# UNIT II
# ARITHMETIC OPERATIONS

Addition and subtraction of signed numbers – Design of fast adders – Multiplication of positive numbers - Signed operand multiplication- fast multiplication – Integer division – **Floating point numbers and operations**

# Recap the previous Class

# Floating point numbers

- Representation for non - integral numbers
  - Including very small and very large numbers
- Like scientific notation
  - $-2.34 \times 10^{56}$ - Normalized
  - $+0.002 \times 10^{-4}$ – Not Normalized
  - $+987.02 \times 10^{9}$ – Not Normalized
- In binary
  - $\pm 1.xxxxxxx_2 \times 2^{yyyy}$

# Floating point Standard

- Defined by IEEE Std754 - 1985
- Two representations
  - Single precision (32 - bit)
  - Double precision (64 - bit)

| | Sign | Exponent | Fraction | Bias |
|---|---|---|---|---|
| Single precision | 1[31] | 8[30-23] | 23[22-00] | 127 |
| Double precision | 1[63] | 11[62-52] | 52[51-00] | 1023 |

# IEEE Floating point Representation

$$x = (-1)^S \times (1 + \text{Fraction}) \times 2^{(\text{Exponent} - \text{Bias})}$$

**Exponent : excess representation: actual exponent + Bias**

Ensures exponent is unsigned

single: 8 bits
double: 11 bits

single: 23 bits
double: 52 bits

S: sign bit

0 –non - negative

1 - negative

Single: Bias = 127
Double: Bias = 1023

# Floating Point Example

Represent 0.75 in Single and Double precession

Step 1: Convert decimal to binary Number 0.75 ------ 0.11

Step 2: Scientific Notation 0.11--------$0.11 \times 2^{0}$

Step 3: Normalize the Scientific Notation $0.11 \times 2^{0}$ ------- $1.1 \times 2^{-1}$

| 0 | 01111110 | 10000000000000000000000 |
|---|---|---|

$$x = (-1)^{S} \times (1 + \text{Fraction}) \times 2^{(\text{Exponent} - \text{Bias})}$$

Exponent = Actual Exponent +Bias
= -1 +127 = 126

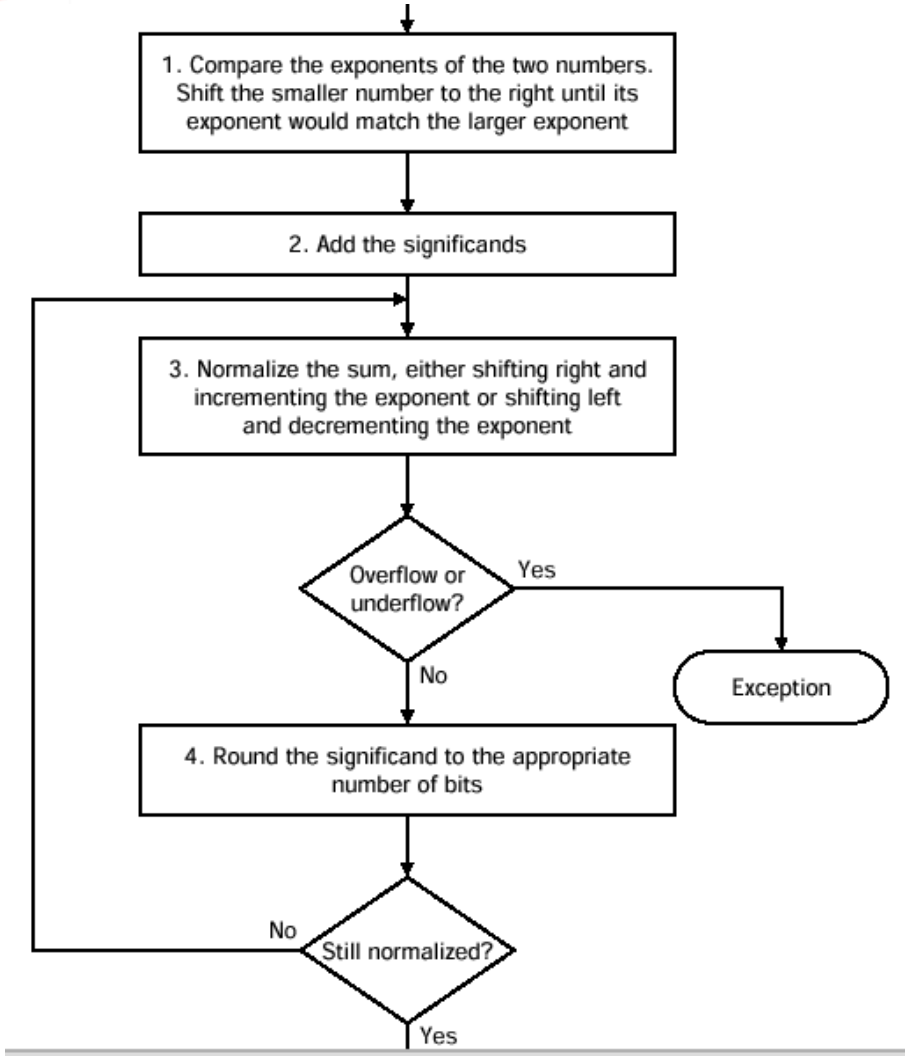
$$X = 1 \times (1.1) \times 2^{126-127}$$

# Example for Floating Point Representation

- Represent –0.75
  - $-0.75 = (-1)^1 \times 1.1_2 \times 2^{-1}$
    $= -1 \times 1.\ ½ \times ½$
    $= -1.5 * .5 = -0.75$
  - S = 1
  - Fraction = $1000\ldots00_2$
  - Exponent = –1 + Bias
    - Single: $-1 + 127 = 126 = 01111110_2$
    - Double: $-1 + 1023 = 1022 = 01111111110_2$

- Single: 10111111010000…00
- Double: 10111111111010000…00

# Floating Point Operation - Addition




OVERFLOW – Positive Exponent too larger to fit in the exponent Field

UNDERFLOW – negative Exponent too larger to fit in the exponent Field

# Example -Floating Point Operation - Addition

Consider a 4-digit decimal example $9.999 \times 10^{1} + 1.610 \times 10^{-1}$

1. Align decimal points
   - Shift number with smaller exponent $9.999 \times 10^{1} + 0.016 \times 10^{1}$
2. Add significands
   - $9.999 \times 10^{1} + 0.016 \times 10^{1} = 10.015 \times 10^{1}$
3. Normalize result & check for over/underflow
   - $1.0015 \times 10^{2}$
4. Round and renormalize if necessary
   - $1.002 \times 10^{2}$

# Example -Floating Point Operation - Addition

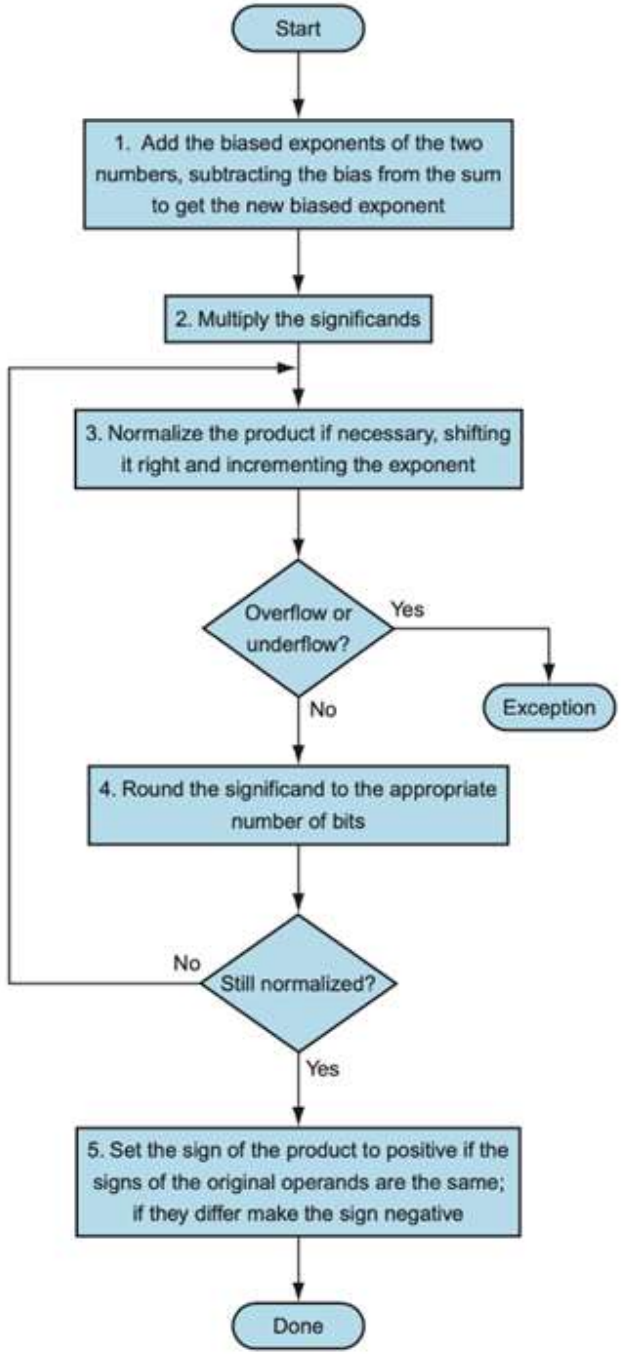
Now consider a 4-digit binary example (0.5 + –0.4375)

$1.000_2 \times 2^{-1} + -1.110_2 \times 2^{-2}$

1. Align binary points
   - Shift number with smaller exponent $1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-1}$
2. Add significands
   - $1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-}1 = 0.001_2 \times 2^{-1}$
3. Normalize result & check for over/underflow
   - $1.000_2 \times 2^{-4}$, with no over/underflow
4. Round and renormalize if necessary

$1.000_2 \times 2^{-4}$ (no change) = 0.0625

# Floating Point Operation - Multiplication



# Floating Point Multiplication

Consider a 4-digit decimal $1.110 \times 10^{10} \times 9.200 \times 10^{-5}$

**Add exponents** - For biased exponents, subtract bias from sum

New exponent = 10 + –5 = 5 Unbiased = 5+127 = 132

**Multiply significands**

$1.110 \times 9.200 = 10.212$ = $10.212 \times 10^{5}$

**Normalize result & check for over/underflow**

$1.0212 \times 10^{6}$

**Round and renormalize if necessary**

$1.021 \times 10^{6}$

**Determine sign of result from signs of operands**

$+1.021 \times 10^{6}$

# Floating Point Multiplication

Consider a 4-digit binary $1.000_2 \times 2^{-1} \times -1.110_2 \times 2^{-2}$ $(0.5 \times -0.4375)$

**Add exponents**

Unbiased: $-1 + -2 = -3$ ; Biased: $= -3 + 127 = 124$

**Multiply significands**

$1.000_2 \times 1.110_2 = 1.1102$ => $1.110_2 \times 2^{-3}$

**Normalize result & check for over/underflow**

$1.110_2 \times 2^{-3}$ (no change) with no over/underflow

**Round and renormalize if necessary**

$1.110_2 \times 2^{-3}$ (no change)

**Determine sign:** +ve × −ve => −ve

$-1.1102 \times 2^{-3} = -0.21875$

# Guard bit and Truncation

- Guard bits
  - Extra bits during intermediate steps to yield maximum accuracy in the final result
- They need to be removed when generating the final result
  - Chopping
    - simply remove guard bits
  - Von Neumann rounding
    - if all guard bits 0, chop, else 1
  - Rounding
    - Add 1 to LSB if guard MSB = 1

# TEXT BOOK

Carl Hamacher, Zvonko Vranesic and Safwat Zaky, “Computer Organization”, McGraw-Hill, 6th Edition 2012.

# REFERENCES


1. David A. Patterson and John L. Hennessey, “Computer organization and design”, MorganKauffman ,Elsevier, 5th edition, 2014.
2. William Stallings, “Computer Organization and Architecture designing for Performance”, Pearson Education 8th Edition, 2010
3. John P.Hayes, “Computer Architecture and Organization”, McGraw Hill, 3rd Edition, 2002
4. M. Morris R. Mano “Computer System Architecture” 3rd Edition 2007
5. David A. Patterson “Computer Architecture: A Quantitative Approach”, Morgan Kaufmann; 5th edition 2011


# THANK YOU