200    ; Condition (3-1)
 LDS.L      @R15+,PR
 RTS
---------------------------------------------------------------
```

If the above result is processed by the optimizing linkage editor,

the following symptom may arise:
The lower 16 bits, "B200," of the literal value that is placed
4 bytes before the RTS instruction is exchanged by the LDS instruction
(operation code 4F26) each other. After this exchange, 10004F26 is
referenced as a literal value, and the lower 16 bits, "B200,"
of the literal value is referenced as an operation code; not a literal value.

5. Workarounds

To avoid this problem, do either of the following:
(1) Do not use -optimize, but use -nooptimize.
(2) Do not use -optimize=register, =speed, and =safe.
For -optimize=speed and =safe, change these sub-options as follows:

For -optimize=speed:
-optimize=string_unify,symbol_delete,branch
For -optimize=safe:
-optimize=string_unify, branch

6. Schedule of Fixing Problem

We have no plan to fix this problem.
