taleemzone.com

CLASS 9TH • COMPUTER SCIENCE

1

Fundamentals of Computer Systems

Comprehensive Study Material

SECTION 1: SYSTEM CONCEPTS

1. Objective of a System

Every system has a purpose or goal that it wishes to fulfil. Analyzing a system's operation requires understanding its aim. This insight improves the efficiency and efficacy of the present system.

EXAMPLE

- A transport system aims to transfer people and products securely and effectively between locations.
- A computer system's principal goal is to process data and provide useful information to users.

2. Components of a System

Components are the building blocks of any system. Each component plays a specific role and contributes to the overall functionality of the system. Understanding the role of each component is essential to understand how the entire system works. This helps in identifying problems, improving performance, and refining system design. The smooth and proper working of these components together ensures the system meets its objectives.

3. Environment of a System

The environment of a system includes everything external to the system that interacts with it. It consists of all external factors that affect the system's operation. Understanding the environment of a system is important as it influences the system's performance and behavior by providing inputs and receiving outputs. Intelligent systems adapt to changes in their environment to continue their functionality.

4. Communication in a System

Communication and interaction among system components is key to the functioning of a system. It ensures that components work together in an organized manner to achieve the system's objectives.

EXAMPLE

- In a computing system, the CPU communicates with memory to fetch and store data.
- In a biological system, the brain sends signals to muscles to initiate movement.

SECTION 2: SOFTWARE

What is Software?

Software is a collection of programs and instructions that tell a computer what to do and how to do it. Without software, computers would be useless machines.

First Computer Virus:
The first computer virus, called "Creeper," was created in 1971 as an experimental self-replicating program. It simply displayed the message, "I'm the creeper, catch me if you can!"

Types of Software

Software is classified into two main categories:

1. System Software

System software is designed to manage the system resources and provide a platform for application software to run. It acts as a bridge between the hardware and the user applications.

EXAMPLE

- **Operating Systems:** Manage computer hardware and resources.
  - Examples: Microsoft Windows, macOS, and Linux
- **Device Drivers:** Allow the computer to communicate with hardware devices.
  - Examples: Printer drivers, graphics card drivers, and sound card drivers
- **Utility Programs:** Designed to maintain and protect the computer system.
  - Examples: Antivirus software, disk cleanup tools, and backup software

2. Application Software

Application software is designed to help users perform specific tasks. These programs are built to fulfill user needs and are typically more varied than system software.

EXAMPLE

- **Word Processors:** Used to create and edit text documents.
  - Examples: Microsoft Word and Google Docs
- **Web Browsers:** Used to access and browse the internet.
  - Examples: Google Chrome, Mozilla Firefox, and Safari
- **Games:** Designed for entertainment purposes.
  - Examples: Minecraft, Fortnite, and Among Us
- **Media Players:** Used to play audio and video files.
  - Examples: VLC Media Player and Windows Media Player

Difference Between System Software and Application Software

Aspect	System Software	Application Software
Purpose	**Manages and operates computer hardware. Provides a platform for other software to run.**	**Helps users perform specific tasks such as writing documents or browsing the internet.**
Installation	Usually pre-installed on the computer by the manufacturer.	Installed by the user according to their needs.
Function	Acts as an interface between hardware and application software.	Directly interacts with the user to complete tasks.
Dependency	Essential for the computer to function properly.	Depends on system software to run.
Examples	Operating systems (Windows, Linux), device drivers, and utility programs.	Word processors, web browsers, games, media players, etc.

SECTION 3: VON NEUMANN ARCHITECTURE

Definition

Von Neumann architecture is a computer paradigm that describes a system in which the hardware of the computer has four primary components:

1. Memory
2. Central Processing Unit (CPU)
3. Input mechanisms
4. Output mechanisms

This model is named after mathematician and physicist John von Neumann, who contributed to its development during the 1940s.

Primary Components

1. Memory

Contains both input data and the instructions (program) required for CPU processing.

Example: Consider the RAM of your computer: when a program starts, it is loaded into RAM to enable faster execution compared to when it runs from the hard disk.

2. Central Processing Unit (CPU)

Performs addition and subtraction, and executes commands provided by the memory. The CPU has two main components:

a. Arithmetic Logic Unit (ALU)

Performs mathematical computations and logical operations.

b. Control Unit (CU)

Governs the activities of the CPU by instructing the ALU and memory to execute tasks according to the program instructions. It ensures the proper and timely execution of duties by all the other components.

Example: When doing the calculation 2 + 2 on a calculator application, the Arithmetic Logic Unit (ALU) handles the numerical values while the Control Unit (CU) supervises the whole procedure.

3. Input Devices

Enable users to input data and instructions into the computer system.

Examples: Keyboard, mouse, and microphone.

Example: Entering text on the keyboard transmits data to the CPU for subsequent processing.

4. Output Devices

Present or communicate the outcomes of the tasks executed by the computer.

Examples: Monitor and printer.

Example: Upon completion of data processing, the CPU transmits the outcome to the monitor for visual display.

Working of Von Neumann Architecture

The Von Neumann architecture encompasses four essential stages for a CPU to carry out instructions:

1. Fetching

The central processing unit retrieves an instruction from the computer's memory.

Components Involved: Memory, CPU, Program Counter (PC), Instruction Register (IR)

Description: The Program Counter (PC) stores the memory address of the subsequent instruction. Once the address is stored in memory, the instruction located at that location is retrieved and placed into the Instruction Register (IR).

2. Decoding

The Control Unit (CU) decodes the instruction to determine the necessary action.

Components Involved: Control Unit (CU)

Description: The Control Unit (CU) decodes the opcode (operation code) of the instruction and determines the required procedures and data.

3. Execution

The CPU processes the instruction.

Components Involved: ALU and CU

Description: When the instruction involves a computation, it is executed by the Arithmetic Logic Unit (ALU). Any task that requires transferring data between several locations is managed by the CU.

4. Storing

The result of the execution is either returned to memory or sent to an output device.

Description: This allows the CPU to save or display the outcome of the processed instruction.

SECTION 4: SYSTEM BUS

Definition

A system bus is a communication mechanism that facilitates the movement of data between components inside a computational system.

Types of System Bus

1. Data Bus

- A set of wires or lines used to transfer actual data between the CPU, memory, and input/output devices.
- It is a **two-way communication bus**, meaning data can flow in both directions.
- Used to transport data.

2. Address Bus

- Used to carry memory addresses from the CPU to memory or input/output devices.
- It is a **one-way bus**, meaning information flows only from the CPU to other components.
- Maintains data destination information.

3. Control Bus

- Carries control signals from the CPU to other components of the computer.
- Helps manage and regulate data flow between components.
- Transports control electrical signals.

SECTION 5: NUMBER SYSTEMS

Why are Number Systems Essential?

Number systems form the basis for representing, storing, and processing data in computers. They allow computers to perform calculations, store information, transfer data, represent text, colors, and memory addresses.

Decimal Number System

- Base-10 system
- Uses digits from 0 to 9
- Each digit represents a power of 10

EXAMPLE

text

$523_{10} = 5 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$

Binary Number System

- Base-2 system
- Uses digits 0 and 1
- Each position represents a power of 2
- Computers use it because electronic devices have two states: ON (1) and OFF (0)

EXAMPLE

text

$1011_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 0 + 2 + 1 = 11_{10}$

Octal Number System

- Base-8 system
- Uses digits from 0 to 7
- Each digit represents a power of 8

EXAMPLE

text

$157_8 = 1 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 = 64 + 40 + 7 = 111_{10}$

Hexadecimal Number System

- Base-16 system
- Uses digits 0-9 and letters A-F
- Each digit represents a power of 16

EXAMPLE

text

$1A3_{16} = 1 \times 16^2 + 10 \times 16^1 + 3 \times 16^0 = 256 + 160 + 3 = 419_{10}$

Conversions

Decimal to Binary Conversion

Method: Divide the decimal number by 2 repeatedly and record the remainders. Read remainders from bottom to top.

Example: Convert 83_{10} to binary

Division	Quotient	Remainder
83 ÷ 2	**41**	**1**
41 ÷ 2	20	1
20 ÷ 2	10	0
10 ÷ 2	5	0
5 ÷ 2	2	1
2 ÷ 2	1	0
1 ÷ 2	0	1

Reading remainders from bottom to top: **1010011_2**

Decimal to Octal Conversion

Method: Divide the decimal number by 8 repeatedly and record the remainders. Read remainders from bottom to top.

Example: Convert 83_{10} to octal

Division	Quotient	Remainder
83 ÷ 8	**10**	**3**
10 ÷ 8	1	2
1 ÷ 8	0	1

Reading remainders from bottom to top: **123_8**

Decimal to Hexadecimal Conversion

Method: Divide the decimal number by 16 repeatedly and record the remainders. Read remainders from bottom to top.

Example: Convert 2297_{10} to hexadecimal

Division	Quotient	Remainder
2297 ÷ 16	**143**	**9**
143 ÷ 16	8	F
8 ÷ 16	0	8

Reading remainders from bottom to top: **$8F9_{16}$**

Relationship Between Number Systems

Octal to Binary

Each octal digit represents three binary digits (bits) because $8 = 2^3$.

Correspondence Table:

Octal	Binary
0	**000**
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Example: Convert 653_8 to binary

text

6 → 110

5 → 101

3 → 011

Therefore, 653_8 = **110101011_2**

Hexadecimal to Binary

Each hexadecimal digit represents four binary digits (bits) because $16 = 2^4$.

Correspondence Table:

Hex	Binary	Hex	Binary
0	**0000**	**8**	**1000**
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Example: Convert $D6B2_{16}$ to binary

text

D → 1101

6 → 0110

B → 1011

2 → 0010

Therefore, $D6B2_{16}$ = **1101011010110010_2**

SECTION 6: DATA REPRESENTATION IN COMPUTING SYSTEMS

Whole Numbers (W)

Whole numbers are a set of non-negative integers. They include zero and all positive integers.

Mathematically:

text

W = {0, 1, 2, 3, ...}

Storage Capacity:

- 1-byte whole number (8 bits): Maximum value = $2^8 - 1$ = **255**
- 2-byte whole number (16 bits): Maximum value = $2^{16} - 1$ = **65,535**
- 4-byte whole number (32 bits): Maximum value = $2^{32} - 1$ = **4,294,967,295**

Integers (Z)

Integers extend the concept of whole numbers to include negative numbers. In programming, these are called signed integers.

Mathematically:

text

Z = {..., -3, -2, -1, 0, 1, 2, 3, ...}

Storage:

- One bit is reserved as the sign bit (MSB - Most Significant Bit)
- If the sign bit is ON (1), the value is negative
- If the sign bit is OFF (0), the value is positive

Maximum Positive Values

For an n-bit signed integer:

text

Maximum positive value = 2^(n-1) - 1

- 1-byte signed integer: Maximum = $2^7 - 1$ = **127**
- 2-byte signed integer: Maximum = $2^{15} - 1$ = **32,767**
- 4-byte signed integer: Maximum = $2^{31} - 1$ = **2,147,483,647**

Minimum Negative Values

For an n-bit signed integer:

text

Minimum value = -2^(n-1)

- 1-byte signed integer: Minimum = -2^7 = **-128**
- 2-byte signed integer: Minimum = -2^{15} = **-32,768**
- 4-byte signed integer: Minimum = -2^{31} = **-2,147,483,648**

1's Complement

1's Complement is a method to represent negative numbers by inverting all bits of a positive number (changing 0s to 1s and 1s to 0s).

Example: 1's complement of 1010_2 is 0101_2

2's Complement

2's Complement is the most common method used by computers to represent negative numbers.

Steps to find 2's complement:

1. Invert all the bits (1's complement)
2. Add 1 to the Least Significant Bit (LSB)

Example: Convert -5 to 8-bit 2's complement

1. Binary representation of 5: 00000101
2. Invert all bits: 11111010
3. Add 1: 11111010 + 1 = 11111011

Therefore, -5 in 8-bit 2's complement = **11111011_2**

SECTION 7: BINARY ARITHMETIC OPERATIONS

Binary Addition Rules

Rule	Result
0 + 0	**0**
0 + 1	1
1 + 0	1
1 + 1	0 (with carry 1 to next higher bit)

EXAMPLE

text

1011_2

$+ 0110_2$

10001_2

Binary Subtraction

Binary subtraction is carried out by adding the 2's complement of the subtrahend to the minuend.

Formula: A - B = A + (2's complement of B)

Binary Multiplication

Binary multiplication is similar to decimal multiplication:

1. Multiply each bit of one number by each bit of the other
2. Shift partial results one place left for each new row
3. Add the partial results

Example: $101_2 \times 11_2$

text

101

× 11

-----

101 (101 × 1)

101 (101 × 1, shifted left)

1111_2

Binary Division

Binary division follows the same steps as decimal long division:

1. Compare
2. Subtract if possible
3. Bring down the next digit
4. Repeat until all digits are used

Example: $1100_2 \div 10_2 = 110_2$

SECTION 8: COMMON TEXT ENCODING SCHEMES

ASCII (American Standard Code for Information Interchange)

- Standard ASCII uses 7 bits to represent **128 characters** (0-127)
- Extended ASCII uses 8 bits to represent **256 characters** (0-255)

Unicode Encoding Schemes

UTF-8

- Variable-length encoding scheme (1 to 4 bytes)
- Backward compatible with ASCII
- Used for modern text representation

EXAMPLE

- Letter 'A' (U+0041) = 01000001 (1 byte)
- Urdu letter 'ب' (U+0628) = 11011000 10101000 (2 bytes)

UTF-16

- Variable-length encoding scheme (2 or 4 bytes)
- Not compatible with ASCII
- Used for Unicode text

EXAMPLE

- Letter 'A' in UTF-16 = 00000000 01000001 (2 bytes)
- Urdu letter 'ب' in UTF-16 = 00000110 00101000 (2 bytes)

UTF-32

- Fixed-length encoding scheme (4 bytes per character)
- Simple but uses more space

Example: Letter 'A' in UTF-32 = 00000000 00000000 00000000 01000001 (4 bytes)

SECTION 9: FILE STORAGE

How Files Are Stored

All files such as images, audio, and video are stored in a computer as binary data.

- **Images:** Stored as pixels using formats like JPEG, PNG, or GIF
- **Audio:** Stored as sound waves encoded in formats like MP3 or WAV
- **Video:** Stored as sequences of frames using formats like MP4 or AVI

SECTION 10: MULTIPLE CHOICE QUESTIONS (MCQs)

System Concepts

1. What are the building blocks of a system called?
    - (a) Rules
    - (b) Signals
    - **(c) Components**
    - (d) Inputs

Software

1. What is software?
    - (a) Computer parts
    - **(b) Set of programs**
    - (c) Electric signal
    - (d) Machine tool
2. Software tells the computer:
    - (a) How to build
    - (b) Where to store
    - **(c) What to do**
    - (d) When to stop
3. Without software, a computer would be:
    - (a) Faster

  - (b) Advanced
  - **(c) Useless**
  - (d) Portable

4. Which of the following is a type of software?
  - (a) Hardware
  - (b) Device
  - **(c) System software**
  - (d) Monitor

5. Which of the following is another type of software?
  - (a) Motherboard
  - (b) Printer
  - **(c) Application software**
  - (d) Keyboard

6. Which of the following is an example of an operating system?
  - (a) Google Docs
  - (b) Minecraft
  - **(c) Microsoft Windows**
  - (d) VLC Player

7. Which of the following is also an operating system?
  - (a) Safari
  - **(b) Linux**
  - (c) Fortnite
  - (d) Word

8. Which of the following is an example of a device driver?
  - **(a) Printer driver**
  - (b) Word processor
  - (c) Web browser
  - (d) Media player

9. A graphics card driver is a type of:
    - (a) Application
    - (b) Browser
    - **(c) Device driver**
    - (d) Game
10. Antivirus software is an example of:
    - (a) Browser
    - **(b) Utility program**
    - (c) Driver
    - (d) Game
11. Disk cleanup tools belong to:
    - (a) Games
    - (b) Drivers
    - **(c) Utility programs**
    - (d) Browsers
12. Which of the following is an example of application software?
    - (a) Device driver
    - **(b) Word processor**
    - (c) Operating system
    - (d) Utility tool
13. Mozilla Firefox is a:
    - (a) Media player
    - **(b) Web browser**
    - (c) Game
    - (d) Driver
14. Minecraft is an example of:
    - (a) Browser
    - **(b) Game**
    - (c) Driver
    - (d) Utility

15. Application software mainly helps users to:
    - (a) Control hardware
    - (b) Install drivers
    - **(c) Perform tasks**
    - (d) Build systems
16. Which software is more focused on user tasks?
    - (a) System software
    - **(b) Application software**
    - (c) Device driver
    - (d) Utility program

Von Neumann Architecture

1. How many primary components are in the Von Neumann computer?
    - (a) 2
    - (b) 3
    - **(c) 4**
    - (d) 5
2. Which of the following is NOT a primary component of Von Neumann architecture?
    - (a) Memory
    - (b) CPU
    - **(c) Network card**
    - (d) Input/Output devices
3. Which component performs addition, subtraction, and logical operations?
    - (a) Control Unit
    - (b) Memory
    - **(c) Arithmetic Logic Unit (ALU)**
    - (d) Input Device
4. Which of these is an example of an input device?
    - (a) Printer

- **(b) Keyboard**
- (c) Monitor
- (d) Speaker

5. Which of these is an example of an output device?
   - (a) Mouse
   - (b) Microphone
   - **(c) Monitor**
   - (d) Keyboard
6. The system bus includes which of the following?
   - **(a) Data bus, Address bus, Control bus**
   - (b) Network bus
   - (c) Power bus
   - (d) ALU bus
7. The function of the data bus is to:
   - (a) Store instructions
   - **(b) Transport data**
   - (c) Decode commands
   - (d) Display output
8. The function of the address bus is to:
   - (a) Execute programs
   - (b) Transfer output
   - **(c) Maintain data destination info**
   - (d) Store memory
9. The function of the control bus is to:
   - (a) Display results
   - (b) Move data
   - **(c) Transport control signals**
   - (d) Store programs
10. The first stage of CPU instruction processing is:
    - (a) Execution

- **(b) Fetching**
- (c) Decoding
- (d) Storing

11. During the fetching stage, the instruction is retrieved from:
    - (a) ALU
    - (b) Input device
    - **(c) Memory**
    - (d) Output device
12. The Program Counter (PC) stores:
    - (a) Input data
    - (b) Output data
    - **(c) Address of next instruction**
    - (d) Control signals
13. During decoding, the instruction is interpreted by:
    - (a) ALU
    - (b) Memory
    - **(c) Control Unit (CU)**
    - (d) Input device
14. Which CPU component executes mathematical and logical operations?
    - (a) Program Counter
    - (b) Control Unit
    - **(c) Arithmetic Logic Unit (ALU)**
    - (d) Instruction Register
15. During the storing stage, the result of execution is sent to:
    - (a) Keyboard
    - (b) ALU
    - **(c) Memory or Output device**
    - (d) Control Unit
16. Which stage involves transferring data between memory locations?
    - (a) Fetching

- **(b) Execution**
- (c) Decoding
- (d) Storing

17. Which stage ensures the instruction is placed in the Instruction Register?
    - (a) Decoding
    - (b) Execution
    - **(c) Fetching**
    - (d) Storing
18. Von Neumann architecture developed in which decade?
    - (a) 1920s
    - (b) 1930s
    - **(c) 1940s**
    - (d) 1950s

Number Systems

1. Computers use number systems because they:
    - (a) Write letters
    - **(b) Represent and process data**
    - (c) Store hardware
    - (d) Control power
2. Decimal system is based on:
    - (a) 2
    - (b) 8
    - **(c) 10**
    - (d) 16
3. Decimal digits range from:
    - (a) 0-7
    - **(b) 0-9**
    - (c) 1-10
    - (d) 1-16

4. Binary system is based on:
   - (a) 10
   - (b) 8
   - **(c) 2**
   - (d) 16
5. Binary digit 1 represents:
   - (a) OFF
   - **(b) ON**
   - (c) 2
   - (d) 0
6. Binary digit 0 represents:
   - **(a) OFF**
   - (b) ON
   - (c) 1
   - (d) 2
7. Computers work in binary because:
   - (a) Decimal is easier
   - **(b) Electronics have two states (0,1)**
   - (c) Octal is faster
   - (d) Hexadecimal is simpler
8. Octal system is base:
   - (a) 2
   - (b) 10
   - **(c) 8**
   - (d) 16
9. Octal digits range from:
   - (a) 0-9
   - (b) 1-8
   - **(c) 0-7**
   - (d) 0-15

10. Hexadecimal system is base:
    - (a) 2
    - (b) 8
    - (c) 10
    - **(d) 16**
11. Hexadecimal digits include:
    - (a) 0-7 only
    - (b) 0-9 only
    - **(c) 0-9 and A-F**
    - (d) 1-16
12. How many binary bits does one octal digit represent?
    - (a) 1
    - (b) 2
    - **(c) 3**
    - (d) 4
13. How many binary bits does one hexadecimal digit represent?
    - (a) 2
    - (b) 3
    - **(c) 4**
    - (d) 5
14. When converting decimal to binary, the binary number is read:
    - (a) Left to right
    - **(b) Bottom to top**
    - (c) Top to bottom
    - (d) Right to left
15. When converting numbers to binary or hex, extra zeros are added to:
    - (a) Right side
    - (b) Middle
    - **(c) Left side**
    - (d) Not needed

Data Representation

1. Whole numbers include:
    - (a) Negative integers
    - (b) Fractions
    - **(c) 0 and positive integers**
    - (d) Decimals
2. Whole numbers in computing are used to represent:
    - (a) Negative values
    - **(b) Quantities that can't be negative**
    - (c) Fractions
    - (d) Characters
3. A 1-byte whole number has how many bits?
    - (a) 4
    - (b) 16
    - **(c) 8**
    - (d) 32
4. Maximum value of a 1-byte whole number is:
    - (a) 127
    - (b) 128
    - **(c) 255**
    - (d) 256
5. Maximum value of a 2-byte whole number is:
    - (a) 65,555
    - (b) 32,768
    - **(c) 65,535**
    - (d) 64,000
6. Maximum value of a 4-byte whole number is:
    - (a) 2,147,483,647
    - (b) 4,000,000,000

- **(c) 4,294,967,295**
- (d) 2,147,483,648

7. Integers differ from whole numbers because they:
   - (a) Include fractions
   - **(b) Can be negative**
   - (c) Include decimals
   - (d) Exclude zero
8. In signed integers, the sign bit indicates:
   - (a) Size
   - **(b) Positive or negative**
   - (c) Decimal value
   - (d) Binary code
9. Maximum positive value in 1-byte signed integer is:
   - (a) 128
   - **(b) 127**
   - (c) 255
   - (d) 256
10. Minimum value in 1-byte signed integer is:
    - (a) -127
    - **(b) -128**
    - (c) -255
    - (d) -256
11. Minimum value in 2-byte signed integer is:
    - (a) -32,767
    - (b) 32,768
    - **(c) -32,768**
    - (d) -65,535
12. Minimum value in 4-byte signed integer is:
    - (a) -2,147,483,647

- **(b) -2,147,483,648**
- (c) 2,147,483,647
- (d) -4,294,967,295

13. 1's complement represents negative numbers by:
    - (a) Adding 1
    - **(b) Inverting all bits**
    - (c) Shifting left
    - (d) Setting last bit to 0
14. To find 2's complement of a number, you:
    - (a) Add 1 only
    - (b) Invert bits only
    - **(c) Invert bits and add 1**
    - (d) Subtract 1
15. The Most Significant Bit (MSB) is used as:
    - (a) LSB
    - **(b) Sign bit**
    - (c) Parity
    - (d) Memory

Binary Arithmetic

1. Binary arithmetic uses digits:
    - (a) 0-9
    - (b) 0-7
    - **(c) 0 and 1**
    - (d) 1-2
2. In binary addition, 1 + 1 equals:
    - (a) 2
    - (b) 1
    - (c) 0

- **(d) 0 with carry 1**

3. Binary addition of 0 + 1 equals:
   - (a) 0
   - **(b) 1**
   - (c) 2
   - (d) Carry 1
4. Binary subtraction is done by:
   - (a) Borrowing like decimal
   - **(b) Adding 2's complement**
   - (c) Subtracting directly
   - (d) Shifting bits
5. Binary division is similar to:
   - (a) Decimal addition
   - **(b) Decimal long division**
   - (c) Hexadecimal division
   - (d) Octal division
6. Carry in binary addition occurs when:
   - (a) 0 + 0
   - (b) 0 + 1
   - **(c) 1 + 1**
   - (d) 1 + 0
7. Binary multiplication aligns numbers by:
   - (a) Leftmost bit
   - **(b) Least Significant Bit (rightmost bit)**
   - (c) Middle bit
   - (d) Random

Text Encoding

1. ASCII stands for:
   - (a) American Standard Code for Information Exchange
   - (b) American Standard Code for Internal Use

  - **(c) American Standard Code for Information Interchange**
  - (d) Advanced Standard Code for International

2. Standard ASCII includes how many characters?
  - (a) 64
  - (b) 127
  - **(c) 128**
  - (d) 256

3. Extended ASCII uses how many bits?
  - (a) 7
  - (b) 9
  - **(c) 8**
  - (d) 16

SECTION 11: KEY FORMULAS AND EQUATIONS

Number System Conversions

Decimal to Binary

text

Decimal ÷ 2 → Record remainder → Continue until quotient = 0 → Read remainders bottom to top

Decimal to Octal

text

Decimal ÷ 8 → Record remainder → Continue until quotient = 0 → Read remainders bottom to top

Decimal to Hexadecimal

text

Decimal ÷ 16 → Record remainder → Continue until quotient = 0 → Read remainders bottom to top

Data Representation

Whole Numbers

text

Maximum value for n-bit whole number = $2^n - 1$

Signed Integers

text

Maximum positive value = 2^(n-1) - 1

Minimum negative value = -2^(n-1)

2's Complement

text

2's complement = (1's complement) + 1

Binary Arithmetic

Binary Addition Rules

text

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 (carry 1)

Binary Subtraction

text

A - B = A + (2's complement of B)

SECTION 12: IMPORTANT DEFINITIONS

System Concepts

- **System:** A set of components working together to achieve a common objective
- **Software:** Collection of programs and instructions that tell a computer what to do

Types of Software

- **System Software:** Software that manages system resources and provides a platform for applications
- **Application Software:** Software designed to help users perform specific tasks
- **Operating System:** System software that manages computer hardware and resources
- **Device Driver:** System software that allows communication with hardware devices
- **Utility Program:** System software designed to maintain and protect the computer system

Architecture Components

- **Memory:** Stores data and instructions
- **CPU:** Processes instructions and performs calculations
- **ALU:** Performs arithmetic and logical operations
- **Control Unit:** Governs CPU activities and coordinates operations
- **Input Device:** Enables users to input data and instructions
- **Output Device:** Presents results of computer processing

System Bus

- **Data Bus:** Transfers data between components (two-way)
- **Address Bus:** Carries memory addresses (one-way)
- **Control Bus:** Transports control signals

Number Systems

- **Decimal System:** Base-10 system using digits 0-9
- **Binary System:** Base-2 system using digits 0 and 1
- **Octal System:** Base-8 system using digits 0-7
- **Hexadecimal System:** Base-16 system using digits 0-9 and letters A-F

Data Representation

- **Whole Numbers:** Non-negative integers (0, 1, 2, 3, ...)
- **Integers:** All positive and negative whole numbers and zero
- **1's Complement:** Binary representation by inverting all bits
- **2's Complement:** Binary representation by inverting all bits and adding 1

Text Encoding

- **ASCII:** 7-bit encoding scheme for 128 characters

- **UTF-8:** Variable-length Unicode encoding (1-4 bytes)
- **UTF-16:** Variable-length Unicode encoding (2-4 bytes)
- **UTF-32:** Fixed-length Unicode encoding (4 bytes)