1. Introduction

Q.2.2.1.1 Distingush between the concepts: computer architecture and computer organization.
Answer: Computer architecture refers to attributes such as the instruction set, the number of bits used to represent various data types, I/O mechanisms, and techniques for addressing memory. These attributes have a direct impact on the logical execution of a program. On the otherhand, computer organization refers to the operational units and their interconnections that realize the architectural specifications. Some examples of organizational attributes are control signals, interfaces between the computer and peripherals, and the memory technology used. These hardware details are transparent to the programmer.

Q.2.2.1.2 Computer being a complex system, discuss concepts of structure and function associated with it.
Answer: A computer system may be perceived as a collection of complex systems, where each sytem is a set of interrelated subsystems, connected in hierarchical manner. At each level, the system consists of a set of components and their interrelationships. At each level, the designer is concerned with structure and function. Here, *structure* refers to the way in which the components are interrelated, and *function* refers to the operation of each individual component as part of the structure.
In general, there are four functions: data processing, data storage, data movement and control. A computer has four main structural components: central processing unit (CPU), main memory, I/O devices, system interconnection. The most interesting and complex unit is CPU. It has four components: arithmetic and logic unit (ALU), control unit, registers and CPU interconnection. Again, the control unit has three components: sequencing logic, control unit registers and decoders, and control memory.

Q.2.2.1.3 What is multicore machine?
Answer: Designers of machine find difficulty in making powerful processor by increasing clock rate. The faster the clock rate, the greater the amount of power to be dissipated. Thus, designers have turned to a fundamentally new approach to improving performance by placing multiple processors on the same chip, with a large shared cache. The use

of multiple processors on the same chip, referred to as multiple cores, or multicore. It provides the potential to increase performance without increasing the clock rate. Chip manufacturers are now in the process of making a huge leap forward by increasing more number of cores per chip.

Q.2.2.1.4 Discuss different levels of memory available in a computer.
Answer: Normally, four different levels of memory are available in a computer: register, cache memory, primary memory and secondary memory. (see Fig. 1.1)
Register: CPU registers are at the top of the memory hierarchy. Registers usually consist of small amount of storage, may be read only or write only. They provide fastest way to access data. Some example of registers are accumulator, program counter, instruction register, memory buffer register and memory address register.

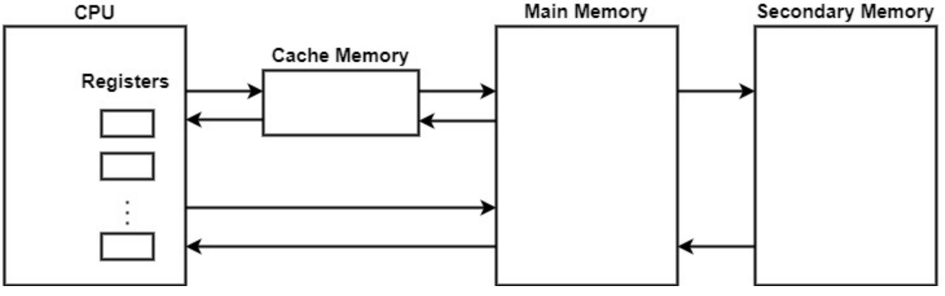

Figure 1.1: Hierarchy of memory

Cache memory: Cache is a fast memory block placed between CPU and main memory. Highly-accessed data are placed in cache memory. It allows CPU to get data faster. Cache memory can be considered at the second level in the memory hierarchy.
Primary memory: It is the primary storage of data and program in a computer. When a program is in execution, program, data and a part of operating system reside in the main memory. It is volatile in nature in the sense that when power goes off, the content of maim memory is lost.
Secondary memory: Secondary memory is at the lowest level. Some example of secondary memory are hard disk, optical disk, tape drive. It is non-volatile in nature. They are mainly used to take backup of data and programs. They are normally least expensive.

Q.2.2.1.5 Discuss different types of registers available in a CPU.
Answer: Some important registers are discussed here.
Accumulator: Intermediate arithmetic and logic results are stored here. Modern computer systems often have multiple general purpose registers that operate as accumulators. This is the most frequently used register to store data taken from memory.
Memory address registers (MAR): It holds the address of the location to be accessed from memory.
Memory data registers (MDR): It contains data to be written into or to be read out from the addressed location. MAR and MDR (Memory data register) together facilitate the communication of the CPU and the main memory.
General purpose registers: These registers are numbered as $R0, R1, R2, \ldots, Rn$, and used to store temporary data during any ongoing operation. Its content can be accessed through assembly language programming.
Program counter (PC): It is used to keep the track of execution of a program. PC contains the address of the next instruction to be fetched from the main memory when the previous instruction has been successfully completed.
Instruction register (IR): It holds the instruction which is currently being executed.

Q.2.2.1.6 What is the purpose of clock?
Answer: The system clock is a device used in computers which regulates the internal components of the computer by issuing a high-frequency signal. This signal ensures that all components are synchronized.
The length of a clock cycle is the time required for one complete clock pulse. The clock rate (R) is the number of clock cycles per second, i.e., R= $\frac{1}{\rho}$, where ρ is the length (in second) of one clock cycle.

Q.2.2.1.7 Compare and contrast computers that follow RISC and CISC philosophies.
Answer: RISC is abbreviation of reduced instruction set computer. RISC is a computer with a small, highly optimized set of instructions, rather than the more specialized set often found in other types of architecture. It is a type of microprocessor architecture that uses a small set of instructions of uniform length. These are simple instructions which are generally executed in one clock cycle. RISC chips are relatively simple to design and inexpensive.A disadvantage of this design is that the computer has to repeatedly perform simple operations to execute a larger program

2. Computer Evolution and Performance

Q.2.2.2.1 Name the first general purpose digital computer.
Answer: Electronic numerical integrator and computer (ENIAC) was the first programmable, electronic, general-purpose digital computer. ENIAC was designed by John Mauchly and J. Presper Eckert of the University of Pennsylvania, U.S. and was formally dedicated at the University of Pennsylvania on February 15, 1946. The project was financed by the United States Army.
It was a general-purpose machine, built using vacuum tubes. The machine was primarily built to help compute artillery firing tables during World War II. It was programmable through manual setting of switches and plugging of cables.

Q.2.2.2.2 Discuss some performance measures of a computer.
Answer: We shall discuss here a number of performance measures.
1. *CPU time required to execute a job (program)*
Let $nCC(J)$ be the number of CPU clock cycles required to excute job J. Let $t(C)$ be the cycle time of CPU C.
Then CPU time required to execute job J is $nCC(J) \times t(C)$ (2.1)
Let $f(C)$ be the clock frequency of CPU C. Then $f(C) = \frac{1}{t(C)}$ (2.2)
Using (2.1) and (2.2) we get,
CPU time required to execute job J is $\frac{nCC(J)}{f(C)}$ (2.3)

2. *CPU time required to execute a job (program)*
It may be easier to count the number of instructions executed in a given program. Let $avg\ (CPI)$ be the average number of CPU clock cycles per instruction.
$avg\ (CPI) = \frac{nCC(J)}{number\ of\ instructions\ for\ job\ J}$ (2.4)
Then CPU time required to execute job J
$= number\ of\ instructions\ \times\ avg\ (CPI) \times t(C)$
$= \frac{number\ of\ instructions\ \times\ avg\ (CPI)}{f(C)}$ (2.5)

3. *Million of instructions per second (MIPS)*
$1\ million = 10,00,000 = 10^6$.
Then $MIPS = \frac{Number\ of\ instructions}{Execution\ time\ \times 10^6}$ (2.6)

4. *Million of floating point operations per second (MFLOPS)*

$1\ million = 10,00,000 = 10^6$.
Then $MFLOPS = \frac{Number\ of\ floating\ point\ operations}{Execution\ time\ \times 10^6}$ (2.7)

Q.2.2.2.3 State features of Von Naumann model. Draw a picture of the proposed model of computer.
Answer: The task of entering and modifying programs for the ENIAC (see Q.2.2.2.1) was extremely tedious. If a program can be represented in a form suitable for storing in memory alongside the data, the computer could get its instructions by reading them from memory. Also, the program could be modified easily, by modifying some instructions stored in a portion of memory. This idea, known as the stored-program concept, was proposed by John von Neumann, who was a consultant of the ENIAC project. The proposed model has been depicted in Fig. 2.1.

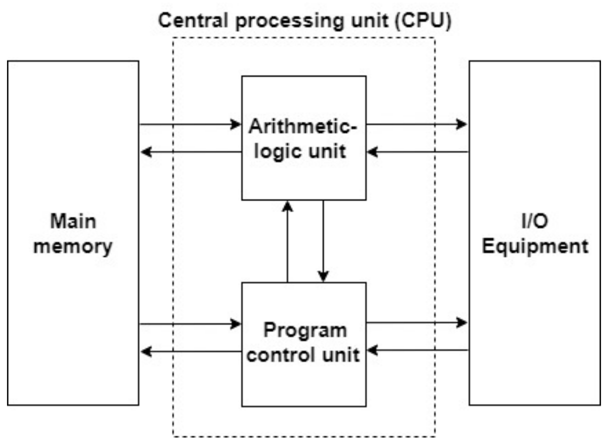

Figure 2.1: Von Naumann model

Some imp[ortant features of the proposed model are given below:
* There is a main memory, which stores both data and instructions.
* An arithmetic and logic unit (ALU) is placed, which is capable of operating on binary data.
* A control unit is attached, which interprets the instructions in memory and causes them to be executed.
* There is input / output (I/O) equipment that is operated by the control unit.
Most of today's computers have this same general structure and function.

Q.2.2.2.4 State characteristics of first generation of computers. Name a few of such computers.
Answer: First generation of computers
∗ were made of vacuum tube technology
∗ supported machine language only
∗ were very expensive
∗ produced a lot of heat
∗ associated with slow I / O devices
∗ were of large size
∗ were expensive
A few first generation of computers are mentioned in Table 2.1.

Table 2.1 Some first generation of computers

Model Name	Year	Manufacturer	Country
UNIVAC I	1951	Eckert-Mauchly Computer Corporation	USA
701	1953	IBM	USA
UNIVAC II	1958	Eckert-Mauchly Computer Corporation	USA

Q.2.2.2.5 Give some features of second generation of computers. Mention a few of those computers.
Answer: These computers used transistors, had magnetic memory and magnetic storage disks with high speed I/O devices. They were of reduced size, and generated much lesser heat. Overall, speed and reliability were improved. These computers were programmed using high level languages such as Fortran and Cobol. Refer Table 2.2 for some second generation of computers.

Table 2.2 Examples of second generation of computers

Model Name	Year	Manufacturer	Country
7090	1960	IBM	USA
System/360	1964	IBM	USA
PDP-8	1965	DEC	USA

Q.2.2.2.6 Describe features of third generation of computers. List a few third generation of computers.
Answer: These computers used integrated circuits from small-scale integration CPUs (SSI) to large-scale integration CPUs (LSI). They are

3. Computer Functions and Internections

Q.2.2.3.1 Specify the functions of the following registers: memory address register, memory data register, memory buffer register, I/O address register, I/O buffer register

Answer: *Memory address register (MBR)*: In a computer, the memory address register is the CPU register that either stores the memory address from which data will be fetched to the CPU, or the address to which data will be sent and stored. When reading from memory, data addressed by MAR is fed into the MDR (memory data register) and then used by the CPU.

Memory buffer register (MBR): It is also known as memory data register (MDR). A memory buffer register is the register in a computer's processor, or CPU, that stores the data being transferred to and from the immediate access storage. It contains the copy of designated memory locations specified by the MAR. It acts as a buffer allowing the processor and memory units to act independently without being affected by minor differences in operation. A data item will be copied to the MBR ready for use at the next clock cycle, when it can be either used by the processor for reading or writing or stored in main memory after being written.

I/O address register (I/O AR): It specifies a particular I/O device.

I/O buffer register (I/O BR): It is used for the exchange of data between an I/O module and the processor. An I/O module transfers data from external devices to processor and memory, and vice versa. It contains internal buffers for temporarily holding this data until it can be sent on. It is used for the exchange of data between an I/O module and the CPU.

Q.2.2.3.2 With the help of a neat diagram, provide a top-level view of a computer.

Answer: A program is a set finite instructions that is stored in the main memory of a computer. The CPU executes the program by executing one instruction at a time. Key computing components of a simple computer are shown in Fig. 3.1. Some basic steps are fetch, decode, execute and store operations. All these operations fall under fetch and execute cycles. Program execution can be halted by any of the following events: turning off the computer, encountering an error and executing an instruction to stop the execution.

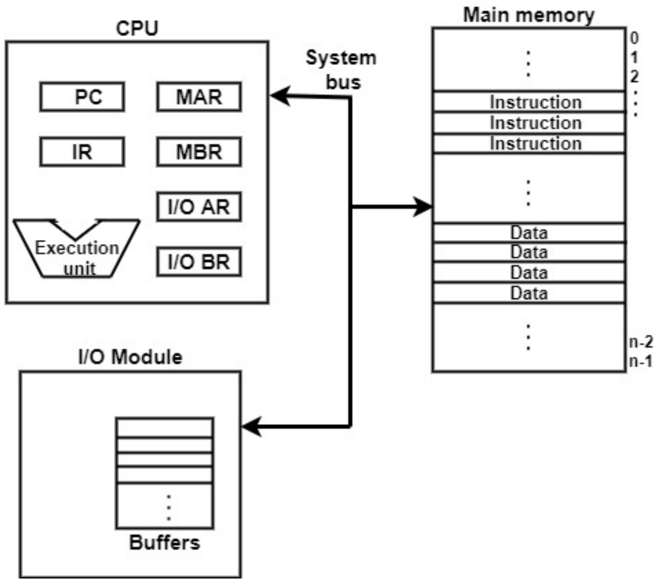

Figure 3.1: Computing components of a computer

Q.2.2.3.3 Discuss different actions that processor has to take based on decoding an instruction.
Answer: The processor interprets an instruction and performs the required action. An action can be one of the following types:
• Data transfer between processor and memory: Data may be transferred from processor to memory or from memory to processor.
• Data transfer between processor and I/O: Data may be transferred to or from a peripheral device by transferring between the processor and an I/O module.
• Data processing: The processor may perform some arithmetic or logic operation on data.
• Control: An instruction may specify that the sequence of execution is needed to be altered. For example, the processor may fetch an instruction from location 300, which specifies that the next instruction be from location 400. The processor will remember this fact by setting the program counter to 400. Thus, on the next fetch cycle, the instruction will be fetched from location 400 rather than 301.

Q.2.2.3.4 Write the steps performed for the instruction: *ADD B*, *A*
Answer: This instruction states that the contents of memory locations *B* and *A* are to be added, and the resultant sum is to be stored into memory location *A*. The following steps are followed.
- Fetch the *ADD* instruction.
- Read the contents of memory location *A* into the processor.
- Read the contents of memory location *B* into the processor.
- Add the two values in accumulator.
- Store the result into memory location *A*.

Q.2.2.3.5 What is an interrupt? How does processor handle an interrupt.
Answer: An interrupt is a signal, sent to the processor that interrupts the execution of current process. It may be generated by a hardware device or a software program.
An interrupt is a condition that alerts the processor. It is a request for the processor to interrupt the currently executing code when permitted, and serve the requested service (event) in a timely manner. If the request is accepted, the processor responds by suspending its current activities, saving its state, and executing a function called, *interrupt service routine*, to deal with the event. This interruption is temporary, and, unless the interrupt indicates a fatal error, the processor resumes normal activities after the interrupt handler finishes.

Q.2.2.3.6 Discuss different types of interrupt.
Answer: There are different types of interrupt, stated as follows:
Software: This type of interrupts are generated by some conditions that occur as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space. Software generated interrupts are also called *traps*.
I/O: I/O devices communicate with the CPU to acknowlege certain events. They are generated by I/O controllers, to signal normal completion of an operation, request service from the processor, or to signal a variety of error conditions.
Hardware failure: These are generated by failures such as power failure and memory parity error.
Timer: It is generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis. During context switching, timer interrupts the CPU, and the CPU saves the