Assignment - 1

Q-1 Explain Functional units of computer.

↳ A computer system performs different types of operations such as input, processing, storage and output. To perform these tasks efficiently, a computer is divided into several functional units. Each unit has a specific role in the working of the computer system.

→ The basic functional units of a computer are:

1) Input unit
2) Memory unit
3) Arithmethic Logic unit (ALU)
4) Control unit
5) output unit.

① Input Unit:-

The input unit is used to enter data and instructions into the computer system. It acts as a communication link between the user and the computer.

function:-

↳ Accepts data and instructions from the user.
↳ converts the input data into binary form
↳ Transfers the converted data to memory.

ex:- keyboard, mouse, scanner etc.

② Memory Unit:-

The memory unit stores data, instructions and intermediate results.

It is divided into two main types:

ⓐ Primary memory:-
Also called main memory. It directly communicates with the CPU.
↳ RAM (Random Access Memory)
↳ ROM (Read only Memory)

ⓑ secondary memory:-
Used for permanent storage of data.
↳ Hard DISK, SSD, Pen Drive

Function:-
↳ stores input data.
↳ stores programs.
↳ stores intermediate and final results.

③ Arithmetic Logic Unit (ALU):
ALU is a major part of the CPU. It performs all arithmatic and logical operations.

Arithmetic operation:-
- Addition
- subtraction
- multiplication
- Division.

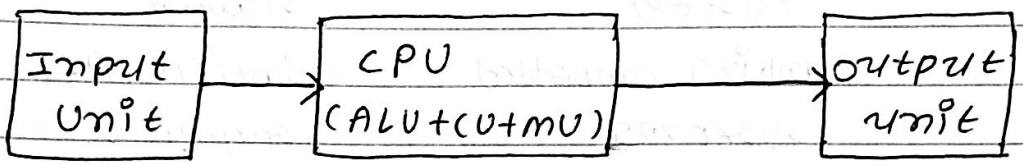
Logical operation:-
- AND
- OR
- NOT
- comparison operations (<, >, =)

This ALU is responsible for data processing.

④ Control unit (CU):-
The control unit provides processed information to the user.

Function:-
↳ Receives result from memory.
↳ converts binary data into human readable form.
↳ Display the result.

ex monitor, printer, speaker.



Q-2 Compare fixed and floating point Numbers.

→ In computer systems, real numbers can be represented in two ways: Fixed point representation and Floating point representation. These methods differ in the way decimal (or binary) point is stored and handled.

① Fixed Point Representation:-
In fixed point representation, the position of the decimal (binary) point is fixed. It does not change during calculations.

→ Features:
↳ Decimal point is fixed at a predefined position.

↳ simple hardware implementation.
↳ Fast computation.
↳ Limited range.
↳ suitable for embedded systems and financial calculation

Ex If decimal point is fixed after 2 digits:
123.45 → stored as 12345 (with fixed scaling factor).

② Floating point Representation.
In floating point representation the decimal (binary) point position can vary. It is represented using mantissa and exponent form.

It follows the standard like IEEE 754 format.
Number is represented as:
Number = Mantissa x (Base)$^{(Expone)}$

Features:
→ Decimal point "floats" depending on exponent.
→ Can represent very large and very small number
→ more complex hardware.
→ slightly slower than fixed point.
→ Used in scientific and engineering calculations.

Advantages:-
Fixed point
↳ simple design
↳ Faster computation
↳ Less memory required.

Floating point:-
↳ wide range
↳ can handle very small & very large num.
↳ High accuracy.

Parameter	fixed Point	floating Point
Decimal point	Fixed position	Variable position.
~~Number~~ Range	limited	very large.
precision	limited	high precision
Hardware complexity	simple	complex
Speed	Faster	slower
Application	Banking, embedded systems	scintific, AI, graphics.

Q-3 Explain Von Neumann architecture.

→ John Von Neumann proposed the Von Neumann Architecture in 1945.

→ It is a computer architecture design in which data and instructions are stored in the same memory

→ This architecture is also known as the stored program concept.

↳ Basic concept:-

In Von Neumann architecture:

↳ Program instructions and data share the same memory.

↳ Both use the same data bus and address bus.

↳ Instructions are executed sequentially one after another.

↳ Functional Units:-

The main components of Von Neumann architecure are

① Input unit

② output unit

③ Memory Unit

④ Central processing Unit (CPU)

↳ Arithmetic Logic Unit (ALU)

↳ control Unit (CU)

↳ Input → memory ↔ CPU (ALU + CU) → output.

↳ memory stores both data and instructions.

↳ CPU fetches instruction from memory.

↳ Control Unit decodes instruction.

↳ ALU performs required operation.

↳ Result is stored back in memory or sent to output.

→ Working of Von Neumann Architecture.
It follows the Fetch - Decode - Execute cycle:
① Fetch:- Instruction is fetched from memory.
② Decode:- Control Unit decodes the instructions.
③ Execute:- ALU performs operation.
④ Store:- Result is stored back in memory.

This cycle continues until program execution is complete.

→ Advantages:-
↳ simple design & Easy to implement.
↳ cost effective & flexibility

→ Disadvantages:-
→ only one data bus → causes bottleneck.
→ CPU can't fetch instruction and data at the same time
→ Slower compared to Harvard architecture.
This limitation is called Von Neumann Bottleneck.

Q-4 Describe parity and Hamming codes.
→ During data transmission, errors may occur due to noise or interference. To detect and correct errors, error control techniques like parity & Hamming code are used.

① Parity:-
It is a simple error detection technique used to detect single bit errors in data transmission.

Types of parity.

① Even parity:-
↳ Total number of 1's in the data (including parity bit) must be even.
↳ If number of 1's is odd, parity bit is set to 1.

② odd parity:-
↳ Total number of 1's must be odd.
↳ If number of 1's is even, parity bit is set to 1.

ex

Data:- 1011001
Number of 1's = 4 (even)
↳ for even parity → Parity bit = 0.
↳ for odd parity → Parity bit = 1.

Advantages:-
↳ simple and low cost.
↳ Easy to implement.

Limitations:-
↳ can only detect single bit errors.
↳ can't correct errors.
↳ can't detect multiple errors properly.

② Hamming code:-
It is an error detection and correction technique developed by Richard Hamming.
It can detect and correct single bit errors.

Basic idea:-

↳ Extra bits called parity bits are added at specific positions.

↳ Position are powers of 2: 1, 2, 4, 8, ...

↳ Remaining positions are data bits.

Number of Parity Bits:-

To find number of parity bits(r):

$$2^r \geq m + r + 1$$

where:

m = number of data bits

r = number of parity bits.

Working:-

1. Insert parity bits at position 1, 2, 4, 8, ...
2. Each parity bit check specific bit position.
3. At receiver side, parity bits are recalculated.
4. If error occurs, binary value of parity bits indicates error position.
5. That bit is corrected by flipping it.

→ Advantages:-

↳ Detects and corrects single bit errors

↳ more reliable than parity & memory systems (RAM)

→ Limitations:-

↳ more complex than simple parity.

↳ can't correct double bit errors.

Q-5 Define RTL with example.

RTL (Register Transfer Language) is a symbolic language used to describe the transfer of data between registers and the operations performed on data in digital systems.

→ RTL is mainly used in computer organization & Architecture to represent micro operations inside the CPU.

↳ RTL shows:
- ↳ which register transfer data
- ↳ to which register data is transferred.
- ↳ What operation is performed.

→ Basic format of RTL:-

The general form is:

Destination Register ← source Register.

The symbol "←" indicates transfer of data.

⇝ Types of Register Transfer operations:

1. Register Transfer
2. Arithmetic operations
3. Logic operations
4. shift operations.

① Register Transfer Example

If content of register R_1 is transferred to R_2:

$R_2 \leftarrow R_1$

meaning: Data stored in R_1 is copied to R_2.

② Arithmetic operation Example
Addition of two registers:
$R3 \leftarrow R1 + R2$
Meaning: Add contents of R1 and R2 and store result in R3.

③ Logic operation Example.
AND operation:
$R3 \leftarrow R1$ AND $R2$

④ Conditional Transfer Example.
Transfer only if control signal P = 1:
$P: R2 \leftarrow R1$
meaning: if P = 1 then transfer R1 to R2.

important points:-
↳ RTL describes micro operations.
↳ It is used to design and understand CPU operations.
↳ It does not represent full programming language instructions.
↳ Used in control unit design.

Q-6 Explain arithmetic micro operations.
Arithmetic micro operations are basic operations performed on binary numbers stored in registers. These operations are executed by the Arithmetic Logic Unit (ALU) of the CPU.

→ Arithmetic micro-operations deal with numerical data and perform mathematical calculations.

These operations are represented using Register Transfer Language (RTL)

Types of Arithmetic micro operations

① Addition :-
Adds the contents of two registers and stores the result in a register.

RTL Ex :- $R_3 \leftarrow R_1 + R_2$.
Add contents of R_1 and R_2 and store result R_3.

② Subtraction :-
subtracts one register from another.
subtraction is usually performed using 2's complement method.

ex :- $R_3 \leftarrow R_1 - R_2$.

③ Increment :-
Adds 1 to the content of a register.
ex :- $R_1 \leftarrow R_1 + 1$.
Used in program counter (PC) to move to next instruction.

④ Decrement :-
subtract 1 from the content of a register.
ex :- $R_1 \leftarrow R_1 - 1$.

⑤ Add with carry.
Adds two registers along with carry bit.
ex $R_3 \leftarrow R_1 + R_2 + C_{in}$

Hardware implementation:-

→ Arithmetic micro-operations are performed using:
- ↳ Binary Adder
- ↳ Parallel Adder.
- ↳ full Adder
- ↳ Adder-subtractor circuit.

The ALU performs these operations using combinational circuits.

→ Application:-
- ↳ Address calculation
- ↳ Loop control
- ↳ Program execution
- ↳ Numerical computations.

Q-7 Explain tri-state bus buffer.

In a digital computer system, many registers and devices share a common bus to transfer data. If multiple devices try to send data on the bus at the same time, it causes conflict to avoid this problem, a Tri state Bus Buffer is used.

→ A Tri-state Bus Buffer is a digital device that allows data to the bus only when enabled. It has three output states:
1. Logic 0
2. Logic 1
3. High impedance(z)

The third state disconnects the device from the bus.

Why It is Needed.

↳ To prevent bus contention
↳ To allow multiple devices to share a common bus
↳ To control data flow.

Working of Tri-state Buffer:-

A tri state buffer has:

↳ Data input
↳ control (enable) input
↳ output.

operation:

1> when Enable = 1
    ↳ output follows input (0 or 1)
2> when Enable = 0.
    ↳ output becomes high impedance (z)

High impedance means the device is electrically disconnected from the bus.

Truth Table:-

Enable(E)	Input(A)	output(Y)
0	X ← don't care	Z ← high impedance
1	0	0
1	1	1

↳ Advantages:-

↳ Allow multiple registers to share common bus.
↳ Reduces number of wires
↳ prevents short circuit condition.

Applications:

- System bus design
- Register transfer operation
- Memory and I/O interfacing
- Data communication lines.

Q-8 Differentiate logical & arithmatic shift

→shift operations are micro operations that move the bits of a register left or right. They are mainly used for multiplication, division and data manipulation.

→There are two important types of shift operations:
① Logical shift
② Arithmatic shift

① Logical shift :-
In logical shift, bits are shifted left or right and the vacant position is filled with 0.
It doesn't consider the sign of the numbers.

TYPES: ① Logical Left shift (LSL)
② Logical Right shift (LSR)

Ex original : 1011
LSL → 0110 } zero is inserted in empty position.
RSL → 0101

characteristics:
↳ used for unsigned numbers.
↳ Left shift multiplies by 2.
↳ Right shift divided by 2.

② Arithmetic shift (2's compliment)
Arithmetic shift is used for signed numbers
It preserves the sign bit.

Types:-
Arithmetic left shift (same as logical left shift)
Arithmetic Right shift (sign bit is preserved)

Original:- 1011 (negetive number in 2's compliment
Arithmetic Right shift → 1101

characteristics:-
Used for signed numbers.
preserves sign of number
Right shift maintain sign.

Q-9 write RTL for fetch cycle.
The fetch cycle is the first step of the introduction cycle. In this cycle, the CPU fetches the instruction from memory whose address is stored in the program counter (PC).

RTL (Register Transfer Language) is used to represent micro-operations performed during the fetch cycle.

Registers Used:-
PC → Program Counter
MAR → memory Address Register
MDR → memory Data Register
IR → Instruction Register.

RTL for Fetch cycle:-
The fetch cycle consists of the following micro operations:

S-1] Transfer address from PC to MAR.
Pc contains address of next instruction.
MAR ← PC.

S-2] Read instruction from memory
memory is accessed using MAR
MDR ← M[MAR]

S-3] Transfer instruction to IR.
instruction is stored in instruction Register.
IR ← MDR.

S-4] Increment program counter.
PC ← PC+1

MAR ← PC
MDR ← M[MAR]
IR ← MDR
PC ← PC+1.

Q-10 Explain performance metrics of computer.
→ Performance metrics are used to measure how efficiently and quickly a computer system execute s programs. These metrics help compare different computer systems based on speed, efficiency, & cost effectiveness.

(1) Execution Time (CPU Time)
Execution time is the total time required by the CPU to compelete a program.

It is of two types:
↳ User CPU Time
↳ System CPU Time.

Formula
CPU Time = Instruction Count x CPI x Clock cycle Time
where:-
↳ Instruction count(IC) = Total number of instruction executed
↳ CPI = cycles per Instruction.
↳ Clock cycle Time = $\frac{1}{\text{clock Rate}}$.

Execution time is the most import performance metric.

② Throughput:-
It is the number of tasks completed per unit time.
↳ Higher throughput means better performance
↳ Important in multi-user system and servers
ex
If a system completes 100 jobs per second, throughput = 100 jobs/sec.

③ Clock Rate (clock speed)
Clock rate is the number of clock cycles per second.
Measured in:
↳ MHz (Megahertz) & GHz (Gigahertz)
Higher clock Rate generally means faster processing, but it does not always guarantee better performance.

④ CPI (Cycles Per Instruction)
CPI indicates how many clock cycles are required to execute one instruction.

Lower CPI → Better performance.
Different instructions may have different CPI

⑤ MIPS (Million Instructions per second)
MIPS measures how many millions of instructions are executed per second.

$$MIPS = \frac{\text{Clock Rate}}{CPI \times 10}$$

Limitation:-
→ Doesn't consider instruction complexity.

⑥ MFLOPS
MFLOPS stands for million Floating Point Operations Per Second.
→ Used in scientific & engineering applications.

⑦ Benchmark Programs:-
Benchmark are standard programs used to evaluate System performance.
ex
SPEC benchmark (standard Performance Evaluation Corporation)
→ Benchmarks provide realistic comparison between systems.

-> Factors Affecting performance

- CPU speed
- memory access time
- cache memory
- I/O speed
- Compilers efficiency