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.

H8SX Family

Multiple-Bit Shifting

Introduction

This application note describes the multiple-bit shift function, which is one enhancement to the instruction set for the H8SX family relative to the set for the H8S.

Target Devices

H8SX family

Contents

1. Overview

The H8SX CPU used in H8SX-family products is a 32-bit CPU having an architecture that maintains upward compatibility with the H8/300, H8/300H, and H8S CPUs, and an instruction set that has been strengthened for better CPU performance. This leads to greatly improved code efficiency relative to the earlier series. This improved code efficiency reduces the amount of space that programs take up in ROM and the number of instruction-fetching cycles in program execution.

In the H8SX CPU, instructions for bit shift operations incorporate the capability of shifting by 1, 2, 4, 8, or 16 bits. This is one way to realize programs that take up less space in ROM and requires less time for instruction fetching. This application note describes this enhancement to the instruction set, i.e., the availability of the multiple-bit shift function.

2. Applicable Conditions

Table 1 Applicable Conditions

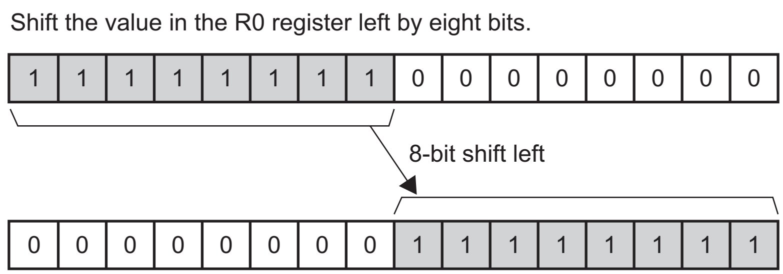
Item	Contents
Development tool	High-performance Embedded Workshop Version 4.00.03
C/C++ compiler	H8S, H8/300 Series C/C++ Compiler Version 6.01.01 (from Renesas Technology Corp.)
H8SX compiler options	-cpu = h8sxa:24:md, -code = machinecode, -optimize = 1, -regparam = 3 -speed = (register,shift,struct,expression)
H8S compiler options	-cpu = 2600a:24, -code = machinecode, -optimize = 1, -regparam = 3 -speed = (register,shift,struct,expression)

Table 2 Section Settings

Address	Section Name	Description
H'001000	P	Program area
H'FF2000	B	RAM area

3. Configuration

The earlier H8/300, H8/300H, and H8S CPUs provide bit shift instructions for only 1- and 2-bit shift operations. In contrast, the H8SX CPU has additional 2-byte-code instructions for 1-, 2-, 4-, 8-, and 16-bit shift operations and 4-byte-code instructions for up to 32-bit shift operations. For example, with the earlier H8S CPU, an 8-bit shift operation is done by repeating a 2-bit shift instruction four times. With the H8SX CPU, however, the same operation is achieved with a single 8-bit shift instruction. This is illustrated in figure 1.



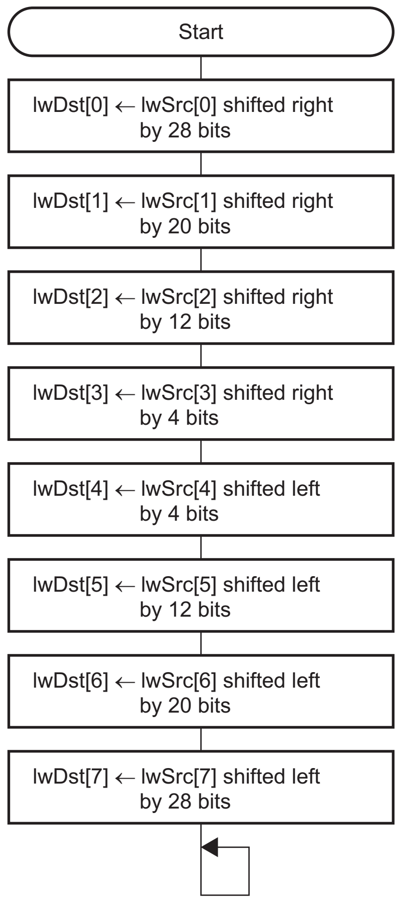
Shift Operation with the H8S CPU	Shift Operation with the H8SX CPU
SHLL.W #2, R0 SHLL.W #2, R0 SHLL.W #2, R0 SHLL.W #2, R0	SHLL.W #8, R0

Figure 1 Example: 8-Bit Shifting

4. Sample Program

4.1 Flowchart

This sample program is intended to convey an understanding of the multiple-bit shift function, one way in which the H8SX instruction set has been enhanced relative to that of the H8S. Shown below is a flowchart of the sample program, which performs right and left shift operations.



Note: * The results of compilation given in the following pages are to allow comparison with the H8S CPU. However, the results of compiling actual application programs will vary significantly with the source program and conditions of compilation.
Therefore, the results of compilation in this application note are only for reference.

4.2 Program Listing

A listing of the sample program in the C programming language is shown below. The results of compilation for the H8S CPU and H8SX CPU are given in section 4.3.

```
/*************************************************************/
/*  Application Note                                         */
/*************************************************************/

#include    <machine.h>

/*************************************************************/
/* RAM allocation                                            */
/*************************************************************/
unsigned long lwSrc[8];                                        /* Shift data          */
unsigned long lwDst[8];                                        /* Execute data shifing*/

/*************************************************************/
/* function prototype                                        */
/*************************************************************/
void main ( void );

/*************************************************************/
/*  Vector Address                                           */
/*************************************************************/
#pragma entry main(sp=0xFFC000,vect=0)                         /* H'0000 : Reset      */

#pragma section                                                /* P                   */
/*************************************************************/
/*  Main Program                                             */
/*************************************************************/
void main ( void )
{
    unsigned char   i;

    for ( i = 0; i < 8; i++ ) {
        lwSrc[i] = 0x12345678;
    }

    lwDst[0] = lwSrc[0]>>28;                                   /* 28-bit right shift  */
    lwDst[1] = lwSrc[1]>>20;                                   /* 20-bit right shift  */
    lwDst[2] = lwSrc[2]>>12;                                   /* 12-bit right shift  */
    lwDst[3] = lwSrc[3]>>4;                                    /* 4-bit right shift   */
    lwDst[4] = lwSrc[4]<<4;                                    /* 4-bit left shift    */
    lwDst[5] = lwSrc[5]<<12;                                   /* 12-bit left shift   */
    lwDst[6] = lwSrc[6]<<20;                                   /* 20-bit left shift   */
    lwDst[7] = lwSrc[7]<<28;                                   /* 28-bit left shift   */

    while(1);
}
```

4.3 Results of Compilation

4.3.1 Results for the H8S CPU

The assembly code is shown below.

```
00000000 _main:                                    ; function: main
00000000     MOV.L       #H'00FFC000,SP
00000006     MOV.B       #8:8,R3L
00000008     SUB.L       ER2,ER2
0000000A     MOV.L       #H'12345678,ER1
00000010 L23:
00000010     MOV.L       ER2,ER0
00000012     SHLL.L      #2,ER0
00000014     MOV.L       ER1,@(_lsrc:32,ER0)
0000001E     INC.L       #1,ER2
00000020     DEC.B       R3L
00000022     BNE         L23:8
00000024     MOV.L       #_lsrc,ER2
0000002A     MOV.L       @ER2,ER1
0000002E     MOV.W       E1,R0
00000030     MOV.B       R0H,R0L
00000032     SUB.B       R0H,R0H
00000034     SUB.W       E0,E0
00000036     SHLR.L      #2,ER0
00000038     SHLR.L      #2,ER0
0000003A     MOV.L       #_ldst,ER3
00000040     MOV.L       ER0,@ER3
00000044     MOV.L       @(4:16,ER2),ER0
0000004A     MOV.W       E0,R0
0000004C     SUB.W       E0,E0
0000004E     SHLR.L      #2,ER0
00000050     SHLR.L      #2,ER0
00000052     MOV.L       ER0,@(4:16,ER3)
00000058     MOV.L       @(8:16,ER2),ER0
0000005E     SHLR.L      #2,ER0
00000060     SHLR.L      #2,ER0
00000062     SHLR.L      #2,ER0
00000064     SHLR.L      #2,ER0
00000066     SHLR.L      #2,ER0
00000068     SHLR.L      #2,ER0
0000006A     MOV.L       ER0,@(8:16,ER3)
00000070     MOV.L       @(H'000C:16,ER2),ER0
00000076     SHLR.L      #2,ER0
00000078     SHLR.L      #2,ER0
0000007A     MOV.L       ER0,@(H'000C:16,ER3)
00000080     MOV.L       @(H'0010:16,ER2),ER0
00000086     SHLL.L      #2,ER0
00000088     SHLL.L      #2,ER0
0000008A     MOV.L       ER0,@(H'0010:16,ER3)
```

```
  00000090     MOV.L       @(H'0014:16,ER2),ER0
  00000096     SHLL.L      #2,ER0
  00000098     SHLL.L      #2,ER0
  0000009A     SHLL.L      #2,ER0
  0000009C     SHLL.L      #2,ER0
  0000009E     SHLL.L      #2,ER0
  000000A0     SHLL.L      #2,ER0
  000000A2     MOV.L       ER0,@(H'0014:16,ER3)
  000000A8     MOV.L       @(H'0018:16,ER2),ER0
  000000AE     MOV.W       R0,E0
  000000B0     SUB.W       R0,R0
  000000B2     SHLL.L      #2,ER0
  000000B4     SHLL.L      #2,ER0
  000000B6     MOV.L       ER0,@(H'0018:16,ER3)
  000000BC     MOV.L       @(H'001C:16,ER2),ER1
  000000C2     MOV.W       R1,R0
  000000C4     MOV.B       R0L,R0H
  000000C6     SUB.B       R0L,R0L
  000000C8     MOV.W       R0,E0
  000000CA     SUB.B       R0H,R0H
  000000CC     SHLL.L      #2,ER0
  000000CE     SHLL.L      #2,ER0
  000000D0     MOV.L       ER0,@(H'001C:16,ER3)
  000000D6 L25:
  000000D6     BRA         L25:8

B                                                    ; section
  00000000 _lsrc:                                    ; static: lsrc
  00000000     .RES.L      8
  00000020 _ldst:                                    ; static: ldst
  00000020     .RES.L      8
$VECT0                                               ; section
  00000000     .DATA.L     _main
```

4.3.2 Results for the H8SX CPU

The assembly code is shown below.

```
  00000000 _main:                                          ; function: main
  00000000      MOV.L       #H'00FFC000,SP
  00000006      MOV.B       #8:8,R1L
  00000008      SUB.L       ER0,ER0
  0000000A L23:
  0000000A      MOV.L       #H'12345678:32,@(_lsrc:32,ER0.L)
  00000016      INC.L       #1,ER0
  00000018      DEC.B       R1L
  0000001A      BNE         L23:8
  0000001C      MOV.L       #_lsrc,ER1
  00000022      MOV.L       @ER1,ER0
  00000026      SHLR.L      #28:5,ER0
  0000002A      MOV.L       #_ldst,ER2
  00000030      MOV.L       ER0,@ER2
  00000034      MOV.L       @(4:2,ER1),ER0
  00000038      SHLR.L      #20:5,ER0
  0000003C      MOV.L       ER0,@(4:2,ER2)
  00000040      MOV.L       @(8:2,ER1),ER0
  00000044      SHLR.L      #12:5,ER0
  00000048      MOV.L       ER0,@(8:2,ER2)
  0000004C      MOV.L       @(12:2,ER1),ER0
  00000050      SHLR.L      #4,ER0
  00000052      MOV.L       ER0,@(12:2,ER2)
  00000056      MOV.L       @(H'0010:16,ER1),ER0
  0000005C      SHLL.L      #4,ER0
  0000005E      MOV.L       ER0,@(H'0010:16,ER2)
  00000064      MOV.L       @(H'0014:16,ER1),ER0
  0000006A      SHLL.L      #12:5,ER0
  0000006E      MOV.L       ER0,@(H'0014:16,ER2)
  00000074      MOV.L       @(H'0018:16,ER1),ER0
  0000007A      SHLL.L      #20:5,ER0
  0000007E      MOV.L       ER0,@(H'0018:16,ER2)
  00000084      MOV.L       @(H'001C:16,ER1),ER0
  0000008A      SHLL.L      #28:5,ER0
  0000008E      MOV.L       ER0,@(H'001C:16,ER2)
  00000094 L25:
  00000094      BRA         L25:8

B                                                          ; section
  00000000 _lsrc:                                          ; static: lsrc
  00000000      .RES.L      8
  00000020 _ldst:                                          ; static: ldst
  00000020      .RES.L      8
$VECT0                                                     ; section
  00000000      .DATA.L     _main
```

4.4 Comparison of the Results of Compilation

The portions of the compilation results of the right shift processing for the H8S CPU and H8SX CPU are shown in tables 3 and 4, respectively. As shown in the tables, a single instruction enables the right shift processing with the H8SX CPU, reducing the total length of the instructions from 36 to 14 bytes and the execution time from 18 to 11 cycles.

Table 3 Results for the H8S CPU

Number of Bits Shifted Right	Assembly Code		Instruction Length (Bytes)	Execution Time (Number of Cycles)
28	MOV.W	E1,R0	2	1
	MOV.B	R0H,R0L	2	1
	SUB.B	R0H,R0H	2	1
	SUB.W	E0,E0	2	1
	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
20	MOV.W	E0,R0	2	1
	SUB.W	E0,E0	2	1
	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
12	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
4	SHLR.L	#2,ER0	2	1
	SHLR.L	#2,ER0	2	1
Total			36	18

Table 4 Results for the H8SX CPU

Number of Bits Shifted Right	Assembly Code		Instruction Length (Bytes)	Execution Time (Number of Cycles)
28	SHLR.L	#28:5,ER0	4	4
20	SHLR.L	#20:5,ER0	4	3
12	SHLR.L	#12:5,ER0	4	3
4	SHLR.L	#4,ER0	2	1
Total			14	11

Website and Support

Renesas Technology Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Sep.11.06	—	First edition issued

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.