UNIT 1 Computer System

Structure **Page Nos.**

1.0 INTRODUCTION

In the present competitive world, a business can survive if it uses most advanced Information technology to support various businesses processes. In this digital world, computers are an important part your daily life. You use computer to use health services, banking services, teaching and learning services, online services made available by the Government and many more such services. Computer technology can be used to make all these processes more efficient and user friendly.

This Unit introduces you to some of the basic terminologies used to define computer system of today. In addition, the breakthrough in the history of computer systems has also been included in this Unit. This Unit also introduces you to some of the popular computer architectures, like von Neumann Architecture, which was one of the first computer architecture, and other contemporary architectures of Computer System. Also, a simple novel idea for execution of instructions has also been introduced. This basic process of instruction execution will be explained in more details in the later Units of the course. You can get more information on these principles from the further readings.

1.1 OBJECTIVES

After going through this unit, you will be able to:

- explain the basic structure of computer system
- list and explain the features of the von Neumann architecture of the computer;
- identify some of the important breakthroughs in history of computers;
- identify the process of instruction execution;
- define some of the contemporary computer organization

1.2 A Brief History

This section traces a brief historical background of computer. You should be aware of that the push to construct Computer has not come from one person or group or organisation. There were many attempts to develop an automatic and programmable computing device. In 1944, the University of Pennsylvania developed the first automatic computing device, which was called Electronic Numerical Integrator and Calculator (ENIAC). It was a general-purpose machine fabricated utilizing vacuum tubes. This machine was planned basically to compute the shooting scope of weapons during World War II. It was arranged by manual setting of exchanging and associating links. An improved ENIAC model was called Electronic Discrete Variable Automatic Computer (EDVAC), which was completed in 1952. Meanwhile, the specialists at the Institute for Advanced Study (IAS) in Princeton assembled (1946) an IAS machine, which was multiple times quicker than ENIAC.

A similar project was started in 1946 at Cambridge University. The objective of this project was to design a computer that stores instruction and data in the computer memory. This proposed computer was known as the Electronic Delay Storage Automatic Calculator (EDSAC). In 1949, EDSAC was completed and became the first computer of the world that stored the program instructions and related data in the memory of the computer. The architecture of these machines is discussed in section 1.6. Thereafter, Harvard launched a series of computes named Harvard Mark I, Mark II, Mark III, and Mark IV. The machines have different architecture than that of other contemporary machines. This architecture was called the Harvard Architecture. The Harvard architecture is discussed in section 1.6. The term Harvard architecture, now, used for machines, which have separate instruction and data cache memory. The term cache memory is explained in Block 2.

In 1951, UNIVersal Automatic Computer (UNIVAC I) was developed. This can be called the first commercial computer. In 1952, IBM announced its first computer, the IBM701. Later in 1964, IBM announced a family of computers, called IBM 360 series. The basic concept of family of computers was to keep the similar instruction set. However, the performance and price of the family increased with higher models. This allowed businesses to migrate to higher performance computer while retaining their investments in software. Another important computer series of this time was PDP-8 by Digital Equipment Corporation (DEC).

Advancement of technology lead to development of microprocessor on a single silicon chip. In 1971, Intel produced its first microprocess, called Intel 4004. The first personal computer (PC) was introduced by the Apple computer in 1977. Further, in this year itself world again saw the installation of a mainframe computer. A mainframe computer had many user terminal that used to solve the processing power of a single mainframe computer VAX-11/780 by the DEC. The year 1977 may be considered as an important year for computer industry, as in this year in addition to the developments as given above, one of the important microprocessor 8086, which you will study in Block 4, was introduced. This led to development of micro-computers. Different organisations launched their micro-computers based on Intel 8088/8086 microprocessors or many other microprocessors. In the subsequent years, the computers developed by Compaq, Apple, IBM, Dell, and many others became popular in the academia and

industry. A powerful computer that could perform millions of computation, called super computers were developed. The first such computer, the CDC 6600, was introduced in 1961 by the Control Data Corporation. Cray Research Corporation introduced the most expensive but most efficient super computer Cray-1, in 1976. During the decades of 1980s and 1990s, more powerful super computers were launched. These super computers had larger number of more powerful processors. These super computers were categorised on the basis of the sharing of memory by various processors, viz. shared memory and distributed memory super computers. A super computer differs from a simple multi-processor system, as it can have hundreds or even more processors. Examples of computers at this time included Intel iPSC, nCUBE, Imaging Equipment (CM-2, CM-5), and many others. The perfect design of computer is to install central servers through computer networks. These networks connect cheap desktop machines that have the ability to compensate for unmatched computing power. However, in late 1990s, with the popularity of Local area networks (LANs), which allowed sharing of resources, mainframes and minicomputers were replaced by network computers or personal computers, called desktop computers. These individual desktop computers were then connected to large computers over broadband (WAN) networks. The inclusiveness of the Internet has developed keen attention in the areas like network and grid computing. "Grids are geographically distributed platforms of computation". Grid computing provides reliable, consistent, persistent, and low-cost access to high-end computational facilities.

Table 1.1 highlights the development of Computers, in terms of type of circuitry used, the size of Random Access Memory (RAM), the speed in terms of Instruction execution per second and Programming languages of those era. The Table also lists some of important computers of those days.

Subject	1st generation	2nd generation	3rd generation	4th generation	5th generation
Period	1940-1956	1956-1963	1964-1971	1971-present	present & beyond
Circuitry	Vacuum tube	Transistor	Integrated chips (IC)	Microprocessor (VLSI)	ULSI (Ultra Large Scale Integration) technology
Memory Capacity	20 KB	128KB	1MB	Magnetic core memory, LSI and VLSI. High Capacity	ULSI
Processing Speed	300 IPS instructions Per sec.	300 IPS	1MIPS (1 million inst. Per sec.)	Faster than 3rd generation	Very fast
Programming Language	Machine, Language	Assembly language & early high-level languages(FORTRAN, COBOL, ALGOL)	C,C++	Higher level languages,C,C++,Java	All the Higher level languages,,Neural networks,
Example of computers	UNIVAC, EDVAC	IBM 1401, IBM 7094, CDC 3600,D UNIVAC 1108	IBM 360 series, 1900 series	Pentium series,Multimedia,	Artificial Intelligence, Robotics

TABLE 1.1 Decades of Computing
["Ref: https://www.discovertips.in/2017/06/computer-study-notes-history-and.html"]

Semi-Conductor Chips and Computer

A computer system consists of number of components, which are fabricated using semicoductor materials. The basic unit of these semi-conductor materials is an electronic transisters, which are used to integrate large computer circuitry on a single semi-conductor chip. Over the last five decades computer has shown an exponential rate of improvement due to growth and advancement in circuit integration technology on a semi-conductor chip; from a handful of transistors to millions of transistors on a single integrated chip. This resulted in multiple

fold increase in the size of computer memory and very large increase in processing capacities of computer processor. The integration technology has shown a rapid growth. Initially, only a few transistors were fabricated on a computer chip. This was called small-scale integration (SSI). Subsequently, with the advancements of technology, computer chips were made using the medium-scale integration (MSI), then using the large-scale integration (LSI), then using the very large-scale integration (VLSI), and currently to ultra large-scale integration (ULSI). Figure 1.2 includes various technologies and number of transistors per chip, also called as chip complexity. The integration growth can be more understandable in terms of "feature size". Feature size is that dimension of transistor that is required to be minimized without changing the functionality of the device. So, a decrease in feature size will eventually increase the number of transistors per chip which in turn, will result in development of new advanced chip having advanced functionality. This has resulted in growth of semiconductor memories (RAM memories) so that now, designers can trade off speed with memory size.

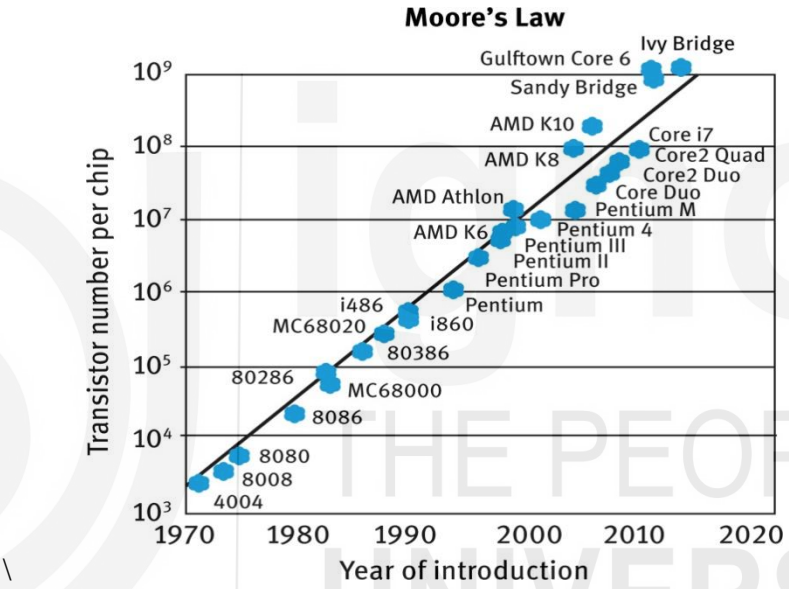


Figure 1.2: Numbers of Transistors per Chip
["Ref: https://www.ncbi.nlm.nih.gov/books/NBK321721/figure/oin_tutorial.F3/"]

1.3 STRUCTURE OF A COMPUTER

As discussed in the previous section, computer is fabricated using semi-conductor chips. This technology primarily have two stable states of representation, such as presence or absence of output. These two states are used to denote bits 0 and bit 1 respectively. Thus, the computer is a digital device, which performs all the operations using binary digits (bits). Therefore, computer instructions, character set, data in computation etc everything is to be represented and processed using binary codes or binary number system. The minimal processing requirements of a computers system are:

- It should allow input of instructions or commands and data on which these comamands are to perfomed.
- There can be large number of instructions that may be required for data manipulation. Some of these instructions may involve decision making and/or repetitions, therefore, it may be a good idea to store instruction and data. Thus, a computer should have memory.

- The computer should be able to process data as per the command/instruction. Thus, it will require components, which may process data, store temporary results and transfer data among different units.

- It should be able to store the results of the processed data using some output unit.

Therefore, a computer system should have a processor to perform computations as per the instructions, a memory for data storage/instructions, Input/Output units to input the data and insturction and output the results and a data path to transfer data. In order to define a simple structure of a computer system, the role and functions of its basic components should be studied. This section defines the basic components of the computer, like CPU, the Memory, the Input/output devices, data paths etc.

1.3.1 The CPU

The CPU is responsible for executing a sequence of instructions, which are part of a computer program. A computer program along with the data is stored in the main memory of a computer system. The CPU consists of following components: (1) registers, (2) an arithmetic logic unit (ALU), and (3) a control unit (CU). The role of each component of CPU is well defined. The registers are used to store - (1) commands or instruction which is presently being executed by the Arithmetic logic unit, (2) they can also store addresses such as address of operands /data, "address of the next instruction to executed", (3) the data or operands itself. Arithmetic logic unit is designed to perform binary computations.

The control unit (CU) controls the execution of instructions by the computer. CU controls the fetching, interpreting and execution of the commands or instructions on a computer, which primarily results in processing of the data stored in registers or the main memory. Figure 1.3 shows the structure of the CPU and its interaction with the memory system and input / output devices. The CPU "reads commands form the memory, reads and writes data from and to the memory, and transfers the data to the input / output devices". The most common and simple commands/instruction processing can be summarized as follows:

1. Get the next instruction or command to be executed from the main memory to the CPU registers
    a. In general, the address of the next instruction is stored in the program counter register (PC).
    b. The CU causes the instruction fetch operation using the PC.
    c. The fetched instruction is stored in the CPU, in general, in an instruction Register (IR)
2. The instruction in the IR is interpreted by the control unit.
3. In addition to step 2, the operands are brought from the main memory to CPU registers.
4. The operation as interpreted at point 2, is performed on the data obtained in step 3.
5. Results of the operation in step 4 are stored in the CPU registers. In case the instruction specifies that the result of the operation is to be stored in the memory, then the result data in CPU registers are transferred to the specified memory locations.

The execution cycle is repeated as long as there are more commands to be executed. Sometimes the execution of a program is required to be terminated abnormally due to occurrence of certain error or other conditions, like division by zero. Thus, there may be need of a mechanism that can interrupt the execution

of a sequence of instruction. This mechanism is known as an interrupt mechanism, which uses an interrupt signal. On occurrence of an interrupt signal, CPU suspends the execution of next instruction to be executed, though it completes the execution of the current instruction. More specifically, when a request for interruption occurs, a move to an interrupt management process occurs. Interrupt management systems are the set of programs that are used to address the cause of the interruption and restores the system to the last instruction, where interruption was acknowledged.

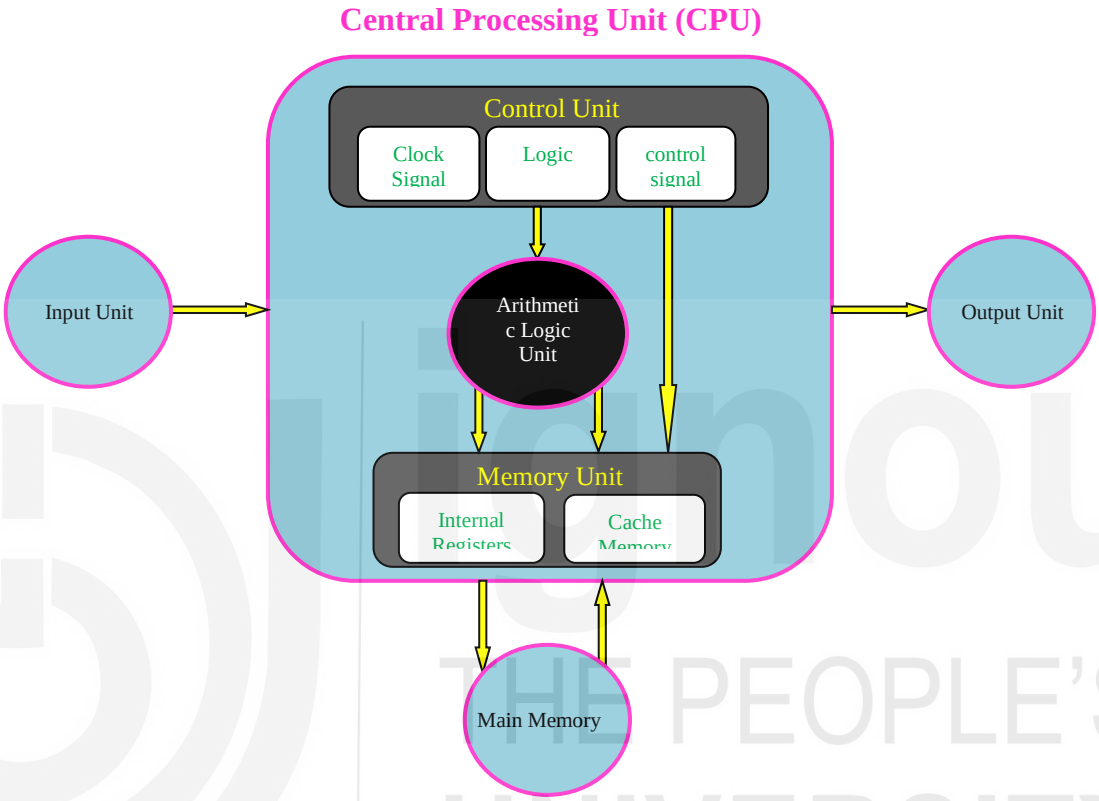


Figure 1.3: Central processing Unit and its components

1.3.2. Register Set

Registers are extremely fast, but temporary storage locations, within the CPU. Registers store the intermediate results, therefore, are used to enhance the CPU performance for performing arithmetic; logical or other operations. The names, number, type, and length of registers are different for different computers. In general, general purpose registers can be used for storing the data for an operation or address of an operand or any other purpose of various tasks in a program. Special purpose registers are limited to designated functionalities only. There are cases when some registers are used only for performing operations on data and cannot be used for operand address computations. These data registers length should be long enough to hold values for most types of data. Some machines allow two integrated registers to hold longer length data. Address registers are used for computing the address of an operand in the memory. This task may be assigned to specific types of register or the general address purpose register may be used for this purpose too. Address registers should be long enough to handle a very large address, which in general depends on the size of the memory. The number of registers in a computer affects the size of the instructions. Why? you will find answer to this question in Block 3. Some important registers are given below:

Register for Fetching and storing Instructions

In a computer system the instructions of a program are stored in the main memory. Two main registers, which are involved for transferring instructions from main memory to CPU are: Program counter (PC) and instruction/command or instruction register (IR). PC contains the address of the memory location from which the next command may be executed. Instructions are fetched to IR, so that it can be interpreted and executed.

Condition Registers and Status registers

In some computers, status registers or flag registers are used to store status information of various operations. Some computers have a special program status register (PSW). PSW contains the present status of the processor flags. A detailed discussion on flags is given in Block 3 and Block 4.

1.3.3. Datapath

As discussed earlier, a computer performs instruction execution using differnet components. But, how does the data and instructions get communicated from one component to other. This is achieved by datapaths. There can be two diferent types of datapaths:

(i) The datapaths which are internal to CPU: Such datapaths transfer two different categories of data. Data category, which includes data contained in different registers of ALU. The control data, which essentially pulls control signals from the datapaths for the use of control unit. In addition, the data is moved between two registers or between ALU and a register. This internal data migration is done by local buses, which may carry control information, instructions and addresses.

(ii) Externally, data may be transferred to and from registers to memory and I / O devices, usually using a special set of circuit called the **system bus**.

Internal data transfer between registers and between ALUs and registers can be done through various organizations including one, or more buses. Dedicated datapaths can also be used for data transfer devices between the CPU components on a regular basis. For example, Program counter register (PC) content is transferred to Memory Address Register (MAR) to fetch new commands at the commencement of each command cycle. Therefore, dedicated datapaths from PC to MAR can help speed up this part of command execution.

1.3.4. Control Unit

The control unit is primary unit which commands operations of the system by giving control signals to various units of computer. The control unit is also responsible to regulate internal and external flow of data from CPU. The data transfer from CPU to/from memory and CPU to/from I/O is also controlled by these signals. A continuous pulse sequence is generated by a system clock over a set period of time. This sequence of steps, identified as t0, t1, t2, ..., ($t0 \leq t1 \leq t2$, ...), is used to perform a specific command by enabling control signals in a specific order. The details on various types of control units and their operation are discussed in Block 3 of this course.

1.3.5 Memory Unit and I/O Devices

An interesting part of computer is the memory of a computer, which stores the data as well as instructions. Computer stores binary digits 0 and 1 called bits. However, bit is a very elementary, therefore, a meaningful combination of bits is generally, required to be stored in memory. A group of 8 bits is traditionally called a “Byte”. However present day data may include 16 bits (2 bytes), 32 bits (4 bytes), 64 bits (8 bytes) and so on.

Interestingly the size of the memory is represented as a byte in many situations, which was equal to one character in earlier data representation (Please refer to Unit 2 for data representation). Therefore, higher units are needs to measure the size of the memory. As computer is a binary device, an interesting combination as given in the following table is used to measure the memory capacity.

Unit	Equivalent to
1 Kilobyte (KB)	2^{10} Byte = 1024 bytes ≈ 1000 bytes
1 Megabyte (MB)	2^{10} KB = 20^{20} byte
1 Gigabyte (GB)	2^{10} MB = 20^{30} byte
1 Terabyte (TB)	2^{10} GB = 20^{40} byte
1 Petabyte (PB)	2^{10} TB = 20^{50} byte
1 Exabyte (EB)	2^{10} PB = 20^{60} byte
1 Zettabyte (ZB)	2^{10} EB = 20^{70} byte
1 Yottabyte (YB)	2^{10} ZB = 20^{70} byte
and beyond	

As stated earlier, the purpose of memory in computer is to store the instructions of a program and the related data. These instructions and related data to these instructions are fetched by the CPU, which then executes theses instructions. This memory is also called the Random Access Memory (RAM) as any location of the memory can be addressed randomly. Details on memory system are given in Block 2.

I/O devices are used to input data and programs, e.g. a keyboard can be used to input a program. Present day computer use pointing devices and screens to select options from Graphical user Interfaces (GUI) like Microsoft Windows. The output devices like printer, monitor etc. display the output either in printed (also called hard copy) form or are displayed on the details on I/O devices are also given in Block 2.

1.3.6 What is an Instruction?

In the entire discussion on CPU, one term was used consistently - an instruction. What is an instruction in the context of a computer system?

In general, you write programs in a high level computer language like C, Python, JAVA, C++ and so on. Most of these programming language requires you to compile the program into an object program, which is linked with library programs and loaded in the memory of the computer. A program also include some basic data and I/O from keyboard or files. These loaded programs contains binary instructions, which operate on binary data.

Some of the common high level statements include arithmetic instructions like addition, subtraction; decision like if-then-else; repetitive loops which also involves decisions like while, for etc.; and procedure/function calls. In all these statements few common characteristics stand out, which are made part of binary instructions as given below:

(1) Each binary instruction may define one operation using a binary operation code also called opcode.

(2) Each instruction may have one or more operands which may be data itself or can be used to compute the address of operand. A computer instructions, depending on machines, may consist of one to three operands.

(3) The result of operation of machine instruction can be stored in some machine register.

(4) Some instructions, like branch, function call etc., result in transfer of execution to a new instruction, which may be at a different address in the program. This, can be achieved by changing the value in program counter register, which stores the next instruction to be executed.

Binary instruction design of a computer system is a very complex task. In fact the machine instructions of different computers are different, that is why you require a separate compiler for a separate type of computing machine. A detailed discussion on computer instructions is given in Block 3.

Check Your Progress 1

1) State True or False

		T/F
a)	Byte consists of 8 bits, and it may represent a character.	☐
b)	A character representation requires at least one byte.	☐
c)	One bit is an independent unit and can be accessed independently.	☐
d)	A machine can have two paths called internal and external.	☐

2) Explain the concept of an Instruction.

……

……

……

3) Why does a computer need CU, ALU, memory and I/O devices?

…… ……

……..

……..

1.4 HOW ARE INSTRUCTION EXECUTED?

After getting a fair idea about basic structure of computer system, let us now understand about execution of a program. To learn about how a computer executes a program, let us take an example from high level language.

Consider the following program segment of a high level programming language:

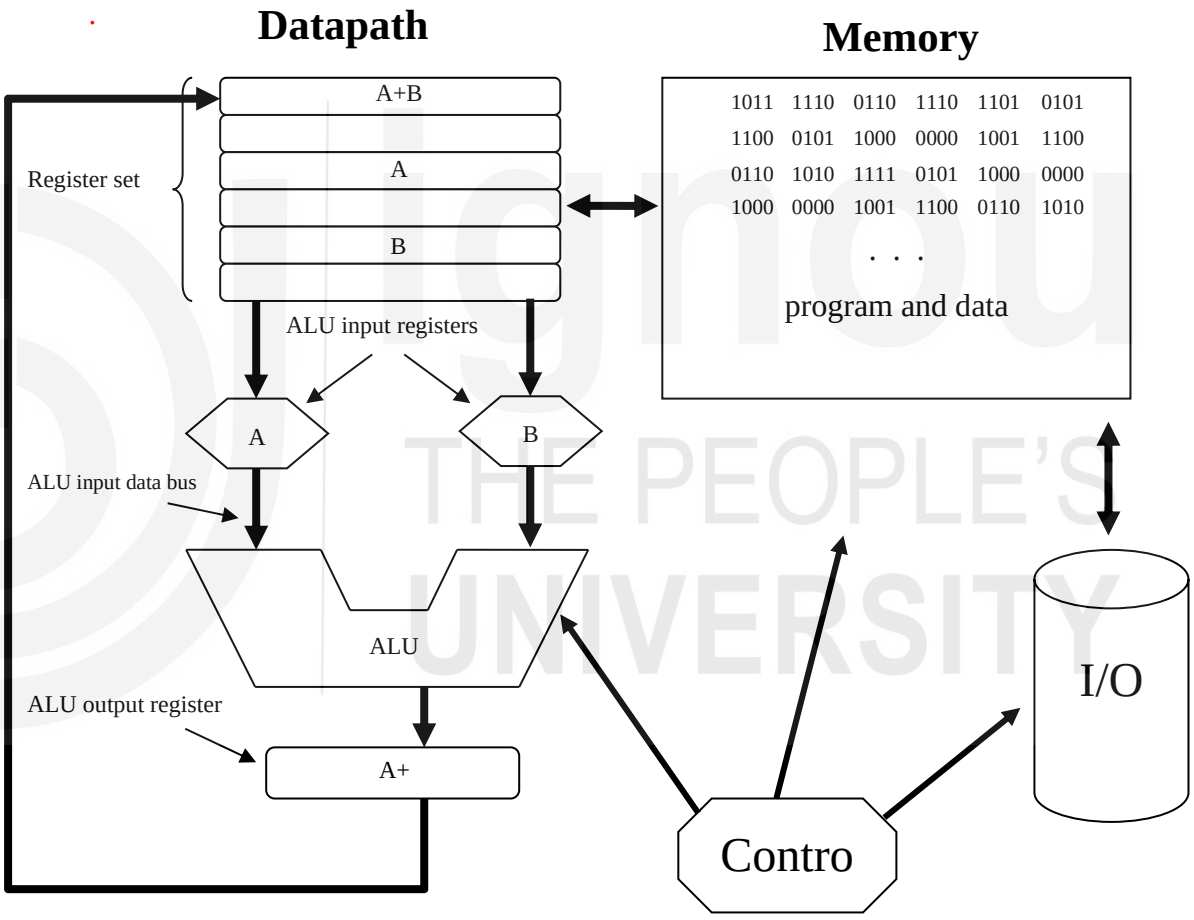
```
int x=10, y=5, z;
z=x+y;
```

Please note that the instruction *int x=10, y=5, z;* will allocate three memory locations of type integer (how big will be one integer). The *x, y* and *z,* in the context of programming languages, are called variables. This instruction will also assign integer value 10 in first location, identified by variable name *x* and integer value 5 in second location identified by variable name *y*.

The instruction will also allocate a third memory location named *z*. Thus, first instruction, technically will be translated to create three integer location named *x*, *y* and *z*. Please note that these locations, when loaded in the memory will be identified as three separate addresses. The second instruction *z=x+y;* will be executed by the CPU to produce the desired results. But, how does these set of instructions be executed by computer? As a first step, a compiler program will be executed by computer and all the High level programming statements will be translated to a machine language program consisting of data in the form of variable locations and values, and instructions as binary operation codes and operand addresses.

Please note in the C program segment, the declaration results in creation of variable locations, with data values.

It is the instruction *z=x+y;* which gets converted to one or more machine instructions, which will have binary codes indicating addition operation, opened address on which this addition is to be performed, and where the result will be stored. Assuming a typical machine is shown in Figure 1.4



Fetch – Decode – Execute Cycle

Figure 1.4: "Instruction and data format of an assumed machine[https://www.cise.ufl.edu/~mssz/CompOrg/CDA-lang.html"]

Please notice the role of ALU registers. They get values of locations *x* and *y* first to be added by the ALU and the answer of this addition is stored in a register, which is sent back to the memory location *z*.

In general, a machine consists of several registers as shown in figure 1.5. The details on these registers will be discussed in Block 3 of this course.

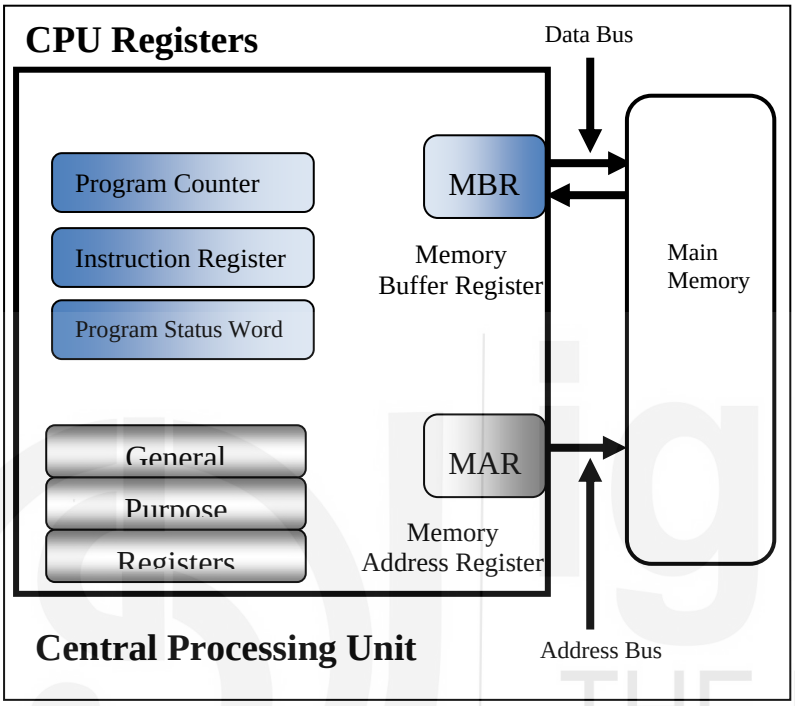


Figure 1.5: "Central Processing Unit with registers [http://digitalthinkerhelp.com/what-is-cpu-processor-register-in-computer-its-types-with-functions/]"

1.5 INSTRUCTION CYCLE

A program is having a set of instruction which are stored in computer's memory. In general, the instructions of a program are executed sequentially by the processor. A computer system executes an instruction in the following four phases:

1. Fetching.
2. Decoding
3. Read the operands from the memory.
4. Execution.

These steps are explained in details in Block 3. In this unit they are just being introduced.

Fetching the instruction: An instruction is placed in the memory location or locations in RAM. Instruction fetch will bring this instruction into the Instruction register. This step requires the information about where the instruction is in the memory. In general, PC register contains this information.

Decode the Instruction: Decoding the instruction requires to interpret the operation code of the instruction, this will be followed by finding locations of the operands in the memory.

Read the operands from memory: Once the addresses of the operands are known, they are brought into the CPU registers, such that the required operation may be performed.

Execute the Instruction: Finally the instructions are executed and results are stored in the local temporary locations.

The process is shown in figure 1.6

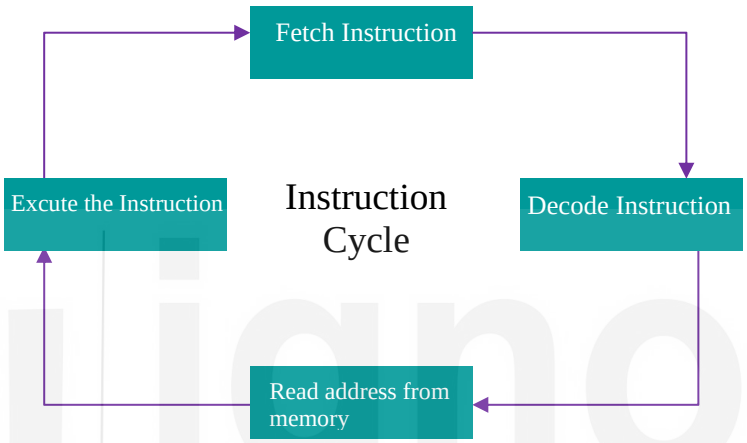


Figure 1.6: Instruction Cycle [https://www.javatpoint.com/instruction-cycle].

Can a program be interrupted while it is getting executed? A program may be required to be stopped, during the execution due to occurrence of errors or occurrence of completion of certain other important – I/O events. This process has been explained in the next subsection:

Interrupts

The meaning of word interrupt is to stop the ongoing activity. In computer, an interrupt, stops execution of next instruction in the instruction cycle, once the execution of ongoing instruction is completed. Why? Your will get an answer with this question in Block 3.

Why does an interrupt occurs in a computer?

Interrupt occurs due to:	**Example**
Program instruction execution itself. Causes an interrupt.	❑ Division by Zero occurs ❑ The result exceeds the limit of the number allowed. ❑ Security violations.
Clock initiated.	• Expiry of time limit of a program.
I/O devices	❑ Input/Output starting or completion ❑ Error due to an I/O device
Hardware generated.	❑ Memory errors

Figure 1.7: Example of Interrupts in a Computer.

Interrupt mechanism is a very useful mechanism for increasing the efficiency of program execution. The instruction cycle many continue, till the time an interrupt occurs. As a result of an interrupt, the CPU knows that some event has occurred and then CPU stops the execution of the current program and goes on to process that event. CPU then uses the following steps to process the interrupt:

- The CPU identifies the source of the interrupt.
- CPU executes and Interrupt Servicing Routine (ISR), which services the event that has occurred.
- Meanwhile, the program which was being executed is moved to a hold state.
- Once the CPU completes the ISR, i.e. executes it till its completion, it resumes the execution of program it has put on hold.

Interrupts and Instruction Cycle

The interrupt process is summarised below:

- CPU is executing a program say "X". and is in the decode stage of i^{th} instruction of the program X.
- Assume an interrupt due to an event occurred at this time.
- CPU waits till the i^{th} instruction execution is complete, and then stores the register values & PC, which has address of $(i+1)^{th}$ instructions into memory or special storage area.
- CPU indentifies the interrupt and executes the interrupt servicing program till it is complete.
- CPU then return to execution of the $(i+1)^{th}$ instruction by restoring the register and PC.

Thus, after interrupt processing, the execution of the interrupt program is resumed. Figure 1.8 shows Interrupt cycle.

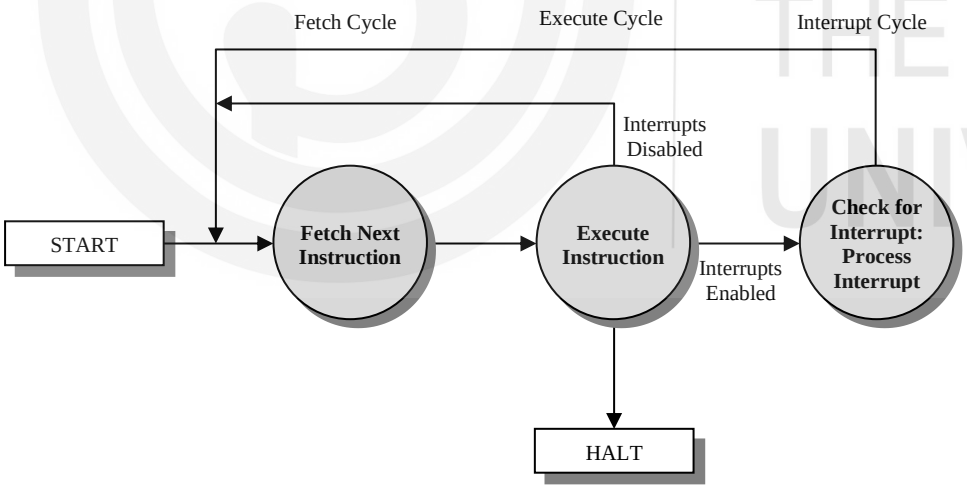


Figure 1.8: "Instruction Cycle with Interrupt Cycle [https://www.ece.ucdavis.edu/~vojin/CLASSES/EEC70/W2004/presentations/Ch_03.pdf"]

Please note an important point in the figure 1.8, which is - interrupt enable and interrupt disable conditions. If CPU is executing very important instruction, the response to an interrupt can be disabled. In this situation interrupts will not be processed or acknowledged by the CPU till they are enabled again.

Check Your Progress 2

1) **State True or False** T/F

i) Assume a computer has one-byte long instructions, its PC contains a decimal value 205 and only one byte is fetched from the memory at a time. For this machine, after fetching an instruction, the value of PC will become 206.

ii) PC register is needed to fetch the data from memory. ☐

iii) A clock signal cannot cause as interrupt. ☐

iv) The interrupt are answered by the hardware. ☐

v) The processor processes one interrupt, even if, multiple interrupts may occur at a time. ☐

2) Explain the term interrupt and its cause.

.....

.....

.....

3) Explain the interrupt processing in a computer.

.....

.....

.....

1.6 VARIOUS COMPUTER ARCHITECTURE

Architecture of a computer contains procedures or processes to explain the user about the functionality of a computer system. This is analogous to design of a building, i.e. the construction of buildings is tailored to the needs of the user by taking into account the cost threshold. The original design is designed on paper. Likewise the computer, after it was constructed using the logic of transistors and integrated circuits, the construction was tested and built in a hardware way. Computers can be evaluated based on their performance, efficiency, reliability, and cost of computer software, which works with software and computer technology standards. In this section, you will learn about various computer architectures.

1.6.1 von-Neumann Architecture

John von – Neumann has invented this architecture. Several computers are based on the construction of this architecture and other improvements on this architecture. The basic building blocks of this architecture are-

1. CPU
2. Memory Unit
3. I/O Devices
4. Connection Structure

Each memory unit having multiple locations will have different addresses. Same memory is having instructions and data, which has multiple locations with each location having unique address. A computer program consists of instructions and data. CPU process the data stored in the memory or obtained term input devices as per the instruction the program. The results are stored in the memory or output devices. The data processing operation in CPU use registers.

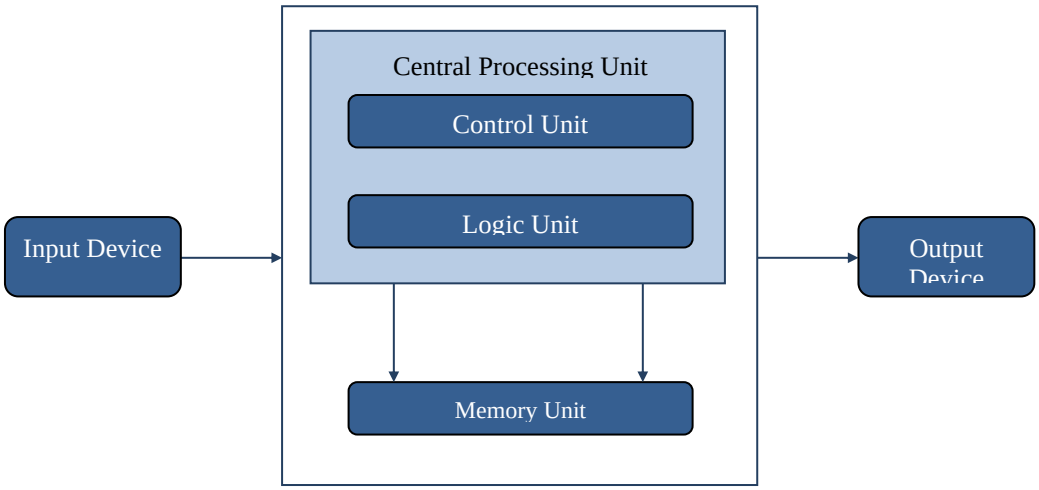


Figure 1.9: Von Neumann Machine Ref: www.educba.com

Buses (address bus / data bus / control bus) are used for communicating the address, data and control signals. The input device fetches data or commands and the Central processing unit (CPU) executes it. When the task is completed, the result is stored on output device.

As discussed in the previous section, a computer system executes instruction/commands using processor and memory while the registers provide the processor's temporary storage requirements, whereas memory works with medium-term and long-term data storage requirements. Any data processing system consists of a sequence of instructions that enable the processor to perform the functionality you want. These instructions operate on data, which may be input from input devices. The processed data is the output of the data processing systems. We may store instruction and data together or separately. In the Princeton architecture, data and instruction share the same memory space. In Harvard architecture, system or instruction memory space differs from data memory space. Princeton configuration can lead to easy hardware connection to memory, as only one connection is required. Harvard configuration requires, dual connections, can simultaneously read instruction and operand data, so it can lead to improved performance. Most machines have a Princeton design. Intel 8051 is a well-known Harvard architecture. Figure 1.10 is a diagrammatic representation of Harvard architecture, whereas figure 1.11 represents the Princeton architecture of a computer.

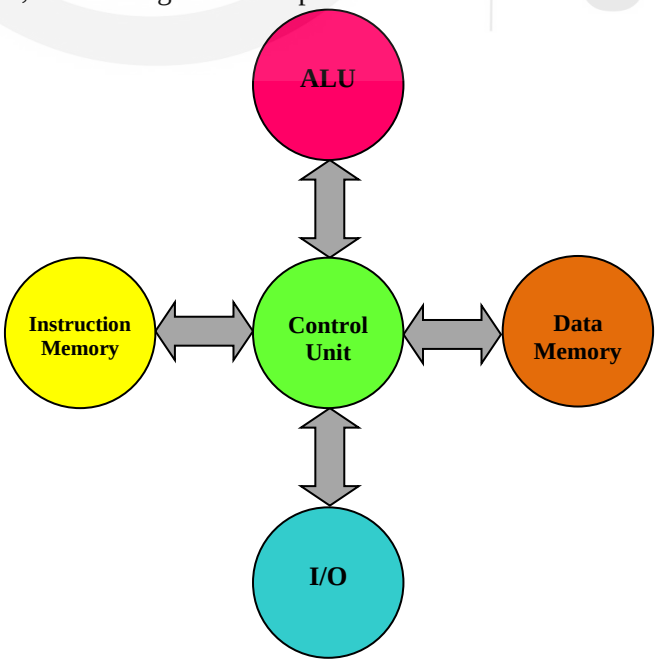


Figure 1.10: Harvard Architecture

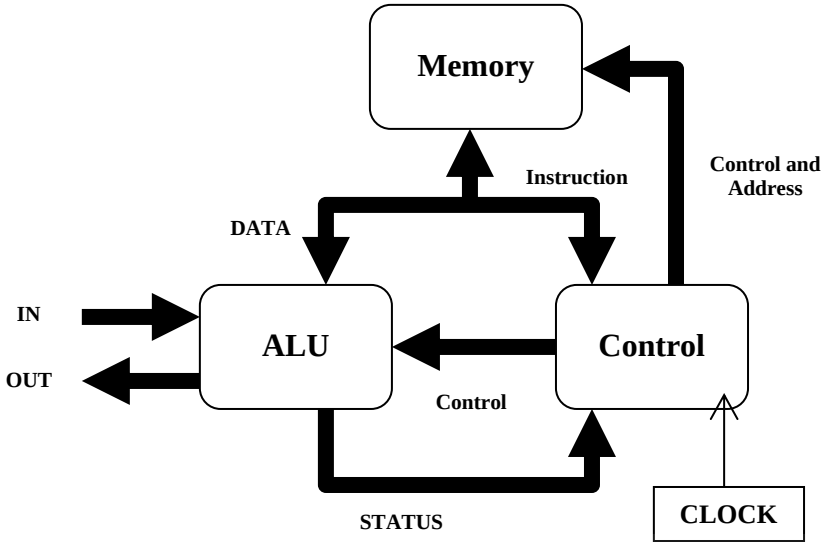


Figure 1.11: Princeton (von Neumann) Architecture

Stored System Computers –

Such computers can be programmed to do various tasks and also, applications are kept on them that is why they are named as Stored System Computers. This concept of a stored system was presented by John von Neumann. In such computer systems, a single memory is used to store programs and data and is treated in the same way. A computer built in this way will be easy to imitate.

The von Neumann architecture, also called as Princeton model, is a computer architecture designed by the 1945 definition of John von Neumann and others in the First Draft of a Report on the EDVAC. The draft defines the design of a digital computer. As per this document a general purpose computer should contain:

- a Processing unit comprising of arithmetic unit, logic unit and processor registers
- A control unit comprising of command register and a program counter register
- Data storage in data memory
- Limited external storage
- Input and output methods

The name "von Neumann architecture" has been attributed to any computer where the stored program and data are not transmitted simultaneously because they share the same medium of data transfer, called system bus. This is called a von Neumann bottleneck, as it restricts simultaneous access of data and instructions, thus, may result in reduction in the performance of the system. However, the single path simplifies the design of the von Neumann machine. In comparison, a Harvard architecture machine uses one dedicated bus each for address memory and data memory respectively, therefore, is more complex.

The basic characteristics of Von Neumann architecture are summarized below.

(1) A computer has the following operational units:

- The control unit (CU), which interprets commands to be executed, and generates control signal which instructs other units about how to perform the operation.

- Registers and other circuitry.
- Input/output system, which are used for input of instructions and output of results.
- A memory, which stores instructions as well as data (both).
- The inter connection structures, which allow communication between various components.

(2) von Neumann machine works on a concept of stored program, which requires that data and commands both must be loaded into the memory of the main computer prior to execution &called as Random Access Memory (RAM).

(3) The instruction/commands in a Von Neumann machine are executed in a sequence, therefore, to change this order of execution, a special instruction may be used.

(4) A Von Neumann machine has a single path from control unit to main memory so only one of two-data or instruction can be transferred between the two at a time.

1.6.2 Harvard Architecture

When data and instructions are kept in separate memory then it is Harvard architecture. Data is fetched from one memory location and instruction is retrieved from a separate location. Pipelining can be done in this architecture. It is complicated to design this architecture. The CPU can fetch, decode and execute instructions and data. The construction of this architecture has separate access for codes and data address.

The modified Harvard architecture is similar to the Harvard architecture and has a standard address space for a separate data and instruction cache. It has digital signal processors that can handle audio and video data efficiently. It also has microcontrollers, the processing circuits that process small number of applications and has small data memory and speed up processing by performing the commands and data access simultaneously.

Figure 1.12 shows different connection paths for modified Harvard architecture. All these four units are contained in CPU. It can perform simultaneous input / output operations and has a separate mathematical and logic component.

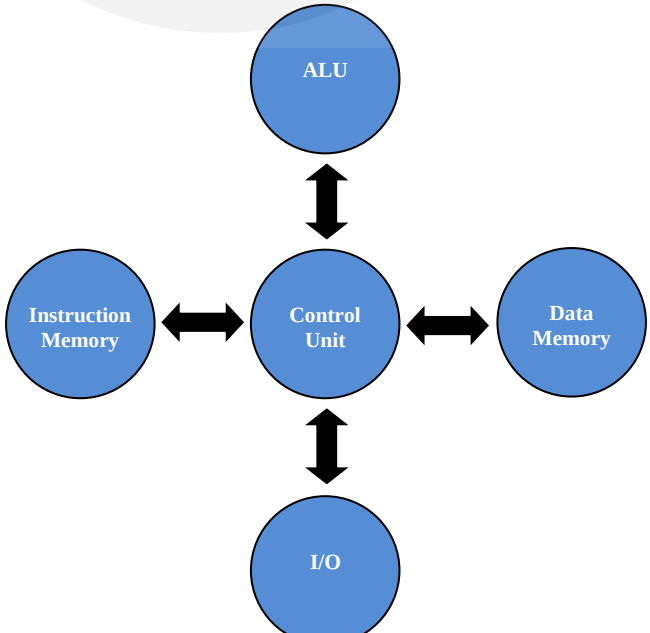


Figure 1.12: The modified Harvard architecture.

1.6.3 Instruction Set Architecture (ISA)

Instruction set architecture (ISA) defines a set of instructions that are to be supported by a processor. From system point of view, ISA does not care about certain computer implementation details. It is only about setting up or collecting the basic functions that a computer should support. Some of the common examples of ISA are Intel upgraded x86, ARM, MIPS, and AMD.

An ISA includes different kinds of instructions, sizes of differ instruction formats. These concepts will be explained in more details in the Block 3. The following example of MIPS ISA very briefly defines the description that should be supported by an ISA. You may refer to further readings for more details in this ISA.

1. ISA defines the types of commands that the processor will support.

Depending on class of work they are doing MIPS Instructions are divided into three types:

- Arithmetic / Logic Instructions
- Data transfer instructions
- Branch and Jump Instructions

2. Maximum length of each type of instruction has been defined in ISA.

3. I*nstruction format for each type of instruction has been defined in ISA.*

Various formats in MIPS ISA:

- R-Instruction format
- I-Instruction Format
- J-Instruction Format

As every format is having separate command coding schemes in terms of operation code, number of operands etc., so they are required to be understood differently by the processor.

The following diagram shows the hierarchy of abstraction of Architecture:

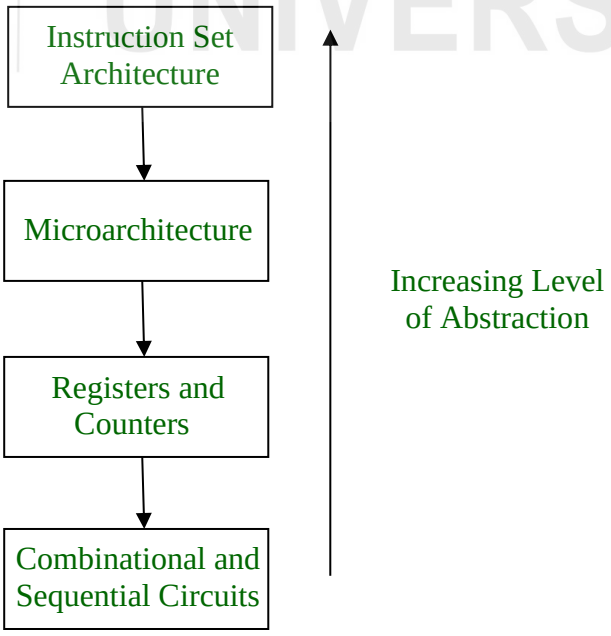


Figure 1.13: The Abstraction Hierarchy

As Micro-architectural standard is placed just below the ISA standard and is therefore deals with the implementation of computer-based core functions as suggested by ISA.

What is the need to differentiate between Micro-architecture and ISA? It is required to establish and maintain system compliance across all ISA-based hardware applications. Adapting different machines to the same set of basic commands (ISA) allows the same system to run smoothly on multiple different machines, thus, making it easier for editors to write and store code for many different machines simultaneously and efficiently.

This abstraction hierarchy supports flexibility, which is why first the ISA is developed and then various micro architectures are created that are compatible with this machine-operated ISA. The micro-architecture is implemented using various logic circuits.

1.6.4 RISC

RISC formulation is being used by ARM core. RISC is a strong design and it delivers simple commands in a single cycle with high clock speeds. RISC targets to reduce the hardware complexity of instructions because hardware has less flexibility in comparison to software. As a result, the structure of the RISC places expectations on the compiler. Conversely, complex instructions (CISC) rely heavily on hardware for operational performance, and as a result CISC commands are complex. Figure 1.14 shows this major difference.

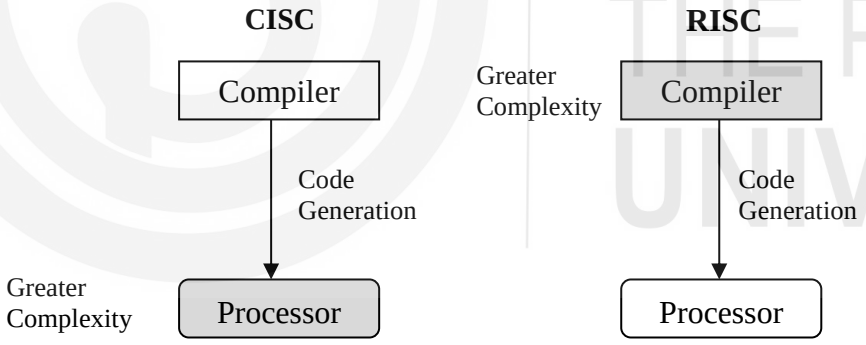


Figure 1.14: RISC vs. CISC

CISC emphasizes the complexity of hardware. The RISC emphasizes the complexity of the compiler.

The RISC philosophy is based on four main building codes:

1. *Instructions* - Number of instructions has been reduced in RISC. An instruction of RISC is executed in several segments. Instruction execution is overlapped in these segments (called pipeline segment). Each segment takes about one clock cycle time. For example, a processor performs complex tasks by combining several simple commands. Each command is of a limited length to allow the “pipeline to pick up future commands before deciding on current commands”. CISC instructions are usually of varied length.

2. *Pipes* - The processing of the instructions is divided into smaller units that can be made similar to pipes. Appropriately the pipe continues one step in each cycle of execution.
3. *Registers* – Large registers for general purposes have been included in RISC which may contain data or address. CISC processors have been provided with specific registers for specific tasks.
4. *Load-store creation* - The processor operates on data managed in registers. Memory access is expensive, so separating memory access to data processing provides an opportunity. With CISC design the data processing functionality can work in direct memory.

The RISC is explained in more details in Block 3.

1.6.5 Multiprocessor and Multicore Architecture

One of the ways to improve computer system efficiency is to have processors with faster clock speeds. But, owing to the thermal wall problem, when the clock speeds started hitting the thermal barrier, focus was on to extract maximum work out of the available system as an alternative of increasing processor speed.\

One of the ways to introduce parallelism in computers is to have more than one processor available in the single computer system referred to as multiprocessor system. It is very likely that at times the job (application) to be executed on a processor can be divided into various tasks with two or more tasks being capable of running independent of each other. Multiprocessor system corresponds to the architecture in which rather than having only one processor we have more than one processor available on the chip. Thus, if the job demanding execution comes with independent tasks, a processor can be dedicated to each task in order to exploit the parallelism in the job. In a way, multiprocessor system can be looked as a solution to offer hardware parallelism to match the available software parallelism in the job. This architecture corresponds to instruction level parallelism allowing independent instructions (or sub tasks) being assigned to different processors in the multiprocessor system allowing their parallel execution. If all the processors are same, the system is referred to as a Symmetric Multi-Processor (SMP). The multiprocessor architecture shares the computing environment viz. memory, OS, system clock etc.

In a conventional computer system, you had only one CPU available, which is responsible for the job execution focusing on only one task at a time. However, now a days, you hear about terms like a computer with quad core processor or an octa core processor. This architecture is referred to as the multicore architecture corresponding to both instruction level and thread level parallelism in a uniprocessor system. In this case, a single processor CPU is fabricated to have more than one CPU cores inside it. Each core acts an independent processing unit (CPU) and can execute independent threads, if permitted by the program. Thus, a quad core processor comprises for four cores whereas an octa core processor has eight cores capable of working independently on the same processor.

The difference between multiprocessor architecture and multicore architecture lies in the fact that multiprocessor architecture has more than one processor available whereas multicore processors have a single processor with multiple cores used as independent CPUs. Thus, multicore architecture is multiprocessing in a single packaged unit. Both multiprocessor and multicore architectures correspond to the MIMD category of Flynn's classification with multiprocessor architecture being more pure form of parallel processing. Practically, for improved performance, you can have multiprocessor machines with each processor having multiple cores.

1.6.6 Mobile Architecture

Nowadays, mobile handsets are no longer a means of enabling conversations but have turned into a small but powerful computer having many features like fast memory, support for software for numerous applications like chatting, document editing, entertainment, news etc. These phones have become really handy equipped with efficient performance and it has become really difficult to have a demarcation between a computer and a mobile phone.

Just like any computer system, mobiles too have an input unit, a Central Processing Unit (CPU) and an output unit as shown in Figure 1.15. The input unit could be a keyboard based or a touch screen based input mechanism and the output unit can be the screen or audio or video system. However, the CPU in the case of mobiles can be considered to be having two processing units viz. Communications Processing Unit and an Applications Processing Unit to cater to mobile calls and handling various applications available in the form of Apps. In addition, some other functional units like Display management, Memory management, Power management, Data management can be considered to be associated with the mobile system architecture but the discussion is beyond the scope of this course. All these units, under the mobile CPU can be understood to be working under the control of a mobile Control Unit for control and synchronization purposes.

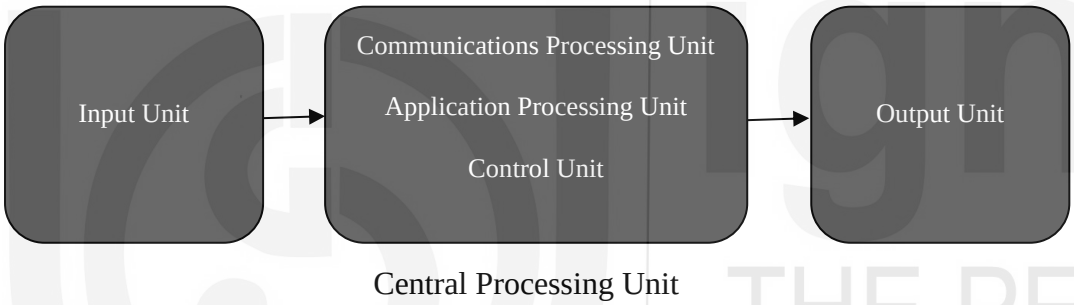


Figure 1.14: Block diagram of Mobile System

Check your progress 3:

Q1: Differentiate between the micro-architecture and ISA.

.....

.....

.....

.....

Q2: Differentiate von Neumann architecture and Harvard architectures.

.....

.....

.....

.....

Q3: Differentiate multiprocessor and mullticore processor

.....

.....

1.7 SUMMARY

This unit introduces you to some of the basic architectural features of a computer system. A computing device consists of some basic units, like processing unit, which includes control unit and arithmetic logic unit, memory, input and output devices and data paths. This unit also explains the concepts of various computer architecture, which represents, how these various component of computer can be used to execute an instruction set. The unit also introduces to the concept of ISA, multiprocessor and multicore and mobile architecture. A detailed discussion on the various aspects of Computer organization has not been included in this unit, they will be discussed in the subsequent blocks and units. This unit also introduces you to the concept of instruction and its execution by a computer.

1.8 SOLUTIONS/ANSWERS

Check Your Progress 1

1) (a) True (b) True (c) False (d) True

2) An instruction in a computer system is a binary code, which is interpreted by the control unit of a computer and it communicates the following information to CPU:

- the operation which is to performed
- the operands and their location
- the possible information of storage of results

etc.

3) The CU controls - (1) the sequence of instruction execution, (2) the communication through the data paths, (3) the communication among different units (4) the operation to be performed by ALU (5) what to do in case of errors etc. Thus, in general, CU is responsible for complete control of a computer

The ALU primarily performs the arithmetic and logical and shift operations on the specified data. Memory stores the data as well as instructions and I/O devices are used to input data or instructions as well as display of results.

Check Your Progress 2

1) (i) True (ii) False (iii) False (iv) False (v) True

2) An interrupt is the process of stopping the execution of currently executing program due to occurrence of some event, which requires attention of the CPU. The causes of the interrupt may be division by zero; arithmetic overflow, program address space violation, starting or completion of I/O, errors etc.

3) Interrupt can be processed by suspending the execution of currently executing program and executing the ISR of the interrupting event. It may be noted that interrupts can be acknowledged only if the interrupts are enabled.

Check your progress 3:

1) An ISA defines the set of instructions, but a micro-architecture provides details on how various functions will be performed by various units. An ISA is a top level design, whereas micro-architecture is a detailed design. A faster compatible computer can be designed with same ISA but more efficient

micro-architecture. ISA simplifies the job of programmers, who may use same instruction set to write programs.

2) The von Neumann architecture uses same memory for data and instruction, the Harvard architecture may have separate memories for instructions and data. In von Neumann architecture data and instructions cannot be accessed simultaneously but in Harvard architecture it is possible. The von Neumann architecture is simpler to implement in comparison to Harvard architecture.

3) A multiprocessor system may have multiple processors, whereas multicore processor have multiple CPUs in a single processor. Today, a multiprocessor system can be constructed using multicore processor chips. Both these technology are of the type MIMD, which is multiple instruction and multiple data form of multiprocessing, thus, allow multiple sections of programs being executed at the same time operating on separate data.

ignou
THE PEOPLE'S
UNIVERSITY