
UNIT 1 THE BASIC COMPUTER

Structure	Page Nos.
1.0 Introduction	5
1.1 Objectives	5
1.2 The von Neumann Architecture	5
1.3 Instruction Execution: An Example	9
1.4 Instruction Cycle	12
1.4.1 Interrupts	
1.4.2 Interrupts and Instruction Cycle	
1.5 Computers: Then and Now	18
1.5.1 The Beginning	
1.5.2 First Generation Computers	
1.5.3 Second Generation Computers	
1.5.4 Third Generation Computers	
1.5.5 Later Generations	
1.6 Summary	29
1.7 Solutions/Answers	29

1.0 INTRODUCTION

The use of Information Technology (IT) is well recognised. IT has become a must for the survival of all business houses with the growing information technology trends. Computer is the main component of an Information Technology network. Today, computer technology has permeated every sphere of existence of modern man. From railway reservations to medical diagnosis, from TV programmes to satellite launching, from matchmaking to criminal catching — everywhere we witness the elegance, sophistication and efficiency possible only with the help of computers.

In this unit, you will be introduced to one of the important computer system structures: the von Neumann Architecture. In addition, you will be introduced to the concepts of a simple model of Instruction execution. This model will be enhanced in the later blocks of this course. More details on these terms can be obtained from further reading. We have also discussed about the main developments during the various periods of computer history. Finally, we will discuss about the basic components of microprocessors and their uses.

1.1 OBJECTIVES

After going through this unit you will be able to:

- define the logical structure of the computer;
 - define the instruction cycle;
 - define the concept or Interrupt;
 - discuss the basic features of computers; and
 - define the various components of a modern computer and their usage.
-

1.2 THE VON NEUMANN ARCHITECTURE

The von Neumann architecture was the first major proposed structure for a general-purpose computer. However, before describing the main components of von Neumann

architecture, let us first define the term 'computer' as this will help us in discussing about von Neumann architecture in logical detail.

Computer is defined in the Oxford dictionary as "An automatic electronic apparatus for making calculations or controlling operations that are expressible in numerical or logical terms".

The definition clearly categorises computer as an electronic apparatus although the first computers were mechanical and electro-mechanical apparatuses. The definition also points towards the two major areas of computer applications viz., data processing's and computer assisted controls/operations. Another important aspect of the definition is the fact that the computer can perform only those operations/calculations, which can be expressed in Logical or Numerical terms.

Some of the basic questions that arise from above definition are:

How are the data processing and control operations performed by an electronic device like the computer?

Well, electronic components are used for creating basic logic circuits that are used to perform calculations. These components are further discussed in the later units. However, for the present discussion, it would be sufficient to say that there must be a certain unit that will perform the task of data processing and control.

What is the basic function performed by a computer? The basic function performed by a computer is the execution of the program. A program is a sequence of instructions, which operates on data, to perform certain tasks such as finding a prime number. The computer controls the execution of the program.

What is data in computers? In modern digital computers data is represented in binary form by using two symbols 0 and 1. These are called **binary digits** or bits. But the data which we deal with consists of numeric data and characters such as decimal digits 0 to 9, alphabets A to Z, arithmetic operators (e.g. +, -, etc.), relations operators (e.g. =, >, etc.), and many other special characters (e.g. :, @, {, }, etc.). Therefore, there has to be a mechanism for data representation. Old computers use eight bits to represent a character. This allows up to $2^8 = 256$ different items to be represented uniquely. This collection of eight bits is called a byte. Thus, one byte is used to represent one character internally. Most computers use two bytes or four bytes to represent numbers (positive and negative) internally. The data also includes the operational data such as integer, decimal number etc. We will discuss more about data representation in the next unit.

Thus, the prime task of a computer is to perform instruction execution. The key questions, which can be asked in this respect, are: (a) how are the instructions supplied to the computer? and (b) how are the instructions interpreted and executed?

Let us answer the second question first. All computers have a Unit that performs the arithmetic and logical functions. This Unit is referred to as the Arithmetic Logic Unit (ALU). But how will the computer determine what operation is to be performed by ALU or in other words who will interpret the operation that is to be performed by ALU?

This interpretation is done by the Control Unit of the computer. The control unit accepts the binary form of instruction and interprets the instruction to generate control signals. These control signals then direct the ALU to perform a specified arithmetic or logic function on the data. Therefore, by changing the control signal the desired function can be performed on data. Or conversely, the operations that need to be performed on the data can be obtained by providing a set of control signals. Thus, for a new operation one only needs to change the set of control signals.

The unit that interprets a code (a machine instruction) to generate respective control signals is termed as Control Unit (CU). A program now consists of a sequence of codes. Each code is, in effect, an instruction, for the computer. The hardware

interprets each of these instructions and generates respective control signals such that the desired operation is performed on the data.

The Arithmetic Logic Unit (ALU) and the Control Unit (CU) together are termed as the Central Processing Unit (CPU). The CPU is the most important component of a computer's hardware.

All these arithmetic and logical Operations are performed in the CPU in special storage areas called registers. The size of the register is one of the important considerations in determining the processing capabilities of the CPU. Register size refers to the amount of information that can be held in a register at a time for processing. The larger the register size, the faster may be the speed of processing.

But, how can the instructions and data be put into the computers? The instructions and data to a computer are supplied by external environment; it implies that input devices are needed in the computer. The main responsibility of input devices will be to put the data in the form of signals that can be recognised by the system. Similarly, we need another component, which will report the results in proper format. This component is called output device. These components together are referred to as input/output (I/O) devices.

In addition, to transfer the information, the computer system internally needs the system interconnections. At present we will not discuss about Input/Output devices and system interconnections in details, except the information that most common input/output devices are keyboard, monitor and printer, and the most common interconnection structure is the Bus structure. These concepts are detailed in the later blocks.

Input devices can bring instructions or data only sequentially, however, a program may not be executed sequentially as jump, looping, decision-making instructions are normally encountered in programming. In addition, more than one data element may be required at a time. Therefore, a temporary storage area is needed in a computer to store temporarily the instructions and the data. This component is referred to as memory.

The memory unit stores all the information in a group of memory cells such as a group of 8 binary digits (that is a byte) or 16 bits or 32 bits etc. These groups of memory cells or bits are called memory locations. Each memory location has a unique address and can be addressed independently. The contents of the desired memory locations are provided to the CPU by referring to the address of the memory location. The amount of information that can be held in the main memory is known as memory capacity. The capacity of the main memory is measured in Mega Bytes (MB) or Giga Bytes (GB). One-kilo byte stands for 2^{10} bytes, which are 1024 bytes (or approximately 1000 bytes). A Mega byte stands for 2^{20} bytes, which is approximately a little over one million bytes, a giga byte is 2^{30} bytes.

Let us now define the key features of von Neumann Architecture:

- The most basic function performed by a computer is the execution of a program, which involves:
 - ❑ the execution of an instruction, which supplies the information about an operation, and
 - ❑ the data on which the operation is to be performed.

The control unit (CU) interprets each of these instructions and generates respective control signals.

- The Arithmetic Logic Unit (ALU) performs the arithmetic and logical Operations in special storage areas called registers as per the instructions of control unit. The size of the register is one of the important considerations in determining the processing capabilities of the CPU. Register size refers to the

amount of information that can be held in a register at a time for processing. The larger the register size, the faster may be the speed of processing.

- An Input/ Output system involving I/O devices allows data input and reporting of the results in proper form and format. For transfer of information a computer system internally needs the system interconnections. One such interconnection structure is BUS interconnection.
- Main Memory is needed in a computer to store instructions and the data at the time of Program execution. Memory to CPU is an important data transfer path. The amount of information, which can be transferred between CPU and memory, depends on the size of BUS connecting the two.
- It was pointed out by von-Neumann that the same memory can be used for Storing data and instructions. In such a case the data can be treated as data on which processing can be performed, while instructions can be treated as data, which can be used for the generation of control signals.
- The von Neumann machine uses **stored program concept**, i.e., the program and data are stored in the same memory unit for execution. The computers prior to this idea used to store programs and data on separate memories. Entering and modifying these programs was very difficult as they were entered manually by setting switches, plugging, and unplugging.
- Execution of instructions in von Neumann machine is carried out in a sequential fashion (unless explicitly altered by the program itself) from one instruction to the next.

Figure 1 shows the basic structure of a conventional von Neumann machine

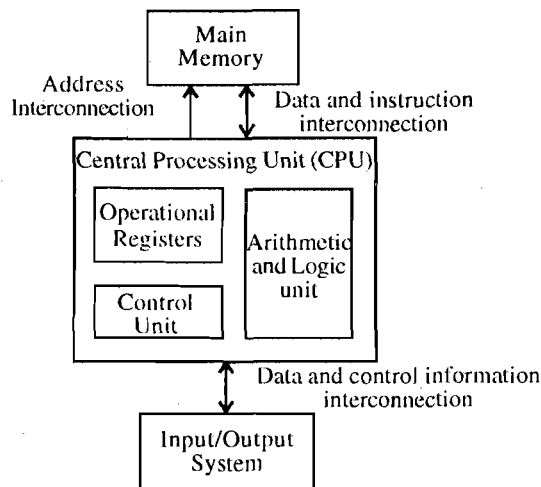


Figure 1: Structure of a Computer

A von Neumann machine has only a single path between the main memory and control unit (CU). This feature/constraint is referred to as von Neumann bottleneck. Several other architectures have been suggested for modern computers. You can know about non von Neumann architectures in further readings.

Check Your Progress 1

1) State True or False

- | | | |
|----|--|---|
| a) | A byte is equal to 8 bits and can represent a character internally. | <div style="border: 1px solid black; width: 30px; height: 20px; float: right; text-align: center; line-height: 20px;">T/F</div> |
| b) | von Neumann architecture specifies different memory for data and instructions. The memory, which stores data, is called data memory and the memory, which stores instructions, is called instruction memory. | <div style="border: 1px solid black; width: 30px; height: 20px; float: right;"></div> |
| c) | In von Neumann architecture each bit of memory can be accessed independently. | <div style="border: 1px solid black; width: 30px; height: 20px; float: right;"></div> |
| d) | A program is a sequence of instructions designed for achieving a task/goal. | <div style="border: 1px solid black; width: 30px; height: 20px; float: right;"></div> |

- e) One MB is equal to 1024KB. ☐
- f) von Neumann machine has one path between memory and control unit. ☐
- This is the bottleneck of von Neumann machines.

2) What is von Neumann Architecture?

.....

.....

.....

3) Why is memory needed in a computer?

.....

.....

.....

1.3 INSTRUCTION EXECUTION: AN EXAMPLE

After discussing about the basic structure of the computer, let us now try to answer the basic question: "How does the Computer execute a Program?" Let us explain this with the help of an example from higher level language domain.

Problem: Write a program to add two numbers.

A sample C program (Assuming two fixed values of numbers as a = 5 and b = 2)

```
1.  #include <stdio.h>
2.  main ()
3.  {
4.  int a=5, b=2, c;
5.  c= a+b;
6.  printf ("The added value is: %d", c);
7.  }
```

The program at line 4 declares variables that will be equivalent to 3 memory locations namely a, b and c. At line 5 these variables are added and at line 6 the value of c is printed.

But, how will these instructions be executed by CPU?

First you need to compile this program to convert it to machine language. But how will the machine instructions look like?

Let us assume a hypothetical instruction set of a machines of a size of 16 binary digits (bits) instructions and data. Each instruction of the machine consists of two components: (a) Operation code that specifies the operation that is to be performed by the instruction, and (b) Address of the operand in memory on which the given operation is to be performed.

Let us further assume that the size of operation code is assumed to be of six bits; therefore, rest 10 bits are for the address of the operand. Also the memory word size is assumed to be of 16 bits. Figure 2 shows the instruction and data formats for this machine. However, to simplify our discussion, let us present the operation code using Pnemonics like LOAD, ADD, STORE and decimal values of operand addresses and signed decimal values for data.

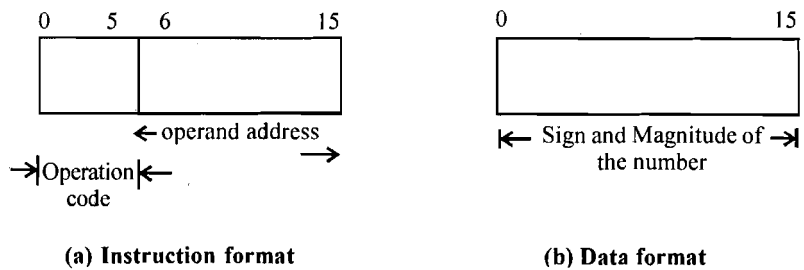


Figure 2: Instruction and data format of an assumed machine

The instruction execution is performed in the CPU registers. But before we define the process of instruction execution let us first give details on Registers, the temporary storage location in CPU for program execution. Let us define the minimum set of registers required for von Neumann machines:

Accumulator Register (AC): This register is used to store data temporarily for computation by ALU. AC is considered to contain one of the operands. The result of computation by ALU is also stored back to AC. It implies that the operand value is over-written by the result.

Memory Address Register (MAR): It specifies the address of memory location from which data or instruction is to be accessed (read operation) or to which the data is to be stored (write operation). Refer to figure 3.

Memory Buffer Register (MBR): It is a register, which contains the data to be written in the memory (write operation) or it receives the data from the memory (read operation).

Program Counter (PC): It keeps track of the instruction that is to be executed next, that is, after the execution of an on-going instruction.

Instruction Register (IR): Here the instructions are loaded prior to execution.

Comments on figure 3 are as follows:

- All representation are in decimals. (In actual machines the representations are in Binary).
- The Number of Memory Locations = 16
- Size of each memory location = 16 bits = 2 Bytes (Compare with contemporary machines word size of 16,32, 64 bits)
- Thus, size of this sample memory = 16 words (Compare it with actual memory size, which is 128 MB, 256 MB, 512 MB, or more).
- In the diagram MAR is pointing to location 10.
- The last operation performed was “read memory location 10” which is 65 in this. Thus, the contents of MBR is also 65.

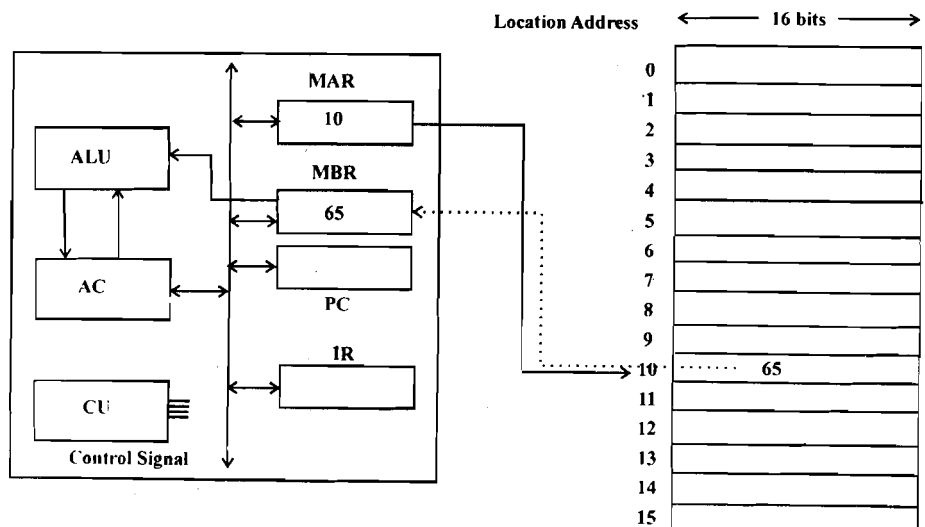


Figure 3: CPU registers and their functions

The role of PC and IR will be explained later.

Now let us define several operation codes required for this machine, so that we can translate the High level language instructions to assembly/machine instructions.

Operation Code		Definition/Operation (please note that the address part in the Instruction format specifies the Location of the Operand on whom operation is to be performed)
LOAD	as	"Load the accumulator with the content of memory"
STORE	as	"Store the current value of Accumulator in the memory"
ADD	as	"Add the value from memory to the Accumulator"

A sample machine instructions for the assumed system for line 5 that is $c = a + b$ in the program would be:

LOAD	A	; Load the contents of memory location A to Accumulator register
ADD	B	; Add the contents of B to contents of Accumulator and store result in Accumulator.
STORE	C	; Store the content into location C

Please note that a simple one line statement in 'C' program has been translated to three machine instructions as above. Please also note that these translated instructions are machine dependent.

Now, how will these instructions execute?

Let us assume that the above machine instructions are stored in three consecutive memory locations 1, 2 and 3 and the PC contains a value (1), which in turn is address of first of these instructions. (Please refer to figure 4 (a)).

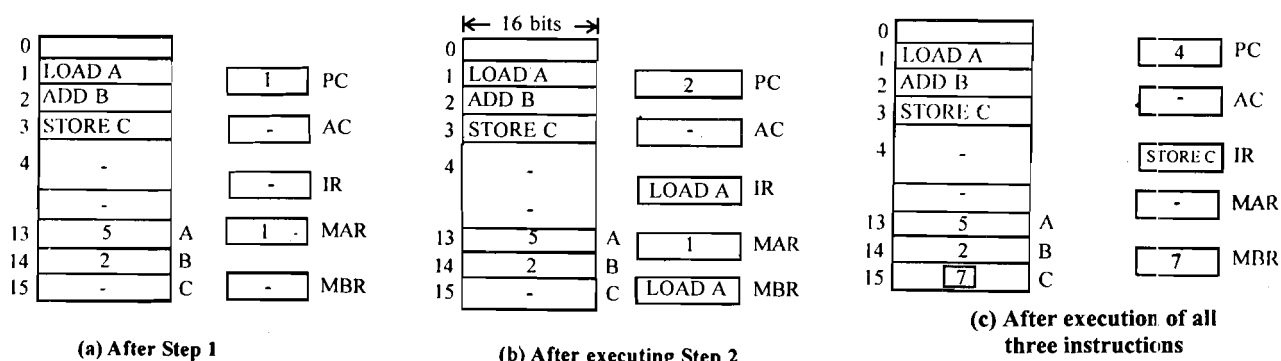


Figure 4: Memory and Registers Content on execution of the three given Consecutive Instructions (All notations in Decimals)

Then the execution of the instructions will be as follows:

Fetch First Instruction into CPU:

Step 1: Find or calculate the address of the first instruction in memory: In this machine example, the next instruction address is contained in PC register. It contains 1, which is the address of first instruction to be executed. (figure 4 a).

Step 2: Bring the binary instruction to IR register. This step requires:

- Passing the content of PC to Memory Address Registers so that the instruction pointed to by PC is fetched. That is location 1's content is fetched.
- CPU issues "Memory read" operation, thus, brings contents of location pointed by MAR (1 in this case) to the MBR register.
- Content of MBR is transferred to IR. In addition PC is incremented to point to next instruction in sequence (2 in this case).

Execute the Instruction

- Step 3: The IR has the instruction LOAD A, which is decoded as “Load the content of address A in the accumulator register”.
- Step 4: The address of operand that is 13, that is A, is transferred to MAR register.
- Step 5: The content of memory location (specified by MAR that is location 13) is transferred to MBR.
- Step 6: The content of MBR is transferred to Accumulator Register.

Thus, the accumulator register is loaded with the content of location A, which is 5. Now the instruction 1 execution is complete, and the next instruction that is 2 (indicated by PC) is fetched and PC is incremented to 3. This instruction is ADD B, which instruct CPU to add the contents of memory location B to the accumulator. On execution of this instruction the accumulator will contain the sum of its earlier value that is A and the value stored in memory location B.

On execution of the instruction at memory location 3, PC becomes 4; the accumulator results are stored in location C, that is 15, and IR still contains the third instruction. This state is shown in figure 4 (C).

Please note that the execution of the instructions in the above example is quite simple and requires only data transfer and data processing operations in each instruction. Also these instructions require one memory reference during its execution.

Some of the problems/limitations of the example shown above are?

1. The size of memory shown in 16 words, whereas, the instruction is capable of addressing $2^{10} = 1 \text{ K}$ words of Memory. But why 2^{10} , because 10 bits are reserved for address in the machine instruction format.
2. The instructions shown are sequential in nature, however, a machine instruction can also be a branch instruction that causes change in the sequence of instruction execution.
3. When does the CPU stop executing a program? A program execution is normally completed at the end of a program or it can be terminated due to an error in program execution or sometimes all running programs will be terminated due to catastrophic failures such as power failure.

1.4 INSTRUCTION CYCLE

We have discussed the instruction execution in the previous section, now let us discuss more about various types of instruction execution.

What are the various types of operations that may be required by computer for execution of instruction? The following are the possible steps:

S.No.	Step to be performed	How is it done	Who does it
1	Calculate the address of next instruction to be executed	The Program Counter (PC) register stores the address of next instruction.	Control Unit (CU).
2.	Get the instruction in the CPU register	The memory is accessed and the desired instruction is brought to register (IR) in CPU	Memory Read operation is done. Size of instruction is important. In addition, PC is incremented to point to next instruction in sequence.
3.	Decode the instruction	The control Unit issues necessary control signals	CU.

4.	Evaluate the operand address	CPU evaluates the address based on the addressing mode specified.	CPU under the control of CU
5.	Fetch the operand	The memory is accessed and the desired operands brought into the CPU Registers	Memory Read
Repeat steps 4 and 5 if instruction has more than one operands.			
6.	Perform the operation as decoded in steps 3.	The ALU does evaluation of arithmetic or logic, instruction or the transfer of control operations.	ALU/CU
7.	Store the results in memory	The value is written to desired memory location	Memory write

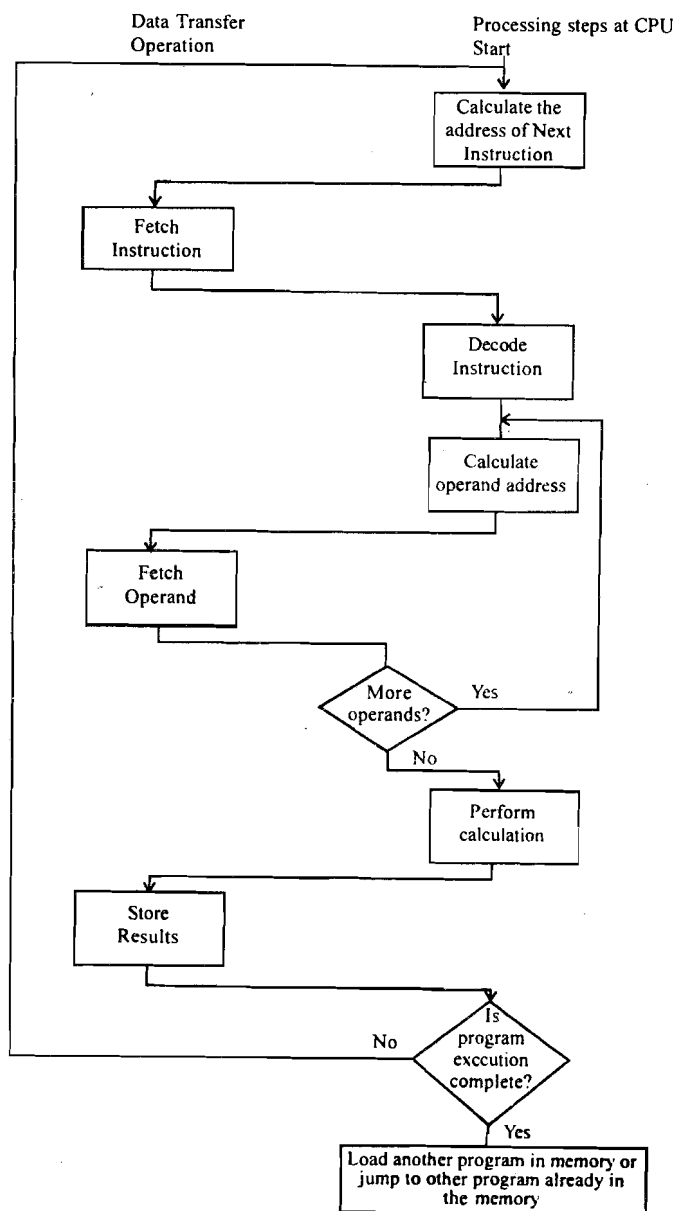


Figure 5: Instruction Cycle

Thus, in general, the execution cycle for a particular instruction may involve more than one stage and memory references. In addition, an instruction may ask for an I/O operation. Considering the steps above, let us work out a more detailed view of instruction cycle. Figure 5 gives a diagram of an instruction cycle.

Please note that in the preceding diagram some steps may be bypassed while some may be visited more than once. The instruction cycle shown in figure 5 consists of following states/stages:

- First the address of the next instruction is calculated, based on the size of instruction and memory organisation. For example, if in a computer an instruction is of 16 bits and if memory is organized as 16-bits words, then the address of the next instruction is evaluated by adding one in the address of the current instruction. In case, the memory is organized as bytes, which can be addressed individually, then we need to add two in the current instruction address to get the address of the next instruction to be executed in sequence.
- Now, the next instruction is fetched from a memory location to the CPU registers such as Instruction register.
- The next state decodes the instruction to determine the type of operation desired and the operands to be used.
- In case the operands need to be fetched from memory or via Input devices, then the address of the memory location or Input device is calculated.
- Next, the operand is fetched (or operands are fetched one by one) from the memory or read from the Input devices.
- Now, the operation, asked by the instruction is performed.
- Finally, the results are written back to memory or Output devices, wherever desired by first calculating the address of the operand and then transferring the values to desired destination.

Please note that multiple operands and multiple results are allowed in many computers. An example of such a case may be an instruction ADD A, B. This instruction requires operand A and B to be fetched.

In certain machines a single instruction can trigger an operation to be performed on an array of numbers or a string of characters. Such an operation involves repeated fetch for the operands without fetching the instruction again, that is, the instruction cycle loops at operand fetch.

Thus, a Program is executed as per the instruction cycle of figure 5. But what happens when you want the program to terminate in between? At what point of time is an interruption to a program execution allowed? To answer these questions, let us discuss the process used in computer that is called interrupt handling.

1.4.1 Interrupts

The term interrupt is an exceptional event that causes CPU to temporarily transfer its control from currently executing program to a different program which provides service to the exceptional event. It is like you asking a question in a class. When you ask a question in a class by raising hands, the teacher who is explaining some point may respond to your request only after completion of his/her point. Similarly, an interrupt is acknowledged by the CPU when it has completed the currently executing instruction. An interrupt may be generated by a number of sources, which may be either internal or external to the CPU.

Some of the basic issues of interrupt are:

- What are the different kinds of interrupts?
- What are the advantages of having an interruption mechanism?
- How is the CPU informed about the occurrence of an interrupt?
- What does the CPU do on occurrence of an interrupt?

Figure 6 Gives the list of some common interrupts and events that cause the occurrence of those interrupts.

Interrupt Condition	Occurrence of Event
Interrupt are generated by executing program itself (also called traps)	<input type="checkbox"/> Division by Zero <input type="checkbox"/> The number exceeds the maximum allowed. <input type="checkbox"/> Attempt of executing an illegal/privileged instruction. <input type="checkbox"/> Trying to reference memory location other than allowed for that program.
Interrupt generated by clock in the processor	Generally used on expiry of time allocated for a program, in multiprogramming operating systems.
Interrupts generated by I/O devices and their interfaces	<input type="checkbox"/> Request of starting an Input/Output operation. <input type="checkbox"/> Normal completion of an Input/Output operation. <input type="checkbox"/> Occurrence of an error in Input/Output operation.
Interrupts on Hardware failure	<input type="checkbox"/> Power failure <input type="checkbox"/> Memory parity error.

Figure 6: Various classes of Interrupts

Interrupts are a useful mechanism. They are useful in improving the efficiency of processing. How? This is to the fact that almost all the external devices are slower than the processor, therefore, in a typical system, a processor has to continually test whether an input value has arrived or a printout has been completed, in turn wasting a lot of CPU time. With the interrupt facility CPU is freed from the task of testing status of Input/Output devices and can do useful processing during this time, thus increasing the processing efficiency.

How does the CPU know that an interrupt has occurred?

There needs to be a line or a register or status word in CPU that can be raised on occurrence of interrupt condition.

Once a CPU knows that an interrupt has occurred then what?

First the condition is to be checked as to why the interrupt has occurred. That includes not only the device but also why that device has raised the interrupt. Once the

interrupt condition is determined the necessary program called ISRs (Interrupt servicing routines) must be executed such that the CPU can resume further operations.

For example, assume that the interrupt occurs due to an attempt by an executing program for execution of an illegal or privileged instruction, then ISR for such interrupt may terminate the execution of the program that has caused this condition. Thus, on occurrence of an Interrupt the related ISR is executed by the CPU. The ISRs are pre-defined programs written for specific interrupt conditions.

Considering these requirements let us work out the steps, which CPU must perform on the occurrence of an interrupt.

- The CPU must find out the source of the interrupt, as this will determine which interrupt service routine is to be executed.
- The CPU then acquires the address of the interrupt service routine, which are stored in the memory (in general).
- What happens to the program the CPU was executing before the interrupt? This program needs to be interrupted till the CPU executes the Interrupt service program. Do we need to do something for this program? Well the context of this program is to be saved. We will discuss this a bit later.
- Finally, the CPU executes the interrupt service routine till the completion of the routine. A RETURN statement marks the end of this routine. After that, the control is passed back to the interrupted program.

Let us analyse some of the points above in greater detail.

Let us first discuss saving the context of a program. The execution of a program in the CPU is done using certain set of registers and their respective circuitry. As the CPU registers are also used for execution of the interrupt service routine, it is highly likely that these routines alter the content of several registers. Therefore, it is the responsibility of the operating system that before an interrupt service routine is executed the previous content of the CPU registers should be stored, such that the execution of interrupted program can be restarted without any change from the point of interruption. Therefore, at the beginning of interrupt processing the essential context of the processor is saved either into a special save area in main memory or into a stack. This context is restored when the interrupt service routine is finished, thus, the interrupted program execution can be restarted from the point of interruption.

1.4.2 Interrupts and Instruction Cycle

Let us summarise the interrupt process, on the occurrence of an interrupt, an interrupt request (in the form of a signal) is issued to the CPU. The CPU on receipt of interrupt request suspends the operation of the currently executing program, saves the context of the currently executing program and starts executing the program which services that interrupt request. This program is also known as interrupt handler. After the interrupting condition/ device has been serviced the execution of original program is resumed.

Thus, an interrupt can be considered as the interruption of the execution of an ongoing user program. The execution of user program resumes as soon as the interrupt processing is completed. Therefore, the user program does not contain any code for interrupt handling. This job is to be performed by the processor and the operating system, which in turn are also responsible for suspending the execution of the user program, and later after interrupt handling, resumes the user program from the point of interruption.

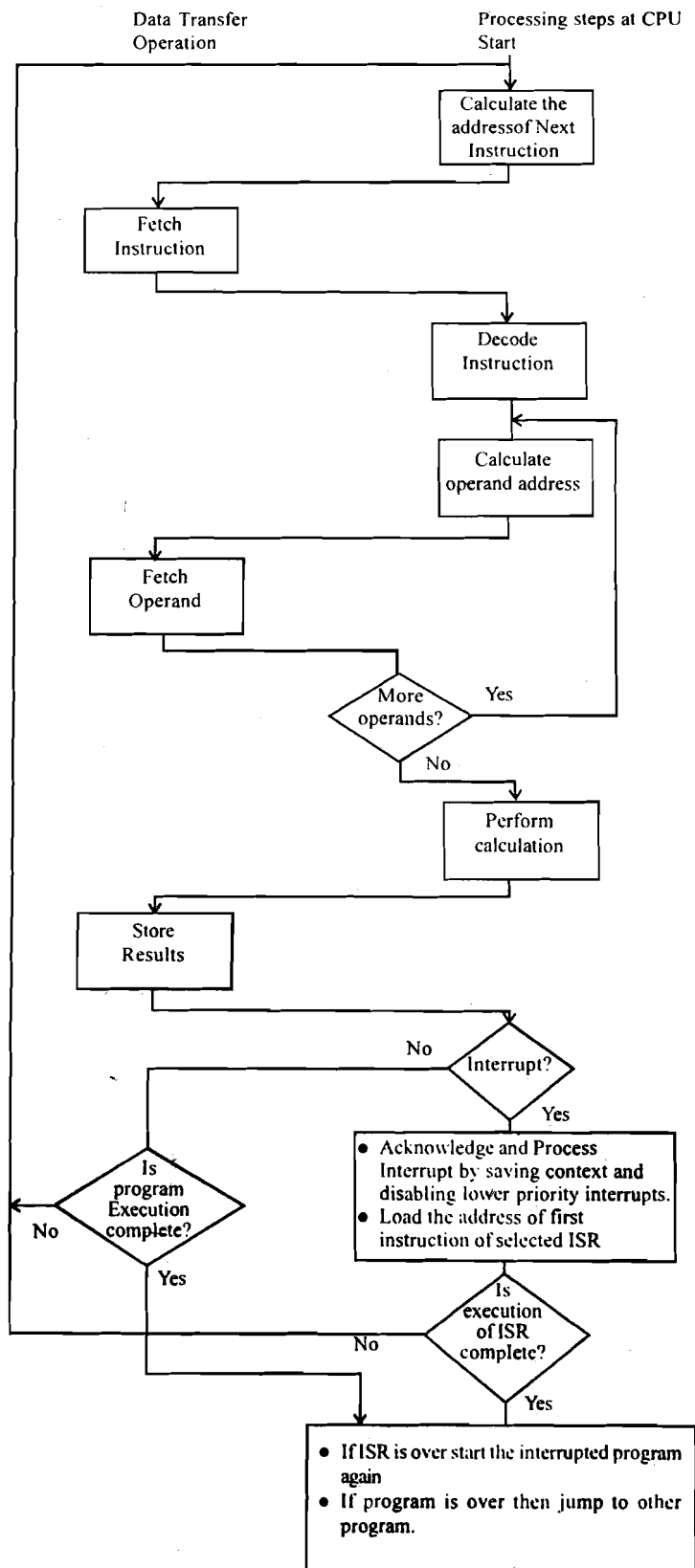


Figure 7: Instruction Cycle with Interrupt Cycle

But when can a user program execution be interrupted?

It will not be desirable to interrupt a program while an instruction is getting executed and is in a state like instruction decode. The most desirable place for program

interruption would be when it has completed the previous instruction and is about to start a new instruction. Figure 7 shows instruction execution cycle with interrupt cycle, where the interrupt condition is acknowledged. Please note that even interrupt service routine is also a program and after acknowledging interrupt the next instruction executed through instruction cycle is the first instruction of interrupt servicing routine.

In the interrupt cycle, the responsibility of the CPU/Processor is to check whether any interrupts have occurred checking the presence of the interrupt signal. In case no interrupt needs service, the processor proceeds to the next instruction of the current program. In case an interrupt needs servicing then the interrupt is processed as per the following.

- Suspend the execution of current program and save its context.
- Set the Program counter to the starting address of the interrupt service routine of the interrupt acknowledged.
- The processor then executes the instructions in the interrupt-servicing program. The interrupt servicing programs are normally part of the operating system.
- After completing the interrupt servicing program the CPU can resume the program it has suspended in step 1 above.

Check Your Progress 2

1) State True or False

T/F

- i) The value of PC will be incremented by 1 after fetching each instruction if the memory word is of one byte and an instruction is 16 bits long. ☐
- ii) MAR and MBR both are needed to fetch the data /instruction from the memory. ☐
- iii) A clock may generate an interrupt. ☐
- iv) Context switching is not desired before interrupt processing. ☐
- v) In case multiple interrupts occur at the same time, then only one of the interrupt will be acknowledged and rest will be lost. ☐

2) What is an interrupt?

.....

.....

.....

.....

3) What happens on the occurrence of an interrupt?

.....

.....

.....

.....

1.5 COMPUTERS: THEN AND NOW

Let us now discuss the history of computers because this will give the basic information about the technological development trends in computer in the past and its projections for the future. If we want to know about computers completely, then we

must look at the history of computers and look into the details of various technological and intellectual breakthroughs. These are essential to give us the feel of how much work and effort has been done in the past to bring the computer to this shape. Our effort in this section will be to describe the conceptual breakthroughs in the past.

The ancestors of modern age computer were the mechanical and electromechanical devices. This ancestry can be traced as far back as the 17th Century, when the first machine capable of performing four mathematical operations, viz. addition, subtraction, division and multiplication, appeared. In the subsequent subsection we present a very brief account of Mechanical Computers.

1.5.1 The Beginning

Blaise Pascal made the very first attempt towards automatic computing. He invented a device, which consisted of lots of gears and chains which used to perform repeated additions and subtractions. This device was called Pascaline. Later many attempts were made in this direction.

Charles Babbage, the grandfather of the modern computer, had designed two computers:

The Difference Engine: It was based on the mathematical principle of finite differences and was used to solve calculations on large numbers using a formula. It was also used for solving the polynomial and trigonometric functions.

The Analytical Engine by Babbage: It was a general purpose-computing device, which could be used for performing any mathematical operation automatically. The basic features of this analytical engine were:

- It was a general-purpose programmable machine.
- It had the provision of automatic sequence control, thus, enabling programs to alter its sequence of operations.
- The provision of sign checking of result existed.
- A mechanism for advancing or reversing of control card was permitted thus enabling execution of any desired instruction. In other words, Babbage had devised the conditional and branching instructions. The Babbage's machine was fundamentally the same as the modern computer. Unfortunately, Babbage work could not be completed. But as a tribute to Charles Babbage his Analytical Engine was completed in the last decade of the 20th century and is now on display at the Science Museum at London.

The next notable attempts towards computers were electromechanical. Zuse used electromechanical relays that could be either opened or closed automatically. Thus, the use of binary digits, rather than decimal numbers started, in computers.

Harvard Mark-I and the Bug

The next significant effort towards devising an electromechanical computer was made at the Harvard University, jointly sponsored by IBM and the Department of UN Navy, Howard Aiken of Harvard University developed a system called Mark I in 1944. Mark I was a decimal machine, that is, the computations were performed using decimal digits.

Some of you must have heard a term called "bug". It is mainly used to indicate errors in computer programs. This term was coined when one day, a program in Mark-I did not run properly due to a moth short-circuiting the computer. Since then, the moth or the bug has been linked with errors or problems in computer programming. Thus, the process of eliminating error in a program is known as 'debugging'.

The basic drawbacks of these mechanical and electromechanical computers were:

- Friction/inertia of moving components limited the speed.
- The data movement using gears and liners was quite difficult and unreliable.
- The change was to have a switching and storing mechanism with no moving parts. The electronic switching device “triode” vacuum tubes were used and hence the first electronic computer was born.

1.5.2 First Generation Computers

It is indeed ironic that scientific inventions of great significance have often been linked with supporting a very sad and undesirable aspect of civilization, that is, fighting wars. Nuclear energy would not have been developed as fast, if colossal efforts were not spent towards devising nuclear bombs. Similarly, the origin of the first truly general-purpose computer was also designed to meet the requirement of World War II. The ENIAC (the Electronic Numerical Integrator And Calculator) was designed in 1945 at the University of Pennsylvania to calculate figures for thousands of gunnery tables required by the US army for accuracy in artillery fire. The ENIAC ushered in the era of what is known as first generation computers. It could perform 5000 additions or 500 multiplications per minute. It was, however, a monstrous installation. It used thousands of vacuum tubes (18000), weighed 30 tons, occupied a number of rooms, needed a great amount of electricity and emitted excessive heat.

The main features of ENIAC can be summarised as:

- ENIAC was a general purpose-computing machine in which vacuum tube technology was used.
- ENIAC was based on decimal arithmetic rather than binary arithmetic.
- ENIAC needed to be programmed manually by setting switches and plugging or unplugging. Thus, to pass a set of instructions to the computer was difficult and time-consuming. This was considered to be the major deficiency of ENIAC.

The trends, which were encountered during the era of first generation computers were:

- Centralised control in a single CPU; all the operations required a direct intervention of the CPU.
- Use of ferrite-core main memory was started during this time.
- Concepts such as use of virtual memory and index register (you will know more about these terms later) started.
- Punched cards were used as input device.
- Magnetic tapes and magnetic drums were used as secondary memory.
- Binary code or machine language was used for programming.
- Towards the end, due to difficulties encountered in use of machine language as programming language, the use of symbolic language that is now called assembly language started.
- Assembler, a program that translates assembly language programs to machine language, was made.
- Computer was accessible to only one programmer at a time (single user environment).
- Advent of von-Neumann architecture.

1.5.3 Second Generation Computers

Silicon brought the advent of the second generation computers. A two state device called a transistor was made from silicon. Transistor was cheaper, smaller and dissipated less heat than vacuum tube, but could be utilised in a similar way to vacuum tubes. A transistor is called a solid state device as it is not created from wires, metal glass capsule and vacuum which was used in vacuum tubes. The transistors were invented in 1947 and launched the electronic revolution in 1950.

But how do we characterise the future generation of computers?

The generations of computers are basically differentiated by the fundamental hardware technology. The advancement in technology led to greater speed, large memory capacity and smaller size in various generations. Thus, second generation computers were more advanced in terms of arithmetic and logic unit and control unit than their counterparts of the first generation and thus, computationally more powerful. On the software front at the same time use of high level languages started and the developments were made for creating better Operating System and other system software.

One of the main computer series during this time was the IBM 700 series. Each successful member of this series showed increased performance and capacity and reduced cost. In these series two main concepts, I/O channels - an independent processor for Input/Output, and Multiplexer - a useful routing device, were used. These two concepts are defined in the later units.

1.5.4 Third Generation Computers

The third generation has the basic hardware technology: the Integrated Circuits (ICs). But what are integrated circuits? Let us first define a term called discrete components. A single self-contained transistor is called discrete component. The discrete components such as transistors, capacitors, resistors were manufactured separately and were soldered on circuit boards, to create electronic components or computer cards. All these cards/components then were put together to make a computer. Since a computer can contain around 10,000 of these transistors, therefore, the entire mechanism was cumbersome. The basic idea of integrated circuit was to create electronic components and later the whole CPU on a single Integrated chip. This was made possible by the era of microelectronics (small electronics) with the invention of Integrated Circuits (ICs).

In an integrated circuit technology the components such as transistors, resistors and conductors are fabricated on a semiconductor material such as silicon. Thus, a desired circuit can be fabricated in a tiny piece of silicon. Since, the size of these components is very small in silicon, thus, hundreds or even thousands of transistors could be fabricated on a single wafer of silicon. These fabricated transistors are connected with a process of metalisation, thus, creating logic circuits on the chip.

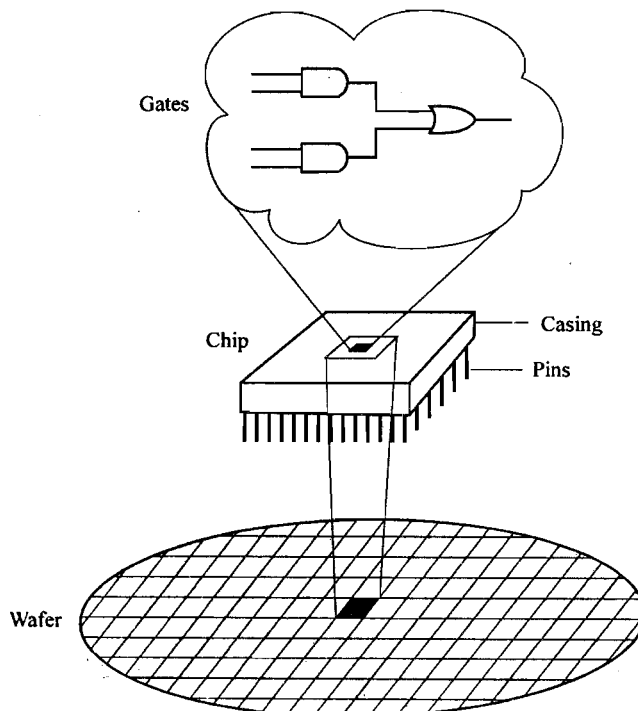


Figure 8: Silicon Wafer, Chip and Gates

An integrated circuit is constructed on a thin wafer of silicon, which is divided into a matrix of small areas (size of the order of a few millimeter squares). An identical circuit pattern is fabricated in a dust free environment on each of these areas and the wafer is converted into chips. (Refer figure 8). A chip consists of several gates, which are made using transistors. A chip also has a number of input and output connection points. A chip then is packaged separately in a housing to protect it. This housing provides a number of pins for connecting this chip with other devices or circuits. For example, if you see a microprocessor, what you are looking and touching is its housing and huge number of pins.

Different circuits can be constructed on different wafers. All these packaged circuit chips then can be interconnected on a Printed-circuit board (for example, a motherboard of computer) to produce several complex electronic circuits such as in a computer.

The Integration Levels:

Initially, only a few gates were integrated reliably on a chip. This initial integration was referred to as small-scale integration (SSI).

With the advances in microelectronics technologies the SSI gave way to Medium Scale Integration where 100's of gates were fabricated on a chip.

Next stage was Large Scale Integration (1,000 gates) and very large integration (VLSI 1000,000 gates on a single chip). Presently, we are in the era of Ultra Large Scale Integration (ULSI) where 100,000,000 or even more components may be fabricated on a single chip.

What are the advantages of having densely packed Integrated Circuits? These are:

- **Reliability:** The integrated circuit interconnections are much more reliable than soldered connections. In addition, densely packed integrated circuits enable fewer inter-chip connections. Thus, the computers are more reliable. In fact, the two unreliable extremes are when the chips are in low-level integration or extremely high level of integration almost closer to maximum limits of integration.
- **Low cost:** The cost of a chip has remained almost constant while the chip density (number of gates per chip) is ever increasing. It implies that the cost of computer logic and memory circuitry has been reducing rapidly.
- **Greater Operating Speed:** The more is the density, the closer are the logic or memory elements, which implies shorter electrical paths and hence higher operating speed.
- **Smaller computers provide better portability**
- **Reduction in power and cooling requirements.**

The third generation computers mainly used SSI chips. One of the key concept which was brought forward during this time was the concept of the family of compatible computers. IBM mainly started this concept with its system/360 family.

A family of computers consists of several models. Each model is assigned a model number, for example, the IBM system/360 family have, Model 30,40, 50,65 and 75. The memory capacity, processing speed and cost increases as we go up the ladder. However, a lower model is compatible to higher model, that is, program written on a lower model can be executed on a higher model without any change. Only the time of execution is reduced as we go towards higher model and also a higher model has more

number of instructions. The biggest advantage of this family system was the flexibility in selection of model.

For example, if you had a limited budget and processing requirements you could possibly start with a relatively moderate model. As your business grows and your processing requirements increase, you can upgrade your computer with subsequent models depending on your need. However, please note that as you have gone for the computer of the same family, you will not be sacrificing investment on the already developed software as they can still be used on newer machines also.

Let us summarise the main characteristics of a computer family. These are:

S.No.	Feature	Characteristics while moving from lower member to higher member
1.	Instruction set	<input type="checkbox"/> Similar instructions. <input type="checkbox"/> Normally, the instructions set on a lower end member is a subset of higher end member. A program written on lower end member can be executed on a higher end member, but program written on higher end member may or may not get executed on lower end members.
2	Operating System	<input type="checkbox"/> Same may have some additional features added in the operating system for the higher end members.
3	Speed of instruction execution	<input type="checkbox"/> Increases
4	Number of I/O ports	<input type="checkbox"/> Increases
5	Memory size	<input type="checkbox"/> Increases
6	Cost	<input type="checkbox"/> Increases

Figure 9: Characteristics of computer families

But how was the family concept implemented? Well, there were three main features of implementation. These were:

- Increased complexity of arithmetic logic unit;
- Increase in memory - CPU data paths; and
- Simultaneous access of data in higher end members.

The major developments which took place in the third generation, can be summarized as:

- Application of IC circuits in the computer hardware replacing the discrete transistor component circuits. Thus, computers became small in physical size and less expensive.
- Use of Semiconductor (Integrated Circuit) memories as main memory replacing earlier technologies.
- The CPU design was made simple and CPU was made more flexible using a technique called microprogramming (will be discussed in later Blocks).
- Certain new techniques were introduced to increase the effective speed of program execution. These techniques were pipelining and multiprocessing. The details on these concepts can be found in the further readings.
- The operating system of computers was incorporated with efficient methods of sharing the facilities or resources such as processor and memory space automatically. These concepts are called multiprogramming and will be discussed in the course on operating systems.

1.5.5 Later Generations

One of the major milestones in the IC technology was the very large scale integration (VLSI) where thousands of transistors can be integrated on a single chip. The main impact of VLSI was that, it was possible to produce a complete CPU or main memory or other similar devices on a single IC chip. This implied that mass production of CPU, memory etc. can be done at a very low cost. The VLSI-based computer architecture is sometimes referred to as fourth generation computers.

The Fourth generation is also coupled with Parallel Computer Architectures. These computers had shared or distributed memory and specialized hardware units for floating point computation. In this era, multiprocessing operating system, compilers and special languages and tools were developed for parallel processing and distributed computing. VAX 9000, CRAY X-MP, IBM/3090 were some of the systems developed during this era.

Fifth generation computers are also available presently. These computers mainly emphasise on Massively Parallel Processing. These computers use high-density packaging and optical technologies. Discussions on such technologies are beyond the scope of this course.

However, let us discuss some of the important breakthroughs of VLSI technologies in this subsection:

Semiconductor Memories

Initially the IC technology was used for constructing processor, but soon it was realised that the same technology can be used for construction of memory. The first memory chip was constructed in 1970 and could hold 256 bits. The cost of this first chip was high. The cost of semiconductor memory has gone down gradually and presently the IC RAM's are quite cheap. Although the cost has gone down, the memory capacity per chip has increased. At present, we have reached the 1 Gbits on a single memory chip. Many new RAM technologies are available presently. We will give more details on these technologies later in Block 2.

Microprocessors

Keeping pace with electronics as more and more components were fabricated on a single chip, fewer chips were needed to construct a single processor. Intel in 1971 achieved the breakthrough of putting all the components on a single chip. The single chip processor is known as a microprocessor. The Intel 4004 was the first microprocessor. It was a primitive microprocessor designed for a specific application. Intel 8080, which came in 1974, was the first general-purpose microprocessor. This microprocessor was meant to be used for writing programs that can be used for general purpose computing. It was an 8-bit microprocessor. Motorola is another manufacturer in this area. At present 32 and 64 bit general-purpose microprocessors are already in the market. Let us look into the development of two most important series of microprocessors.

S.No.	Processor	Year	Memory size	Bus width	Comment
1	4004	1971	640 bytes	4 bits	Processor for specific applications
2.	8080	1974	64 KB	8 bits	First general-purpose micro-processor. It was used in development of first personal computer
3.	8086	1978	1 MB	16 bits	<input type="checkbox"/> Supported instruction cache memory or queue <input type="checkbox"/> Was the first powerful machine

4	80386	1985-1988 various versions.	4 G Byte Processor	32 bits	<input type="checkbox"/> First 32 bit <input type="checkbox"/> The processor supports multitasking
5	80486	1989-1991	4 g Byte	32 bits	<input type="checkbox"/> Use of powerful cache technology. <input type="checkbox"/> Supports pipeline based instruction execution <input type="checkbox"/> Contains built-in facility in the term of built-in math co-processor for floating point instructions
6	Pentium	1993-1995	64 G Bytes	32-bits and 64 bits	Uses superscalar techniques, that is execution of multiple instructions in parallel.
7	Pentium II	1997	64 G Bytes	64 bits	Contains instruction for handling processing of video, audio, graphics etc. efficiently. This technology was called MMX technology.
8	Pentium III	1999	64 B bytes	64 bits	Supports 3 D graphics software.
9	Pentium IV	2000	64 G Bytes	64 bits	Contains instructions for enhancement of multimedia. A very powerful processor
10	Itaium	2001	64 G bytes	64 bits	Supports massively parallel computing architecture.
11	Xeon	1999	64 G bytes	64 bits	Support hyper threading explained after this diagrams Outstanding performance and dependability: ideal for low cost servers

Figure 10: Some Important Developments in Intel Family of Microprocessors

Hyper-threading:

Nonthreaded program instructions are executed in a single order at a time, till the program completion. Suppose a program have 4 tasks namely A, B, C, D. Assume that each task consist of 10 instructions including few I/O instructions. A simple sequential execution would require $A \rightarrow B \rightarrow C \rightarrow D$ sequence.

In a threaded system these tasks of a single process/program can be executed in parallel provided is no data dependency. Since, there is only one processor these tasks will be executed in threaded system as interleaved threads, for example, 2 instructions of A 3 instruction of B, 1 instruction of C, 4 instruction of D, 2 instruction of C etc. till completion of the threads.

Hyper-threading allows 2 threads A & B to execute at the same time. How? Some of the more important parts of the CPU are duplicated. Thus, there exists 2 executing threads in the CPU at the exact same time. Please note that both these sections of the CPU works on the same memory space (as threads are the same program). Eventually dual CPUs will allow the computer to execute two threads in two separate programs at the same time.

Thus, Hyper-threading technology allows a single microprocessor to act like two separate threaded processors to the operating system and the application program that use it.

Hyper-threading requires software that has multiple threads and optimises speed of execution. A threaded program executes faster on hyper threaded machine. However, it should be noted that not all programs can be threaded.

The other architecture that has gained popularity over the last decade is the power PC family. These machines are reduced set instruction computer (RISC) based technologies. RISC technologies are finding their application because of simplicity of Instructions. You will learn more about RISC in Block 3 of this course.

The IBM made an alliance with Motorola and Apple who has used Motorola 68000 chips in their Macintosh computer to create a POWER PC architecture. Some of the processors in this family are:

S.No.	Processor	Year	Bus Width	Comment
1	601	1993	32 bits	The first chip in power PC
2	603/603e	1994	32 bits	❑ Low cost machine, intended for low cost desktop
3	604/604e	1997	64 bits	❑ Low end server having superscalar architecture
4	G3	1997	64 bits	❑ Contains two levels of cache ❑ Shows good performance
5	G4	1999	64 bits	Increased speeds & parallelism of instruction execution
6	G6	2003	64 bits	Extremely fast multimedia capability rated very highly.

Figure 11: Power PC Family

The VLSI technology is still evolving. More and more powerful microprocessors and more storage space now is being put in a single chip. One question which we have still not answered, is: Is there any classification of computers? Well-for quite sometime computers have been classified under the following categories:

- Micro-controllers
- Micro-computers
- Engineering workstations
- Mini computers
- Mainframes
- Super computers
- Network computers.

Micro-controllers: These are specialised device controlling computers that contains all the functions of computers on a single chip. The chip includes provision for processing, data input and output display. These chips then can be embedded into various devices to make them more intelligent. Today this technology has reached

great heights. In fact it has been stated that embedded technology computing power available even in a car today is much more than what was available in the system on first lunar mission”.

Microcomputers

A microcomputer's CPU is a microprocessor. They are typically used as single user computer although present day microcomputers are very powerful. They support highly interactive environment specially like graphical user interface like windows. These computers are popular for home and business applications. The microcomputer originated in late 1970's. The first microcomputers were built around 8-bit microprocessor chips. What do we mean by an 8-bit chip? It means that the chip can retrieve instructions/data from storage, manipulate, and process an 8-bit data at a time or we can say that the chip has a built-in 8-bit data transfer path.

An improvement on 8-bit chip technology was seen in early 1980s, when a series of 16-bit chips namely 8086 and 8088 were introduced by Intel Corporation, each one with an advancement over the other.

8088 was an 8/16 bit chip i.e. an 8-bit path is used to move data between chip and primary storage (external path), but processing was done within the chip using a 16-bit path (internal path) at a time. 8086 was a 16/16-bit chip i.e. the internal and external paths both were 16 bits wide. Both these chips could support a primary basic memory of storage capacity of 1 Mega Byte (MB).

Similar to Intel's chip series exists another popular chip series of Motorola. The first 16-bit microprocessor of this series was MC 68000. It was a 16/32-bit chip and could support up to 16 MB of primary storage. Advancement over the 16/32 bit chips was the 32/32 chips. Some of the popular 32-bit chips were Intel's 80486 and MC 68020 chip.

Most of the popular microcomputers were developed around Intel's chips, while most of the minis and super minis were built around Motorola's 68000 series chips. With the advancement of display and VLSI technology a microcomputer was available in very small size. Some of these are laptops, note book computers etc. Most of these are of the size of a small notebook but equivalent capacity of an older mainframe.

Workstations

The workstations are used for engineering applications such as CAD/CAM or any other types of applications that require a moderate computing power and relatively high quality graphics capabilities. Workstations generally are required with high resolution graphics screen, large RAM, network support, a graphical user interface, and mass storage device. Some special type of workstation comes, without a disk. These are called diskless terminals/ workstations. Workstations are typically linked together to form a network. The most common operating systems for workstations are UNIX, Windows 2003 Server, and Solaris etc.

Please note that networking workstation means any computer connected to a local area network although it could be a workstation or a personal computer.

Workstations may be a client to server Computers. Server is a computer that is optimised to provide services to other connected computers through a network. Servers usually have powerful processors, huge memory and large secondary storage space.

Minicomputer

The term minicomputer originated in 1960s when it was realised that many computing tasks do not require an expensive contemporary mainframe computers but can be solved by a small, inexpensive computer.

The mini computers support multi-user environment with CPU time being shared among multiple users. The main emphasis in such computer is on the processing power and less for interaction. Most of the present day mini computers have proprietary CPU and operating system. Some common examples of a mini-computer are IBM AS/400 and Digital VAX. The major use of a minicomputer is in data processing application pertaining to departments/companies.

Mainframes

Mainframe computers are generally 32-bit machines or higher. These are suited to big organisations, to manage high volume applications. Few of the popular mainframe series were DEC, IBM, HP, ICL, etc. Mainframes are also used as central host computers in distributed systems. Libraries of application programs developed for mainframe computers are much larger than those of the micro or minicomputers because of their evolution over several decades as families of computing. All these factors and many more make the mainframe computers indispensable even with the popularity of microcomputers.

Supercomputers

The upper end of the state of the art mainframe machine are the supercomputers. These are amongst the fastest machines in terms of processing speed and use multiprocessing techniques, where a number of processors are used to solve a problem. There are a number of manufacturers who dominate the market of supercomputers-CRAY, IBM 3090 (with vector), NEC Fujitsu, PARAM by C-DEC are some of them. Lately, a range of parallel computing products, which are multiprocessors sharing common buses, have been in use in combination with the mainframe supercomputers. The supercomputers are reaching upto speeds well over 25000 million arithmetic operations per second. India has also announced its indigenous supercomputer. They support solutions to number crunching problems.

Supercomputers are mainly being used for weather forecasting, computational fluid dynamics, remote sensing, image processing, biomedical applications, etc. In India, we have one such mainframe supercomputer system-CRAY XMP-14, which is at present, being used by Meteorological Department.

Let us discuss about PARAM Super computer in more details

PARAM is a high-performances, scalable, industry standard computer. It has evolved from the concepts of distributes scalable computers supporting massive parallel processing in cluster of networked of computers. The PARAM's main advantages is its Scalability. PARAM can be constructed to perform Tera-floating point operations per second. It is a cost effective computer. It supports a number of application software.

PARAM is made using standard available components. It supports Sun's Ultra SPARC series servers and Solaris Operating System. It is based on open environments and standard protocols. It can execute any standard application available on Sun Solaris System.

Some of the applications that have been designed to run in parallel computational mode on PARAM include numerical weather forecasting, seismic data processing, Molecular modelling, finite element analysis, quantum chemistry.

It also supports many languages and Software Development platforms such as:

Solaris 2.5.1 Operating system on I/O and Server nodes, FORTRAN 77, FORTRAN 90, C and C++ language compilers, and tools for parallel program debugging, Visualisation and parallel libraries, Distributed Computing Environment, Data warehousing tools etc.

- 1) What is a general purpose machine?

.....

.....

.....

- 2) List the advantages of IC technology over discrete components.

.....

.....

.....

- 3) What is a family of computers? What are its characteristics?

.....

.....

1.6 SUMMARY

This completes our discussion on the introductory concepts of computer architecture. The von-Neumann architecture discussed in the unit is not the only architecture but many new architectures have come up which you will find in further readings. The information given on various topics such as interrupts, classification, history of computer although is exhaustive yet can be supplemented with additional reading. In fact, a course in an area of computer must be supplemented by further reading to keep your knowledge up to date, as the computer world is changing with leaps and bounds. In addition to further readings the student is advised to study several Indian Journals on computers to enhance his knowledge.

1.7 SOLUTIONS / ANSWERS

Check Your Progress 1

- 1.
- a) True
 - b) False
 - c) False
 - d) True
 - e) True
 - f) True
2. von Neumann architecture defines the basic architectural (logical) components of computer and their features. As per von Neumann the basic components of a computer are CPU (ALU+CU + Registers), I/O Devices, Memory and interconnection structures. von Neumann machine follows stored program concept that is, a program is loaded in the memory of computer prior to its execution.
- 3.
- The instructions are not executed in sequence
 - More than one data items may be required during a single instruction execution.
 - Speed of CPU is very fast in comparison to I/O devices.

Check Your Progress 2

1.
 - i) False
 - ii) True
 - iii) True
 - iv) False
 - v) False, they may be acknowledged as per priority.
2. An interrupt is an external signal that occurs due to an exceptional event. It causes interruption in the execution of current program of CPU.
3. An interrupt is acknowledged by the CPU, which executes an interrupt cycle which causes interruption of currently executing program, and execution of interrupt servicing routine (ISR) for that interrupt.

Check Your Progress 3

1. A machine, which can be used for variety of applications and is not modeled only for specific applications. von Neumann machines are general-purpose machines since they can be programmed for any general application, while microprocessor based control systems are not general-purpose machines as they are specifically modeled as control systems.
2.
 - Low cost
 - Increased operating speed
 - Reduction in size of the computers
 - Reduction in power and cooling requirements
 - More reliable
3. The concept of the family of computer was floated by IBM 360 series where the features and cost increase from lower end members to higher end members.