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.

RENESAS

Notice

H8SX Family

Direct Operation on Values in Memory

Introduction

This application note describes direct operations on operands in memory. Such operations are 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 arithmetic and logical operations incorporate the capability of directly performing operations on operands in memory. This is one way in which programs that take up less space in ROM and shorter times for instruction fetching are realized. This application note describes this enhancement to the instruction set, i.e. the availability of direct operations on values in memory.

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

"Direct operation on values in memory" refers to functionality for the addressing of operands in memory by the direct specification of absolute addresses etc. for both the source and destination operands of instructions for arithmetic and logical operations. For example, with the earlier H8S CPU, two values stored in memory are added by using multiple instructions in this way: transfer the two data values in memory to respective CPU registers (loading), add the values in the registers, and transfer the result back to memory (storage). With the H8SX CPU, however, the same operation is achieved with a single instruction. This is illustrated in figure 1.

The rest of this application note describes a sample program that adds pairs of data values with three sizes (byte, word, and longword), and then compares the results of compilation for the H8S and H8SX CPUs. The sample program is written in the C language and compiled for the respective CPUs. Listings in assembly code of the results of compilation are given and the results for instruction-code length of the relevant generated code segments are compared.

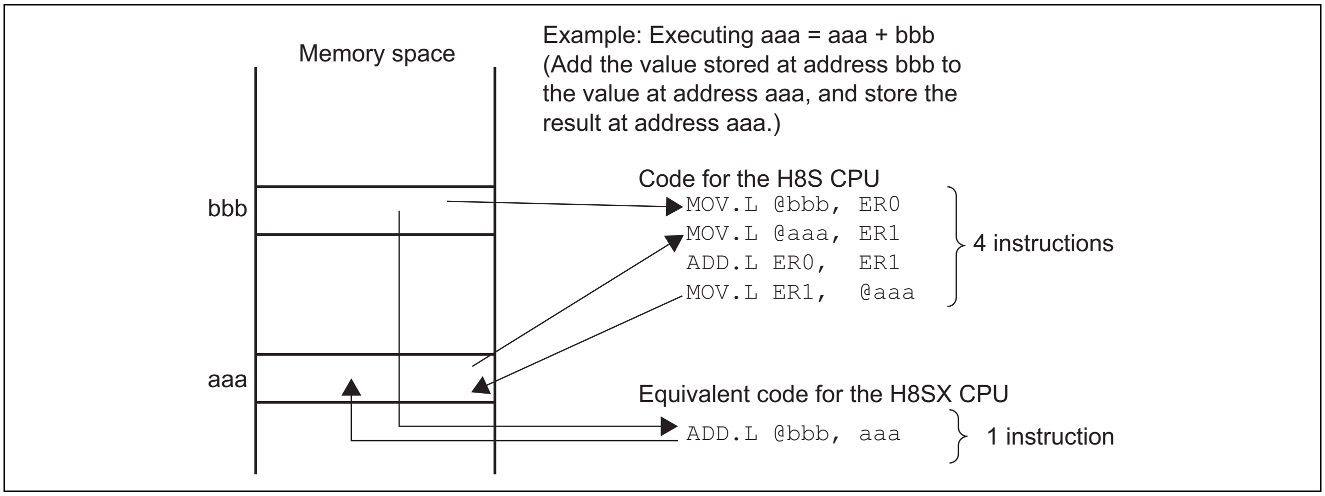
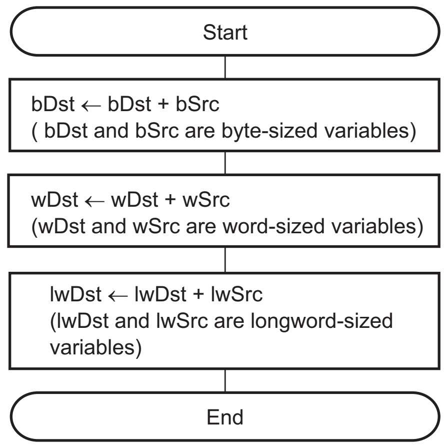


Figure 1 Example: Adding Data Values Stored in Memory

4. Sample Program

4.1 Flowchart

This sample program is intended to convey an understanding of direct operations on values in memory, one way in which the H8SX instruction set has been enhanced relative to that of the H8S. Shown below is the flowchart of the sample program, which performs byte-wise, word-wise, and longword-wise addition of pairs of data values in memory.



Note: * The results of compilation given in the following pages are to allow comparison with the H8S CPU. However, 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>

/*************************************************************/
/*  Static variables                                         */
/*************************************************************/
unsigned char   bSrc, bDst;
unsigned short  wSrc, wDst;
unsigned long   lwSrc, lwDst;

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

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

#pragma section                         /* P                */
/*************************************************************/
/*  Main program                                             */
/*************************************************************/
void main ( void )
{
    bDst += bSrc;
    wDst += wSrc;
    lwDst += lwSrc;

    while(1);
}
```

4.3 Results of Compilation

4.3.1 Results for the H8S CPU

The assembly code is shown below.

```
P                                         ; section
_main:                                    ; function: main
    MOV.L       #H'00FFC000,SP
    MOV.B       @_bDst:32,R0L
    MOV.B       @_bSrc:32,R1L
    ADD.B       R1L,R0L
    MOV.B       R0L,@_bDst:32
    MOV.W       @_wDst:32,R1
    MOV.W       @_wSrc:32,E1
    ADD.W       E1,R1
    MOV.W       R1,@_wDst:32
    MOV.L       @_lwDst:32,ER1
    MOV.L       @_lwSrc:32,ER0
    ADD.L       ER0,ER1
    MOV.L       ER1,@_lwDst:32
L28:
    BRA         L28:8

B                                         ; section
_wSrc:                                    ; static: wSrc
    .RES.W      1
_wDst:                                    ; static: wDst
    .RES.W      1
_lwSrc:                                   ; static: lwSrc
    .RES.L      1
_lwDst:                                   ; static: lwDst
    .RES.L      1
_bSrc:                                    ; static: bSrc
    .RES.B      1
_bDst:                                    ; static: bDst
    .RES.B      1

$VECT0                                    ; section
    .DATA.L     _main
```

4.3.2 Results for the H8SX CPU

The assembly code is shown below.

```
P                                      ; section
_main:                                 ; function: main
    MOV.L       #H'00FFC000,SP
    ADD.B       @_bSrc:32,@_bDst:32
    ADD.W       @_wSrc:32,@_wDst:32
    ADD.L       @_lwSrc:32,@_lwDst:32
L28:
    BRA         L28:8

B                                      ; section
_wSrc:                                 ; static: wSrc
    .RES.W      1
_wDst:                                 ; static: wDst
    .RES.W      1
_lwSrc:                                ; static: lwSrc
    .RES.L      1
_lwDst:                                ; static: lwDst
    .RES.L      1
_bSrc:                                 ; static: bSrc
    .RES.B      1
_bDst:                                 ; static: bDst
    .RES.B      1

$VECT0                                 ; section
.DATA.L     _main
```

4.4 Comparison of the Results of Compilation

The key portions of the results of compilation for the H8S CPU and H8SX CPU are shown in tables 3 and 4, respectively. As is shown in the tables, directly specifying the address in the ADD instructions eliminates the need for MOV instructions, reducing the total length of the instructions from 66 to 38 bytes and the execution time from 45 to 17 cycles.

Table 3 Results for the H8S CPU

Assembly Code			Instruction Length (Bytes)	Execution Time (Number of Cycles)
Byte-data addition	MOV.B	@_bDst:32,R0L	6	4
	MOV.B	@_bSrc:32,R1L	6	4
	ADD.B	R1L,R0L	2	1
	MOV.B	R0L,@_bDst:32	6	4
Word-data addition	MOV.W	@_wDst:32,R1	6	4
	MOV.W	@_wSrc:32,E1	6	4
	ADD.W	E1,R1	2	1
	MOV.W	R1,@_wDst:32	6	4
Longword-data addition	MOV.L	@_lwDst:32,ER1	8	6
	MOV.L	@_lwSrc:32,ER0	8	6
	ADD.L	ER0,ER1	2	1
	MOV.L	ER1,@_lwDst:32	8	6
Total			66	45

Table 4 Results for the H8SX CPU

Assembly Code			Instruction Length (Bytes)	Execution Time (Number of Cycles)
Byte-data addition	ADD.B	@_bSrc:32,@_bDst:32	12	5
Word-data addition	ADD.W	@_wSrc:32,@_wDst:32	12	6
Longword-data addition	ADD.L	@_lwSrc:32,@_lwDst:32	14	6
Total			38	17

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