Computer Organization and Architecture

Floating Point Arithmetic

Veena Thenkanidiyoor

National Institute of Technology Goa

Recap

- Representation for real numbers
- Floating point representation
- Signed exponent

Floating Point Numbers and
Excess-k Format for Signed Integers

Excess-k Representation for Signed Integers

- Signed integers can also be represented using Excess-k format
- Integers obtained after representing the signed integers in excess-k format are called as biased integers
- Biased integer = true integer + k
  - k is called as bias
  - For any n-bit integers, bias, $k=2^{(n-1)}-1$
  - True integer: The actual value of an integer. It can be positive or negative value
  - Biased integer: The positive integer value obtained by adding bias to the actual integer
- This representation is typically used in representing the exponent part of the floating point number

Illustration of Excess-7 Format for 4-bit Signed Integers

- Based integer = true integer + k
- True Integer = X: $x_3\,x_2\,x_1\,x_0$
- Based integer = $\hat{X}$: $\hat{x}_3\hat{x}_2\hat{x}_1\hat{x}_0$
  - For 4-bit signed integers, n=4
  - bias, $k=2^{(n-1)}-1 = 2^3-1 = 7$
- Range of positive values the biased integer can hold is 0 to 15

X	$\hat{X}$	$\hat{X}$ in binary
-7	0	0000
-6	1	0001
-5	2	0010
-4	3	0011
-3	4	0100
-2	5	0101
-1	6	0110
0	7	0111
1	8	1000
2	9	1001
3	10	1010
4	11	1011
5	12	1100
6	13	1101
7	14	1110
8	15	1111

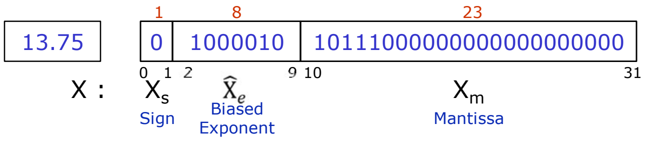
Floating Point Number Representation

- Example: $-6.3245\text{x}10^{-2}$

  $13.75 = 1101.11\text{x}2^0$

  $= 1.10111\text{x}2^3$

  Normalized form
- IEEE Standard 754
- 32-bit single precision



  - Exponent is represented in Excess-k representation
  - bias, $k=2^{(8-1)}-1=2^7-1 = 127$
  - Excess-127
  - Example: True exponent=3,

    Biased exponent= 3+127 = 130

32-bit Single Precision

- Exponent field is 8-bit in length
- Exponent is represented in Excess-k format
- Biased exponent is in the range: $0 \leq \hat{X}_e \leq 255$
- The biased exponent value 0 and 255 is used to represent special values
- Actual biased exponent takes the values from 1 to 254
  - Hence, true exponent is in the range: $-126 \leq X_e \leq +127$

X_e	$\hat{X}_e$	X_m	*Remark*
-	0	0	The value exact 0 is represented

32-bit Single Precision

- Exponent field is 8-bit in length
- Exponent is represented in Excess-k format
- Biased exponent is in the range: $0 \leq \hat{X}_e \leq 255$
- The biased exponent value 0 and 255 is used to represent special values
- Actual biased exponent takes the values from 1 to 254
  - Hence, true exponent is in the range: $-126 \leq X_e \leq +127$

X_e	$\hat{X}_e$	X_m	*Remark*
-	0	0	The value exact 0 is represented
-	255	0	The value ∞ is represented

32-bit Single Precision

- Exponent field is 8-bit in length
- Exponent is represented in Excess-k format
- Biased exponent is in the range: $0 \leq \hat{X}_e \leq 255$
- The biased exponent value 0 and 255 is used to represent special values
- Actual biased exponent takes the values from 1 to 254
  - Hence, true exponent is in the range: $-126 \leq X_e \leq +127$

X_e	$\hat{X}_e$	X_m	*Remark*
-	0	0	The value exact 0 is represented
-	255	0	The value ∞ is represented
-	0	≠0	Denormalized value

32-bit Single Precision

- Exponent field is 8-bit in length
- Exponent is represented in Excess-k format
- Biased exponent is in the range: $0 \leq \hat{X}_e \leq 255$
- The biased exponent value 0 and 255 is used to represent special values
- Actual biased exponent takes the values from 1 to 254
  - Hence, true exponent is in the range: $-126 \leq X_e \leq +127$

X_e	$\hat{X}_e$	X_m	*Remark*
-	0	0	The value exact 0 is represented
-	255	0	The value ∞ is represented
-	0	≠0	Denormalized value
-	255	≠0	Not a number (NaN)

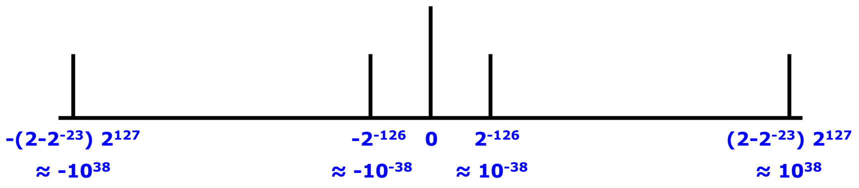
32-bit Single Precision

- Exponent field is 8-bit in length
- Exponent is represented in Excess-*k* format
- Biased exponent is in the range: $0 \leq \hat{X}_e \leq 255$
- The biased exponent value 0 and 255 is used to represent special values
- Actual biased exponent takes the values from 1 to 254
  - Hence, true exponent is in the range: $-126 \leq X_e \leq +127$

X_e	$\hat{X}_e$	X_m	*Remark*
-	0	0	The value exact 0 is represented
-	255	0	The value ∞ is represented
-	0	≠0	Denormalized value
-	255	≠0	Not a number (NaN)
-126 to 127	1 to 254	0 or ≠0	Normalized value

Range and Resolution in 32-bit Single Precision

- Range:



- In 32-bit fixed-point numbers, range is $\pm 4.55 \times 10^{-10}$ to $\pm 2.15 \times 10^{9}$
- Resolution:
  - Different exponent will have different resolution
  - $2^{-23+\text{true exponent}}$

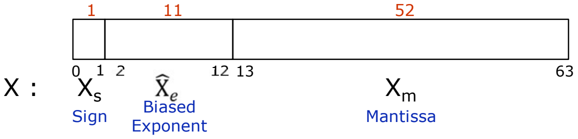
Resolution in 32-bit Single Precision

- Resolution:
  - Different exponent will have different resolution
  - $2^{-23+\text{true exponent}}$

Resolution in 32-bit Single Precision

- Resolution:
  - Different exponent will have different resolution
  - $2^{-23+\text{true exponent}}$

64-bit Double Precision

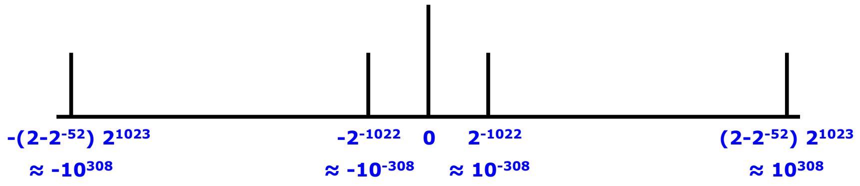
- 

- Exponent field is 11-bit in length
- Exponent is represented in Excess-1023 format
- Biased exponent is in the range: $0 \le \hat{X}_e \le 2047$
- The biased exponent value 0 and 2047 is used to represent special values
- Actual biased exponent takes the values from 1 to 2046
  - Hence, true exponent is in the range:

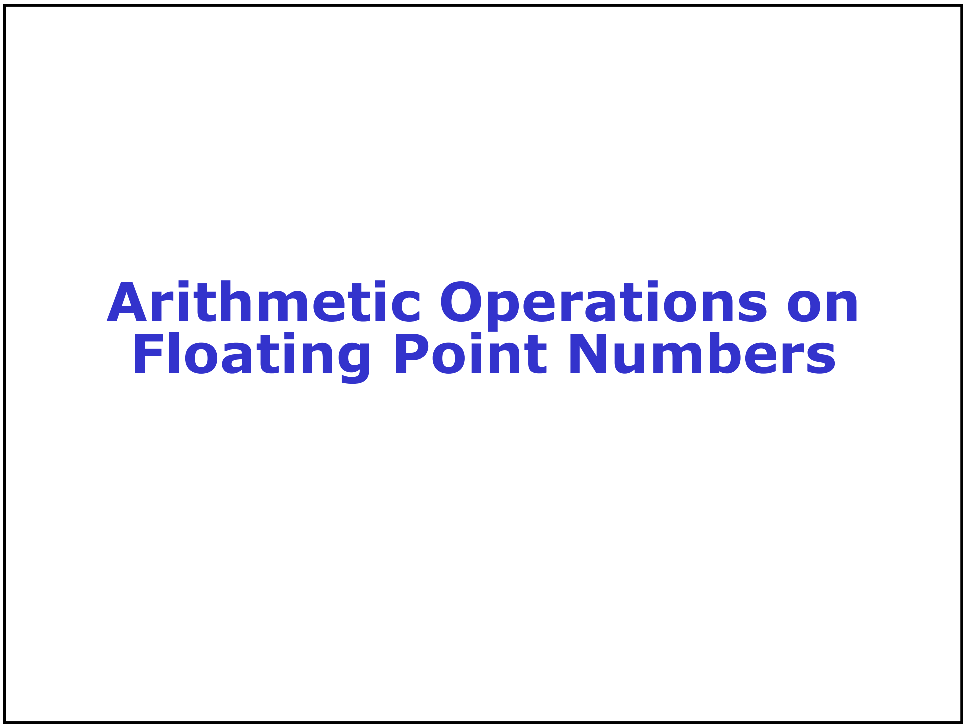
    $-1022 \le X_e \le +1023$
- Resolution: $2^{-52+\text{true exponent}}$

Range and Resolution in 64-bit Double Precision

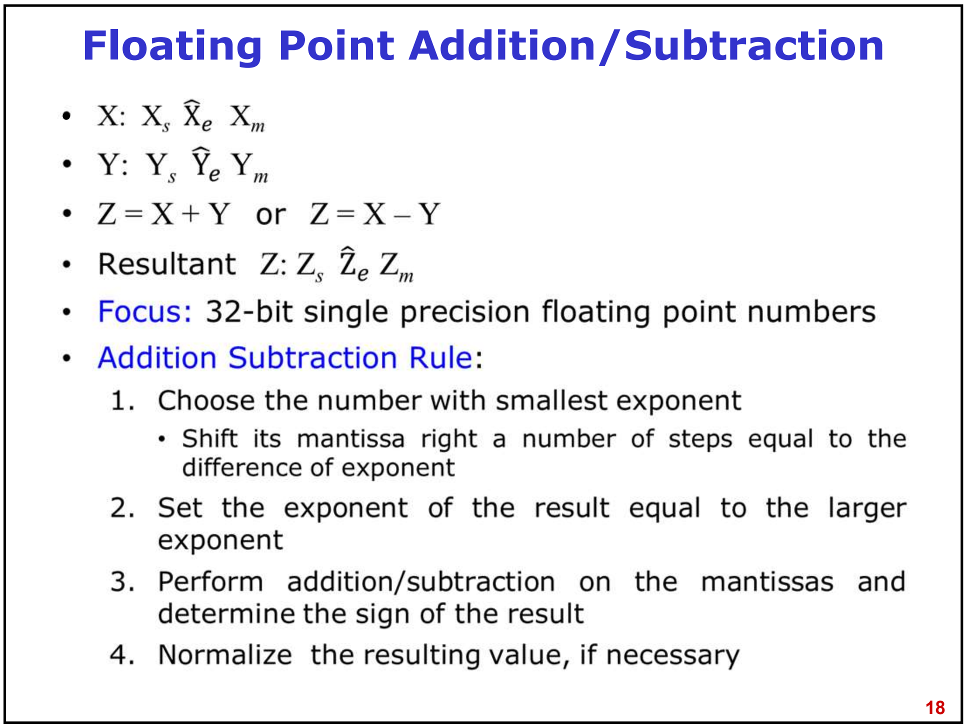
- Range:



- Resolution:
  - Different exponent will have different resolution
  - $2^{-52+\text{true exponent}}$



Arithmetic Operations on Floating Point Numbers



Floating Point Addition/Subtraction

- X: $X_s\ \hat{X}_e\ X_m$
- Y: $Y_s\ \hat{Y}_e\ Y_m$
- $Z = X + Y$ or $Z = X - Y$
- Resultant $Z{:}\ Z_s\ \hat{Z}_e\ Z_m$
- Focus: 32-bit single precision floating point numbers
- Addition Subtraction Rule:
  1. Choose the number with smallest exponent
     - Shift its mantissa right a number of steps equal to the difference of exponent
  2. Set the exponent of the result equal to the larger exponent
  3. Perform addition/subtraction on the mantissas and determine the sign of the result
  4. Normalize the resulting value, if necessary

Floating Point Addition/Subtraction: Example 1

- X: X_s $\hat{X}_e$ X_m X: $1.00000\ldots00x2^{0}$
- Y: Y_s $\hat{Y}_e$ Y_m Y: $1.11110\ldots00x2^{-5}$
- $Z = X + Y$
- Addition Subtraction Rule:
  1. Choose the number with smallest exponent and let that be Y

     Y: $1.11110\ldots00x2^{-5}$

     Shift its mantissa right a number of steps equal to the difference of exponents

     difference = |0+5| = 5

     Y: $0.0000111110\ldots00x2^{0}$
  2. Perform addition/subtraction on the mantissas and determine the sign of the result

     X: $1.0000000000\ldots00x2^{0}$

     Y: $0.0000111110\ldots00x2^{0}$

     Z: $1.0000111110\ldots00x2^{0}$
  3. Normalize the resulting value, if necessary

Floating Point Addition/Subtraction: Example 2

- X: X_s $\hat{X}_e$ X_m X: $-1.00000\ldots00x2^{0}$
- Y: Y_s $\hat{Y}_e$ Y_m Y: $1.11110\ldots00x2^{-5}$
- $Z = X + Y$
- Addition Subtraction Rule:
  1. Choose the number with smallest exponent and let that be Y

     Y: $1.11110\ldots00x2^{-5}$

     Shift its mantissa right a number of steps equal to the difference of exponents

     difference = |0+5| = 5

     Y: $0.0000111110\ldots00x2^{0}$
  2. Perform addition/subtraction on the mantissas and determine the sign of the result

     X: $-1.0000000000\ldots00x2^{0}$

     Y: $0.0000111110\ldots00x2^{0}$

     Z: $-0.1111000010\ldots00x2^{0}$
  3. Normalize the resulting value, if necessary

     Z: $-1.1110000100\ldots00x2^{-1}$

Floating Point Addition/Subtraction: Example 3

- X: X_s $\hat{X}_e$ X_m X: 1.00000...00x2^0
- Y: Y_s $\hat{Y}_e$ Y_m Y: 1.11110...00x2^5
- $Z = X - Y$
- Addition Subtraction Rule:
  1. Choose the number with smallest exponent and let that be Y

     Y: 1.00000...00x2^0

     Shift its mantissa right a number of steps equal to the difference of exponents

     difference = |0-5| = 5

     Y: 0.0000100000...00x2^5
  2. Perform addition/subtraction on the mantissas and determine the sign of the result

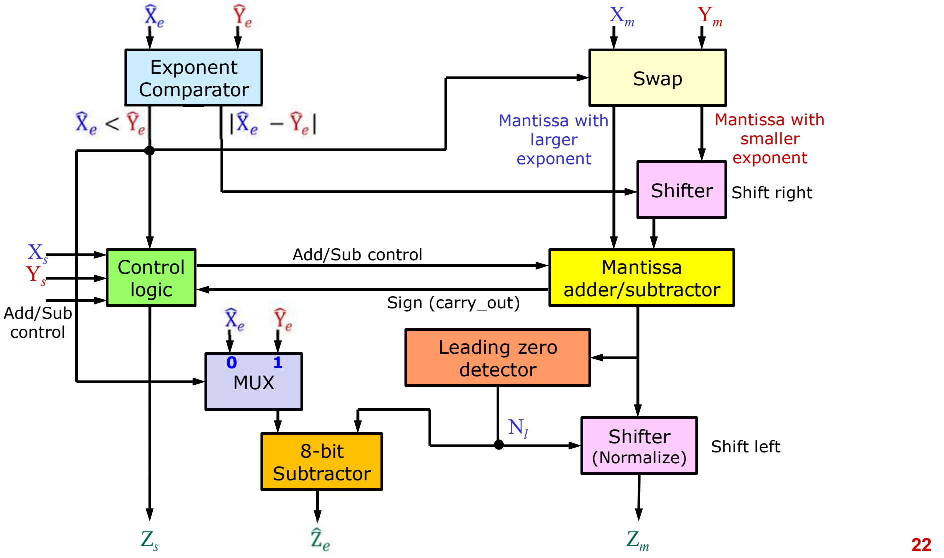
     X: 1.1111000000...00x2^5

     Y: 0.0000100000...00x2^5

     Z:-1.1110100000...00x2^5 Sign, Z_s=1 i.ve negative
  3. Normalize the resulting value, if necessary

Floating Point Addition/Subtraction Circuit

- X: X_s $\hat{X}_e$ X_m Y: Y_s $\hat{Y}_e$ Y_m $Z = X + Y$ or $Z = X - Y$
- Z: Z_s $\hat{Z}_e$ Z_m X, Y and Z are 32-bit operands



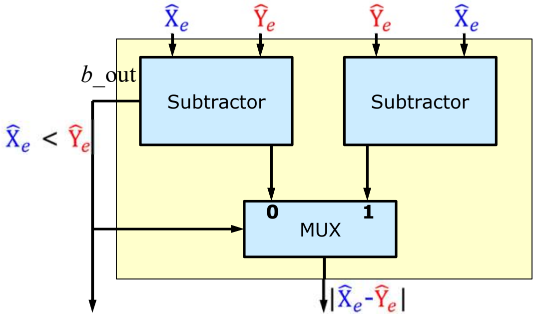
Exponent Comparator

- X: 1.00000…00x2^0

	0	01111111	00000000000000000000000
	0 1	2 9	10 31
X :	X_s	$\hat{X}_e$	X_m

- Y: 1.11110…00x2^{-5}

	0	01111010	11110000000000000000000
	0 1	2 9	10 31
Y :	Y_s	$\hat{Y}_e$	Y_m



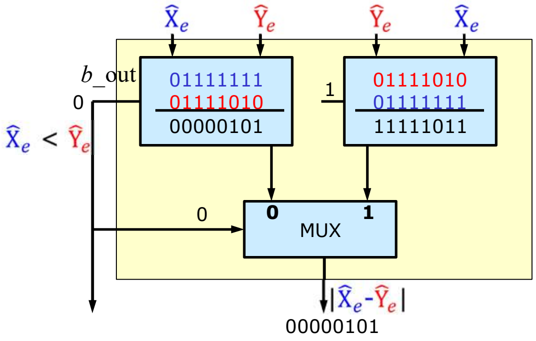
Exponent Comparator

- X: 1.00000…00x2^0

	0	01111111	00000000000000000000000
	0 1	2 9	10 31
X :	X_s	$\hat{X}_e$	X_m

- Y: 1.11110…00x2^{-5}

	0	01111010	11110000000000000000000
	0 1	2 9	10 31
Y :	Y_s	$\hat{Y}_e$	Y_m



Floating Point Multiplication and Division

- 32-bit single precision
- Multiply rule:
  - Add the exponent and subtract 127 (i.e. bias)
  - Multiply the mantissas and determine the sign of the result
  - Normalize the resulting value, if necessary
- Division rule:
  - Subtract the exponent and add 127 (i.e. bias)
  - Divide the mantissas and determine the sign of the result.
  - Normalize the resulting value, if necessary

Reference

- **Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", 5th Edition, Tata McGraw Hill, 2002**

Thank You