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.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

H8/300H Tiny Series

Addition of Single-Precision Floating-Point Numbers (FADD)

Introduction

Adds two single-precision floating-point numbers set in general registers and stores the result in general registers.

Target Device

H8/300H Tiny Series

Contents

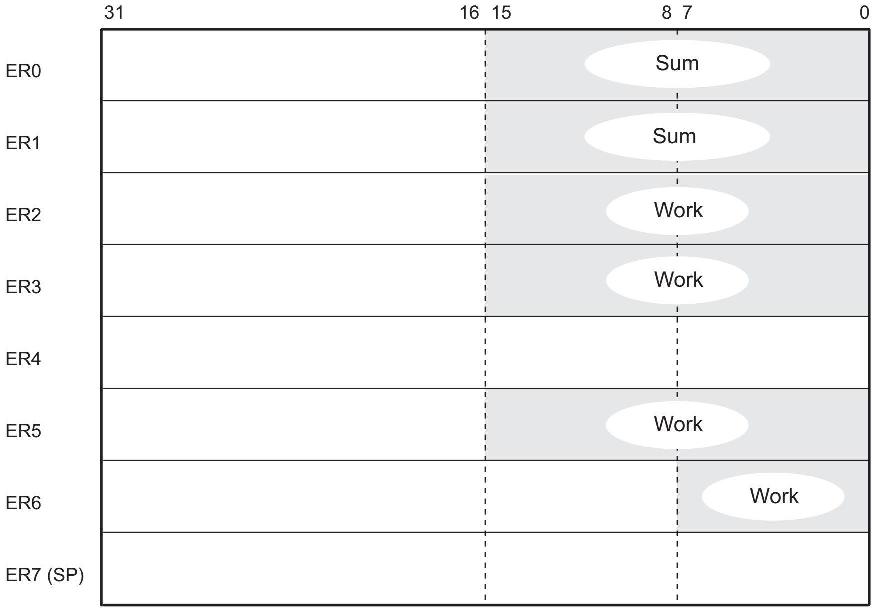
1. Function

1. Adds two single-precision floating-point numbers in general registers and stores the result in general registers.
2. The arguments are all in the single-precision floating-point data format.

2. Arguments

Contents		Storage Location	Data Length (Bytes)
Input	Augend	R0, R1	4
	Addend	R2, R3	4
Output	Sum	R0, R1	4

3. Changes to Internal Registers and Flags



I	UI	H	U	N	Z	V	C
—	—	↕	—	↕	↕	↕	↕

—: No change
↕: Varies
0 : Fixed to 0
1 : Fixed to 1

4. Programming Specifications

Program memory (bytes)	280
Data memory (bytes)	0
Stack (bytes)	0
Number of cycles	268
Re-entrant	Yes
Relocatable	Yes
Interrupts during execution	Yes

5. Notes

The number of cycles in the programming specifications is the value for execution of the example in figure 1. For details on the floating-point data format, refer to Reference: Description of Single-Precision Floating-Point Formats.

6. Descriptions

6.1 Descriptions of Functions

1. The arguments are listed below.
   1) Set the input arguments as follows.
      R0: higher-order two bytes of the augend
      R1: lower-order two bytes of the augend
      R2: higher-order two bytes of the addend
      R3: lower-order two bytes of the addend
   2) The FADD subroutine sets the following output argument.
      R0: higher-order two bytes of the result
      R1: lower-order two bytes of the result

2. The following figure illustrates the execution of the FADD subroutine. When the input arguments are set as shown, FADD places the sum of the input arguments in R0 and R1.

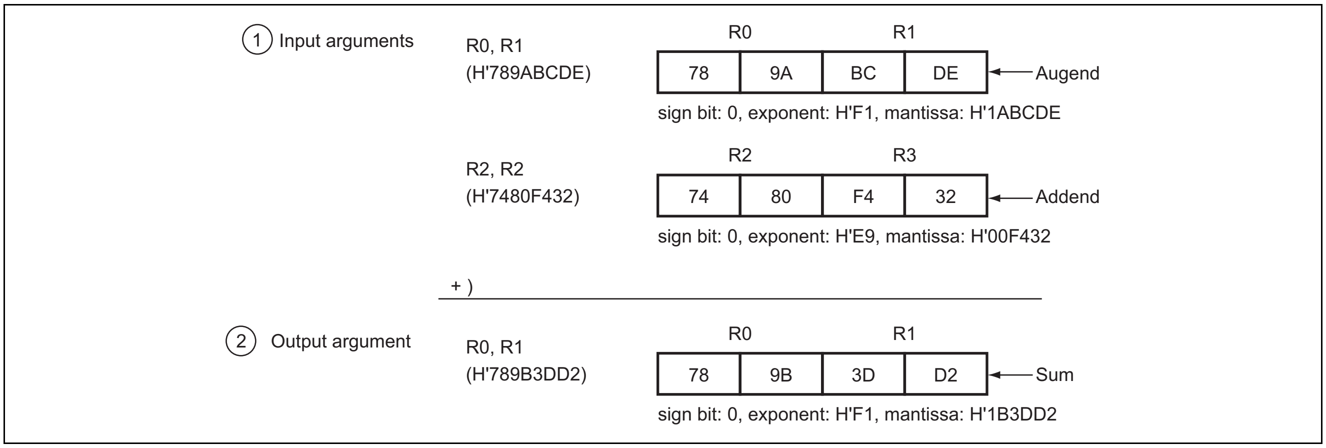


Figure 1 Example of FADD Execution

6.2 Usage Notes

1. The maximum and minimum values of the data that can be handled by the software FADD are as follows.
   Maximum positive value: H'7F80000
   Minimum positive value: H'00000001
   Maximum negative value: H'80000001
   Minimum negative value: H'FF800000
2. Positive single-precision floating-point numbers from H'7F800001 to H'7FFFFFFF are treated as the maximum value, H'7F800000. Negative single-precision floating-point numbers from H'FF80FFFF to H'FFFFFFF are treated as the minimum value, H'FF800000.
3. The maximum value is handled as infinity (∞). Accordingly, the results of value thus does not change if numbers are added to or subtracted from it (see table 1).

Table 1 Results of Addition when Maximum Values are Specified in the Arguments

Augend	Addend	Result
H'7F800000 to H'7FFFFFFF	H'************	H'7F800000
Other than H'7F800000 to H'FFFFFFFF	H'7F800000 to H'7FFFFFFF	H'7F800000
H'FF800000 to H'FFFFFFFF	H'************	H'FF800000
Other than H'7F800000 to H'7FFFFFFF	H'7F800000 to H'FFFFFFFF	H'FF800000

Note: H'******** indicates hexadecimal data.

4. H'80000000 is handled as H'00000000 (zero).
5. The augend and addend in the general registers are lost in the execution of FADD. When you will still require the input arguments, save them elsewhere in memory before running this subroutine.

6.3 Description of Data Memory

No data memory is used by the software FADD.

6.4 Example of Usage

Set the augend and addend in the general registers and then call the FADD subroutine.

```
WORK1   . RES. W 2              ········· Reservaton of the data memory area for setting for the augend by the user program.

WORK2   . RES. W 2              ········· Reservaton of the data memory area for setting for the addend by the user program.

WORK3   . RES. W 2              ········· Reservaton of the data memory area where the will be set for the user program.
             .
             .
             .
        MOV. W @WORK1, R0       ········· Sets, as input aregument, the augend specified by the user program.

        MOV. W @WORK1+2, R1

        MOV. W @WORK2, R2       ········· Sets, as input aregument, the addend specified by the user program.

        MOV. W @WORK2+2, R3

        JSR    @FADD            ········· Subroutine call of the FADD subrutine.

        MOV. W R0, @WORK3       ········· Transfers the result form the output argument to data memory.

        MOV. W R1, @WORK3+2
```

6.5 Principles of Operation

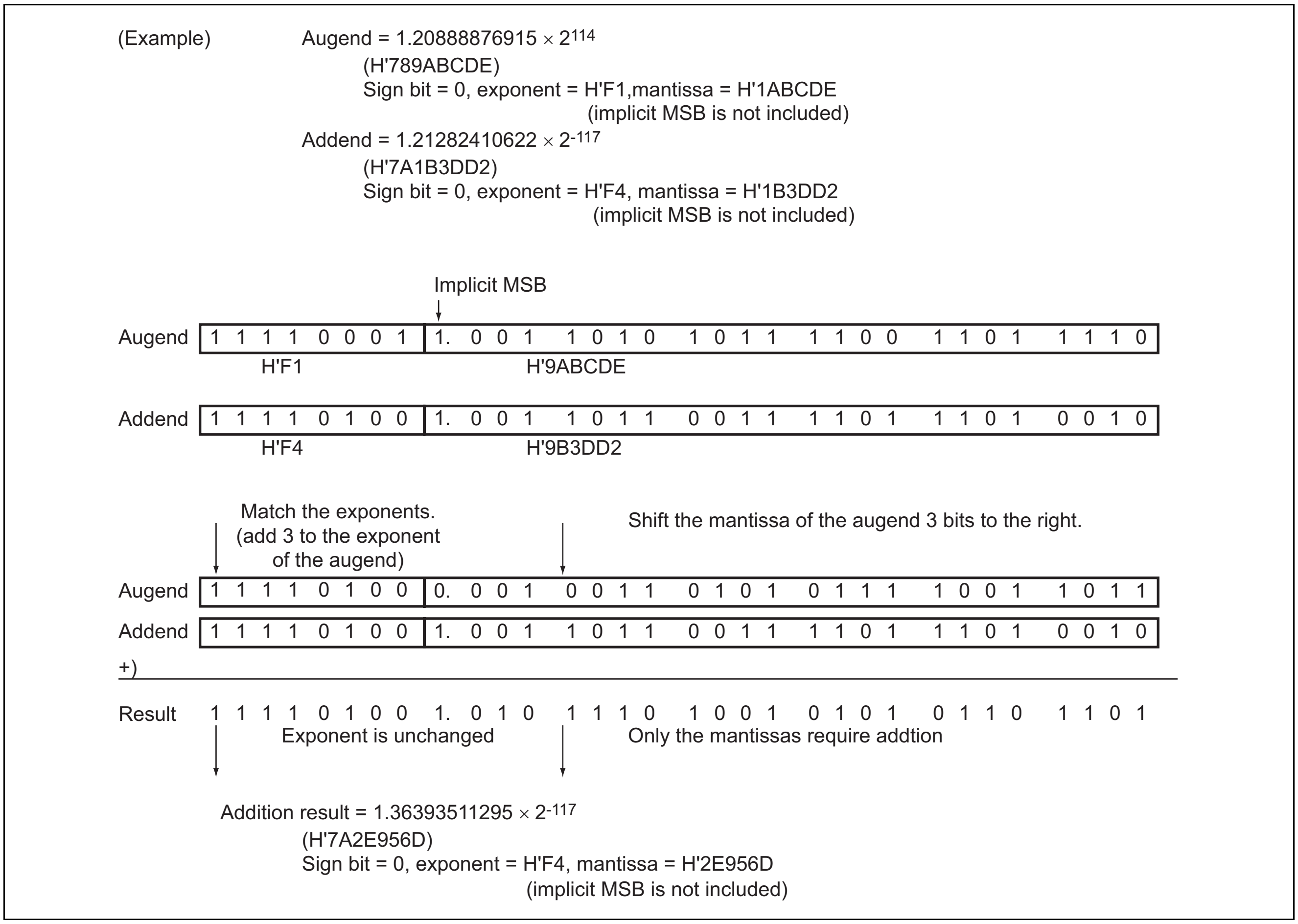
The input single-precision floating-point numbers are added together in the following sequence.

1. The augend and addend are checked for +∞ or -∞ values.
   1) If the exponent of the augend is H'FF, the output is as follows.

Sign Bit	Output Value
0 (positive)	H'7F800000 (+∞)
1 (negative)	H'FF800000 (-∞)

   2) If the augend is not +∞ or -∞ but the exponent of the addend is H'FF, the output will be as indicated in 1) above.

2. The augend and addend are checked for zero values.
   If either the augend or the addend is zero, the output is simply the value of the non-zero argument (if both are zero, the output is H'00000000).
3. The exponents of the augend and addend are matched.
   The smaller exponent is incremented until the exponents are the same, simultaneously shifting the mantissa (including the implicit MSB) one digit to the right per increment. With a number in the denormalized format, 1 is added to the exponent at the beginning of this exponent matching process, and the MSB of the mantissa is taken as implicitly being zero.
4. The mantissas are added.
5. The result of addition is corrected to produce a number in the floating-point data format.



(Example) Augend = 1.20888876915 × 2^{114}
(H'789ABCDE)
Sign bit = 0, exponent = H'F1,mantissa = H'1ABCDE
(implicit MSB is not included)

Addend = 1.21282410622 × 2^{-117}
(H'7A1B3DD2)
Sign bit = 0, exponent = H'F4, mantissa = H'1B3DD2
(implicit MSB is not included)

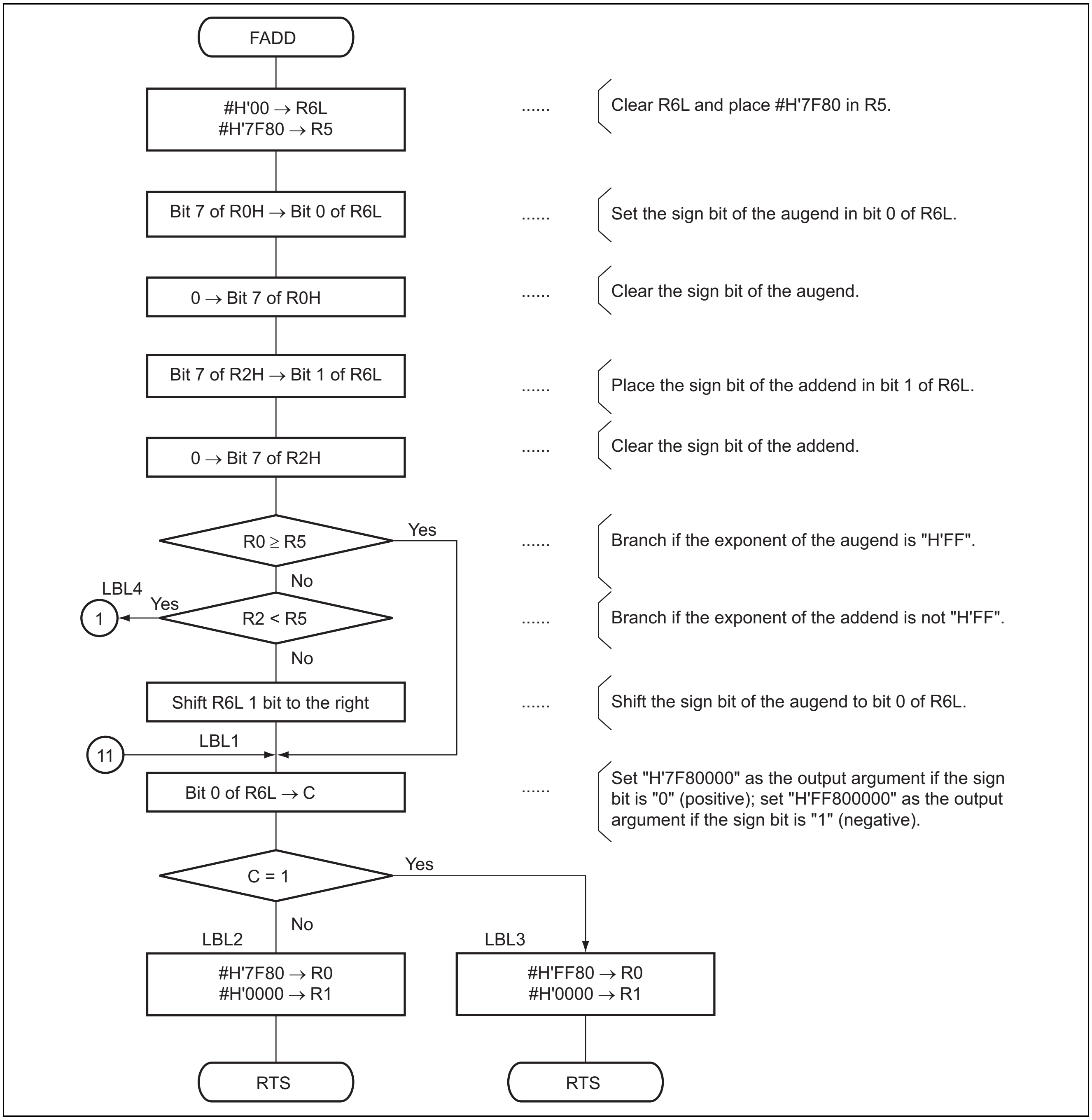
```
                                        Implicit MSB
                                        ↓
Augend | 1 1 1 1  0 0 0 1 | 1.  0 0 1   1 0 1 0   1 0 1 1   1 1 0 0   1 1 0 1   1 1 1 0 |
             H'F1                   H'9ABCDE

Addend | 1 1 1 1  0 1 0 0 | 1.  0 0 1   1 0 1 1   0 0 1 1   1 1 0 1   1 1 0 1   0 0 1 0 |
             H'F4                   H'9B3DD2

         Match the exponents.              Shift the mantissa of the augend 3 bits to the right.
        ↓(add 3 to the exponent       ↓
            of the augend)
Augend | 1 1 1 1  0 1 0 0 | 0.  0 0 1   0 0 1 1   0 1 0 1   0 1 1 1   1 0 0 1   1 0 1 1 |
Addend | 1 1 1 1  0 1 0 0 | 1.  0 0 1   1 0 1 1   0 0 1 1   1 1 0 1   1 1 0 1   0 0 1 0 |
+)
---------------------------------------------------------------------------------------------
Result   1 1 1 1  0 1 0 0   1.  0 1 0   1 1 1 0   1 0 0 1   0 1 0 1   0 1 1 0   1 1 0 1
        ↓   Exponent is unchanged     ↓    Only the mantissas require addtion
```

Addition result = 1.36393511295 × 2^{-117}
(H'7A2E956D)
Sign bit = 0, exponent = H'F4, mantissa = H'2E956D
(implicit MSB is not included)

7. Flowchart



...... Clear R6L and place #H'7F80 in R5.

...... Set the sign bit of the augend in bit 0 of R6L.

...... Clear the sign bit of the augend.

...... Place the sign bit of the addend in bit 1 of R6L.

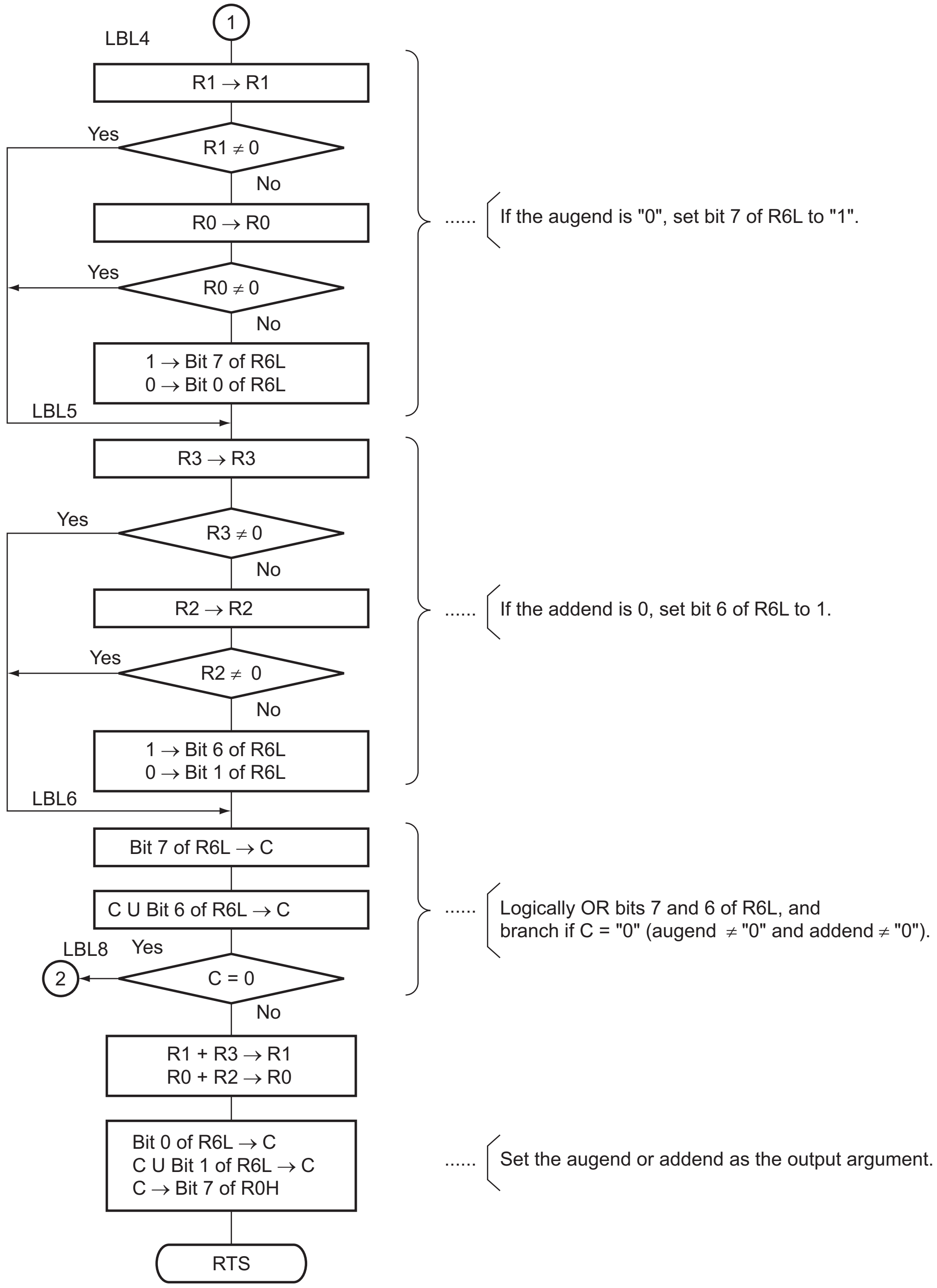
...... Clear the sign bit of the addend.

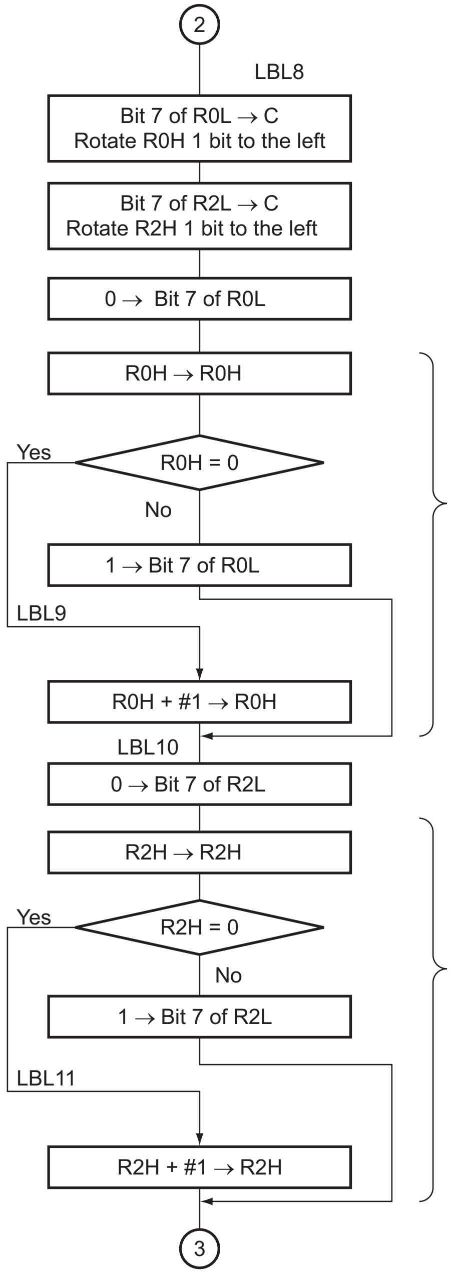
...... Branch if the exponent of the augend is "H'FF".

...... Branch if the exponent of the addend is not "H'FF".

...... Shift the sign bit of the augend to bit 0 of R6L.

...... Set "H'7F80000" as the output argument if the sign bit is "0" (positive); set "H'FF800000" as the output argument if the sign bit is "1" (negative).


1
LBL4
R1 → R1
Yes
R1 ≠ 0
No
R0 → R0
Yes
R0 ≠ 0
No
1 → Bit 7 of R6L
0 → Bit 0 of R6L
LBL5
...... If the augend is "0", set bit 7 of R6L to "1".
R3 → R3
Yes
R3 ≠ 0
No
R2 → R2
Yes
R2 ≠ 0
No
1 → Bit 6 of R6L
0 → Bit 1 of R6L
LBL6
...... If the addend is 0, set bit 6 of R6L to 1.
Bit 7 of R6L → C
C U Bit 6 of R6L → C
LBL8
Yes
2
C = 0
No
...... Logically OR bits 7 and 6 of R6L, and branch if C = "0" (augend ≠ "0" and addend ≠ "0").
R1 + R3 → R1
R0 + R2 → R0
Bit 0 of R6L → C
C U Bit 1 of R6L → C
C → Bit 7 of R0H
...... Set the augend or addend as the output argument.
RTS



...... Set the exponent of the augend in R0H.

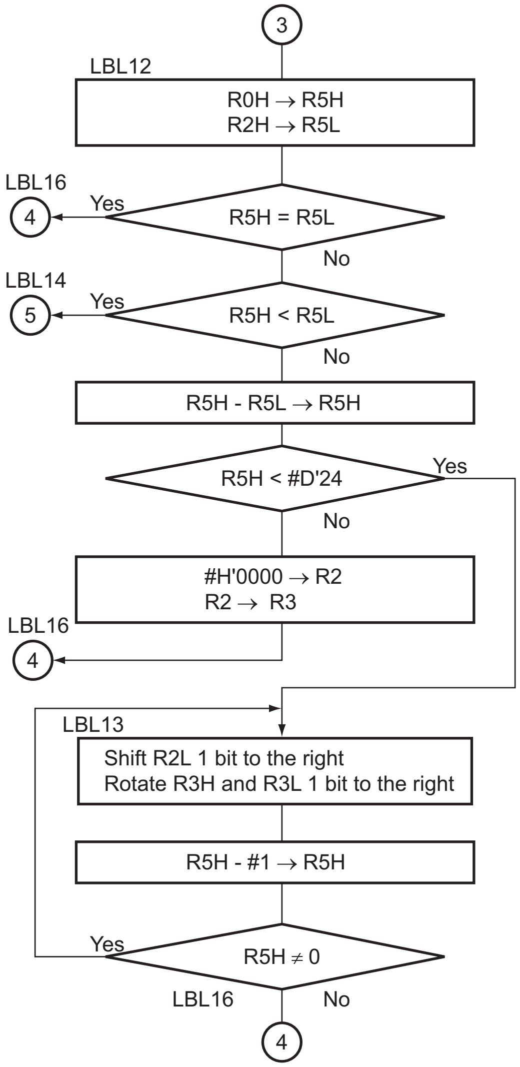
...... Set the exponent of the addend in R2H.

...... Clear bit 7 of R0L to 0.

...... If the augend is in normalized format (R0H ≠ 0), set bit 7 of R0L to "1"; if the augend is in denormalized format (R0H = 0), add #1 to R0H.

...... Clear bit 7 of R2L to 0.

...... If the addend is in normalized format (R2H ≠ 0), set bit 7 of R2L to "1"; if the addend is in denormalized format (R2H = 0), add #1 to R2H.



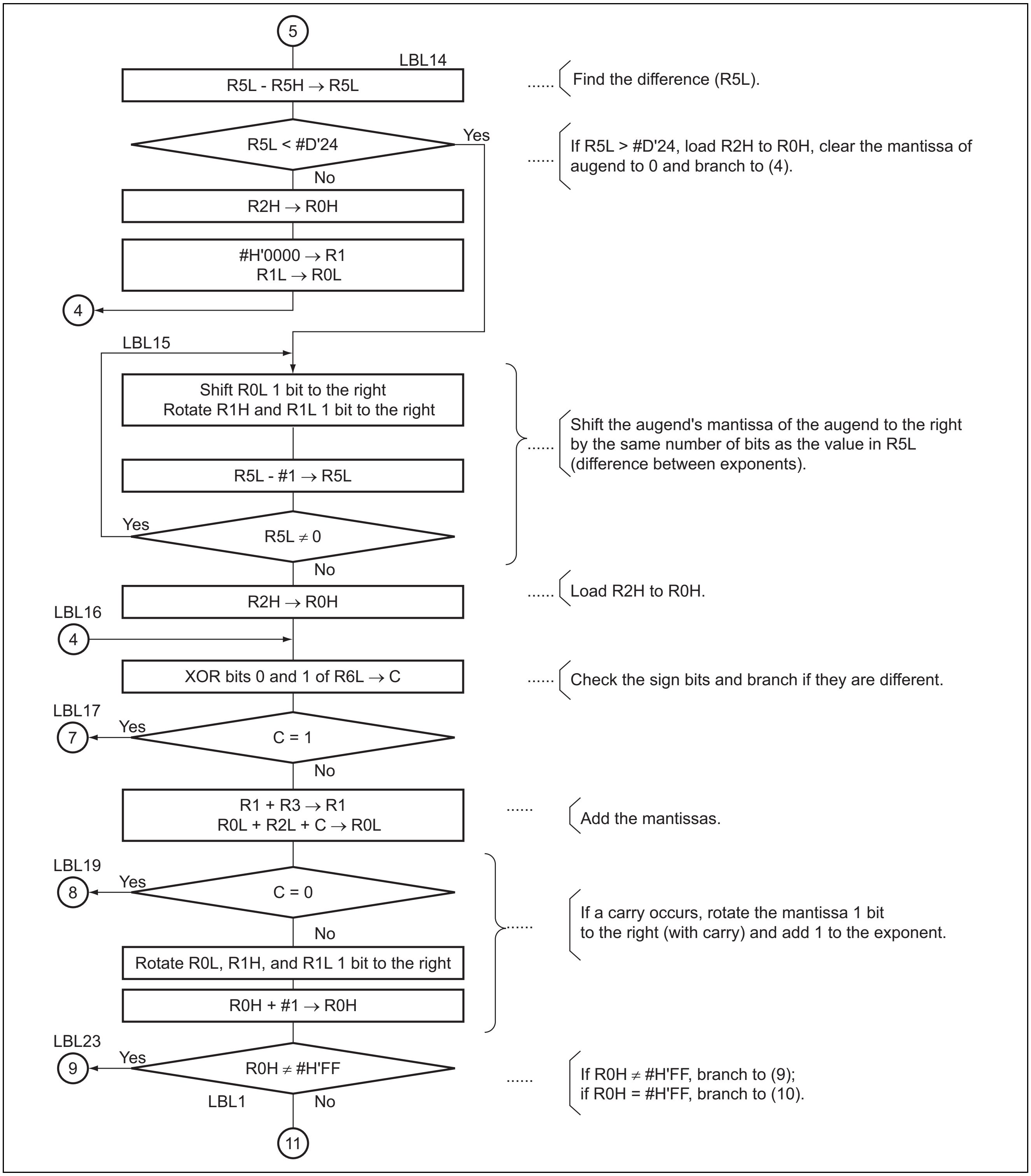
Set the exponent (R0H) of the augend in R5H.
Set the exponent (R2H) of the addend in R5L.

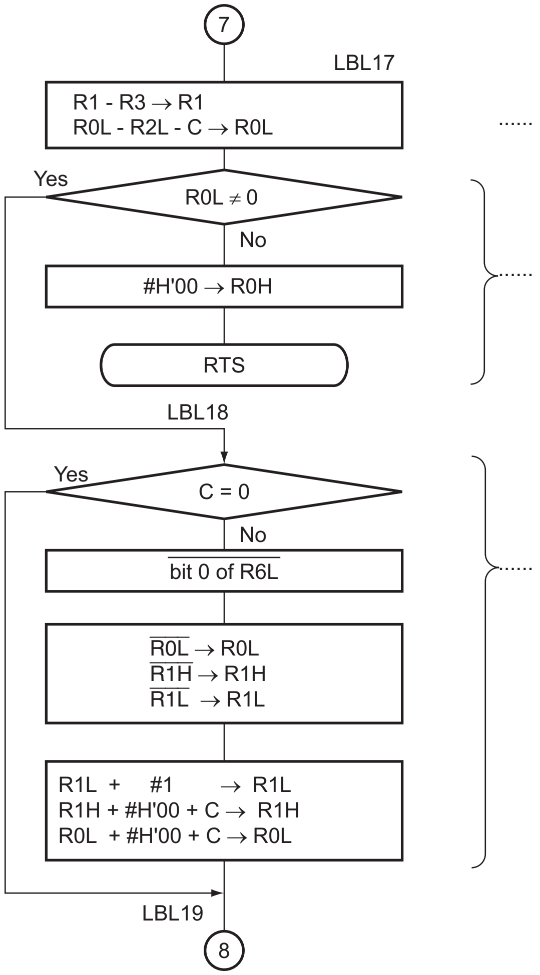
Compare R5H with R5L, and branch to (4)
if R5H = R5L or (5) if R5H < R5L.

If R5H > R5L, find the difference (R5H).

If R5H > #24, clear the addend and branch to (4)

Shift the mantissa of the addend to the right by the same number of bits as the Value in R5H (difference between exponents).

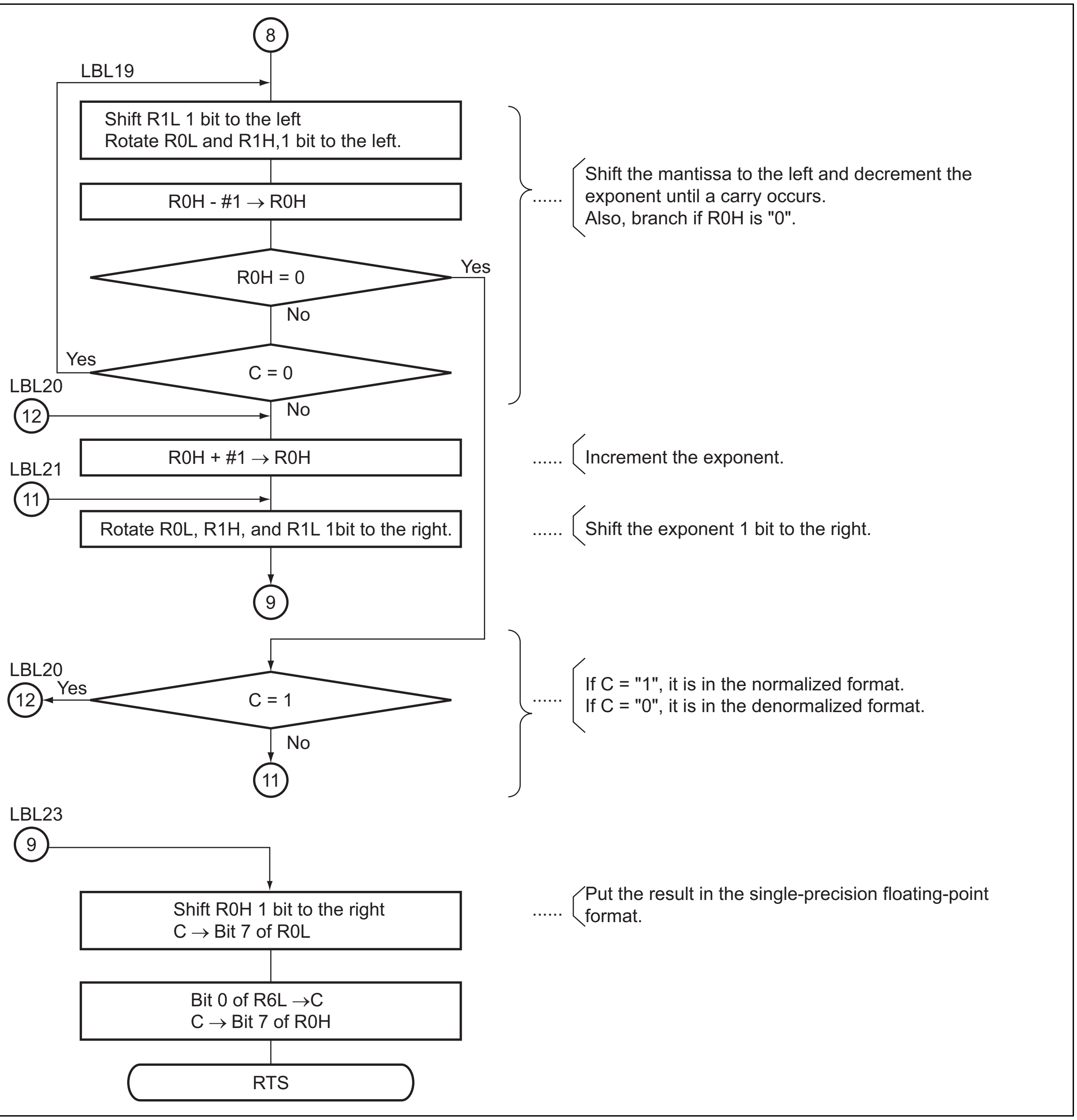

5
LBL14
R5L - R5H → R5L
Find the difference (R5L).
R5L < #D'24
Yes
No
If R5L > #D'24, load R2H to R0H, clear the mantissa of augend to 0 and branch to (4).
R2H → R0H
#H'0000 → R1
R1L → R0L
4
LBL15
Shift R0L 1 bit to the right
Rotate R1H and R1L 1 bit to the right
Shift the augend's mantissa of the augend to the right by the same number of bits as the value in R5L (difference between exponents).
R5L - #1 → R5L
Yes
R5L ≠ 0
No
R2H → R0H
Load R2H to R0H.
LBL16
4
XOR bits 0 and 1 of R6L → C
Check the sign bits and branch if they are different.
LBL17
7
Yes
C = 1
No
R1 + R3 → R1
R0L + R2L + C → R0L
Add the mantissas.
LBL19
8
Yes
C = 0
No
If a carry occurs, rotate the mantissa 1 bit to the right (with carry) and add 1 to the exponent.
Rotate R0L, R1H, and R1L 1 bit to the right
R0H + #1 → R0H
LBL23
9
Yes
R0H ≠ #H'FF
LBL1
No
If R0H ≠ #H'FF, branch to (9);
if R0H = #H'FF, branch to (10).
11



Carry out subtraction of the mantissas.

If the result of subtraction is 0, set H'00 in R0H and end.

If a borrow is generated, invert the sign bit and take the two's compliment of the mantissa.


8
LBL19
Shift R1L 1 bit to the left
Rotate R0L and R1H,1 bit to the left.
R0H - #1 → R0H
R0H = 0
Yes
No
Yes
C = 0
No
LBL20
12
R0H + #1 → R0H
LBL21
11
Rotate R0L, R1H, and R1L 1bit to the right.
9
LBL20
12
Yes
C = 1
No
11
LBL23
9
Shift R0H 1 bit to the right
C → Bit 7 of R0L
Bit 0 of R6L →C
C → Bit 7 of R0H
RTS
......
Shift the mantissa to the left and decrement the exponent until a carry occurs.
Also, branch if R0H is "0".
......
Increment the exponent.
......
Shift the exponent 1 bit to the right.
......
If C = "1", it is in the normalized format.
If C = "0", it is in the denormalized format.
......
Put the result in the single-precision floating-point format.

8. Program Listing

```
                             ;**************************************************************************
                             ;*                                                                        *
                             ;*          NAME : FLOATING POINT ADDITION                                *
                             ;*                 (FADD)                                                 *
                             ;*                                                                        *
                             ;**************************************************************************
                             ;*                                                                        *
                             ;*        ENTRY:   R0     (HIGHER WORD OF AUGEND)                         *
                             ;*                 R1     (LOWER WORD OF AUGEND)                          *
                             ;*                 R2     (HIGHER WORD OF ADDEND)                         *
                             ;*                 R3     (LOWER WORD OF ADDEND)                          *
                             ;*                                                                        *
                             ;*       RETURNS:  R0     (HIGHER WORD OF RESULT)                         *
                             ;*                 R1     (LOWER WORD OF RESULT)                          *
                             ;*                                                                        *
                             ;**************************************************************************
                             ;
                                     .CPU        300HN
0000                                 .SECTION    FADD_code,CODE,ALIGN=2
                                     .EXPORT     FADD
                             ;
     00000000                FADD    .EQU        $           ;Entry point
0000 FE00                            MOV.B       #H'00,R6L   ;Clear R0L
0002 79057F80                        MOV.W       #H'7F80,R5  ;Set H'7F80
                             ;
0006 7770                            BLD         #7,R0H
0008 670E                            BST         #0,R6L      ;Set sign bit to bit 0 of R6L
000A 7270                            BCLR        #7,R0H      ;Bit clear bit 7 of R0H
                             ;
000C 7772                            BLD         #7,R2H      ;Set sign bit to bit 1 of R6L
000E 671E                            BST         #1,R6L
0010 7272                            BCLR        #7,R2H      ;Bit Clear bit 7 of R2H
                             ;
0012 1D05                            CMP.W       R0,R5
0014 4306                            BLS         LBL1        ;Branch if "exponent of augend"= "H'FF"
0016 1D25                            CMP.W       R2,R5
0018 421A                            BHI         LBL4        ;Branch if not "exponent of augend"= "H'FF"
001A 110E                            SHLR        R6L         ;Shift R6L 1 bit right
001C                         LBL1
001C 770E                            BLD         #0,R6L      ;Bit load sign bit
001E 450A                            BCS         LBL3        ;Branch if sign bit=1
0020                         LBL2
0020 79007F80                        MOV.W       #H'7F80,R0  ;Set maximum pos. number
0024 79010000                        MOV.W       #H'0000,R1
0028 5470                            RTS
002A                         LBL3
002A 7900FF80                        MOV.W       #H'FF80,R0  ;Set maximum neg. number
002E 79010000                        MOV.W       #H'0000,R1
0032 5470                            RTS
                             ;
0034                         LBL4
0034 0D11                            MOV.W       R1,R1
```

```
  53  0036 4608           53           BNE         LBL5        ;Branch if Z=0
  54  0038 0D00           54           MOV.W       R0,R0
  55  003A 4604           55           BNE         LBL5        ;Branch if Z=0
  56  003C 707E           56           BSET        #7,R6L      ;Bit set bit 7 of R6L
  57  003E 720E           57           BCLR        #0,R6L      ;Bit clear bit 0 of R6L
  58  0040                58   LBL5
  59  0040 0D33           59           MOV.W       R3,R3
  60  0042 4608           60           BNE         LBL6        ;Branch if Z=0
  61  0044 0D22           61           MOV.W       R2,R2
  62  0046 4604           62           BNE         LBL6        ;Branch if Z=0
  63  0048 706E           63           BSET        #6,R6L      ;Bit set bit 6 of R6L
  64  004A 721E           64           BCLR        #1,R6L      ;Bit clear bit 1 of R6L
  65  004C                65   LBL6
  66  004C 777E           66           BLD         #7,R6L
  67  004E 746E           67           BOR         #6,R6L
  68  0050 440C           68           BCC         LBL8        ;Branch if not augend=addend=0
  69  0052 0931           69           ADD.W       R3,R1       ;Set augend and addend to result
  70  0054 0920           70           ADD.W       R2,R0
  71  0056 770E           71           BLD         #0,R6L
  72  0058 741E           72           BOR         #1,R6L
  73  005A 6770           73           BST         #7,R0H      ;Set sign bit
  74  005C 5470           74           RTS
  75                      75   ;
  76  005E                76   LBL8
  77  005E 7778           77           BLD         #7,R0L
  78  0060 1200           78           ROTXL       R0H         ;Set exponent of augend to R0H
  79                      79   ;
  80  0062 777A           80           BLD         #7,R2L
  81  0064 1202           81           ROTXL       R2H         ;Set exponent of addend to R0L
  82                      82   ;
  83  0066 7278           83           BCLR        #7,R0L
  84  0068 0C00           84           MOV.B       R0H,R0H
  85  006A 4704           85           BEQ         LBL9        ;Branch if augend is normalized
  86  006C 7078           86           BSET        #7,R0L      ;Set implicit MSB to augend
  87  006E 4002           87           BRA         LBL10       ;Branch always
  88  0070                88   LBL9
  89  0070 8001           89           ADD.B       #H'01,R0H
  90  0072                90   LBL10
  91  0072 727A           91           BCLR        #7,R2L
  92  0074 0C22           92           MOV.B       R2H,R2H
  93  0076 4704           93           BEQ         LBL11       ;Branch if addend is normalized
  94  0078 707A           94           BSET        #7,R2L      ;Set implicit MSB to augend
  95  007A 4002           95           BRA         LBL12       ;Branch always
  96  007C                96   LBL11
  97  007C 8201           97           ADD.B       #H'01,R2H
  98                      98   ;
  99  007E                99   LBL12
 100  007E 0C05          100           MOV.B       R0H,R5H
 101  0080 0C2D          101           MOV.B       R2H,R5L
 102  0082 1CD5          102           CMP.B       R5L,R5H
 103  0084 4738          103           BEQ         LBL16       ;Branch if R5H = R5L
 104  0086 451A          104           BCS         LBL14       ;Branch if R5H < R5L
 105                     105   ;
 106  0088 18D5          106           SUB.B       R5L,R5H
```

```
107   008A A518            107             CMP.B     #D'24,R5H   ;Set bit counter
108   008C 4508            108             BCS       LBL13       ;Branch if R5H < D'24
109   008E 79020000        109             MOV.W     #H'0000,R2  ;Clear addend
110   0092 0D23            110             MOV.W     R2,R3
111   0094 4028            111             BRA       LBL16       ;Branch always
112   0096                 112     LBL13
113   0096 110A            113             SHLR      R2L         ;Shift mantissa of addend 1 bit left
114   0098 1303            114             ROTXR     R3H
115   009A 130B            115             ROTXR     R3L
116   009C 1A05            116             DEC.B     R5H         ;Decrement bit counter
117   009E 46F6            117             BNE       LBL13       ;Branch Z=0
118   00A0 401C            118             BRA       LBL16       ;Branch always
119   00A2                 119     LBL14
120   00A2 185D            120             SUB.B     R5H,R5L
121   00A4 AD18            121             CMP.B     #D'24,R5L
122   00A6 450A            122             BCS       LBL15       ;Branch if R5L<D'24
123   00A8 0C20            123             MOV.B     R2H,R0H
124   00AA 79010000        124             MOV.W     #H'0000,R1  ;Clear augend
125   00AE 0C98            125             MOV.B     R1L,R0L
126   00B0 400C            126             BRA       LBL16       ;Branch always
127                        127     ;
128   00B2                 128     LBL15
129   00B2 1108            129             SHLR      R0L         ;Shift mantissa of augend 1 bit right
130   00B4 1301            130             ROTXR     R1H
131   00B6 1309            131             ROTXR     R1L
132   00B8 1A0D            132             DEC.B     R5L         ;Decrement bit counter
133   00BA 46F6            133             BNE       LBL15       ;Branch if Z=0
134   00BC 0C20            134             MOV.B     R2H,R0H
135                        135
136                        136     ;
137   00BE                 137     LBL16
138   00BE 770E            138             BLD       #0,R6L
139   00C0 751E            139             BXOR      #1,R6L
140   00C2 4516            140             BCS       LBL17       ;Branch if different sign bit
141                        141     ;
142   00C4 0931            142             ADD.W     R3,R1       ;Addition mantissa
143   00C6 0EA8            143             ADDX.B    R2L,R0L
144   00C8 442A            144             BCC       LBL19       ;Branch if C=0
145   00CA 1308            145             ROTXR     R0L         ;Rotate mantissa 1 bit right
146   00CC 1301            146             ROTXR     R1H
147   00CE 1309            147             ROTXR     R1L
148   00D0 8001            148             ADD.B     #H'01,R0H   ;Increment exponent
149   00D2 A0FF            149             CMP.B     #H'FF,R0H
150   00D4 4638            150             BNE       LBL23       ;Branch if not exponent=H'FF
151   00D6 5A000000        151             JMP       @LBL1       ;Jump
152   00DA                 152     LBL17
153   00DA 1931            153             SUB.W     R3,R1       ;Subtract mantissa
154   00DC 1EA8            154             SUBX.B    R2L,R0L
155   00DE 4604            155             BNE       LBL18       ;Branch if Z=0
156   00E0 F000            156             MOV.B     #H'00,R0H   ;Clear R0H
157   00E2 5470            157             RTS
158   00E4                 158     LBL18
159   00E4 440E            159             BCC       LBL19       ;Branch if C=0
160   00E6 710E            160             BNOT      #0,R6L      ;Bit not sign bit
```

```
161   00E8 1708           161             NOT        R0L        ;2's complement mantissa
162   00EA 1701           162             NOT        R1H
163   00EC 1709           163             NOT        R1L
164   00EE 8901           164             ADD.B      #H'01,R1L
165   00F0 9100           165             ADDX.B     #H'00,R1H
166   00F2 9800           166             ADDX.B     #H'00,R0L
167                       167     ;
168   00F4                168     LBL19
169   00F4 1009           169             SHLL       R1L        ;Shift mantissa 1 bit left
170   00F6 1201           170             ROTXL      R1H
171   00F8 1208           171             ROTXL      R0L
172   00FA 1A00           172             DEC.B      R0H        ;Decrement exponent
173   00FC 470C           173             BEQ        LBL22      ;Branch if exponent = 0
174   00FE 44F4           174             BCC        LBL19      ;Branch if exponent > 0
175   0100                175     LBL20
176   0100 0A00           176             INC.B      R0H        ;Increment exponent
177   0102                177     LBL21
178   0102 1308           178             ROTXR      R0L        ;Rotate mantissa 1 bit right
179   0104 1301           179             ROTXR      R1H
180   0106 1309           180             ROTXR      R1L
181   0108 4004           181             BRA        LBL23      ;Branch always
182   010A                182     LBL22
183   010A 45F4           183             BCS        LBL20      ;Branch if C=1
184   010C 40F4           184             BRA        LBL21      ;Branch always
185                       185     ;
186   010E                186     LBL23                           ;Correct into floating-point format
187   010E 1100           187             SHLR       R0H
188   0110 6778           188             BST        #7,R0L
189   0112 770E           189             BLD        #0,R6L
190   0114 6770           190             BST        #7,R0H
191   0116 5470           191             RTS
192                       192     ;
193                       193             .END
*****TOTAL ERRORS       0
*****TOTAL WARNINGS     0
```

<Reference> Description of Single-Precision Floating-Point Formats

Single-Precision Floating-Point Formats:

1. Internal Representation of Single-Precision Floating Point Numbers

   One of the following formats is used depending on the value of the single-precision floating-point data in this application note (a real number is indicated as R).

   1) Internal Representation When R=0

31	30	29	......	2	1	0
0	0	0	......	0	0	0

   All the 32 bits are 0.

   2) Normalized Format

31	30 23	22 0
S	α	β

   α is an index number with an 8-bit-long field. β is a mantissa with a 23-bit-long field. Here, the R value can be represented by the expression below (when 1 ≤ α ≤ 254).

   ↓Implicit MSB

   $R = 2^{S} \times 2^{\alpha-126} \times (1 + 2^{-1} \times \beta_{22} + 2^{-2} \times 21 + \ldots\ldots + 2^{-23} \times \beta_{0})$

   where, βi is the value of the i-th bit of β (0 ≤ i ≤ 22), and S is the sign bit.

   3) Denormalized Format

31	30 23	22 0
S	0 0 0 0 0 0 0 0	β

   β is a mantissa with a 23-bit-long field. This format is used to represent a real number that is too small to be represented by the normalized format.

   Here, the R value can be represented by the expression below.

   $R = 2^{S} \times 2^{-126} \times (2^{-1} \times \beta_{22} + 2^{-2} \times 21 + \ldots\ldots + 2^{-23} \times \beta_{0})$

   4) Infinity

31	30 23	22 0
S	1 1 1 1 1 1 1 1	β

   When S = 0: Plus infinity

   $R = +\infty$

   When S = 1: Minus infinity

   $R = -\infty$

2. Internal Representation Examples

S = B'0	(binary)
α = B'10000011	(binary)
β = B'1011100……0	(binary)

   Under the above conditions, the corresponding R value is represented as follows.

   $$R = 2^{0} \times 2^{131-126} \times (1 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-5})$$
   $$= 16 + 8 + 2 + 1 + 0.5 = 27.5$$

   1) Maximum and Minimum Values

   Here, the maximum and minimum values are absolute values. The maximum value is indicated as R_{MAX} and the minimum value is indicated as R_{MIN}. Up to the following values can be represented.

   $$R_{MAX} = 2^{254-127} \times (1 + 2^{-1} + 2^{-2} + 2^{-3} + \ldots\ldots + 2^{-23})$$
   $$\approx 3.27 \times 10^{38}$$
   $$R_{MIN} = 2^{-126} \times 2^{-23} = 2^{-140} \approx 1.40 \times 10^{-45}$$

Revision Record

Rev.	Date	Description Page	Summary
2.00	Feb.28.06	—	Format has been changed from Hitachi version to Renesas version.
3.00	Jun.12.06	13	Error correction

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.