Sub-CLO 1: Basic Computer Functions & Structures

Give an explanation about Moore's law.
Moore's Law asserts that the number of transistors on a semiconductor chip roughly doubles every 18 to 24 months. (Week 1, Page 30)

Consequences of Moore's Law:
- Enhanced performance due to the increasing number of transistors.
- Space-saving benefits from smaller and denser transistors, resulting in reduced space consumption.
- Lower retail prices because of more efficient production.
- Fewer interchip connections leading to improved performance efficiency.
- Reduced signal distances due to smaller transistor sizes, resulting in lower power consumption. Lower power consumption also results in less heat generation, reducing cooling requirements.

What is the key distinguishing feature of a microprocessor?
The primary distinguishing characteristic of a microprocessor is its capability to consolidate all vital CPU components onto a solitary chip.

Give explanation about Von Neumann computer concepts in modern computer systems today
The Von Neumann Architecture is a computer architecture that employs the same memory for storing program instructions and user data. This enables a computer to execute various applications and tasks by loading and running different programs without needing to alter the hardware physically. The Von Neumann Architecture, conceptualised by John von Neumann in the 1940s, forms the basis of modern computer systems. This concept comprises several key components:
- Central Processing Unit (CPU): The computer's brain that handles arithmetic and logic operations. Modern CPUs often feature multiple cores to perform multiple tasks simultaneously.
- Memory: Data and program instructions are stored in Random Access Memory (RAM) for quick access by the CPU.
- Input/Output Devices (I/O): Modern devices like keyboards, monitors, and network interfaces facilitate communication with the external world.

Modern computers have enhanced these concepts by integrating multiple cores, increasing memory capacities, and optimising I/O, resulting in powerful and efficient computer systems used in various applications today. These three components are connected through a communication pathway known as the system bus.

Sub-CLO 2: Processor performance analysis and benchmark

List and briefly define some of the techniques used in contemporary processors to increase speed.
- Pipelining: Concurrent processing of instructions in stages.
- Branch Prediction: Predicting and preloading future instructions to minimise delays caused by branching.
- Superscalar Execution: Simultaneous execution of multiple instructions in a single cycle.
- Data Flow Analysis: Reordering instructions to maximise parallel execution and reduce data dependencies.
- Speculative Execution: Executing instructions based on predictions to reduce idle time in the execution process.

Memory always lags in speed compared to a processor. Describe the efforts need to overcome the imbalance between memory and processor speeds

- Retrieve more bits at once to speed up memory access.
- Use efficient caches between the processor and memory to reduce the need for frequent memory access.
- Improve the DRAM interface by adding caching or buffering on the DRAM chip for faster data access.
- Enhance interconnect bandwidth using high-speed buses and hierarchical structures to accelerate data transfer between processors and memory.

Define the difference between MIPS and FLOPS
MIPS (Million Instructions Per Second) quantifies the number of instructions a CPU can execute in a second. It gauges the CPU's ability to process a set of basic instructions within a given time frame.

FLOPS (Floating-Point Operations Per Second), on the other hand, assesses the number of floating-point operations that an FPU (Floating-Point Unit) can perform in one second. This metric focuses on mathematical calculations involving real numbers, particularly useful for scientific and engineering applications.

Example:
A computer with **MIPS** 100 can execute 100 million instructions in one second.
A computer with **FLOPS** 100 can perform 100 million floating-point operations in one second

Give explanation about benchmark concept
A benchmark is a tool or method used to assess and summarise the performance of a computer system by running actual code examples. Benchmark principles involve desirable characteristics of such a program:

- **High-Level Language:** A benchmark should be written in a high-level programming language, making it portable across various machines and operating systems. This portability ensures that the benchmark can be run on different platforms for comparison.
- **Representative of Domain:** The benchmark should represent a specific programming domain or paradigm, such as systems programming, numerical computing, or commercial applications. This ensures that the benchmark's workload mimics real-world usage scenarios in that domain.
- **Measurability:** The benchmark should be easily measurable, allowing for quantifiable performance assessments. Measuring the time it takes to complete the benchmark or the number of operations performed is common practice.
- **Wide Distribution:** Ideally, benchmark programs should have wide distribution, meaning they are publicly available and widely adopted within the industry. This ensures consistency and allows for fair comparisons across different systems.

Here are some practical applications of benchmarks:

- **Gaming Laptop Selection:** Individuals seeking a new laptop for gaming can utilise benchmarks to compare the performance of different laptops. By running gaming benchmarks, they can determine which laptop offers the best gaming experience based on frame rates, graphics rendering, and overall performance.
- **Server Upgrade Decisions:** Companies looking to upgrade their servers can use benchmarks to evaluate the performance of different server configurations. This helps them identify the most suitable server for their specific needs, considering factors like data processing speed, response times, and overall server efficiency.

- **Game Development Optimization:** Game developers can employ benchmarks to test the performance of their games on various hardware configurations. By running benchmarks on different systems, they can optimise their game to ensure it runs smoothly and delivers an enjoyable gaming experience to a broad user base.

Sub-CLO3: Top level view of computer - instruction execution

Describe the role of the registers (PC, MAR, MBR, IR, AC) during the instruction cycle (fetch and execution)
During the instruction cycle, which comprises the fetch and execution phases, the role of the registers (Program Counter, Memory Address Register, Memory Buffer Register, Instruction Register, and Accumulator) can be described as follows:

Fetch Phase:

- **Program Counter (PC):** The PC stores the address of the next instruction to be executed. During the fetch phase, the CPU reads the instruction from this address, and the address is then sent to the Memory Address Register (MAR).
- **Memory Address Register (MAR):** MAR is responsible for storing the memory address that needs to be accessed or written to. When the CPU needs to read or write data from memory, the memory address is retrieved from MAR.
- **Memory Buffer Register (MBR):** The MBR is a register used to hold data that has been fetched from memory or is to be written to memory. It serves as an intermediary between the CPU and memory for data transfers.
- **Instruction Register (IR):** The instruction fetched from memory is placed into the IR. The IR contains the opcode, which is the operation code that determines the action to be performed, and may also contain memory addresses (operands) needed to execute the instruction.
- **Accumulator (AC):** The AC is a register used to temporarily store the results of arithmetic or logic operations. It acts as a temporary storage location for calculation results and the values in the AC are used in subsequent operations.

Execution Phase:

- **Program Counter (PC):** After fetching the instruction and incrementing the PC by 1 (+1), the CPU proceeds to fetch the next instruction from memory.
- **Memory Address Register (MAR):** MAR retrieves data from the Program Counter and then reads or writes data from memory, forwarding it to the MBR.
- **Memory Buffer Register (MBR):** The MBR retrieves data from the MAR to store the data fetched from memory or to be written to memory for data transfer.Instruction Register (IR): The instruction stored in the MBR is transferred to the IR, which contains the opcode for the operation to be executed.
- **Accumulator (AC):** AC takes data from the IR and temporarily stores it as a working area. The values in the AC are utilised in subsequent operations, possibly as operands or to hold intermediate results.

List and briefly define the types of bus in computer structure

- **Data Bus:** This is a set of signal lines that provides a pathway for the transfer of data between the central processing unit (CPU), memory, and input/output devices. It is responsible for carrying the actual data being processed.

- **Control Bus:** The control bus consists of signal lines used to control the access and utilisation of data and address lines. It carries signals that determine the operation to be performed, such as read, write, or fetch. It helps coordinate and synchronise data transfer operations.
- **Address Bus:** The address bus is a set of signal lines used to designate the source or destination of data on the data bus. It carries memory addresses or I/O addresses, specifying where data is to be read from or written to. The size of the address bus determines the maximum addressable memory capacity.

List and briefly define two approaches to dealing with multiple interrupts.

- **Disabled Interrupts:** During interrupt handling, further interrupts are temporarily ignored and queued for later processing. It doesn't consider priorities.
- **Interrupt Priorities:** Assign priorities to interrupts, allowing higher-priority interrupts to preempt lower-priority ones for more organised handling.

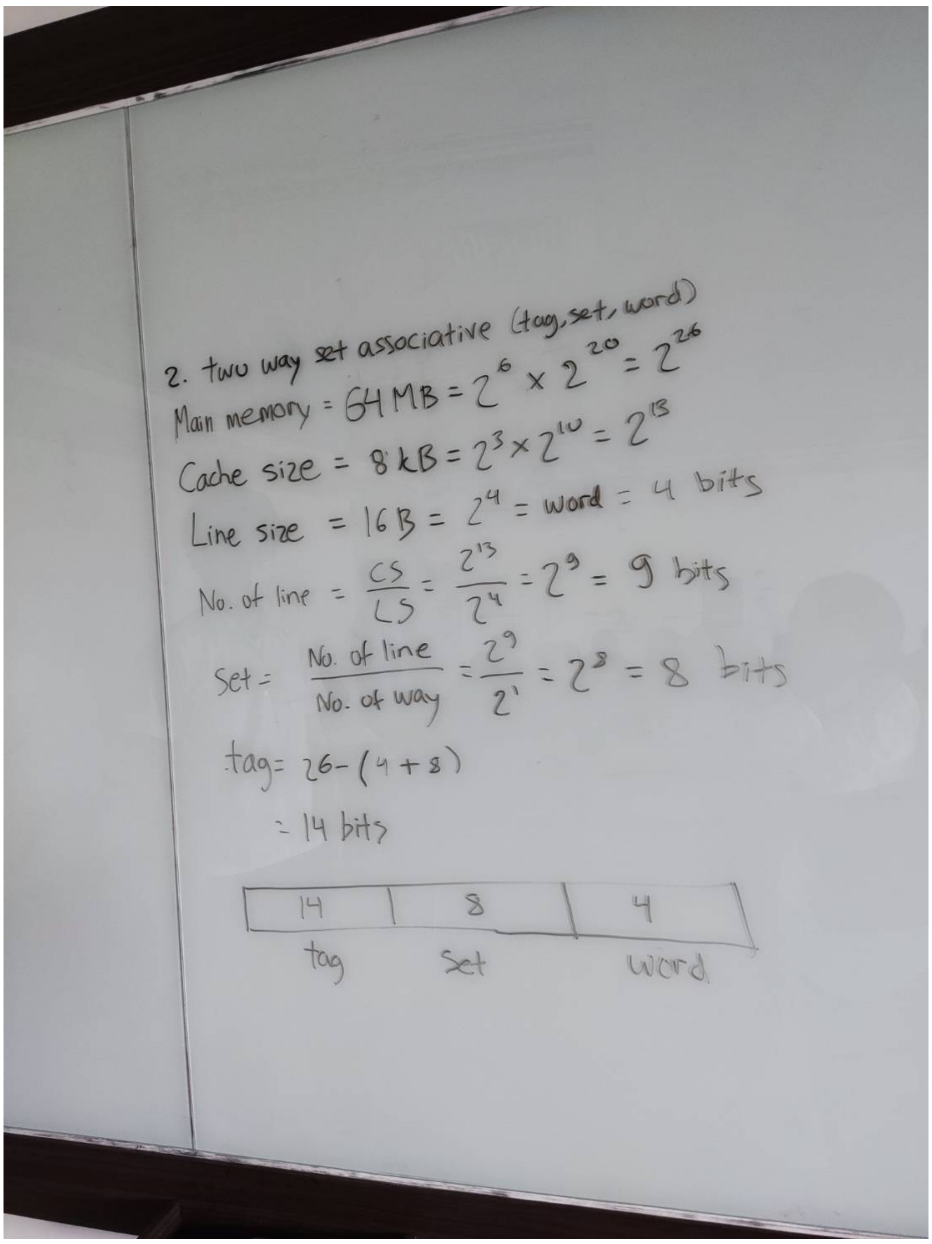
Sub-CLO 4 - cache memory

What is the general relationship among access time, memory cost, and capacity?

The relationship among access time, memory cost, and capacity can be summarised as follows:

- Faster access time generally results in a greater cost per bit of memory.
- Greater capacity typically leads to a smaller cost per bit of memory.
- However, increasing capacity often results in slower access times.

A two-way set-associative cache has lines of 16 bytes and a total size of 8 kB. The 64-MB main memory is byte addressable. Show the format of main memory addresses.



2. two way set associative (tag, set, word)

Main memory = 64 MB = $2^6 \times 2^{20} = 2^{26}$

Cache size = 8 kB = $2^3 \times 2^{10} = 2^{13}$

Line size = 16 B = 2^4 = word = 4 bits

No. of line = $\frac{CS}{LS} = \frac{2^{13}}{2^4} = 2^9$ = 9 bits

Set = $\frac{\text{No. of line}}{\text{No. of way}} = \frac{2^9}{2^1} = 2^8$ = 8 bits

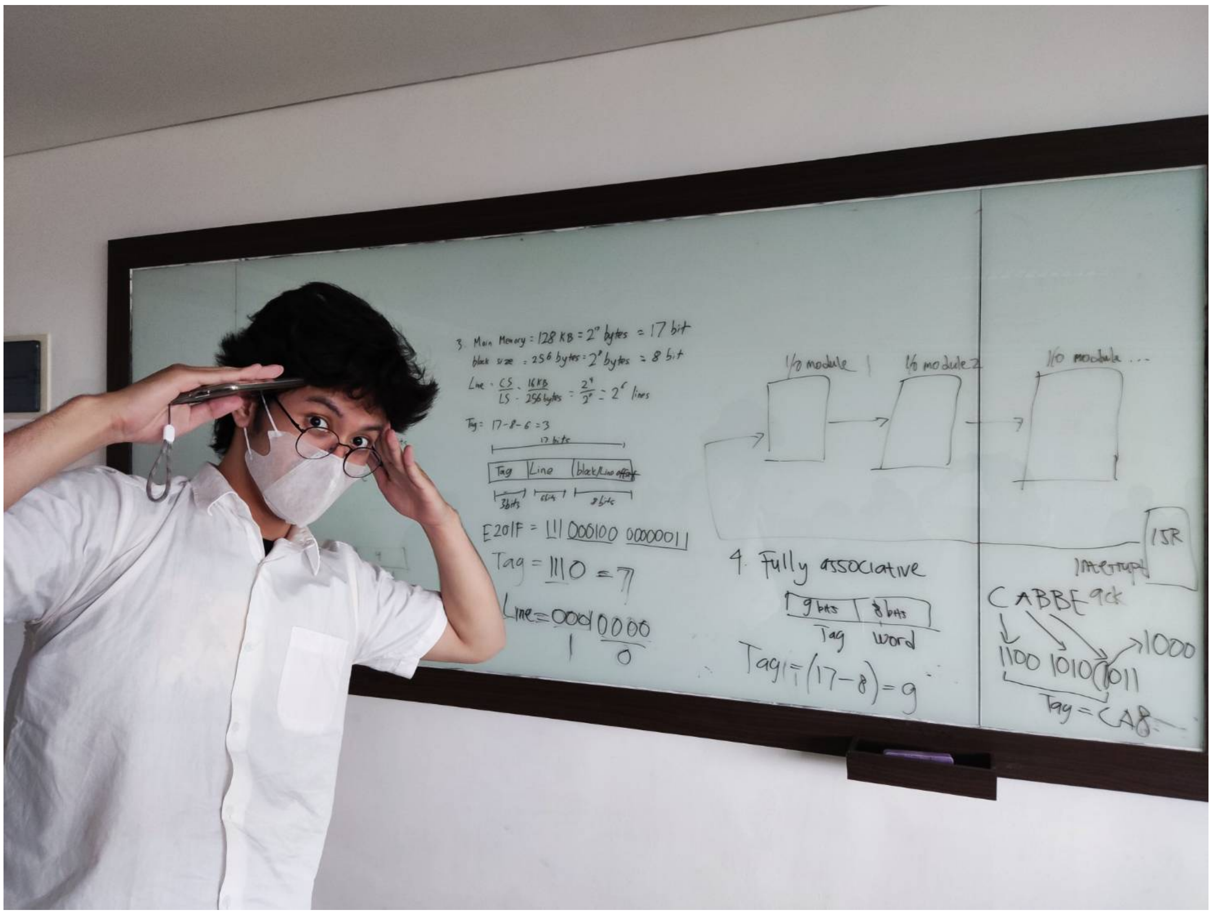
tag = 26 − (4 + 8)

= 14 bits

14	8	4
tag	set	word

Consider a direct mapped cache of size 16 KB with block size of 256 bytes. The size of the main memory is 128 KB. Find the number of bits in the tag. What are the tag and cache line address (in hex) for the main memory address (E201F)?

With the same details as number 3 but the mapping technique change to fully-associative cache, find the tag address (in hex) for main memory address (CABBE)



Sub-CLO 5 internal-external memory

What is the difference between DRAM and SRAM in terms of characteristics such as speed, size, density, cost, and power requirements?

- **Speed:**
  - **DRAM:** Slower due to the need for periodic refresh cycles.
  - **SRAM:** Faster and more efficient as it doesn't require periodic refresh, often used as cache memory for its speed.

- **Size:**
  - **DRAM:** Smaller memory cells, resulting in higher memory density.
  - **SRAM:** Larger memory cells, physically larger than DRAM.

- **Density:**
  - **DRAM:** Higher density due to smaller cell size per bit of data.
  - **SRAM:** Lower density because of larger cell size per bit of data.

- **Cost:**
  - **DRAM:** Generally more cost-effective than SRAM due to its simpler structure and greater physical efficiency, making it common in personal computers.
  - **SRAM:** More expensive than DRAM, attributed to its complex structure and larger cell size.

- **Power Requirements:**
    - **DRAM:** Higher power requirements due to periodic refresh cycles.
    - **SRAM:** More power-efficient because it doesn't require periodic refresh cycles.

In summary, SRAM is faster and more reliable but at a higher cost and power consumption, making it suitable for cache memory. DRAM, while slower, is more cost-effective, denser, and efficient in terms of power usage, commonly used for main system memory in computers and other devices.

What are the differences among EPROM, EEPROM, and flash memory?

EPROM	● Erasable programmable read-only memory ● Erasure process can be performed repeatedly ● More expensive than PROM but it has the advantage of the multiple update capability ● Erasure is performed by shining an intense ultraviolet light
EEPROM	● Electrically erasable programmable read-only memory ● Can be written into at any time without erasing prior contents ● Combines the advantage of non-volatility with the flexibility of being updatable in place ● More expensive than EPROM
Flash Memory	● Intermediate between EPROM and EEPROM in both cost and functionality ● Uses an electrical erasing technology, does not provide byte-level erasure ● Microchip is organised so that a section of memory cells are erased in a single action or "flash"

Briefly define the seven RAID levels along with its advantages and disadvantages.

- **RAID 0 (Striping):**

  - **Description:** RAID 0 distributes data evenly across two or more disks without providing redundancy or fault tolerance.
  - **Advantages:** Increased data transfer speed, as data is striped across multiple disks.
  - D**isadvantages:** No data redundancy, which means if one disk fails, the entire array may be lost.

- **RAID 1 (Mirroring):**
  - **Description:** RAID 1 involves mirroring data by duplicating it across two or more disks. Every write operation is performed on all mirrored disks.
  - **Advantages:** High data redundancy and reliability. Simple recovery in case of disk failure.
  - **Disadvantages:** Requires double the storage capacity. Relatively more expensive.

- **RAID 2:**
  - **Description:** RAID 2 uses bit-level striping and Hamming codes for error correction. It's rarely used in practice.
  - **Advantages:** High data integrity through error correction.
  - **Disadvantages:** Complex, expensive, and rarely implemented.

- **RAID 3:**
  - **Description:** RAID 3 stripes data at the byte level and stores parity information on a dedicated disk.
  - **Advantages:** High data transfer rates. Data on a failed drive can be reconstructed from parity.
  - **Disadvantages:** Poor performance for random access and small transfers.

- **RAID 4:**
  - **Description:** RAID 4 stripes data at the block level and keeps parity information on a dedicated disk.
  - **Advantages:** Suitable for high I/O rates. Parity is stored separately on the dedicated disk.
  - **Disadvantages:** Poor performance for write operations due to the bottleneck of the parity disk.

- **RAID 5:**
  - **Description:** RAID 5 stripes data and parity information across multiple disks, distributing the parity blocks.
  - **Advantages:** Good balance of performance and redundancy. No single parity disk bottleneck.
  - **Disadvantages:** High overhead for parity calculation and disk rebuilds.

- **RAID 6:**
  - **Description:** RAID 6, like RAID 5, stripes data and parity, but it uses two sets of parity information (double parity) for increased fault tolerance.

    - **Advantages:** High data availability and reliability with double parity.
    - **Disadvantages:** Lower performance compared to RAID 5, and higher overhead.

Sub-CLO 6 input output

List and briefly define three techniques for performing I/O

- **Programmed I/O:**
    - **Description:** Data exchange occurs directly between the processor and the I/O module. The processor executes a program that provides it with direct control over the I/O operation.
    - **Characteristics:** The processor issues a command and must wait until the I/O operation is complete, which can be inefficient if the processor is faster than the I/O module.

- **Interrupt-Driven I/O:**
    - **Description:** In this technique, the processor initiates an I/O command, continues executing other instructions, and is interrupted by the I/O module when it has finished its work.
    - **Characteristics:** This approach allows the processor to continue its tasks while waiting for I/O completion, improving overall system efficiency.

- **Direct Memory Access (DMA):**
    - **Description:** DMA enables direct data exchange between the I/O module and main memory without direct processor involvement. The I/O module takes control of memory transfers.
    - **Characteristics:** DMA reduces the load on the processor by allowing I/O devices to access memory directly, increasing overall system performance.

What is the difference between memory-mapped I/O and isolated I/O?

- **Memory-Mapped I/O:**
    - Treats I/O devices as if they were part of the memory space.
    - The CPU can access I/O devices using the same instructions and addresses used for memory access.
    - Easier to implement and faster.
    - May limit the address space available for both memory and I/O devices.

- **Isolated I/O:**
    - Uses separate address spaces and instructions for memory and I/O devices.
    - Requires the CPU to use specific I/O instructions to access I/O devices.
    - More complex to implement and slower.
    - Provides a larger I/O address space and greater flexibility.

When a device interrupt occurs, how does the processor determine which device issued the interrupt?

- **Polling:** The processor checks each connected device individually until it finds the one causing the interrupt. This can be time-consuming for many devices.
- **Multiple Interrupt Lines:** Each device has a unique interrupt line. The active interrupt line indicates which device needs attention, simplifying identification.
- **Daisy Chain Method:** Devices are linked in a sequence, and the interrupt passes through each device until it reaches the intended one, allowing sequential examination.

- **Bus Arbitration:** Devices must request control of the system bus before their interrupt is processed. A defined protocol prioritises interrupts based on who gains bus control first. This efficiently manages and prioritises interrupt requests.

Describe what Direct Memory Access (DMA) is, and what problem it solves.
Direct Memory Access (DMA) is a hardware feature that allows I/O devices to access the computer's memory directly, with minimal involvement from the CPU. DMA is managed by a DMA controller, which communicates with both the CPU and the I/O devices.

The main problem that DMA solves is the inefficiency of traditional I/O methods, where the CPU handles data transfers between I/O devices and memory. DMA addresses this issue by offloading the data transfer task from the CPU to the DMA controller. This has several advantages:

- **Improved Data Transfer Speed:** DMA transfers data more quickly than CPU-managed I/O because it operates independently and is not limited by the CPU's processing speed. This allows for faster I/O operations.
- **Reduced CPU Overhead:** With DMA, the CPU is free to perform other tasks while data is being transferred. This reduces the CPU's involvement in managing I/O, allowing it to be more productive.
- **Efficient Handling of Large Data Volumes:** DMA is particularly useful when large volumes of data need to be moved between memory and I/O devices, as it can do so with greater efficiency than the CPU.

In summary, DMA addresses the inefficiencies of traditional I/O methods by enabling direct data transfer between I/O devices and memory, leading to faster data transfers, reduced CPU overhead, and efficient handling of large data volumes.