**What is a computing?**

**=**Computing is a goal-based operation of a computer. Generally, it means a task that needs computer for completion. It can mean handling, transferring, exchanging, processing data and then producing the output of the task.

Input-> Computing (Goal based operation + processing) -> Output

**What is a computer?**

=A computer is an electronic machine that accepts data from the user, processes the data by performing calculations and operations on it, and generates the desired output results.

**Digital and analog computer:**

A digital computer uses distinct values to represent the data internally. All information is represented using the digits 0s and 1s.

Analog computer is another kind of a computer that represents data as variable across a continuous range of values.

**Characteristics:**

* Speed
* Accuracy
* Diligence (Not fatigued)
* Storage Capability
* Versatility (More than one work at the same type)

**Human is more intelligent than computer:**

Computer can only perform tasks that it has been programmed to do. Computer cannot do any work without instructions from the user. It executes instructions as specified by the user and does not take its own decisions.

**Generation of Computers:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **First generation**  **(Vacuum tube based)** | **Second generation (Transistor based)** | **Third generation (Integrated circuit based)** | **Fourth generation**  **(Microprocessor based)** | **Fifth generation**  **(AI based)** |
| **Timeline** | 1942-1955 | 1955-1964 | 1964-1975 | 1975-1989 | 1989-Present |
| **Hardware** | **\*Circuitry=**vacuum tubes  **\*Memory=**magnetic drums  **\*Input=**punched card + paper tapes  **\*Output=**printouts | **\*Circuitry=**transistors (semiconductor)  **\*Primary memory=**magnetic core technology  **\*Secondary storage=** magnetic tapes +magnetic disks  **\*Input=**punched card  **\*Output=**printouts | **\*Circuitry=Integrated circuit (IC) (**speed + efficiency)  **\*Primary memory=**magnetic core technology  **\*Secondary storage=** magnetic tapes +magnetic disks  **\*Input=** keyboard  **\*Output=** monitor | **\*Circuitry=Microprocessor (LSI + VLSI)**  **\*Primary memory=**Semiconductor memory (RAM, ROM)  **\*Secondary storage(smaller)=** magnetic disks  **\*Development of pointing devices like mouse, and handheld devices**. | **\*Circuitry=**  **AI**  **\*ULSI/SLSI** Ultra/Super Large Scale Integrated chip.  **\*** **Parallel processing (**several instructions to be executed in parallel, instead of serial execution) |
| **Programming language** | machine language | assembly language | High-level languages (FORTAN, BASIC, COBOL, Pascal, C) | High-level languages (Python, C#, Java, JavaScript, Rust, Kotlin, etc.). | understand natural language (human language). |
| **Software** |  |  | keyboard and monitor were interfaced through the operating system. Operating system allowed different applications to run at the same time. | MS-DOS and MS- Windows were developed during this time. This generation of computers supported Graphical User Interface (GUI). |  |
| **Computation time** | milliseconds | microseconds | nanoseconds | picoseconds |  |
| **Speed** | **Low to high** | | | | |
| **Extra** | The first-generation computers could solve one problem at a time. | Used the concept of a stored program, where instructions were stored in the memory of computer. | individual components of the computer were not required to be assembled manually | Development of Network – a group of two or more computer systems linked together**.** | Expert System (ES), Natural Language Processing (NLP), speech recognition, voice recognition, robotics, etc. |
| **Power consumption+Heat + Size** | Each tube half a watt. Per tube=0.5 w | One tenth of a tube. Per transistor=0.05 w | **High to low** | | |
| **Disadvantage** | The first-generation computers used a large number of vacuum tubes and thus generated a lot of heat. They consumed a great deal of electricity and were expensive to operate. The machines were prone to frequent malfunctioning and required constant maintenance. Since first generation computers used machine language, they were difficult to program | | | | |
| **Example** | Universal Automatic Computer (UNIVAC), Electronic Numerical Integrator And Calculator (ENIAC), and Electronic Discrete Variable Automatic Computer (EDVAC) | IBM 1401 and CDC 1604 | IBM 370, IBM 360 PDP-11 PDP-8 | IBM PC and it’s clones, Apple ||, Cray-1 | IBM PC and it’s clones, Apple ||, Cray-1  Desktops, laptops, tablets, smartphones, etc. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Supercomputers** | **Mainframe computers** | **Minicomputers** | **Microcomputers** |
| Supercomputers are most powerful computing machines on the planet and the ultimate engine of the digital age. |  | Minicomputers are standalone mid-sized machines that fall somewhere between smaller mainframe and powerful microcomputers. | Microcomputers are general-purpose computers that are mostly used for daily work that performs all logic and arithmetic operations. |
| The speed of supercomputer is in the range of 100-900 MIPS. The speed of a supercomputer is generally measured in FLOPS (FLoating point Operations Per Second) | Mainframe computers can have a processing speed in the range of 3-4 MIPS to as high as 100 MIPS. | They have speed in the range of 10-30 MIPS. | Microcomputers can have a processing speed in the range of 70 to 100 MIPS. |
| They are primarily used for large and complex mathematical computations in the field of scientific research and forecasting along with scientific simulations, fluid dynamics calculation, nuclear energy research, etc. | Mainframe computers are used as a storage for large databases and serve as a maximum number of users simultaneously in the field of finance and health. Mainframe computers are used in organizations like banks or companies, where many people require frequent access to the same data. | The main purpose of the mini computer is to fulfill the computing needs for several people from small to medium-sized business environment. They are used for real-time applications in industries, research centers, etc. | Microcomputers are used in offices, education systems, database management systems, word processing, etc. |
|  | Mainframe computers support thousands or millions of users simultaneously. The user accesses the mainframe computer via a terminal that may be a dumb terminal (input +output), an intelligent terminal (input +output +process) or a PC. | Minicomputers support (4-200 users) hundreds of users at a time. The users can access the minicomputer through their PCs or terminal. | Minicomputers support single user at a time. |
| Supercomputers have Linux and their variant operating systems. | Mainframe computers can have multiple operating systems simultaneously. |  |  |
| **Size + cost:** High to Low  **Processing speed + Storage**: Low to high | | | |

**Thrashing:** Low performance, exhausted RAM, saturation point, Time required for inter-communication is greater than the time required for calculation**.**

Thrashing occurs when a computer’s virtual memory is overused leading to a constant state of paging and page faults inhibiting most application-level process. Thrash is the poor performance of a virtual memory (or paging) system when the same pages are being loaded repeatedly due to a lack of main memory to keep them in memory. Thrashing is when the page fault and swapping happens very frequently at a higher rate

* **Page fault:** Every program is divided into some pages. A page fault occurs when a program attempts to access data or code in its address space but is not currently located in the system RAM.
* **Swapping:** Whenever a page fault happens, the operating system will try to fetch that page from secondary memory and try to swap it with one of the pages in RAM. This process is called swapping.

**Real time application:** Live video streaming, Gaming

A real time application is one for which timing is critical. Real time means that a human will not recognize the computational period of the program. In other words, the response appears to be instantaneous.

Hard real time is associated with automated control systems that must respond faster than a human can respond. Most humans cannot respond to an event in less than 100 milliseconds. Many control applications require responses far faster than 100 milliseconds.

**What is a computer system?**

= A computer system is a basic, complete and functional hardware and software setup with everything needed to implement computing performance. It includes four distinct parts:

* Hardware: The electronic devices in a computer system that can be seen and touched or the mechanical parts that make up the computer as a machine. The hardware consists of physical devices of the computer. The devices are required for input, output, storage and processing of the data. Keyboard, monitor, hard disk drive, floppy disk drive, printer, processor and motherboard are some of the hardware devices.
* Software: Software is an organized sets of instructions used to operate computers and execute specific tasks. Here using a program means running or executing the program. When a program is on executing situation, it is called process.
* Data: Data are isolated values or raw facts, which by themselves have no much significance. For example, the data like 29, January, and 1994 just represent values. The data is provided as input to the computer, which is processed to generate some meaningful information. For example, 29, January and 1994 are processed by the computer to give the date of birth of a person.
* Users: Users are people who write computer programs or interact with the computer. They are also known as skin ware, liveware, human ware or peopleware. Programmers, data entry operators, system analyst and computer hardware engineers fall into this category.

**What is input-process-output concept?**

=A computer is an electronic device that (1) accepts data, (2) processes data, (3) generates output, and (4) stores data. The concept of generating output information from the input data is also referred to as input-process-output concept.

**Computer System Hardware:**

The computer system hardware comprises of three main components —

* Input/Output (I/O) Unit
* Central Processing Unit (CPU)
* Memory Unit.
* Bus
* Adapter

**CPU:**

Central Processing Unit (CPU) or the processor is also often called the brain of computer. CPU consists of Arithmetic Logic Unit (ALU), Control Unit (CU) and a set of Registers. They are connected to each other through *internal CPU bus* and CPU connects with other components through *system bus*. The CPU is fabricated as a single Integrated Circuit (IC) chip with numerous pins is known as the microprocessor.

**ALU:**

ALU performs all the arithmetic and logic operations on the input data. The computer ALU performs both bitwise (A **bitwise operator** may be used in programming for operating on the individual bits of binary values. For example, NOT 010 would return 101.) and mathematical operations on binary numbers and is the last component to perform calculations in the processor.

ALU is a complex digital circuit that consists of two units—arithmetic unit and logic unit.

• The arithmetic unit performs arithmetic operations on the data that is made available to it. Some of the arithmetic operations supported by the arithmetic unit are—addition, subtraction, multiplication and division.

• The logic unit of ALU is responsible for performing logic operations (AND, NOT, OR). Logic unit performs comparisons of numbers, letters and special characters. Logic operations include testing for greater than, less than or equal to condition.

• The decision-making operations are getting to a conclusion according to condition. Just like if…else in C. The decision-making operations are greater than, less than, equal to.

The ALU comes to the conclusion whether x>y or x<y through subtract operation. If x>y, then x-y gives a positive result which is an unsigned number. If x<y, then x-y gives a negative result which is a signed number.

Data-driven decision-making action is a manner of working that emphasizes the importance of judgments that are well-supported by comprehensive, verifiable, and processed data. At first a team of professionals had to complete a lengthy process of data collection, extraction, formatting, modelling, and analysis. Then gaining insights from the data with highly technical expertise. But thanks to business intelligence components embedded into most software packages. IT teams are no longer burdened by the requirement to provide supporting reports at all hours of the day and night, which would then be examined by hard data analysts afterwards. Data scientists can now focus on assisting stakeholders in making data-driven decisions based on enormous volumes of data, which would have taken a long time to do. Data scientists can communicate complex, weighty data in a form that is easy to understand for non-technical stakeholders.

**Data flow of 3+5 in computer:**

First users input data e.g. 3+5 through the input devices like keyboard and mouse. The operands (3,5) (*an****operand****is a term used to describe any object that is capable of being manipulated. For example, in "1 + 2" the "1" and "2" are the****operands****and the plus symbol is the operator*) then are stored in RAM or directly to the data register in CPU through data bus. The sequence of register is remembered and coordinated by control unit of the register. The control bus is the pathway for the control signal. The **arithmetic/logic unit (ALU)** of a processor performs integer arithmetic and logical operations on the operands. The result of the operation is put into a general-purpose register. From the register the result goes to RAM and then to the output devices through data bus.

**Address generates from microprocessor:**

hard disk ->RAM ->a program that is executing=process ->CPU calculates the address -> address decoder->RAM

(Although both the hard drive and RAM are memory, it's more appropriate to refer to RAM as "memory" or "primary memory" and a hard drive as "storage" or "secondary storage” (nonvolatile).

Two memory operations are:

**1) Load (Read/Fetch) &**  
**2) Store (Write).**

• The Load operation transfers a copy of the contents of a specific memory-location to the processor. The memory contents remain unchanged.

**{complex operation of memory:** hard disk ->RAM ->a program that is executing=process ->CPU calculates the address->address decoder->RAM**}**

• **Steps for Load operation:**  
1) Processor sends the address of the desired location to the memory.  
2) Processor issues “read‟ signal to memory to fetch the data.  
3) Memory reads the data stored at that address. 4) Memory sends the read data to the processor.

• The Store operation transfers the information from the register to the specified memory-location. This will destroy the original contents of that memory-location.

•**Steps for Store operation are:**

1) Processor sends the address of the memory-location where it wants to store data.

2) Processor issues “write‟ signal to memory to store the data.

3) Content of register (MDR) is written into the specified memory-location.

**Registers**

* Registers are high-speed storage areas within the CPU, but have the least storage capacity. Registers are not referenced by their address, but are directly accessed and manipulated by the CPU during instruction execution.
* Registers store data, instructions, addresses and intermediate results of processing. Registers are often referred to as the CPU’s working memory.
* The data and instructions that require processing must be brought in the registers of CPU before they can be processed. For example, if two numbers are to be added, both numbers are brought in the registers, added and the result is also placed in a register.
* The number of registers and the size of each (number of bits) register in a CPU helps to determine the power and the speed of a CPU.
* The overall number of registers can vary from about ten to many hundreds, depending on the type and complexity of the processor.
* The size of register, also called word size, indicates the amount of data with which the computer can work at any given time. The bigger the size, the more quickly it can process data. The size of a register may be 8, 16, 32 or 64 bits. For example, a 32–bit CPU is one in which each register is 32 bits wide and its CPU can manipulate 32 bits of data at a time. Nowadays, PCs have 32–bit or 64–bit registers.
* 32-bit processor and 64-bit processor are the terms used to refer to the size of the registers. Other factors remaining the same, a 64-bit processor can process the data twice as fast as one with 32-bit processor.
* Registers are used for different purposes, with each register serving a specific purpose. Some of the important registers in CPU are as follows—
* Accumulator (ACC) stores the result of arithmetic and logic operations.
* Instruction Register (IR) contains the current instruction most recently fetched.
* Program Counter (PC) contains the address of next instruction to be processed.
* Memory Address Register (MAR) contains the address of next location in the memory to be accessed.
* Memory Buffer Register (MBR) temporarily stores data from memory or the data to be sent to memory.
* Data Register (DR) stores the operands and any other data.

**Control Unit**

The control unit of a computer does not do any actual processing of data. It organizes the processing of data and instructions. It acts as a supervisor and, controls and coordinates the activity of the other units of computer.

CU coordinates the input and output devices of a computer. It directs the computer to carry out stored program instructions by communicating with the ALU and the registers.

CU uses the instructions in the Instruction Register (IR) to decide which circuit needs to be activated. It also instructs the ALU to perform the arithmetic or logic operations. When a program is run, the Program Counter (PC) register keeps track of the program instruction to be executed next.CU tells when to fetch the data and instructions, what to do, where to store the results, the sequencing of events during processing (*though we input sequentially, the register store them arbitrarily; the CU remembers the sequence and directs the ALU to process*) etc. CU also holds the CPU’s Instruction Set, which is a list of all operations that the CPU can perform.

The function of a (CU) can be considered synonymous with that of a conductor of an orchestra. The conductor in an orchestra does not perform any work by itself but manages the orchestra and ensures that the members of orchestra work in proper coordination.

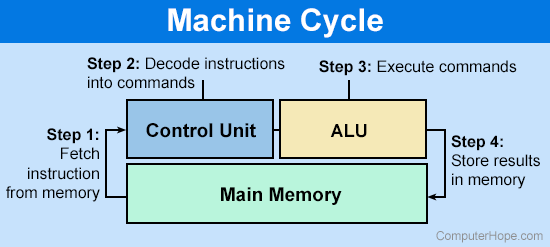
**Instruction Cycle**

The primary responsibility of a computer processor is to execute a sequential set of instructions that constitute a program. CPU executes each instruction in a series of steps, called instruction cycle. A instruction cycle involves four steps.

* Fetching: The processor fetches the instruction from the memory. The fetched instruction is placed in the *Instruction Register*. *Program Counter* holds the address of next instruction to be fetched and is incremented after each fetch. (Fetch=read=copy)
* Decoding: The instruction that is fetched is broken down into parts or decoded. The instruction is translated into commands so that they correspond to those in the CPU’s instruction set. The instruction set architecture of the CPU defines the way in which an instruction is decoded.
* Executing: The decoded instruction or the command is executed. CPU performs the operation implied by the program instruction. For example, if it is an ADD instruction, addition is performed.
* Storing: CPU writes back the results of execution, to the computer’s memory

Machine cycle Instruction Cycle

Execution Cycle



**Instruction set**

An Instruction Set is the set of all the basic operations that a processor can accomplish. The instructions in the instruction set are the language that a processor understands. All programs have to communicate with the processor using these instructions. The instruction set is embedded in the processor (hardwired), which determines the machine language for the processor. Two processors are compatible if the same machine level program can run on both the processors. Therefore, the system software is developed within the processor’s instruction set.

**System Clock**

The oscillator generates a number of electronic pulses, used by the computer to synchronize operations. Transition from +ve to -ve or vice-verse is a cycle and the total amount of cycle/sec is hertz.

The clock speed of a CPU is defined as the frequency with which a processor executes instructions or the data is processed. It synchronizes the operations between components. Higher clock frequencies mean more clock ticks per second. The computer’s operating speed is linked to the speed of the system clock. The clock frequency is measured in millions of cycles per second or megahertz (MHz) or gigahertz (GHz) which is billions of cycles per second. A CPU’s performance is measured by the number of instructions it executes in a second, i.e., MIPS (Million of Instructions Per Second) or BIPS. Microprocessor generally requires 10 clock cycles to complete a single instruction. Speed of a processor is measured with MIPS (million of instructions/sec) PCs nowadays come with a clock speed of more than 1 GHz. In Windows OS, you can select the System Properties dialog box to see the processor name and clock frequency.

**Differences between CISC and RISC:**

|  |  |
| --- | --- |
| **CISC (Complex Instruction Set Computer)** | **RISC (Reduced Instruction Set Computer)** |
| The original microprocessor ISA | Redesigned ISA that emerged in the early 1980s |
| Instructions can take several clock cycles | Single clock cycle instructions |
| Large number of instructions | Small number of instructions. Generally, less than 100 |
| Some instructions with long execution times. These include instructions that copy an entire block from one part of memory to another and others that copy multiple registers to and from memory. | No instruction with a long execution time due to a very simple instruction set. Some early RISC machines did not even have an integer multiply instruction, requiring compilers to implement multiplication as a sequence of additions. |
| Instructions are often of variable size | Fixed length (32 bit) format instructions. |
| CISC supports array. | RISC does not support an array. |
| Condition codes are used. | No condition codes are used. |
| Complex addressing modes. Many different combinations of displacement, base, and index register. | Few simple addressing modes. Only base and displacement addressing is allowed. |
| Single register sets | Multiple register sets |
| Not pipelined or less pipelined | Highly pipelined |
| Complexity in the microprogram | Complexity in the compiler |
|  | RISC chips are relatively simple to design and inexpensive |
| The setback of this design is that the execution time is very high. | The setback of this design is that the computer has to repeatedly perform simple operations to execute a larger program having a large number of processing operations. |
|  | The compiler or programmer synthesizes complicated operations (for example, a divide operation) by combining several simple instructions. Each instruction is a fixed length to allow the pipeline to fetch future instructions before decoding the current instruction. |
| Unified cache for data and instructions | Separate data and instruction cache |
| More efficient use of RAM than RISC | Heavy use of RAM (limited RAM can cause bottlenecks) |
| Arithmetic and logical operations can be applied to both memory and register operands | Arithmetic and logical operations only use register operands. Memory referencing is only allowed by loading and storing instructions, i.e. reading from memory into a register and writing from a register to memory respectively. |
| It requires external memory for calculations | It does not require external memory for calculations |
| CISC processors have dedicated registers for specific purposes. | RISC processors have a large general-purpose register set. Any register can contain either data or an address**.** |
| The stack is being used for procedure arguments and returns addresses | Registers are being used for procedure arguments and return addresses. |
| Code expansion is not a problem but decoding of instructions is complex | Code expansion may be a problem but decoding of instructions is simple |
| Implementation programs are hidden from machine-level programs. | Implementation programs exposed to machine-level programs. |
| Focus on hardware. Uses both hardwired and microprogrammed control unit. | Focus on software. Uses only Hardwired control unit. |
| **Examples:**Intel architecture, AMD | **Examples:** SPARC, POWER PC, etc. |

**CPU Performance**

The performance of a CPU is the number of programs it can run in a given time. The more the number of programs it can run in that time, the faster the CPU is. The performance is determined by the number of instructions that a program has: more instructions, more time to perform them. It also depends upon the number of cycles (clock cycles) per instructions. This means that there are only two ways to improve the performance: Either minimize the number of instructions per program, or reduce the number of cycles per instruction.

**Instruction Format**

An instruction is divided into groups called fields. The common fields of an instruction are— Operation (op) code and Operand code. The operation code represents action that the processor must execute. The operand code defines the parameters of the action and depends on the operation. It specifies the locations of the data or the operand on which the operation is to be performed. It can be data or a memory address.

**Differences between CISC and RISC operation:**

To illustrate a CISC instruction, let’s take the MUL instruction.

This instruction takes two inputs: the memory location of the two numbers to multiply, it then performs the multiplication and stores the result in the first memory location.

***MUL 1200, 1201***

Where MUL takes the value from either two memory locations (say 1200 and 1201) or two registers, finds their product and stores the result in location 1200. This reduces the amount of work that the compiler has to do as the instructions themselves are very high level. The instructions take very little memory in the RAM and most of the work is done by the hardware while decoding instructions. Since in a CISC style instruction, the CPU has to do more work in a single instruction, so clock speeds are slightly slower. Moreover, the number of general-purpose registers is less as more transistors need to be used to decode the instructions.

Multiplication in a RISC architecture cannot be done with a single MUL like instruction. Instead, we have to first load the data from the memory using the LOAD instruction, then multiply the numbers, and the store the result in the memory.

***Load A, 1200***

***Load B, 1201***

***Mul A, B***

***Store 1200, A***

Here the Load instruction stores the data from a memory location like 1200 into a register A or B. Mul multiplies values in the two registers and stores it in A. Then finally we store the value of A in 1200 (or any other memory location). Note that in RISC architectures, we can only perform operations on Registers and not directly on the memory. These are called addressing modes.

This might seem like a lot of work, but in reality, since each of these instructions only take up one clock cycle, the whole multiplication operation is completed in fewer clock cycles (I discuss more on this later).

However, the time advantage is not without its disadvantages. Since RISC has simpler instruction sets, complex High-Level Instructions needs to be broken down into many instructions by the compiler. While the instructions are simple and don’t need complex architectures to decode, it is the job of the compiler to break down complex high level programs into many simple instructions.

This puts a lot of stress on the software and the software designers, while reducing the work needed to be done by the hardware.

Since the decoding logic is simple, transistors required are lesser and a greater number of general-purpose registers can be fit into the CPU

MUL 2 5 in CISC

LOAD A 2 # load the number at the location 2 to register A

LOAD B 5 # load the number at the location 5 to register B

PROD C A B # conduct A x B and save the result back to C

STORE 2 C # store the value at C back at the location 2 in RISC

Assuming each simple instruction can be executed with in one clock cycle, completing a complex instruction MUL 2 2 5 may take the same amount of time as to execute the four simple instructions. However, since all the simple instructions execute in a uniform amount of time, it is possible to leverage pipelining to reduce the execution time.

***#3 operand operation can only be operated in RISC. CISC can do the same in case it is specially manufactured that way, but not generally.***

**Vacuum tube**: The vacuum tube is a glass tube with its gas removed, creating a vacuum. Vacuum tubes contain electrodes for controlling electron flow and were used in early computers as a switch or an amplifier.

**How a vacuum tube works?**

= The simplest version of vacuum tube technology, called a diode, is a very simple device. It is literally two electrodes inside of a glass tube – cathode and anode.  A cathode carries a negative charge, while anodes carry positive charge. This device conducts electricity.  Electricity is nothing more than moving electrons. Thermionic emission is a process where you heat a cathode. As it heats up, the cathode will release negatively charged electrons. Those available electrons float around the tube. That’s where the anode comes into the picture. The positive charge of the anode attracts all of those stray electrons, creating electrical flow. Removing the gas from the tube means that nothing gets in the way of the moving electrons.

**Transistor:** An electronic component that can be used as an amplifier or as a switch. It is used to control the flow of electricity in radios, televisions, computers, etc.

**How a transistor works?**

= Transistors work just like the switch as it can control the current flow. It can be off which is zero state and on which is one state. All information is stored and processed in 0s and 1s.

(Pure silicon is a semiconductor which has four valence electrons and bonds with four silicon atoms and forms tetrahedral crystal. So, in the lattice few electrons get enough energy to escape their bond and travel through the lattice. Having small number of mobile charges make silicon semiconductor. This mobile charge can be increased through doping. There are two types of doping: n-type and p-type.

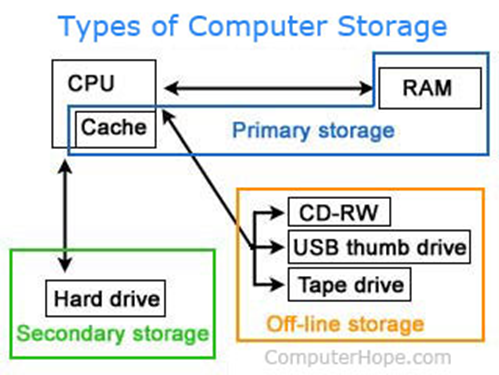
In n-type semiconductor pure silicon is injected with a small amount of element with five valence electrons like phosphorus. Phosphorus is similar enough to silicon that it can fit into the lattice but it brings with it an extra electron. Thus, there are more mobile negative charges to conduct current.

In p-type doping an element with three valence electrons is added to the lattice like boron. This creates a hole - a place where there should be an electron but there isn’t. But this still increases the conductivity of silicon because electrons can move into it. Although the electrons are moving, traditionally it is said that the holes are moving as the holes are fewer. Since the hole is a lack of electron it acts like a positive charge.

n- type and p- type semiconductor both of them are neutrals. The n and p actually just refer to the sign of charge that can move within them.)

A transistor has both n-type and p-type semiconductors. A common configuration is n-p-n.

Just like a switch a transistor has an electrical contact at each end and they are called source and drain. But instead of a mechanical switch there is a third electrical contact called gate which is insulated from the semiconductor by an oxide layer. When a transistor is made, the n-type and p-type don't keep to themselves- electrons actually diffuse from the n-type, where there are more of them into the p-type to fill the holes. This creates a depletion lair between them. Charges that can move are being depleted as there are no free electrons in n-type as they've filled the holes in p-type. This makes the p-type negative due to the added electrons. That's why p-type will repel any electrons that try to come across from the n-type. So, the depletion layer acts like a barrier preventing the flow of electrical current through the transistor. So right now, the transistor is off. It's like an open switch, it's in the zero sate. To turn it on a small (+) voltage has to be applied to the gate. This attracts the electrons over and overcomes the repulsion from the depletion. As a result, it shrinks the depletion layer so that electrons can move through and form a conducting channel. So, the transistor is on and it's on one state.

****

**RAM (random-access memory)** – a type of data storage (memory element) used in computers that temporary stores of programs and data (volatile: its contents are lost when the computer is turned off).

**ROM (read-only memory**) – a type of data storage used in computers that permanently stores data and programs (non-volatile: its contents are retained even when the computer is turned off).

**Note**

The first and maybe the most famous ALU was the 4-bit [**Intel**](https://www.computerhope.com/comp/intel.htm)**74181** released in [1970](https://www.computerhope.com/history/1970.htm). It was the first ALU placed in a single chip.

**Arithmetic operations**

The arithmetic unit handles all of the following computer numerical operations.

* [**Add**](https://www.computerhope.com/jargon/a/add.htm) - add two bits.
* **Add with carry** - add two bits with a carry-in.
* [**Subtract**](https://www.computerhope.com/jargon/s/subtract.htm) - subtract two bits.
* **Subtract with borrow** - subtract two bits with borrow from carry-in.
* [**Negate**](https://www.computerhope.com/jargon/n/negate.htm) - flip the bits values sign (- to + or + to -).
* [**Increment**](https://www.computerhope.com/jargon/i/incremen.htm) - add 1 to a bit.
* [**Decrement**](https://www.computerhope.com/jargon/d/decremen.htm) - subtract 1 from a bit.
* **Pass through** - let bits through without modification.

**Note**

Simple ALU's have no divide or multiply operations. For these ALU's to perform these operations, they use the add and subtraction operations.

**Logic unit operations**

The logic unit performs [logical operations](https://www.computerhope.com/jargon/l/logioper.htm) (e.g., AND, OR, and NOT) and numeric tests like checking if a number is a negative number.

o CU controls the overall operations of the computer i.e., it checks the sequence of execution of instructions, and, controls and coordinates the overall functioning of the units of computer.

**Number Conversion:**

The numbers given as input to computer and the numbers given as output from the computer, are generally in decimal number system, and are most easily understood by humans. However, computer understands the binary number system, i.e., numbers in terms of 0s and 1s. The binary data is also represented, internally, as octal numbers and hexadecimal numbers due to their ease of use.

Therefore, we are concerned with four kinds of number systems, as follows—

• Decimal Number System —Base 10

• Binary Number System —Base 2

• Octal Number System —Base 8

• Hexadecimal Number System—Base 16

**Number System:**

**Base:** The number of unique symbols a number system uses for its digits that are used to make a number or combine to get a number.

In a number, the position of digit starts from the right-hand side of the number. The rightmost digit has position 0, the next digit on its left has position 1, and so on.

The digits of a number have two kinds of values— • Face value • Position value.

The face value of a digit is the digit located at that position. For example, in decimal number 52, face value at position 0 is 2 and face value at position 1 is 5.

The position value of a digit is (baseposition). For example, in decimal number 52, the position value of digit 2 is 100 and the position value of digit 5 is 10.

The number is calculated as the sum of (face value \* position value) = (face value \* baseposition) of each of the digits.

**Conversion from Decimal to Binary, Octal, Hexadecimal:**

A decimal number has two parts—integer part and fraction part. A decimal integer is converted to any other base, by using the division operation where the divisor is the base. A fractional number is a number less than 1. The multiplication operation is used to convert decimal fraction to any other base.

**Ex:** **Convert decimal number 25.0625 to binary**.

**Ans:**

|  |  |  |
| --- | --- | --- |
| To base | Number | Remainder |
| 2 | 25 |  |
| 2 | 12 | 1 |
| 2 | 6 | 0 |
| 2 | 3 | 0 |
| 2 | 1 | 1 |
|  | 0 | 1 |

So, 25(10) = 11001(2)

|  |  |
| --- | --- |
| 0.0625  X 2 |  |
| 0.125  X 2 | 0 |
| 0.25  X 2 | 0 |
| 0.5  X 2 | 0 |
| 1.00 | 1 |

So, 0.0625(10) = 0.001(2)

Hence 25.0625(10) = 11001.001(2)