

H8SX Family

Enhanced Addressing Mode (for Arrays)

Introduction

This application note describes the usage of index register indirect with displacement (offset indexed indirect) addressing. This mode has been included in the instruction set for the H8SX family as an enhancement relative to the set for the H8S. This addressing mode is especially useful in processing arrays of data.

Target Device

H8SX family

Contents

1. Overview	2
2. Applicable Conditions	2
3. Configuration.....	3
4. Sample Program	5

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.

Of the enhanced addressing modes, this application note describes index register indirect with displacement addressing, which is useful in processing arrays of data.

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 compile option	-cpu = h8sxa:24:md, -code = machinecode, -optimize = 1, -regparam = 3, -speed = (register, shift, struct, expression)
H8S compile option	-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

Register indirect with displacement addressing is similar to index register indirect with displacement addressing. The former is provided on both the H8S and H8SX CPUs, but the latter has only been included among the addressing modes of the H8SX CPU.

In register indirect with displacement addressing, the effective address is calculated by adding the displacement to the value in the specified register (limited to the ERn registers). On the other hand, any of the registers, whether it has 8, 16, or 32 bits (i.e. RnL, Rn, and ERn), can be used in index register indirect with displacement addressing. If an RnL or Rn register is specified, the value in the register is zero-extended to form a 32-bit value, to which the displacement is added to produce the effective address. The latter form of addressing is thus more flexible, so it caters to a wider range of applications.

Figure 1 shows how the effective address is calculated in the respective addressing modes. Figure 2 shows an example of access to array data.

The rest of this application note describes the sample program, which is a basic sorting program that accesses a data array, 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.

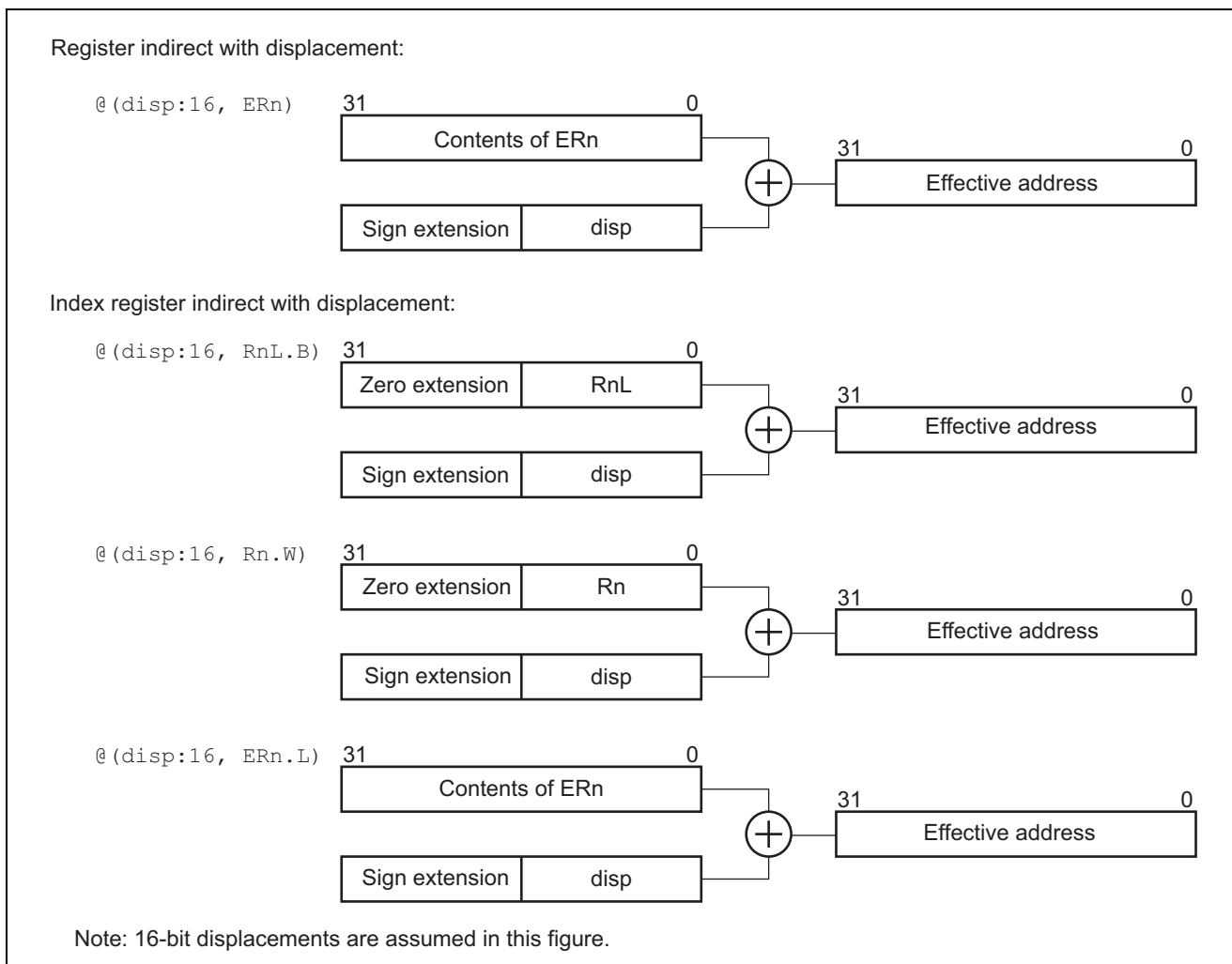


Figure 1 Calculation of the Effective Address

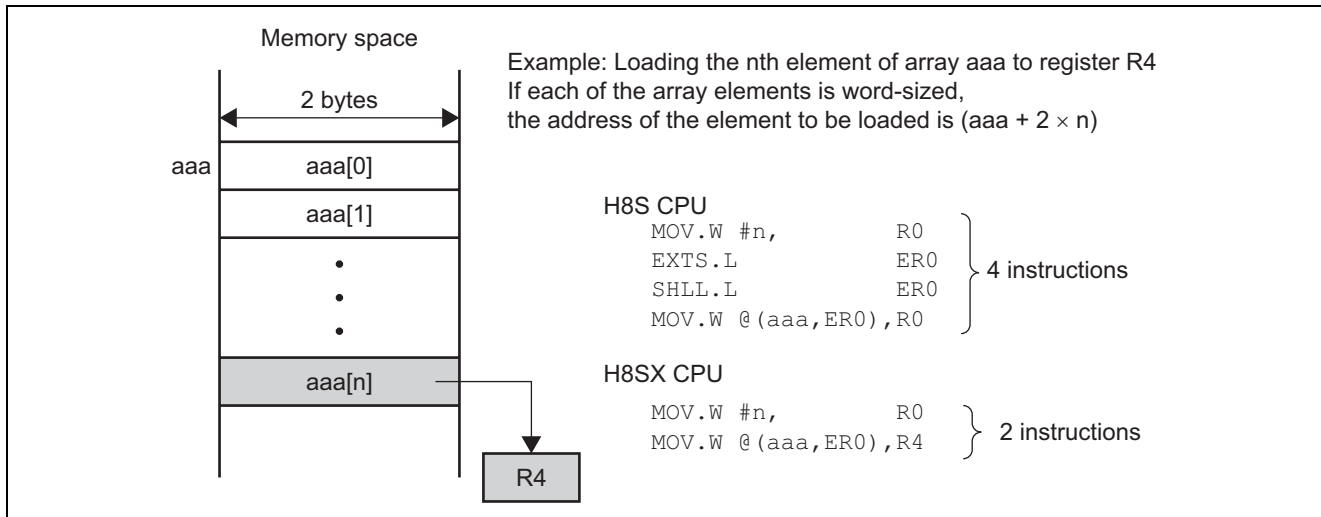
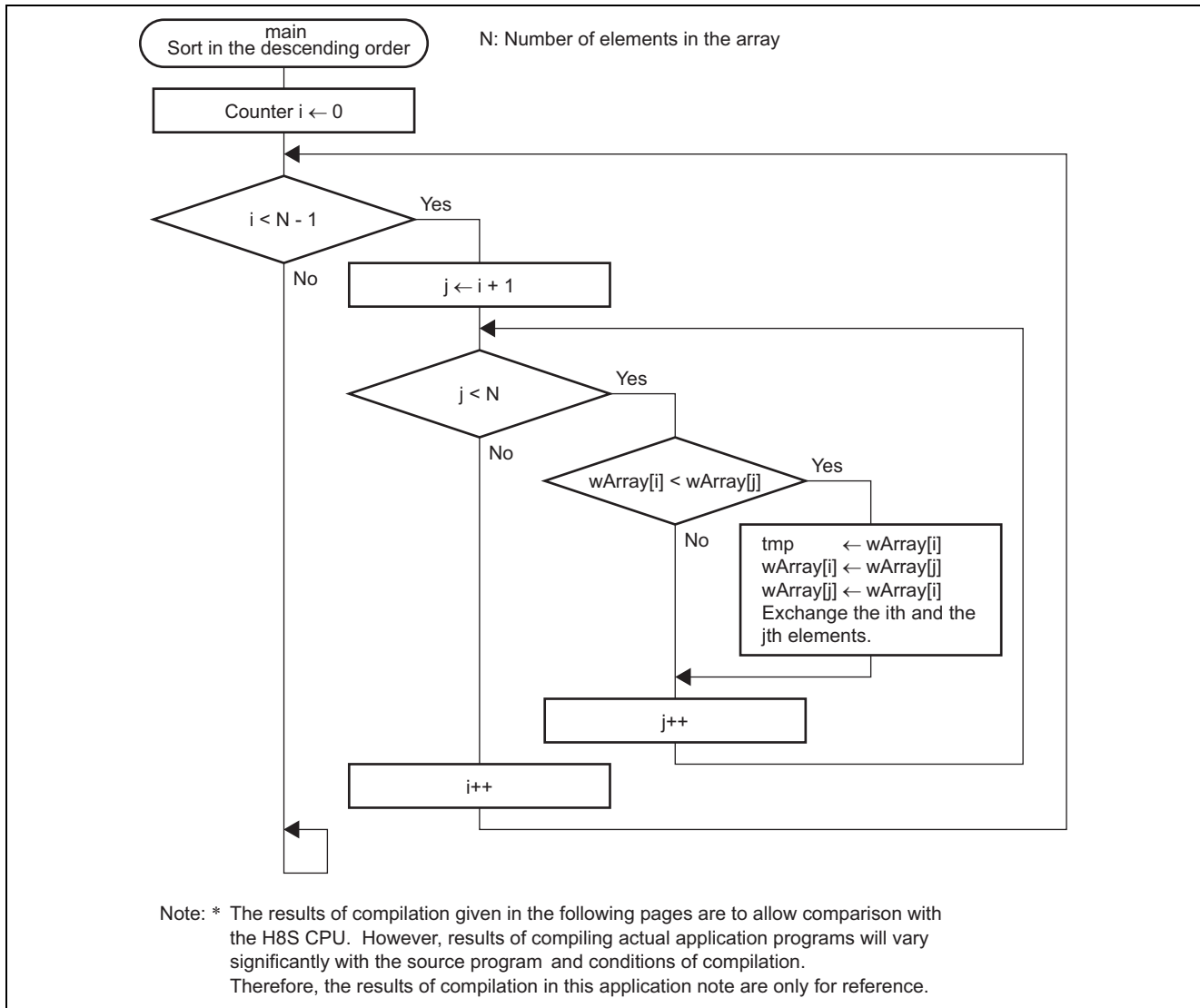


Figure 2 Example of Accessing Array Data

4. Sample Program

4.1 Flowchart

This sample program is a simple sorting program intended to convey an understanding of the index register indirect with displacement addressing, 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 bubble sorting.



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>

/*****/
/*  Array variable                               */
/*****/
#define N    100
short wArray[N];                                /* Data for sorting          */

/*****/
/*  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, j;
    short          tmp;

    for ( i = 0; i < (N-1); i++ ) {
        for ( j = (i+1); j < N; j++ ) {
            if ( wArray[i] < wArray[j] ) {      /* Array element: compare and change*/
                tmp          = wArray[i];
                wArray[i] = wArray[j];
                wArray[j] = tmp;
            }
        }
    }

    while(1);
}

```

4.3 Results of Compilation

4.3.1 Results for the H8S CPU

The assembly code is shown below.

```

P                                     ; section
00000000 _main:                       ; function: main
00000000     MOV.L     #H'00FFC000,SP
00000006     SUB.B    R2H,R2H
00000008     MOV.B    #H'63:8,R6L
0000000A     SUB.L    ER5,ER5
0000000C L22:
0000000C     MOV.B    R2H,R2L
0000000E     INC.B    R2L
00000010     MOV.L    ER5,ER4
00000012     SHLL.L   ER4
00000014     ADD.L    #_wArray,ER4
0000001A     BRA     L24:8
0000001C L25:
0000001C     MOV.W    @ER4,E1
0000001E     SUB.L    ER0,ER0
00000020     MOV.B    R2L,R0L
00000022     SHLL.L   ER0
00000024     ADD.L    #_wArray,ER0
0000002A     MOV.W    @ER0,R1
0000002C     CMP.W    R1,E1
0000002E     BGE     L27:8
00000030     MOV.W    R1,@ER4
00000032     MOV.W    E1,@ER0
00000034 L27:
00000034     INC.B    R2L
00000036 L24:
00000036     CMP.B    #H'64:8,R2L
00000038     BLO     L25:8
0000003A     INC.B    R2H
0000003C     INC.L    #1,ER5
0000003E     DEC.B    R6L
00000040     BNE     L22:8
00000042 L29:
00000042     BRA     L29:8
B                                     ; section
00000000 _wArray:                       ; static: wArray
00000000     .RES.W    100
$VECT0                                     ; section
00000000     .DATA.L   _main

```

4.3.2 Results for the H8SX CPU

The assembly code is shown below.

```

P                                     ; section
00000000 _main:                       ; function: main
00000000     MOV.L     #H'00FFC000,SP
00000006     SUB.B     R0H,R0H
00000008     MOV.B     #H'63:8,R3L
0000000A     SUB.L     ER2,ER2
0000000C L22:
0000000C     MOV.B     R0H,R0L
0000000E     BRA      L31:8
00000010 L24:
00000010     MOV.W     @(_wArray:32,ER2.L),R1
00000018     MOV.W     @(_wArray:32,R0L.B),E0
00000020     CMP.W     E0,R1
00000022     BGE      L31:8
00000024     MOV.W     E0,@(_wArray:32,ER2.L)
0000002C     MOV.W     R1,@(_wArray:32,R0L.B)
00000034 L31:
00000034     INC.B     R0L
00000036     CMP.B     #H'64:8,R0L
00000038     BLO      L24:8
0000003A     INC.B     R0H
0000003C     INC.L     #1,ER2
0000003E     DEC.B     R3L
00000040     BNE      L22:8
00000042 L28:
00000042     BRA      L28:8
B                                     ; section
00000000 _wArray:                       ; static: wArray
00000000     .RES.W     100
$VECT0

```

4.4 Comparison of the Results of Compilation

The following portion of the C source code compares and swaps array elements. The results of compiling this code for the H8S CPU and H8SX CPU are shown in tables 3 and 4, respectively. As shown in the table, the H8SX CPU can access any element of the array with a single instruction. Although the instructions take up more space (24 bytes → 36 bytes), the execution time is reduced from 17 to 15 cycles.

```

if ( wArray[i] < wArray[j] ) {      /* array element:compare and change */
    tmp      = wArray[i];
    wArray[i] = wArray[j];
    wArray[j] = tmp;
}

```

Table 3 Results for the H8S CPU

Assembly Code	Instruction Length (Bytes)	Execution Time (Number of Cycles)
MOV.W @ER4, E1	2	2
SUB.L ER0, ER0	2	1
MOV.B R2L, R0L	2	1
SHLL.L ER0	2	1
ADD.L #_wArray, ER0	6	3
MOV.W @ER0, R1	2	2
CMP.W R1, E1	2	1
BGE L27:8	2	2
MOV.W R1, @ER4	2	2
MOV.W E1, @ER0	2	2
Total	24	17

Table 4 Results for the H8SX CPU

Assembly Code	Instruction Length (Bytes)	Execution Time (Number of Cycles)
MOV.W @(_wArray:32, ER2.L), R1	8	3
MOV.W @(_wArray:32, R0L.B), E0	8	3
CMP.W E0, R1BGE L31:8	2	1
MOV.W E0, @(_wArray:32, ER2.L)	2	2
MOV.W R1, @(_wArray:32, R0L.B)	8	3
Total	36	15

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.