CT 211: COMPUTER ARCHITECTURE AND ORGANIZATION I

TUTORIAL#01

Data Representation in Computer Systems

Binary and Number Systems

1. Convert the decimal number 25 to its binary, octal, and hexadecimal equivalents.
2. Convert the binary number 101101 into decimal and hexadecimal forms.
3. What is the range of values that can be represented with an 8-bit unsigned integer?
4. Represent the number -15 in an 8-bit signed magnitude form.
5. Explain the difference between signed and unsigned numbers in computer systems.

Two's Complement

1. Convert the decimal number -20 to its binary form using two's complement in an 8-bit system.
2. Perform the binary addition of 01100110 and 10011010. Explain the result.
3. Explain how overflow is detected in a two's complement operation.
4. Demonstrate how subtraction can be performed using two's complement addition.
5. Represent -56 and 42 in two's complement form and compute their sum.

Floating-Point Representation

1. Describe the IEEE 754 single-precision floating-point format.
2. Convert the decimal number 18.25 into IEEE 754 single-precision format.
3. What are the components of a floating-point number in computer systems?
4. Explain the concept of "normalization" in floating-point representation.

5. Discuss the difference between single-precision and double-precision floating-point representation.

Character Encoding

1. Convert the string "Data" into ASCII and hexadecimal equivalents.
2. What is the significance of Unicode in modern computer systems?
3. Explain the differences between ASCII, UTF-8, and UTF-16 character encoding schemes.
4. How many characters can be represented using ASCII and UTF-8 encoding?
5. Decode the binary string 01000001 01000010 into its ASCII representation.

Sign Magnitude Representation

1. Represent the decimal number -25 using an 8-bit signed magnitude representation. How does it differ from two's complement representation?
2. Perform binary addition of 01001001 (+73 in sign magnitude) and 11001001 (-73 in sign magnitude). Discuss why the result might not be correct without additional processing.
3. Explain the advantages and disadvantages of using sign magnitude representation in computer systems.
4. Convert the decimal numbers +42 and -42 into their 8-bit signed magnitude representations and compare them.
5. What is the maximum and minimum value that can be represented using an 8-bit signed magnitude system?

Binary Coded Decimal (BCD)

1. Convert the decimal number 47 into its Binary Coded Decimal (BCD) equivalent.
2. Add the BCD numbers for 28 (0010 1000) and 35 (0011 0101). Ensure the result is also in BCD format.
3. Explain the difference between Binary Coded Decimal (BCD) and pure binary representation of numbers.
4. How is a decimal number with a fractional part (e.g., 25.6) represented in BCD format?
5. Discuss the advantages of using BCD representation over binary representation in financial or business applications.

Binary Coded Decimal (BCD) Addition:

1. Perform the BCD addition of the numbers 29 (0010 1001) and 34 (0011 0100). Show all intermediate steps and explain how corrections are applied if necessary.
2. Add the BCD numbers for 58 (0101 1000) and 47 (0100 0111). Explain why the binary addition of these BCD numbers may require an adjustment to maintain valid BCD results.
3. Explain the rules for correcting a BCD sum when a carry occurs or when the result exceeds the valid BCD range (0 to 9 in each nibble). Provide an example with two BCD numbers.