Example Single Precision Floating Number

How would 763.5_{10} be represented as a single precision IEEE floating point number?

$763.5_{10} = 2FB.8_{16} = 1011111011.1_2 = 1.0111110111 \times 2^9$

Hence the sign bit = 0

the characteristic = 9+127 = $136_{10} = 10001000_2$

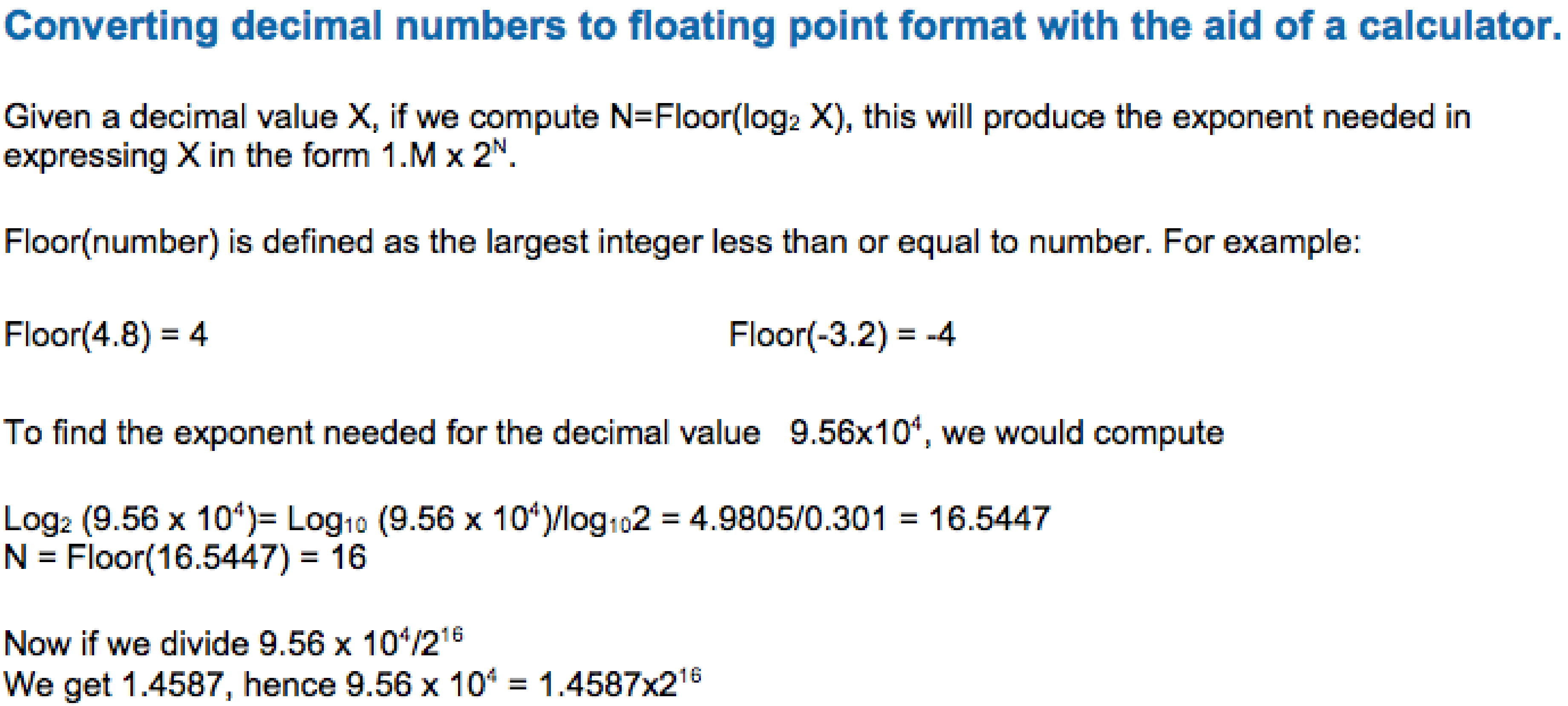
the 23-bit mantissa is 01111101110000000000000

The corresponding IEEE single precision representation is

s	characteristic	mantissa
0	10001000	01111101110000000000000

This 32-bit pattern can be written in short hand form using 8 hex digits: 443EE000

Example Single Precision Floating Number



Converting decimal numbers to floating point format with the aid of a calculator.

Given a decimal value X, if we compute N=Floor($\log_2$ X), this will produce the exponent needed in expressing X in the form 1.M x 2^N.

Floor(number) is defined as the largest integer less than or equal to number. For example:

Floor(4.8) = 4 Floor(-3.2) = -4

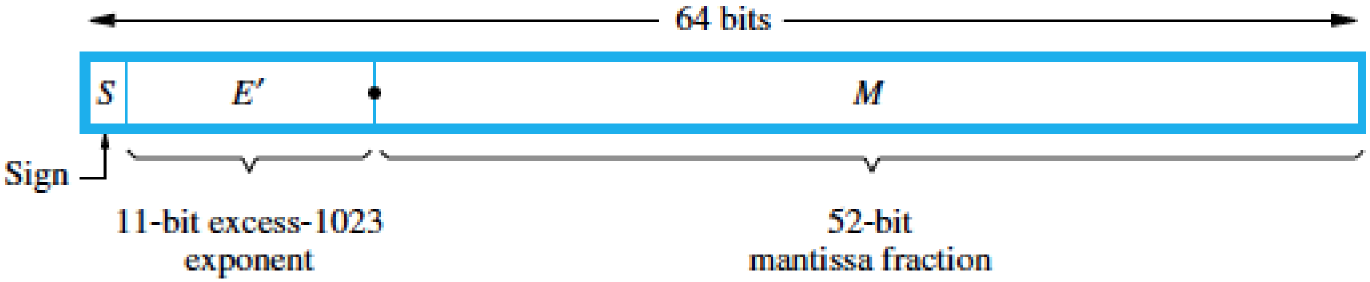
To find the exponent needed for the decimal value $9.56\text{x}10^4$, we would compute

$\text{Log}_2 (9.56 \text{ x } 10^4) = \text{Log}_{10} (9.56 \text{ x } 10^4)/\log_{10}2 = 4.9805/0.301 = 16.5447$
N = Floor(16.5447) = 16

Now if we divide $9.56 \text{ x } 10^4/2^{16}$
We get 1.4587, hence $9.56 \text{ x } 10^4 = 1.4587\text{x}2^{16}$

Multiply fraction to convert to 24-bit integer: 0.4587 x 2^{24} = 7695708
So 0.4587 = 7695708 × 2^{-24} = 0x756D5C × 2^{-24} = 0.756D5C
Mantissa is 011101010110110101011100
Characteristic = 16 + 127 = 143 = 0x8F

Double Precision Floating Point



$$\text{Value represented} = \pm 1.M \times 2^{E'-1023}$$

E’ (characteristic) = exponent + 1023

E’=0 and M≠0 for denormalized numbers
E’=0 and M=0 for true zero
E’=2047 and M≠0 for NaN (e.g. 0/0 or $\sqrt{-1}$)
E’=2047 and M=0 for ±∞

Provides 15 decimal places or precision
Approximate range of $\pm(10^{\pm 308})$

Floating Point Addition & Subtraction

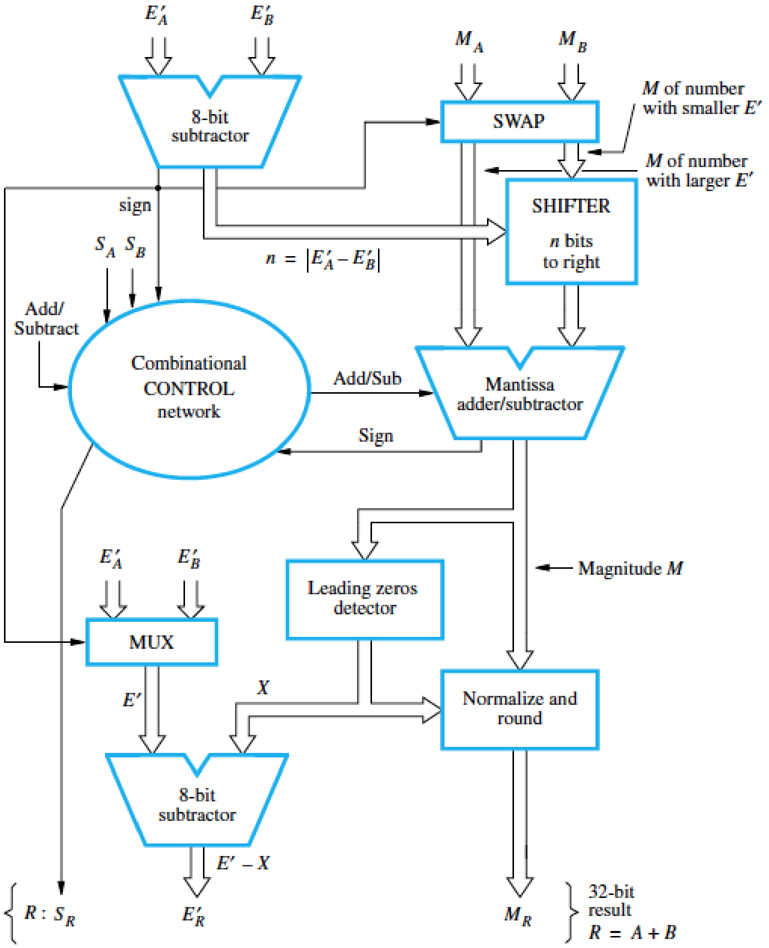
1. Select number with smaller exponent

2. Shift its mantissa right n bits
   where n is the difference between the exponents

3. Add/subtract the two mantissas and set sign of result

4. Normalize and round the result if necessary

Floating Point Addition & Subtraction Unit



Floating Point Addition & Subtraction Unit

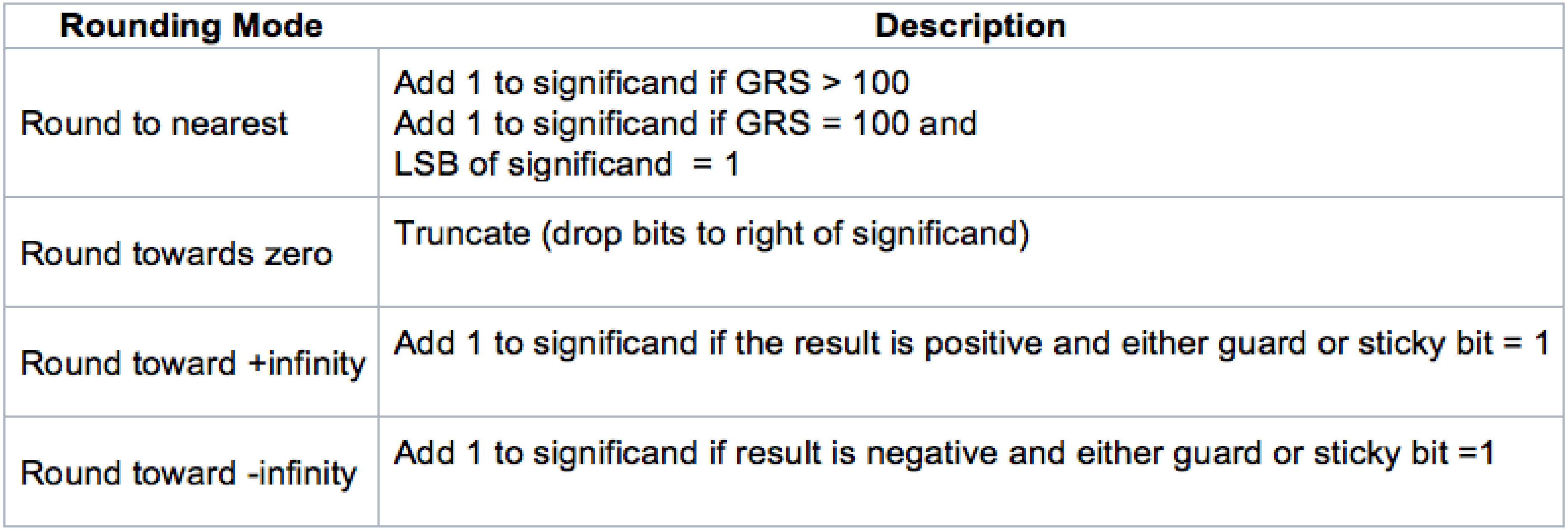
mantissa is shifted right to align for addition or subtraction

- low bits are lost when this is done
- Guard, round and sticky bits increase the accuracy of result
- sticky bit remains a 1 once a non-zero bit passes through
- guard and round retain the two most recent bits shifted out

Rounding of the result can be based on these 3 bits (GRS)

Most systems allow the programmer to specify the rounding mode

Rounding Modes



Rounding Mode	Description
Round to nearest	Add 1 to significand if GRS > 100 Add 1 to significand if GRS = 100 and LSB of significand = 1
Round towards zero	Truncate (drop bits to right of significand)
Round toward +infinity	Add 1 to significand if the result is positive and either guard or sticky bit = 1
Round toward -infinity	Add 1 to significand if result is negative and either guard or sticky bit =1

Floating Point Multiply & Divide

Multiply rule:

1. Exponent of product = sum of exponents of factors (check for exponent overflow)
2. mantissa of product = 1st mantissa times 2nd mantissa
3. Normalize and round result if necessary

Divide rule:

1. Exponent of quotient = dividend exponent – divisor exponent (check for underflow)
2. Quotient mantissa = dividend mantissa / divisor mantissa
3. Normalize and round result if necessary