Module 1

Introduction of Computer Organization and Architecture

Motivation:

Computer Organization and Architecture provides in-depth knowledge of internal working, structuring, and implementation of a computer system. This provides the basis for learning any computer related subjects.

Syllabus:

Lecture no	Content	Duration (Hr)	Self-Study (Hrs)
1	Basic organization of computer and block level description of the functional units, Introduction to computer organization & Architecture	1	1
2	Evolution of Computers, Von Neumann model	1	1
3	Instruction cycle	1	2
4	Addressing Modes, Instruction Format	1	2
5	Introduction to System buses, Multi-bus organization	1	2

Learning Objectives:

Learners shall be able to:

1. To explain the basic units of a computer
2. To compare computer organization and architecture
3. To explain the concept of Von Neumann model
4. To list the various instruction formats
5. To describe the entire process of instruction execution
6. To explain about different Bus Architectures

Theoretical Background:

In order to achieve complete understandings of computer systems, it is always important to consider both hardware and software design of various computer components. In other words, every functionality of the computer has to be studied to increase the performance of the computer.

Computer organization and architecture mainly focuses on various parts of the computer in order to reduce the execution time of the program, improve the performance of each part. Generally, we tend to think computer organization and computer architecture as same but there is slight difference. Computer Organization is study of the system from software point of view and gives overall description of the system and working principles without going into much detail. In other words, it is mainly about the programmer’s or user point of view. Whereas Computer Architecture is study of the system from hardware point of view and emphasis on how the system is implemented. Basically, throws light on the designer’s point of view.

Key Definitions:

Computer: A computer is a programmable machine that receives input, stores and manipulates data, and provides output in a useful format.

Structure: It is the static arrangement of the parts (plan).

Architecture: It is basically a design. The term architecture can refer to either hardware or software, or to a combination of hardware and software. The architecture of a system always defines its broad outlines, and may define precise mechanisms as well.

Organization: The dynamic interaction of these parts and their management (design).

Central Processing Unit (CPU): The main part of the computer, its “brain”, consisting of the central memory, arithmetic logic unit and control unit.

Arithmetic Logic Unit (ALU): It is a digital circuit that performs arithmetic and logical operations. The ALU is a fundamental building block of the central processing unit (CPU) of a computer.

Control Unit (CU): That part of the computer which accesses instructions in sequence interprets them and then directs their implementation.

Accumulator: A special storage register associated with the arithmetic logic unit for storing the results of steps in a calculation or data transfer.

Memory Unit: The part of the computer where data and instructions are held.

Program Counter: It is an address register. It is used to storeaddress of next instruction to be executed hence also referred to an instruction address register.

Address Register: It is a 12- bit address register. It is used to specify the address in the memory of the word to be written into or read from the data register.

Data Register: It is a 40 bit register and used to store any 40 bit word.

Accumulator And Multiplier Quotient: These are two 40 bit register used for the temporary storage of the operands and results.

Chip: It is an alternative name for an integrated circuit.

Clock: It is a special circuit that sends pulses of current to the CPU and other computer components.

Fetch-Execute Cycle: It refers to the process whereby the control unit must first fetch an instruction from main memory before it can execute (interpret) that instruction.

Instruction Decoder: It is a complex circuitry in the control unit designed to decode (interpret) any instruction in the computer's machine code repertoire.

Instruction Register (IR): It is a special register in the CPU that holds the bit pattern corresponding to the next instruction to be performed within the CPU. The Control Unit accesses this register to decide which circuits need to be activated.

Instruction Set: It is a set of assembly language mnemonics which represent the machine code of a particular computer.

Memory Address Register (MAR): When another instruction is needed in the IR, or a value is to be loaded into the accumulator, or an operand is needed to perform some arithmetic or logic instruction, this register contains the memory address where the desired information can be found. It also serves as a pointer to the location in memory where the contents of some CPU register is to be stored.

Memory Buffer Register (MBR): This register serves as an interface between the CPU and main memory. Anything needed by the CPU (instruction or data) is first placed here before it goes to its final destination (such as the accumulator, IR, PC or other registers). Also, anything in the CPU that is to be stored in main memory comes here first before being copied into the main memory at the location specified by the address contained in the MAR.

Program Counter (PC): This CPU register always contains the memory address where the next instruction to be performed by the CPU can be found.

Read Only Memory (ROM): It is the data that is stored permanently inside this chip. It cannot be removed.

Register: It is a special location, which is sometimes protected, used for specific purposes only. Example: accumulator, program counter

Course Content:

Lecture 1

Basic organization of computer and block level description of functional units:

<u>Functional Units</u>

A computer consists of 5 main parts.

- Input
- Memory
- Arithmetic and logic
- Output
- Control Units

Functional units of a Computer

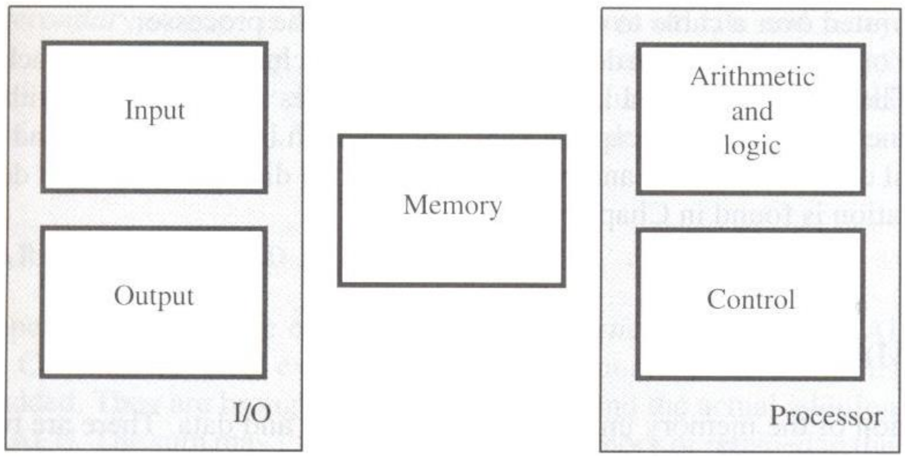


Figure 1.1: Basic functional units of a computer

Input unit accepts coded information from human operators, from electromechanical devices such as keyboards, or from other computers over digital communication lines. The information received is either stored in the computer’s memory for later reference or immediately used by the arithmetic and logic circuitry to perform the desired operations. The processing steps are determined by a program stored in the memory.

Finally the results are sent back to the outside world through the output unit. All of these actions are coordinated by the control unit. The list of instructions that performs a task is called a program. Usually the program is stored in the memory. The processor then fetches the instruction that makes up the program from the memory one after another and performs the desire operations.

i) Input Unit:

Computers accept coded information through input units, which read the data. Whenever a key is pressed, the corresponding letter or digit is automatically translated into its corresponding binary code and transmitted over a cable to either the memory or the processor.

Some input devices are

- Joysticks
- Trackballs
- Mouse’s
- Microphones (Capture audio input and it is sampled & it is converted into digital codes for storage and processing).

ii) Memory Unit:

It stores the programs and data. There are 2 types of storage classes

- Primary
- Secondary

Primary Storage:

It is a fast memory that operates at electronic speeds. Programs must be stored in the memory while they are being executed. The memory contains large no of semiconductor storage cells. Each cell carries 1 bit of information. The Cells are processed in a group of fixed size called Words. To provide easy access to any word in a memory, a distinct address is associated with each word location.

Addresses are numbers that identify successive locations. The number of bits in each word is called the word length. The word length ranges from 16 to 64 bits.

There are 3 types of memory. They are

- RAM(Random Access Memory)
- Cache memory
- Main Memory

RAM:

Memory in which any location can be reached in short and fixed amount of time after specifying its address is called RAM.

Time required to access 1 word is called Memory Access Time.

Cache Memory:

The small, fast, RAM units are called Cache. They are tightly coupled with processor to achieve high performance.

Main Memory:

The largest and the slowest unit is called the main memory.

iii) ALU:

Most computer operations are executed in ALU. Consider a example, Suppose 2 numbers located in memory are to be added. They are brought into the processor and the actual addition is carried out by the ALU. The sum may then be stored in the memory or retained in the processor for immediate use. Access time to registers is faster than access time to the fastest cache unit in memory.

iv) Output Unit:

Its function is to send the processed results to the outside world. Eg.Printer

Printers are capable of printing 10000 lines per minute but its speed is comparatively slower than the processor.

v) Control Unit:

The operations of Input unit, output unit, ALU are co-ordinate by the control unit. The control unit is the Nerve centre that sends control signals to other units and senses their states. Data transfers between the processor and the memory are also controlled by the control unit through timing signals.

The operations of computer are:

- The computer accepts information in the form of programs and data through an input unit and stores it in the memory.

- Information stored in the memory is fetched, under program control into an arithmetic and logic unit, where it is processed.
- Processed information leaves the computer through an output unit.
- All activities inside the machine are directed by the control unit.

Basic operational concepts:

The data/operands are stored in memory. The individual instruction is brought from the memory to the processor, which executes the specified operation.

Eg 1: Add LOC A, R1

Instructions are fetched from memory and the operand at LOC A is fetched. It is then added to the contents of R0, the resulting sum is stored in Register R0.

Eg 2: Load LOC A, R1

Transfer the contents of memory location A to the register R1.

Eg 3: Add R1, R0

Add the contents of Register R1 & R0 and places the sum into R0.

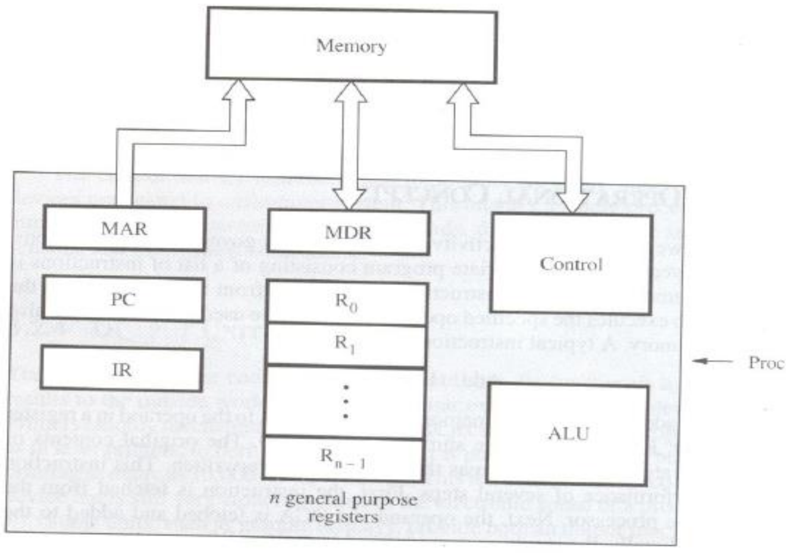


Fig. 1.2: Connection between Processor and Main memory

Instruction Register (IR):

- It holds the instruction that is currently being executed.
- It generates the timing signals.

Program Counter (PC):

It contains the memory address of the next instruction to be fetched for execution.

Memory Address Register (MAR):

It holds the address of the location to be accessed.

Memory Data Register (MDR):

- It contains the data to written into or read out of the address location.

- MAR and MDR facilitates the communication with memory.

Operation Steps:

- The program resides in memory. The execution starts when PC is point to the first instruction of the program.
- MAR read the control signal.
- The Memory loads the address word into MDR. The contents are transferred to Instruction register. The instruction is ready to be decoded & executed.

Interrupt:

Normal execution of the program may be pre-empted if some device requires urgent servicing.
Eg. Monitoring Device in a computer controlled industrial process may detect a dangerous condition.
In order to deal with the situation immediately, the normal execution of the current program may be interrupted & the device raises an interrupt signal. The processor provides the requested service called the Interrupt Service Routine (ISR).
ISR save the internal state of the processor in memory before servicing the interrupt because interrupt may alter the state of the processor. When ISR is completed, the state of the processor is restored and the interrupted program may continue its execution.

Introduction of Computer Organization and Architecture:

Computer architecture refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. **Computer organization** refers to the operational units and their interconnections that realize the architectural specifications. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.
For example, it is an architectural design issue whether a computer will have a multiply instruction. It is an organizational issue whether that instruction will be implemented by a special multiply unit or by a mechanism that makes repeated use of the add unit of the system. The organizational decision may be based on the expected frequency of use of the multiply instruction, the relative speed of the two approaches, and the cost and physical size of a special multiply unit.

Let's check the take away from this lecture

1) Those attributes of the system which is visible to programmer is referred as:

a. Computer organization	**b. Computer architecture**
c. Computer fundamental	d. Computer manufacturing

2) ____ is to fetch the instruction stored in main memory.

a. Output unit	b. Input unit
c. Memory unit	**d. Control unit**

3) Which of the following is the best unit for performing the arithmetic operations?

a. CPU	**b. ALU**
c. MMU	d. PCU

Exercise

Q.1 State the difference between computer organization and computer architecture.

Q.2 Draw the diagram showing the basic units of a computer.

Q.3 Define an interrupt.

Learning from this lecture: Learners will be able to understand the working of different units of computer and also differentiate computer organization and architecture.

Lecture 2

Evolution of Computers, Von Neumann model:

The evolution of computers has been characterized by increasing processor speed, decreasing component size, increasing memory size, and increasing I/O capacity and speed.

The First Generation: Vacuum Tubes

a) ENIAC

The ENIAC (Electronic Numerical Integrator And Computer), designed and constructed at the University of Pennsylvania, was the world's first general purpose electronic digital computer. It was developed for calculating artillery firing tables during World War II to be used by U.S army.

John Mauchly, a professor of electrical engineering at the University of Pennsylvania, and John Eckert, one of his graduate students, proposed to build a general-purpose computer using vacuum tubes. The resulting machine was enormous, weighing 30 tons, occupying 1500 square feet of floor space, and containing more than 18,000 vacuum tubes. When operating, it consumed140 kilowatts of power. It was also substantially faster than any electromechanical computer, capable of 5000 additions per second.

The ENIAC was a decimal rather than a binary machine. That is, numbers were represented in decimal form, and arithmetic was performed in the decimal system. Its memory consisted of 20 “accumulators,” each capable of holding a 10-digitdecimal number. A ring of 10 vacuum tubes represented each digit. At any time, only one vacuum tube was in the ON state, representing one of the 10 digits. The major drawback of the ENIAC was that it had to be programmed manually by setting switches and plugging and unplugging cables.

b) The Von Neumann Machine:

The task of entering and altering programs for the ENIAC was extremely tedious. The programming process could be facilitated if the program could be represented in a form suitable for storing in memory alongside the data. Then, a computer could get its instructions by reading them from memory, and a program could be set or altered by setting the values of a portion of memory.Von Neumann Machine was by developed by John von Neumann in Princeton Institute for Advanced Studies – 1945 to 1952. It is the prototype of all subsequent general-purpose computers and named as IAS computer

Features:

- Stored-Program concept
- Main memory stores both data and instructions
- Arithmetic and logic unit (ALU) capable of operating on binary data
- Control unit, which interprets and executes the instructions in memory
- Input and output (I/O) equipment operated by the control unit

The general structure of the IAS computer / Von Neumann machine is shown below:

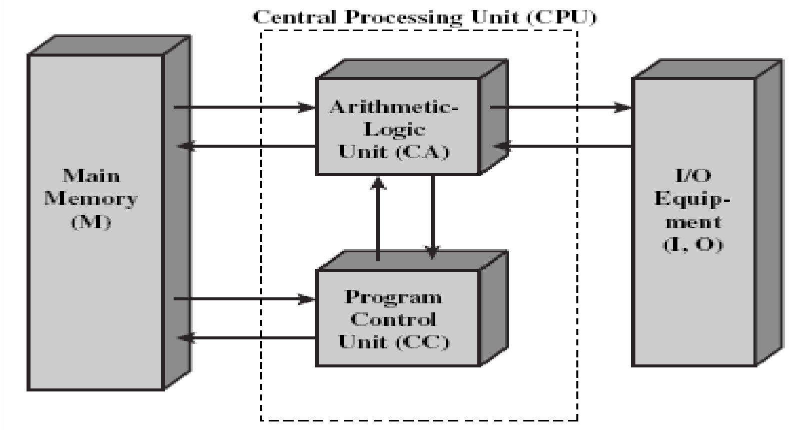
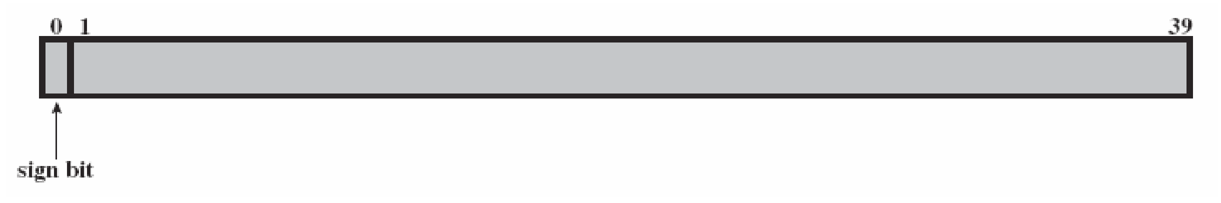

Central Processing Unit (CPU)
Main Memory (M)
Arithmetic-Logic Unit (CA)
Program Control Unit (CC)
I/O Equip-ment (I, O)

Figure 1.3: Von Neumann Machine

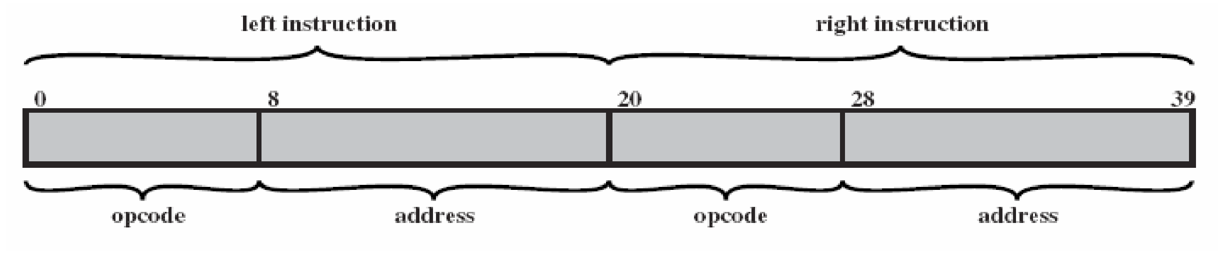
With rare exceptions, all of today's computers have this same general structure and function and are referred to as von Neumann machines.

Expanded Structure of IAS Computer:

The memory of the IAS consists of 1000 storage locations, called *words*, of 40 binary digits (bits) each. Both data and instructions are stored there. Numbers are represented in binary form, and each instruction is a binary code. Each number is represented by a sign bit and a 39-bit value. A word may also contain two 20-bit instructions, with each instruction consisting ofan 8-bit operation code (opcode) specifying the operation to be performed and a12-bit address designating one of the words in memory (numbered from 0 to 999).



(a) Number word



(b) Instruction word

Fig 1.4: IAS Memory Formats

The control unit operates the IAS by fetching instructions from memory and executing them one at a time. The control unit and the ALU contain storage locations, called registers.

• **Memory buffer register (MBR)**: contains a word to be stored in memory, or is used to receive a word
from memory.

• **Memory address register (MAR)**: specifies the address in memory of the word to be written from or
read into the MBR.

• **Instruction register (IR)**: contains the 8-bit opcode instruction being executed.

• **Instruction buffer register (IBR)**: employed to hold temporarily the right-hand instruction from a word in memory.

• **Program Counter (PC)**: contains the address of the next instruction-pair to be fetched from memory.

• **Accumulator (AC) and multiplier quotient (MQ)**: employed to hold temporarily operands and results
of ALU operations.

The expanded structure of IAS Computer is shown below:

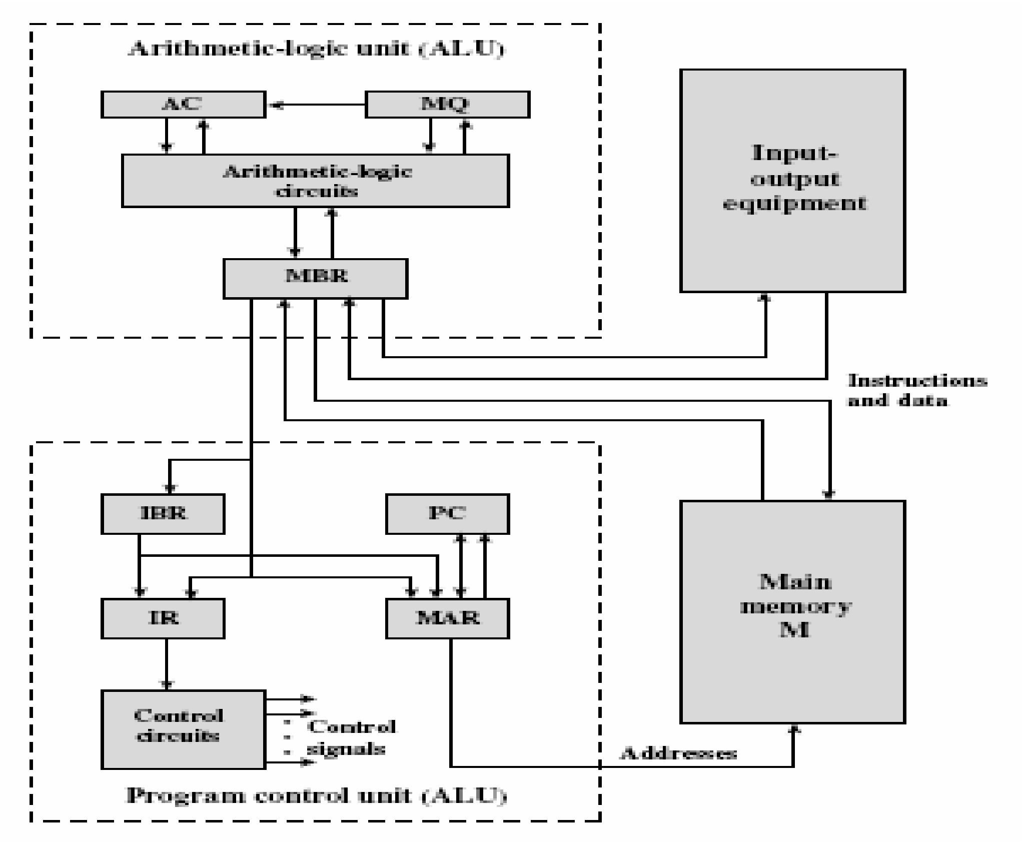


Fig. 1.5: Expanded Structure of IAS Computer

The IAS computer had 21 instructions which can be grouped as follows:

• **Data transfer**: move data between memory and ALU registers or between two ALU registers.

• **Unconditional branch**: used to facilitate repetitive operations.

• **Conditional branch**: branch can be made dependent on a condition, thus allowing decision points.

• **Arithmetic**: operations performed by the ALU.

• **Address modify**: permits address to be computed in the ALU and then inserted into instructions stored in memory.

The Second Generation: Transistors

The first major change in the electronic computer came with the replacement of thevacuum tube by the transistor. The transistor is smaller, cheaper, and dissipates less heat than a vacuum tube but can be used in the same way as a vacuum tube to construct computers. Unlike the vacuum tube, which requires wires, metal plates, a glass capsule, and a vacuum, the transistor is a *solid-state device,* made from silicon.

The transistor was invented at Bell Labs in 1947 and by the 1950s had launched an electronic revolution. The second generation is noteworthy also for the appearance of the Digital Equipment Corporation (DEC). DEC was founded in 1957 and, in that year, delivered its first computer, the PDP-1.This computer and this company began the mini computer phenomenon that would become so prominent in the third generation.

Third Generation: Integrated Circuits

A single, self-contained transistor is called a *discrete component*. Throughout the1950s and early 1960s, electronic equipment was composed largely of discrete components—transistors, resistors, capacitors, and so on. Discrete components were manufactured separately, packaged in their own containers, and soldered or wired together onto masonite-like circuit boards, which were then installed in computers, oscilloscopes, and other electronic equipment. Whenever an electronic device called for a transistor, a little tube of metal containing a pinhead-sized piece of silicon had to be soldered to a circuit board. The entire manufacturing process was expensive.

In 1958 came the achievement that revolutionized electronics and started theera of microelectronics: the invention of the integrated circuit. The basic elements of a digital computer, as we know, must perform storage, movement, processing, and control functions. Only two fundamental types of components are required: gates and memory cells. A gate is a device that implements a simple Boolean or logical function, such as IF *A* AND *B* ARE TRUETHEN *C* IS TRUE (AND gate).The memory cell is a device that can store one bit of data; that is, the device can be in one of two stable states at anytime. By interconnecting large numbers of these fundamental devices, we can construct a computer.

Figure 1.6 depicts the key concepts in an integrated circuit. A thin *wafer* of silicon is divided into a matrix of small areas, each a few millimeters square. The identical circuit pattern is fabricated in each area, and the wafer is broken up into *chips*. Each chip consists of many gates and/or memory cells plus a number of input and output attachment points. This chip is then packaged in housing that protects itand provides pins for attachment to devices beyond the chip. A number of these packages can then be interconnected on a printed circuit board to produce larger and more complex circuits.

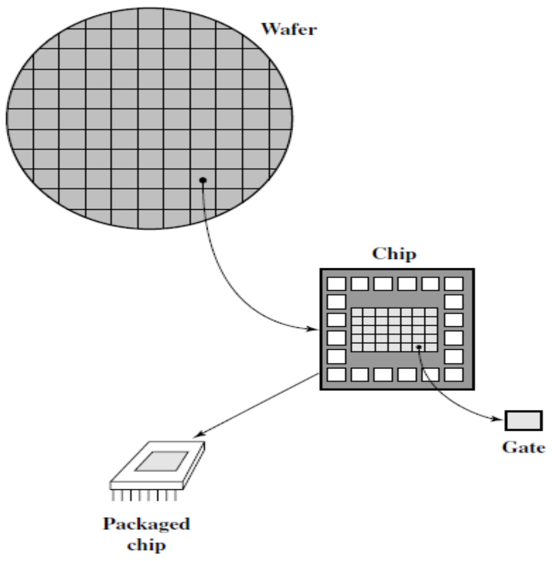


Fig. 1.6: Relationship among Wafer, Chip, Gate

Initially, only a few gates or memory cells could be manufactured and packaged together. These early integrated circuits are referred to as *small-scale integration* (SSI).

Later Generations

Beyond the third generation there is less general agreement on defining generations of computers. There have been a number of later generations, based on advances in integrated circuit technology. With the introduction of large scale integration (LSI), more than 1000 components can be placed on a single integrated circuit chip. Very-large-scale integration (VLSI) achieved more than 10,000components per chip, while current ultra-large-scale integration (ULSI) chips can contain more than one million components.

Semiconductor Memory

The first application of integrated circuit technology to computers was construction of the processor (the control unit and the arithmetic and logic unit) out of integrated circuit chips. Since 1970, semiconductor memory has been through 13 generations: 1K, 4K,16K, 64K, 256K, 1M, 4M, 16M, 64M, 256M, 1G, 4G, and, 16 Gbits on a single chip (1K= 2^{10}, 1M= 2^{20}, 1G= 2^{30}). Each generation has provided four times the storage density of the previous generation, accompanied by declining cost per bit and declining access time.

Just as the density of elements on memory chips has continued to rise, so has the density of elements on processor chips. As time went on, more and more elements were placed on each chip, so that fewer and fewer chips were needed to construct a single computer processor.

Let's check the take away from this lecture

1) Data and instructions are stored in _____.

a. Memory unit	b. Control unit
c. Input unit	d. Output unit

2) The period in which the processor is active is called

a. Processor Time	b. Elapsed Time
c. Response Time	d. Waiting Time

3) First electronic computer was constructed using_________

a) Transistors

b) Vacuum Tubes

c) ICs

d) VLSI

Exercise

Q.1 Explain the structure of IAS computer with diagram.

Q.2 Summarize the technology advancements in third generation of computers from the previous generations.
Q.3 List the registers in Von Neumann model.

Learning from this lecture: Learners will be get knowledge about the evolution of computers and understand the working of Von Neumann model.

Lecture 3

Instruction cycle

An instruction cycle includes the following stages (as shown in fig 1.7):

• **Fetch:** Read the next instruction from memory into the processor.

• **Execute:** Interpret the opcode and perform the indicated operation.

• **Interrupt:** If interrupts are enabled and an interrupt has occurred, save the current process state and service the interrupt.

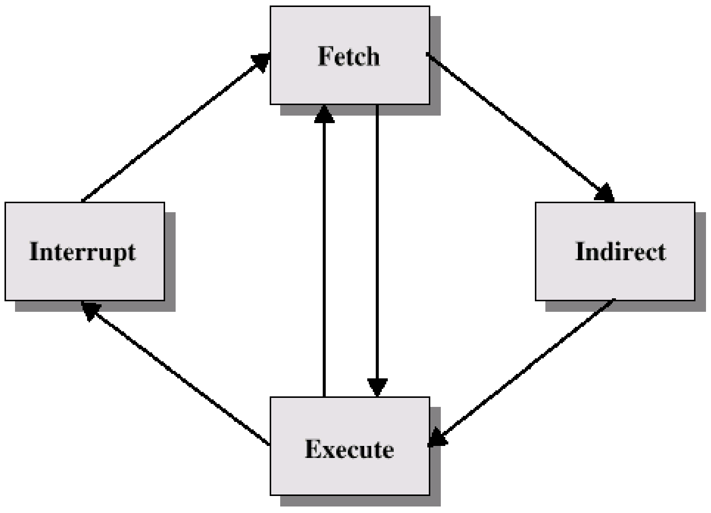
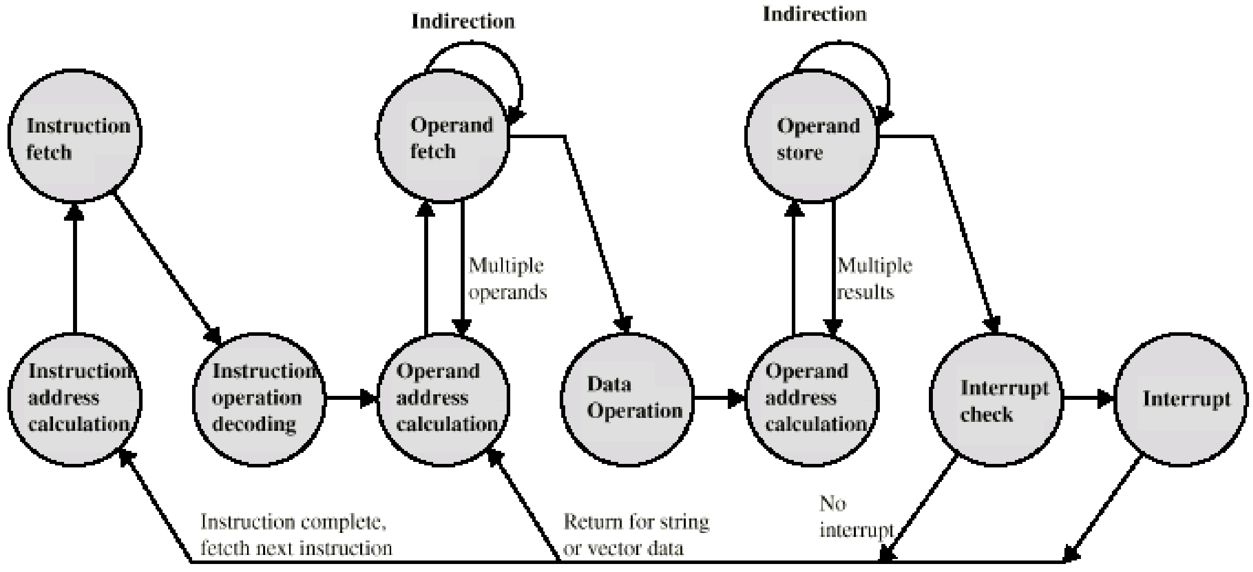


Fig 1.7: The Instruction Cycle

Indirect Cycle

In this case, the execution of an instruction may involve one or more operands in memory, each of which requires a memory access. Further, if indirect addressing is used, then additional memory accesses are required.

Figure 3.4 illustrates more correctly the nature of the instruction cycle. Once an instruction is fetched, its operand specifiers must be identified. Each input operand in memory is then fetched, and this process may require indirect addressing. Register-based operands need not be fetched. Once the opcode is executed, a similar process may be needed to store the result in main memory.



Indirection
Indirection
Instruction fetch
Operand fetch
Operand store
Multiple operands
Multiple results
Instruction address calculation
Instruction operation decoding
Operand address calculation
Data Operation
Operand address calculation
Interrupt check
Interrupt
Instruction complete, fetcth next instruction
Return for string or vector data
No interrupt

Fig 1.8: Instruction Cycle State Diagram

Data Flow (Instruction Fetch)

- Depends on CPU design
- In general:
- Fetch
    - — PC contains address of next instruction
    - — Address moved to MAR
    - — Address placed on address bus
    - — Control unit requests memory read
    - — Result placed on data bus, copied to MBR, then to IR

  Meanwhile PC incremented by 1

Data Flow (Data Fetch)

- IR is examined
- If indirect addressing, indirect cycle is performed
    - — Right most N bits of MBR transferred to MAR
    - — Control unit requests memory read
    - — Result (address of operand) moved to MBR

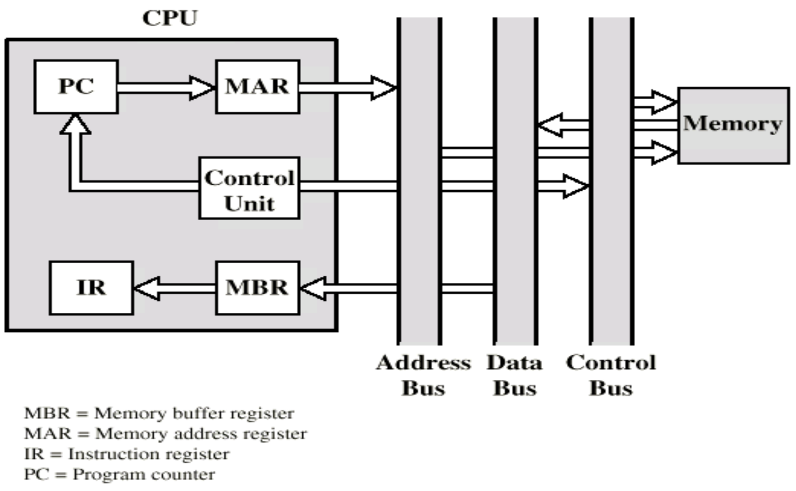

CPU
PC
MAR
Control Unit
IR
MBR
Memory
Address Bus
Data Bus
Control Bus
MBR = Memory buffer register
MAR = Memory address register
IR = Instruction register
PC = Program counter

Fig 1.9: Data Flow Diagram (Fetch)

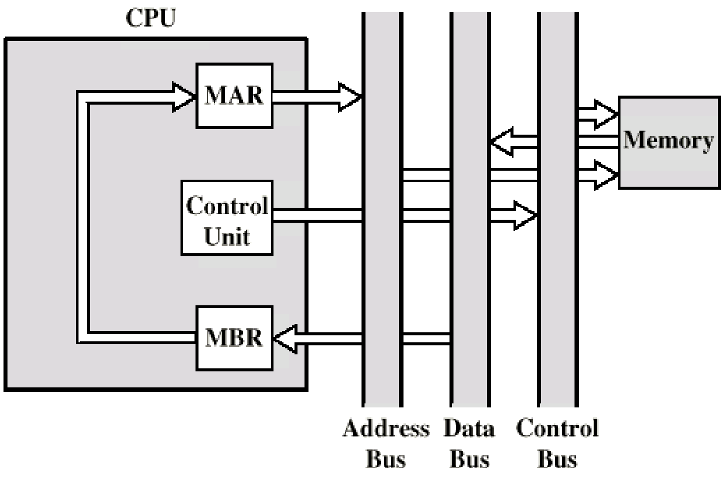


Fig 1.10: Data Flow Diagram (Indirect)

Data Flow (Execute):

- May take many forms
- Depends on instruction being executed
- May include
  - — Memory read/write
  - — Input/output
  - — Register transfers
  - — ALU operations

Data Flow (Interrupt):

- Simple
- Predictable
- Current PC saved to allow resumption after interrupt
- Contents of PC copied to MBR

- Special memory location (e.g. stack pointer) loaded to MAR
- MBR written to memory
- PC loaded with address of interrupt handling routine
- Next instruction (first of interrupt handler) can be fetched

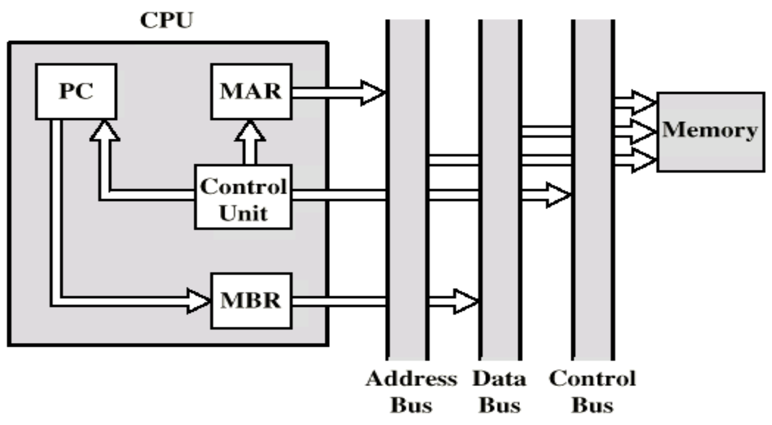


Fig 1.11: Data Flow Diagram (Interrupt)

Let's check the take away from this lecture

1) _____ is a command given to a computer to perform a specified operation on some given data:

a. An instruction

b. Command

c. Code

d. None of these

2) _____is the step during which a new instruction is read from the memory:

a. Decode

b. Fetch

c. Execute

d. None of these

3) ______is the sequence of operations performed by CPU in processing an instruction:

a. Execute cycle

b. Fetch cycle

c. Decode

d. Instruction cycle

Exercise
Q.1 What is an instruction cycle.
Q.2 Draw the state diagram of instruction cycle.
Q.3 State the purpose of MAR and MDR registers.

Learning from this lecture: Learners will be able to understand the operations which happen during instruction cycle.

Lecture 4

Addressing Modes, Instruction Format

Addressing Mode: This is defined as the way to access data from memory, register or immediate data.

i. Register mode:

Both operands are present in register.

e.g MOV R1, R2

ii. Absolute mode or Direct mode:

The address of the location of the operand is given explicitly as a part of instruction.

e.g MOVE A, 2000

iii. Immediate Mode:

The operand is part of instruction.

e.g MOVE A, #20

iv. Indirect Mode:

The EA of the operand is the contents of a register or the main memory location whose address is given explicitly in the instruction.

e.g MOVE A, (R0)

v. Index Mode:

The EA of the operand is generated by adding a constant value (specified in the instruction) to the contents of a register.

e.g. MOVE 20 (R1), R2

vi. Relative Mode:

The EA is determined by the index mode using PC in place of general purpose processor register.The addressing mode is commonly used to specify the target address in branch instruction.

e.g. JNZ BACK

vii. Auto increment Mode:

The EA of the operand is the contents of a register specified in the instruction. After accessing the operand, the contents of this register are incremented to address the next location.

e.g. MOVE (R1), + R0

viii. Auto decrement Mode:

The contents of a register specified in the instruction are decremented & then they are used as an EA to access a memory location.

e.g. MOVE (R1), - R0

Instruction Formats

An instruction consists of bits and these bits are grouped up to make fields.

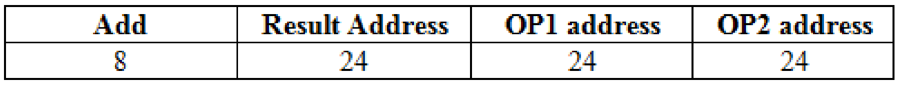
Some fields in instruction format are as follows

1. Opcode which tells about the operation to be performed.
2. Address field designating a memory address or a processor register.
3. Mode field specifying the way the operand or effective address is determined.

Different types of Instruction formats

Some common types are as: Three address instruction format, Two address instruction format, One address instruction format, and Zero address instruction format.

- **Three Address Instruction Format**: This system contains three address fields (address of operand1, address of operand2 and address where result needs to be put). The address of next instruction is held in a CPU register called Program Counter (PC).



Add	Result Address	OP1 address	OP2 address
8	24	24	24

Bits:

Here, the number of bytes required to encode an instruction is 10 bytes.

Each address requires 24 bit = 3 bytes.

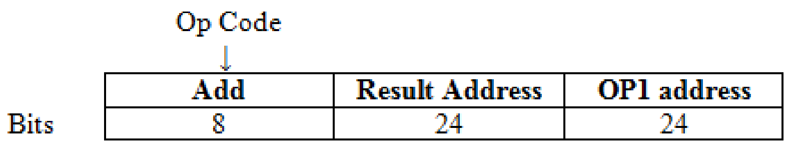
Since, there are three addresses and one opcode field.

Therefore 3 × 3 + 1 = 10 bytes.

The number of memory access required is 7 words.

4 words for instruction fetch, 2 words for operand fetch and 1 word for result to be placed back in memory.

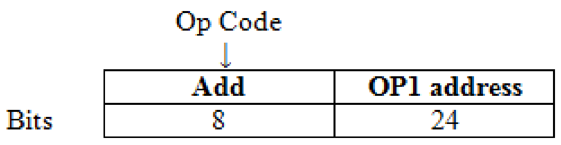
- **Two Address Instruction Format**: In this format, two addresses and an operation field is there. The result is stored in either of the operand address i.e., either in address of first operand or in the address of second operand. CPU register called Program Counter (PC) contains the address of next instruction.



Op Code ↓

	Add	Result Address	OP1 address
Bits	8	24	24

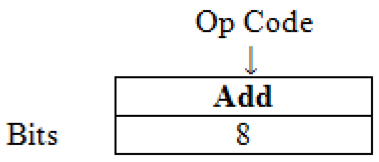
- **One Address Instruction Format:** One address field and an operation field. This address is of the first operand. The second operand and the result are stored in a CPU register called Accumulator Register (AR). Since, a machine has only one accumulator, it needs not be explicitly mentioned in the instruction. A CPU register (i.e., Program Counter (PC) holds the address of next instruction. In this scenario, two extra instructions are required to load and store the accumulator contents.



Number of bits required to encode an instruction is 4 bytes. i.e., each address requires 24 bits = 3 bytes. Since, there are one address and one operation code field, 1* 3 + 1= 4 bytes.

The number of memory access required is 3 words i.e., 2 words for instruction fetch +1 word for code for operand fetch.

- **Zero Address Instruction Format:** Stack is included in the CPU for performing arithmetic and logic instructions with no addresses. The operands are pushed onto the stack from memory and ALU operations are implicitly performed on the top elements of the stack. The address of the next instruction is held in a CPU register called program counter.



e.g., Add

Top of stack ← Top of stack + second top of stack.

Let's check the take away from this lecture

1) Two important fields of an instruction are:

a. Opcode

b. Operand

c. Only a

d. Both a & b

2) The instruction, Add #45, R1 does ________

a) Adds the value of 45 to the address of R1 and stores 45 in that address

b) Adds 45 to the value of R1 and stores it in R1

c) Finds the memory location 45 and adds that content to that of R1

d) None of the mentioned

3) In the case of, Zero-address instruction method the operands are stored in ______

a) Registers

b) Accumulators

c) Push down stack

d) Cache

Exercise
Q.1 Define addressing mode.
Q.2 Give example for register indirect mode.
Q.3 List the different instruction formats.

Learning from this lecture: Learners will be able to understand the concept of addressing modes and different instruction formats.

Lecture 5

Introduction to System buses, Multi-bus organization

The bus that connects major computer components/ module (CPU, memory, I/O) is called system buses. Basically it is a set of conductors that connects the CPU, memory, I/O and it is separated into three functional groups:

- Data bus- the data bus consists of 8, 16, 32 or more parallel signal lines. The data bus lines are bi-directional. It means CPU can read and write the data both. This bus is connected in parallel to all peripherals.
- Address bus- It is an unidirectional bus. The address bus consists of 16, 20, 24 or more parallel lines. On this parallel line the CPU sends out the address of memory location or I/O port that is to be written to or read from.
- Control bus- The control buses regulates the activity on the bus. CPU sends the signal on the control bus to enable the output of addressed memory device or port device.

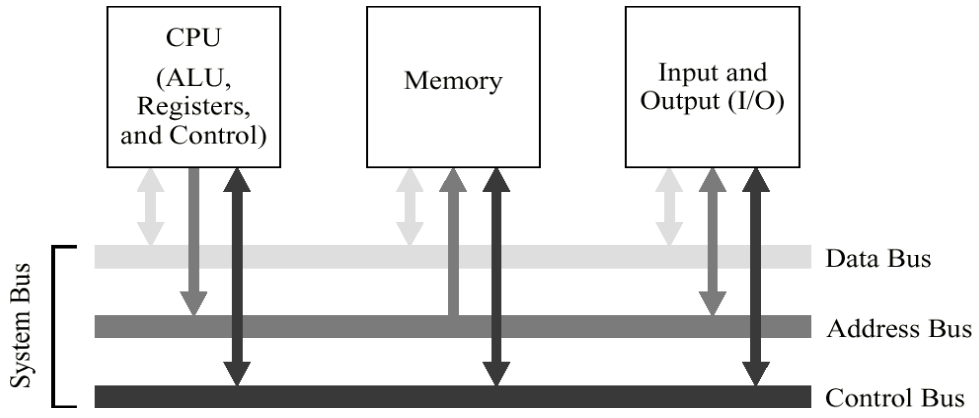


Fig 1.12: System Bus Model

Bus Structure:

A group of lines that serves as the connection path to several devices is called a Bus. A Bus may be lines or wires or one bit per line. The lines carry data or address or control signal.

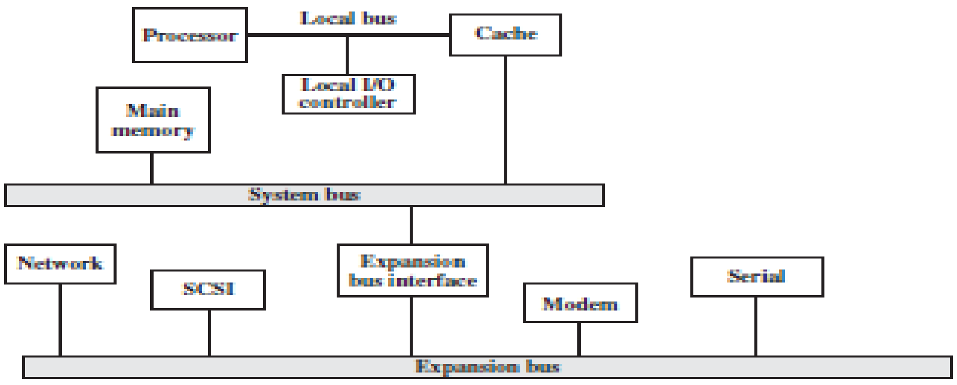


Fig 1.13: Single-bus structure (Traditional bus architecture)

There are 2 types of Bus structures. They are

- Single Bus Structure
- Multiple Bus Structure

i) Single Bus Structure:

- It allows only one transfer at a time.
- It costs low.
- It is flexible for attaching peripheral devices.
- Its Performance is low.

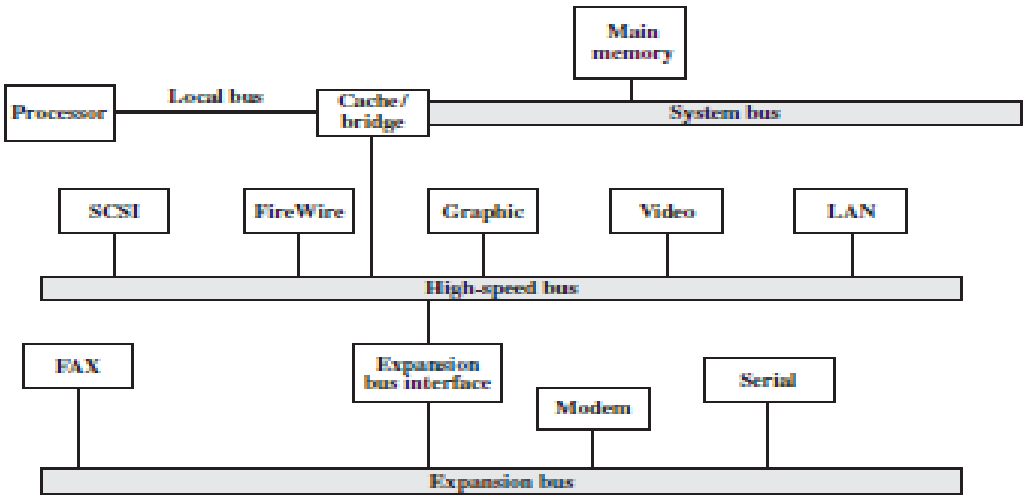


Fig 1.14: Multiple-bus structure

If a great numbers of devices are connected to the bus, performance will suffer. There are two main causes:

1. In general, the more devices attached to the bus, the greater the bus length and hence the greater the propagation delay. This delay determines the time it takes for devices to coordinate the use of the bus. When control of the bus passes from one device to another frequently, these propagation delays can noticeably affect performance.

2. The bus may become a bottleneck as the aggregate data transfer demand approaches the capacity of the bus. This problem can be countered to some extent by increasing the data rate that the bus can carry

and by using wider buses (e.g., increasing the data bus from 32 to 64 bits). However, because the data rates generated by attached devices (e.g., graphics and video controllers, network interfaces) are growing rapidly, the single-bus structure cannot be used.

ii) Multiple Bus Structure:

- It allows two or more transfer at a time.
- It costs high.
- It provides concurrency in operation.
- Its Performance is high.

The use of a cache structure insulates the processor from a requirement to access main memory frequently. Hence, main memory can be moved off the local bus onto a system bus. In this way, I/O transfers to and from the main memory across the system bus do not interfere with the processor's activity.

It is possible to connect I/O controllers directly onto the system bus. A more efficient solution is to make use of one or more expansion buses for this purpose. An expansion bus interface buffers data transfers between the system bus and the I/O controllers on the expansion bus.

Bus Arbitration

A bus cannot be used by more than one device at one time due to electrical properties of the devices (the outputs will be connected together and will cause damage). The solution is arbitration. i.e. A device that starts a read/write operation is called a master device. The device that it "talks" to is called the slave device.

There is always a bus arbitration period preceding a bus utilization period. To speed up the computer, the two operations (bus arbitration and bus utilization) can be and are always performed in parallel. While the current bus utilization period is going on, devices that want to use the bus in the next cycle can decide among themselves who will get to use the system bus in the next period. Therefore, bus arbitration periods and bus utilization periods usually overlap.

Three types of bus arbitration:

1. Daisy Chain
2. Independent Bus Requests and Grant
3. Polling

<u>Daisy Chain</u>

- A method for a centralized bus arbitration process
- The bus control passes from one bus master to the next one, then to the next and so on
- Bus control passes from unit *U*0 to *U*1, then to *U*2, then *U*3, and so on
- U0 has highest priority, U1 next, and so on

Independent Bus Requests and Grant

- The bus control passes from one bus master to another only through the centralized bus controller
- Assume *n* units can be granted bus master status by a centralized processor as bus controller after listening to its request

Polling

- The bus control passes from one processor (bus controller) to another only through the centralized bus controller, but only when the controller sends poll count bits, which correspond to the unit number.
- Assume *n* units can be granted bus master status by a centralized processor.

Let's check the take away from this lecture

1) ------------------- is defined as the communication pathway connecting two or more devices

a. CPU	b. Memory
c. Bus	d. ALU

2) ---------------- provide a path for moving data between system modules

a. Data lines	b. Address lines
b. Control lines	d. None of the above

3) The internal components of the processor are connected by _______

a) Processor intra-connectivity circuitry

b) Processor bus

c) Memory bus

d) Rambus

Exercise
Q.1 Define Bus.
Q.2 Draw the traditional bus architecture and explain.
Q.3 Explain the need for multi-bus organization.

Learning from this lecture: Learners will be able to understand the use of buses in computer.

Conclusion

The study of Computer Organization and Architecture helps to understand the functions of all computer modules and how they communicate through buses. And also about the different processes that happen during an instruction execution.

Short Answer Questions:

1. What are the different generations of computer?

Ans:

Generation 1: Vacuum tube

Generation 2: Transistor

Generation 3: Small- and medium-scale integration

Generation 4: Large-scale integration

Generation 5: Very-large-scale integration

2. Explain (I) ENIAC (II) UNIVAC.

Ans:

ENIAC (Electronic Numerical Integrator and Computer) was the world's first general-purpose electronic digital computer. I was a decimal rather than a binary machine. Its memory consisted of 20 accumulators, capable of holding a 10-digit decimal number. A ring of 10 vacuum tubes represented each digit.

Drawback of ENIAC: Had to be programmed manually by setting switches and plugging and unplugging cables.

UNIVAC: UNIVAC I (Universal Automatic Computer) was commissioned by the Bureau of the Census for the 1950 calculations. Moreover, it was the first commercial computer. It was intended for both scientific and commercial applications. UNIVAC II was then delivered in the late 1950s and had greater memory capacity and higher performance than the UNIVAC I

3. State the difference between computer organization and computer Architecture

Ans:

Computer Architecture: Refers to those attributes of the system which is visible to programmer.

Computer Organization: Refers to the operational units and their interconnections.

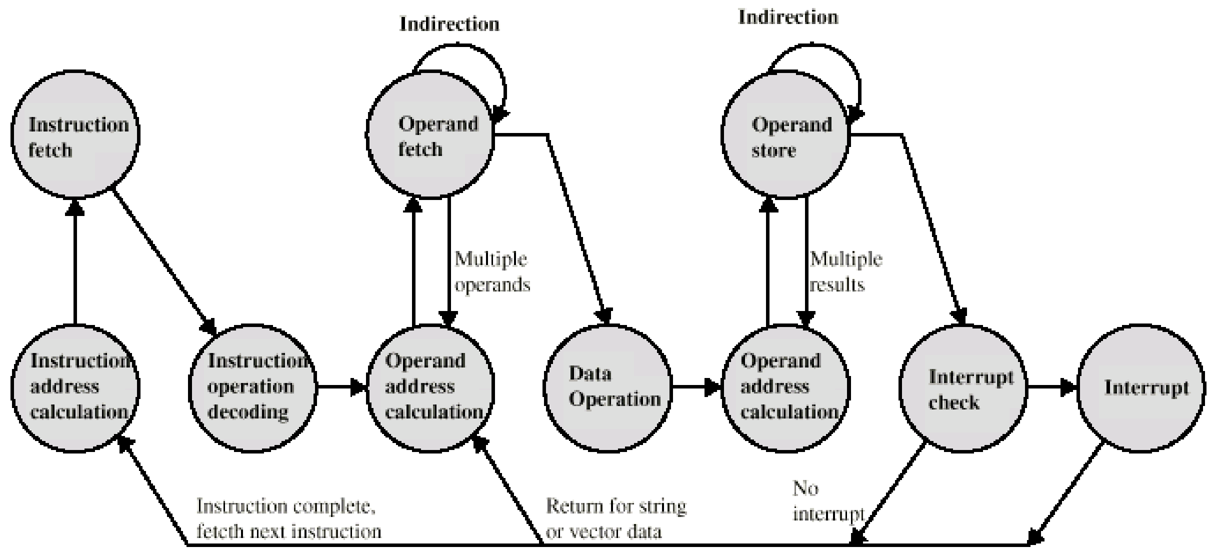
4. Define Bus

Ans.

A bus is a subsystem that is used to connect computer components and transfer data between them. For example, an internal bus connects computer internals to the motherboard.

5. Draw the state diagram of instruction cycle

Ans.


Indirection
Indirection
Instruction fetch
Operand fetch
Operand store
Multiple operands
Multiple results
Instruction address calculation
Instruction operation decoding
Operand address calculation
Data Operation
Operand address calculation
Interrupt check
Interrupt
Instruction complete, fetcth next instruction
Return for string or vector data
No interrupt

6. List the different addressing modes

Ans.

i. Register mode

ii. Absolute mode or Direct mode

iii. Immediate Mode

iv. Indirect Mode

v. Index Mode

vi. Relative Mode

vii. Auto increment Mode

viii. Auto decrement Mode

7. Give examples for one-address and zero-address instruction format.

Ans.

One Address Instruction Format:

Op Code

↓

	Add	**OP1 address**
Bits	8	24

Zero Address Instruction Format:

Op Code

↓

	Add
Bits	8

Long Answer Questions:

1. Write in detail the functional units of a computer.

Ans: Refer the section ‘Description of the functional units’ under lecture 1

2. Write detailed notes on bus interconnection scheme.

Ans: Refer the contents under lecture 5

3. Explain in detail the single-bus structure and multiple-bus structure.

Ans: Refer the contents under lecture 5

4. Draw and explain Von Neumann model?

Ans: Refer the section 'Von Neumann model' under lecture 2

5. Explain why multiple-bus hierarchies are required. Explain traditional bus architecture?

Ans: Refer the contents under lecture 5

6. Explain instruction cycle with state diagram

Ans. Refer the contents under lecture 3

7. Explain the different addressing modes.

Ans. Refer the section 'Addressing modes' under lecture 4

Set of Questions for FA/IA/ESE

1. Explain accumulator based computer with different block of it. (10M)

2. State the difference between computer organization and computer Architecture (5M)

3. Explain general organization of CPU. State the function of following registers: (10M)

   a)MAR

   b)MDR

   c)IR

   d)PC

   e)SP

4. Explain Von Neumann model in detail. (10M)

5. Explain different bus arbitration schemes with suitable diagrams (May 2011 (10M), Dec 2011 (10M))

6. Define the following terms: (10M)

i) Computer Organization

ii) Computer Architecture

iii) MDR

iv) PC

v) SP

7. What is stored program concept in digital computer? (2 M)

8. Explain role of different registers like IR, PC, SP, AC, MAR and MDR used in Von Neumann model. (5 M)

9. What is bus arbitration? Explain any two techniques of bus arbitration. (Dec 2015 (10 M), MAY 2016 (10 M))

10. Draw the state diagram of instruction cycle and explain. (10 M)

References:

1.William Stallings, “Computer Organization and Architecture: Designing for Performance”, Eighth Edition, Pearson.

2. Carl Hamacher, Zvonko Vrasenic and Safwat Zaky, “Computer Organization”, Fifth edition, Tata McGraw-Hill.

Self-assessment

Q.1) What is Computer Organization and Architecture? Describe the difference between them.

Q.2) Explain the functional units of a computer.

Q.3) Explain the structure of IAS computer.

Q.4) Explain the different steps of instruction cycle.

Q.5) Explain the different addressing modes.

Q.6) Describe about the Bus Structure.

Self-evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice
1.	Do you understand the difference between computer organization and architecture?	○ Yes ○ No
2.	Do you understand the structure of Von Neumann model?	○ Yes ○ No
3.	Do you understand the steps involved in instruction cycle?	○ Yes ○ No
4.	Do you understand the concept of addressing modes and instruction format?	○ Yes ○ No

5.	Do you understand the need for buses for the communication between different modules of the computer?	○ Yes ○ No
6.	Do you understand module 1 ?	○ Yes, Completely. ○ Partialy. ○ No, Not at all.