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.

APPLICATION NOTE

H8/300L Super Low Power Series

Conversion from 2-Byte Hexadecimal to 5-Digit Decimal Number (HEX)

Introduction

The software HEX converts a 2-byte hexadecimal number, which is placed in a general-purpose register, to a 5-digit BCD (binary-coded decimal) number and places the result in general-purpose registers.

Target Device

H8/38024

Contents

1. Arguments

Description		Memory area	Data length (bytes)
Input	2-byte hexadecimal number	R0	2
Output	5-digit BCD number (upper 1 digit)	R2L	1
	5-digit BCD number (lower 4 digits)	R3	2

2. Changes to Internal Registers and Flags

R0	R1	R2H	R2L	R3	R4	R5	R6	R7
×	—	×	○	○	—	—	—	—

I	U	H	U	N	Z	V	C
—	—	×	—	×	×	×	×

Legend
—: No change
×: Undefined
○: Result

3. Specifications

Program memory (bytes)
30
Data memory (bytes)
0
Stack (bytes)
0
Clock cycle count
368
Reentrant
Possible
Relocation
Possible
Interrupt
Possible

4. Description

4.1 Details of functions

1. The following arguments are used with the software HEX:
   R0: Sets a 2-byte hexadecimal number as an input argument.
   R2L: The upper 1 digit (1 byte) of the 5-digit BCD number is placed here as an output argument.
   R3: The lower 4 digits (2 bytes) of the 5-digit BCD number are placed here as an output argument.
   Figure 1 shows the formats of the input and output arguments.

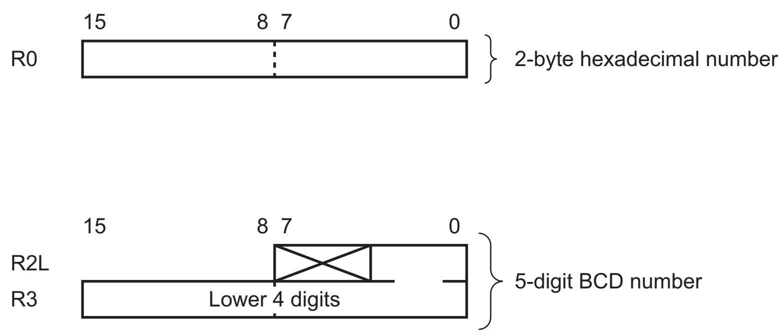


Figure 1 Formats of Input and Output Arguments

2. The following figure illustrates the execution of the software HEX. When the input argument is set as shown in (1), the 5-digit BCD number is placed in R2L and R3 as shown in (2).

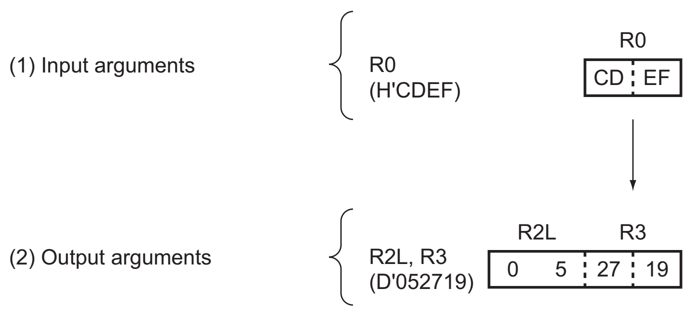


Figure 2 Example of Software HEX Execution

4.2 Note on usage

When upper bits are not used (see figure 3), set them to 0; otherwise, a correct result cannot be obtained because computation is made on numbers including indeterminate data placed in the upper bits.

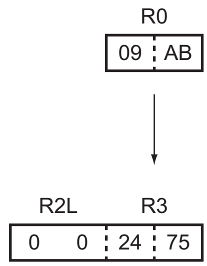


Figure 3 Examples of Operation with Upper Bits Unused

4.3 Description of data memory

The software HEX does not use the data memory.

4.4 Example of usage

Set a 2-byte hexadecimal number in R0 and call the software HEX as a subroutine.

```
WORK1     .RES.W     1           ---------- Reserve a data memory area in which the user program places
                                            a 2-byte hexadecimal number.

WOEK2     .RES.B     3           ---------- Reserve a data memory area in which the user program places
                                            a 5-digit BCD number (3 bytes).
             .
             .
             .
          MOV.W      @WORK1, R0       ------- Place the 2-byte nexadecimal number set by the user
                                              program in the input argument.

          JSR        @HEX             ------- Call the software HEX as a subroutine.

          MOV.B      R2L, @WORK2    }
          MOV.B      R3H, @WORK2+1  }------- Place the 5-digit BCD number set in the output argument
          MOV.B      R3L, @WORK2+2  }        in the data memory area of the user program.
             .
             .
             .
```

4.5 Operation

1. A 4-bit binary number "$B_3B_2B_1B_0$" is represented by equations 1 and 2 below:

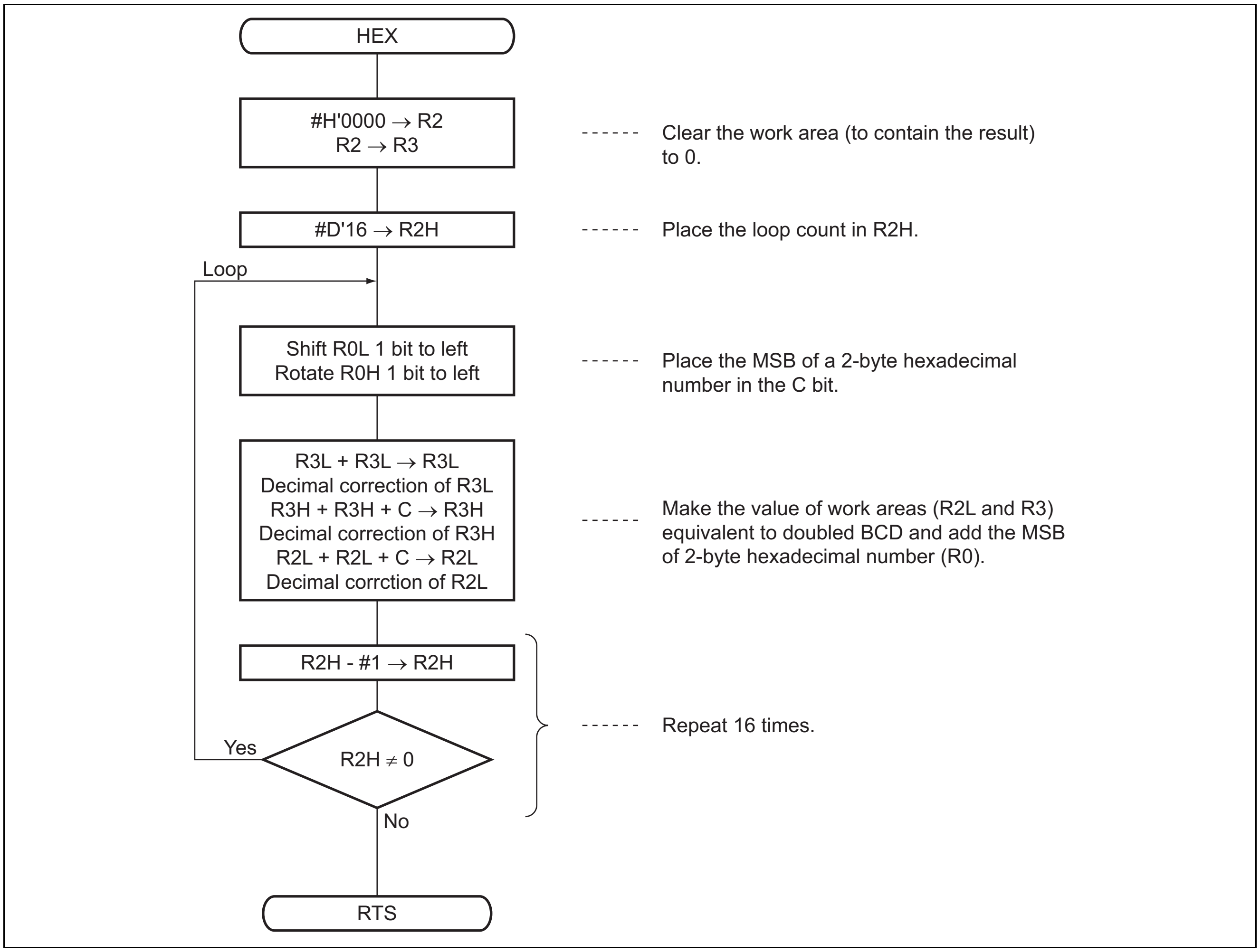
$$B_3\ B_2\ B_1\ B_0 = B_3 \times 2^3 + B_2 \times 2^2 + B_1 \times 2^1 + B_0 \times 2^0 \text{ ------- (equation 1)}$$

$$= \underbrace{(\underbrace{(\underbrace{B_3 \times 2 + B_2}_{\alpha}) \times 2 + B_1}_{\beta}) \times 2 + B_0}_{\gamma} \text{ ------- (equation 2)}$$

Figure 4 4-bit Binary Number "$B_3B_2B_1B_0$"

2. First, equation 2 is used to compute $\alpha = B_3 \times 2 + B_2$ (see figure 4) by executing an add instruction (ADD.B) and decimal adjust instruction (DAA). Next, a series of operations, $\beta = \alpha \times 2 + B_1$, $\gamma = \beta \times 2 + B_0$, etc. are performed to produce a 5-digit BCD number as the result.
3. The software HEX uses R0 and R2L and R3 to compute $\alpha = B_3 \times 2 + B_2$.
   a. R2H is used as the counter that counts the number of bit shifts on R0, which contains the input argument. D'16 is set in R2H for a total of 16 shifts.
   b. R0 containing the 2-byte hexadecimal number is shifted 1 bit to left, and the most significant bit is thus loaded to the C bit.
   c. R2L and R3 containing the 5-digit BCD number are processed in lowest byte-to-highest byte order, as follows:
      R3L + R3L + C $\rightarrow$ R3L; R3L is then decimal-adjusted.
      R3H + R3H + C $\rightarrow$ R3H; R3H is then decimal-adjusted.
      R2L + R2L + C $\rightarrow$ R2L; R2L is then decimal-adjusted.
      Thus, $\alpha = B_3 \times 2 + B_2$ is computed.
   d. In the software HEX, R2H is decremented each time steps b and c are performed. This processing is repeated until R2H reaches "0".

5. Flowchart


HEX
#H'0000 → R2
R2 → R3
Clear the work area (to contain the result) to 0.
#D'16 → R2H
Place the loop count in R2H.
Loop
Shift R0L 1 bit to left
Rotate R0H 1 bit to left
Place the MSB of a 2-byte hexadecimal number in the C bit.
R3L + R3L → R3L
Decimal correction of R3L
R3H + R3H + C → R3H
Decimal correction of R3H
R2L + R2L + C → R2L
Decimal corrction of R2L
Make the value of work areas (R2L and R3) equivalent to doubled BCD and add the MSB of 2-byte hexadecimal number (R0).
R2H - #1 → R2H
Repeat 16 times.
Yes
R2H ≠ 0
No
RTS

6. Program List

```
*** H8/300 ASSEMBLER VER 1.0B ** 08/18/92 10:24:23
PROGRAM NAME =
                              ;**********************************************************
                              ;*
                              ;*     00 - NAME                       :CHANGE 2 BYTE HEXADECIMAL
                              ;*                             TO BCD (HEX)
                              ;*
                              ;**********************************************************
                              ;*
                              ;*     ENTRY                 :R0 (HEXADECIMAL)
                              ;*
                              ;*     RETURNS  :R2L (UPPER 1 CHARACTER (BY BCD))
                              ;*                            R3 (LOWER 4 CHARACTER (BY BCD))
                              ;*
                              ;**********************************************************
                              ;
HEX_code C    0000                   .SECTION              HEX_code,CODE,ALIGN=2
                                     .EXPORT  HEX
                              ;
HEX_code C          00000000  HEX    .EQU $                ;Entry point
HEX_code C    0000  79020000         MOV.W    #H'0000,R2   ;Clear R2
HEX_code C    0004  0D23             MOV.W    R2,R3        ;Clear R3
HEX_code C    0006  F210             MOV.B    #D'16,R2H    ;Set bit counter
HEX_code C    0008                LOOP
HEX_code C    0008  1008             SHLL.B   R0L          ;Shift hexadecimal 1 bit left
HEX_code C    000A  1200             ROTXL.B  R0H
                              ;
HEX_code C    000C  0EBB             ADDX.B   R3L,R3L      ;R3L + R3L -> R3L
HEX_code C    000E  0F0B             DAA      R3L          ;Decimal adjust R3L
HEX_code C    0010  0E33             ADDX.B   R3H,R3H      ;R3H + R3H + C -> R3H
HEX_code C    0012  0F03             DAA      R3H          ;Decimal adjust R3H
HEX_code C    0014  0EAA             ADDX.B   R2L,R2L      ;R2L + R2L + C -> R2L
HEX_code C    0016  0F0A             DAA      R2L          ;Decimal adjust R2L
                              ;
HEX_code C    0018  1A02             DEC.B    R2H          ;Decrement R2H
HEX_code C    001A  46EC             BNE      LOOP         ;Branch Z=0
HEX_code C    001C  5470             RTS
                              ;
                                     .END
*****TOTAL ERRORS 0
*****TOTAL WARNINGS 0
```

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.18.03	—	First edition issued
2.00	Nov.30.06	All pages	Content correction