RAMA UNIVERSITY

www.ramauniversity.ac.in

FACULTY OF ENGINEERING & TECHNOLOGY

CSPS-106 Computer Organization

Lecture-04

Mr. Dilip Kumar J Saini
Assistant Professor
Computer Science & Engineering

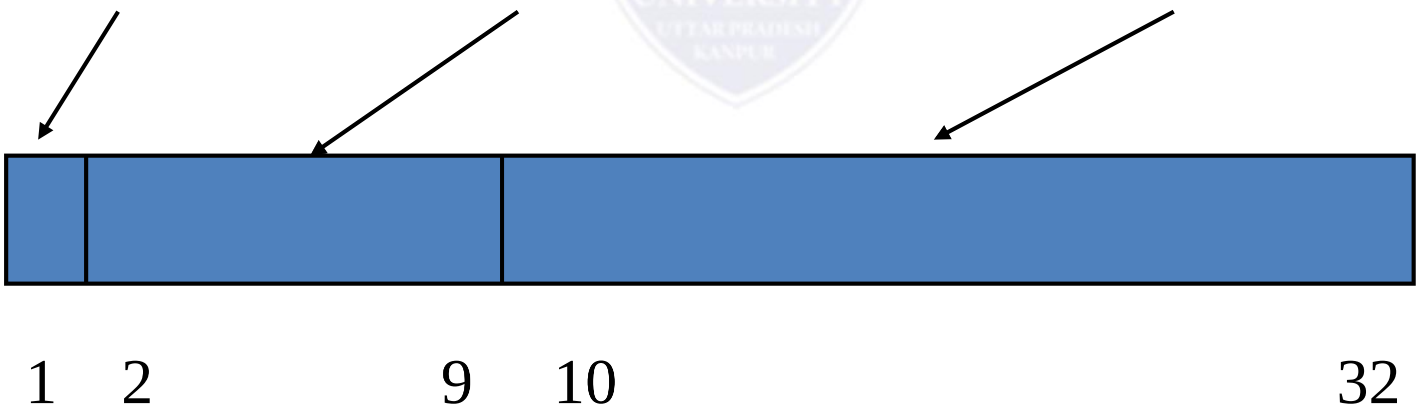
OUTLINE

- IEEE FLOATING POINT REPRESENTATION
- STORING THE BINARY FORM
- SOLUTION IS NORMALIZATION
- DECIMAL FLOATING POINT TO IEEE STANDARD CONVERSION.

IEEE FLOATING POINT REPRESENTATION

- Floating point numbers can be stored into 32-bits, by dividing the bits into three parts:

the **sign**, the **exponent**, and the **mantissa.**



IEEE FLOATING POINT REPRESENTATION

- The first (leftmost) field of our floating point representation will STILL be the sign bit:
  - 0 for a positive number,
  - 1 for a negative number.

STORING THE BINARY FORM

How do we store a radix point?

- All we have are zeros and ones…

Make sure that the radix point is ALWAYS in the same position within the number.

Use the IEEE 32-bit standard

→ the **leftmost** digit must be a 1

SOLUTION IS NORMALIZATION

Every binary number, **except the one corresponding to the number zero**, can be normalized by choosing the exponent so that the radix point falls to the right of the leftmost 1 bit.

$$37.25_{10} = 100101.01_2 = 1.0010101 \times \mathbf{2}^{5}$$

$$7.625_{10} = 111.101_2 = 1.11101 \times \mathbf{2}^{2}$$

$$0.3125_{10} = 0.0101_2 = 1.01 \times \mathbf{2}^{-2}$$

IEEE Floating Point Representation

- The second field of the floating point number will be the **exponent**.
- The exponent is stored as an unsigned 8-bit number, RELATIVE to a **bias of 127.**
  - Exponent 5 is stored as (127 + 5) or 132
    - 132 = 10000100
  - Exponent -5 is stored as (127 + (-5)) or 122
    - 122 = 01111010

IEEE FLOATING POINT REPRESENTATION

- The **mantissa** is the set of 0's and 1's to the right of the radix point of the **normalized** (when the digit to the left of the radix point is 1) binary number.

  Ex: 1.**00101** X 2^3

  (The mantissa is 00101)

- The mantissa is stored in a 23 bit field, so we add zeros to the right side and store:

  00101000000000000000000

IEEE STANDARD CONVERSION

Ex 1: Find the IEEE FP representation of

40.15625

Step 1.

Compute the binary equivalent of the whole part and the fractional part.
(i.e. convert 40 and .15625 to their binary equivalents)

40		.15625	
- 32	**Result:**	-.12500	**Result:**
8	101000	.03125	.00101
- 8		-.03125	
0		.0	

So: $40.15625_{10} = 101000.00101_{2}$

Step 2. Normalize the number by moving the decimal point to the right of the leftmost one.

$$101000.00101 = 1.0100000101 \times 2^5$$

Step 3. Convert the exponent to a biased exponent

$$127 + 5 = 132$$

And convert biased exponent to 8-bit unsigned binary:

$$132_{10} = 10000100_2$$

Step 4. Store the results from steps 1-3:

Sign	Exponent (from step 3)	Mantissa (from step 2)
0	**10000100**	**01000001010000000000000**

Ex 2: Find the IEEE FP representation of **–24.75**

Step 1. Compute the binary equivalent of the whole part and the fractional part.

24		.75	
- 16	**Result:**	- .50	**Result:**
8	11000	.25	.11
- 8		- .25	
0		.0	

So: $-24.75_{10} = -11000.11_{2}$

Step 2.

Normalize the number by moving the decimal point to the right of the leftmost one.

$$-11000.11 = -1.100011 \times 2^4$$

DECIMAL FLOATING POINT TO IEEE STANDARD CONVERSION.

Step 3. Convert the exponent to a biased exponent

$127 + 4 = 131$

$\Longrightarrow \quad 131_{10} = 10000011_{2}$

Step 4. Store the results from steps 1-3

Sign	Exponent	mantissa
1	**10000011**	**1000110..0**

Floating Point Numbers	Arithmetic Operations
$X = X_S \times B^{X_E}$ $Y = Y_S \times B^{Y_E}$	$\left.\begin{array}{l} X + Y = (X_S \times B^{X_E - Y_E} + Y_S) \times B^{Y_E} \\ X - Y = (X_S \times B^{X_E - Y_E} - Y_S) \times B^{Y_E} \end{array}\right\} X_E \leq Y_E$ $X \times Y = (X_S \times Y_S) \times B^{X_E + Y_E}$ $\frac{X}{Y} = \left(\frac{X_S}{Y_S}\right) \times B^{X_E - Y_E}$

Examples:

$$X = 0.3 \times 10^2 = 30$$
$$Y = 0.2 \times 10^3 = 200$$

$$X + Y = (0.3 \times 10^{2-3} + 0.2) \times 10^3 = 0.23 \times 10^3 = 230$$
$$X - Y = (0.3 \times 10^{2-3} - 0.2) \times 10^3 = (-0.17) \times 10^3 = -170$$
$$X \times Y = (0.3 \times 0.2) \times 10^{2+3} = 0.06 \times 10^5 = 6000$$
$$X \div Y = (0.3 \div 0.2) \times 10^{2-3} = 1.5 \times 10^{-1} = 0.15$$

SIGNIFICANT ALIGNMENT

To see the need for aligning exponents, consider the following decimal addition:

$$(123 \times 10^{0}) + (456 \times 10^{-2})$$

Clearly, we cannot just add the significands. The digits must first be set into equivalent positions, that is, the 4 of the second number must be aligned with the 3 of the first. Under these conditions, the two exponents will be equal, which is the mathematical condition under which two numbers in this form can be added. Thus,

$$(123 \times 10^{0}) + (456 \times 10^{-2}) = (123 \times 10^{0}) + (4.56 \times 10^{0}) = 127.56 \times 10^{0}$$

Multiple Choice Question

MUTIPLE CHOICE QUESTIONS:

Sr no	Question	Option A	Option B	OptionC	OptionD
1	The registers, ALU and the interconnection between them are collectively called as _____	process route	Information Tail	information path	Data Path
2	A processor performing fetch or decoding of different instruction during the execution of another instruction is called ______	Pipe-lining	Super-scaling	Parallel Computation	None of the mentioned
3	For a given FINITE number of instructions to be executed, which architecture of the processor provides for a faster execution?	ISA	Super-scalar	ANSA	All of the mentioned
4	The clock rate of the processor can be improved by _________	Improving the IC technology of the logic circuits	By using the overclocking method	Reducing the amount of processing done in one step	All of the mentioned
5	An optimizing Compiler does _________	Better compilation of the given piece of code	Takes advantage of the type of processor and reduces its process time	Does better memory management	None of the mentioned

REFERENCES

- http://www.engppt.com/search/label/Computer%20Organization%20and%20Architecture
- http://www.engppt.com/search/label/Computer%20Architecture%20ppt