

Introduction to Computer, Algorithm and Flowchart

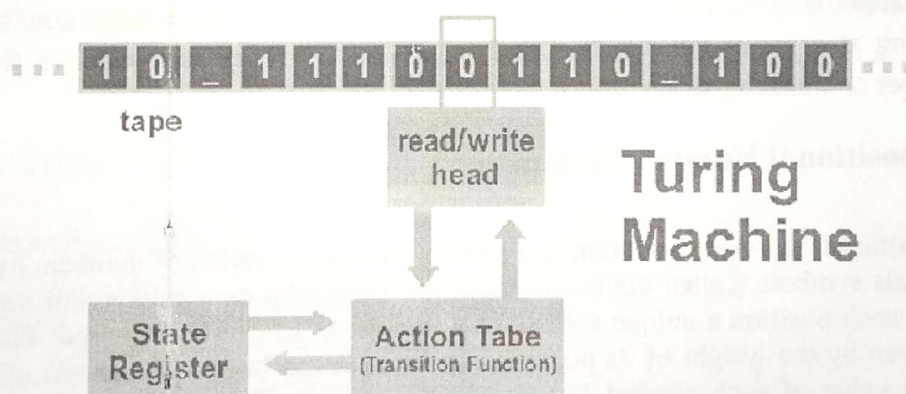
1.1 Basics of Computer

Turing Machine

A Turing Machine (TM) is a mathematical model which consists of an infinite length tape divided into cells on which input is given. It consists of a head which reads the input tape. A state register stores the state of the Turing machine. After reading an input symbol, it is replaced with another symbol, its internal state is changed, and it moves from one cell to the right or left. If the TM reaches the final state, the input string is accepted, otherwise rejected.

A Turing Machine can be formally described as a 7-tuple $(Q, X, \Sigma, \delta, q_0, B, F)$ where –

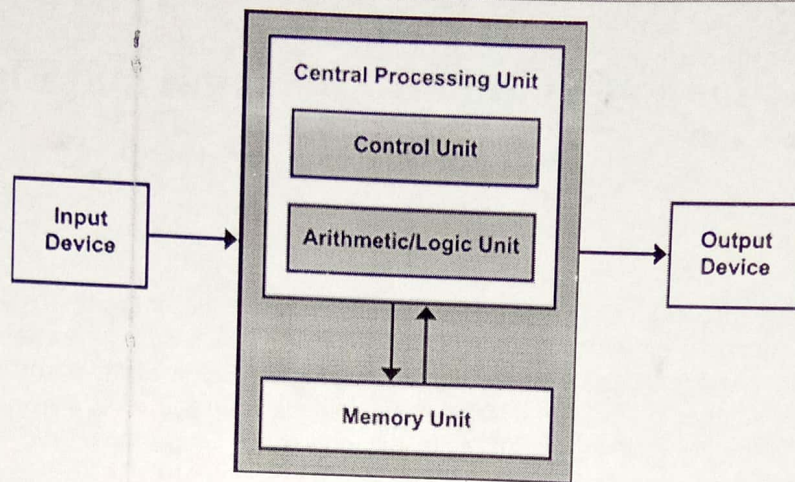
- Q is a finite set of states
- X is the tape alphabet
- Σ is the input alphabet
- δ is a transition function; $\delta : Q \times X \rightarrow Q \times X \times \{\text{Left_shift}, \text{Right_shift}\}$.
- q_0 is the initial state
- B is the blank symbol
- F is the set of final states



Von Neumann Model

VonNeumann conceptualized a computer structure which consists of Input/output devices, a Central Processing Unit containing an Arithmetic and Logic Unit (ALU) and Control Unit, and a Memory Unit, which would store both program and data with necessary communication paths or connections.

In this computer architecture, the program and data are stored at any suitable location in the memory unit. The instruction, of the stored program, are read, by the CPU from Memory, one after another and executed in the sequence. The data outcome is stored in Memory.



Von Neumann Architecture

Properties of von Neumann architecture:

1. *The stored program concept*, which means that the program instructions are stored in the same memory as the data and instruction both can be treated like data.
2. The Memory Unit is *a single memory unit* and is *sequentially addressed*.
3. The *memory unit is one-dimensional*.
4. *The meaning of the data is not stored with it*. In other words, it is not possible to say by looking at a set of bits whether that set of bits represent an integer, a floating point number or character string.

Basics of positional Number System

A **positional (numeral) system** is a system for representation of numbers by an ordered set of numeral symbols (called digits) in which the value of a numeral symbol depends on its position. For each position a unique symbol or a limited set of symbols is used. The value of a symbol is given by the weight of its position expressed in the bases (or radices) of the system. The resultant value of each symbol is given by the value assigned to its position (e.g. by a product of bases) and modified (e.g. multiplied) by the value of the symbol. The total value of the represented number in a positional number is then sum of the values assigned to the symbols of all positions.

Binary Number System

Characteristics of the binary number system are as follows :

- Uses two digits, 0 and 1
- Also called as base 2 number system
- ~~Each~~ ^{First} position in a binary number represents a 0 power of the base (2). Example 2^0

- Last position in a binary number represents a x power of the base (2). Example 2^x where x represents the last position - 1.

ExampleBinary Number: 10101_2

Calculating Decimal Equivalent –

Step	Binary Number	Decimal Number
Step 1	10101_2	$((1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0))_{10}$
Step 2	10101_2	$(16 + 0 + 4 + 0 + 1)_{10}$
Step 3	10101_2	21_{10}

Octal Number System

Characteristics of the octal number system are as follows –

- Uses eight digits, 0,1,2,3,4,5,6,7
- Also called as base 8 number system
- Each position in an octal number represents a 0 power of the base (8). Example 8^0
- Last position in an octal number represents a x power of the base (8). Example 8^x where x represents the last position - 1

ExampleOctal Number: 12570_8

Calculating Decimal Equivalent –

Step	Octal Number	Decimal Number
Step 1	12570_8	$((1 \times 8^4) + (2 \times 8^3) + (5 \times 8^2) + (7 \times 8^1) + (0 \times 8^0))_{10}$
Step 2	12570_8	$(4096 + 1024 + 320 + 56 + 0)_{10}$
Step 3	12570_8	5496_{10}

Hexadecimal Number System

Characteristics of hexadecimal number system are as follows –

- Uses 10 digits and 6 letters, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Letters represent the numbers starting from 10. A = 10, B = 11, C = 12, D = 13, E = 14, F = 15
- Also called as base 16 number system

Sub: SPA

- Each position in a hexadecimal number represents a 0 power of the base (16). Example, 16^0
- Last position in a hexadecimal number represents a x power of the base (16). Example 16^x where x represents the last position - 1

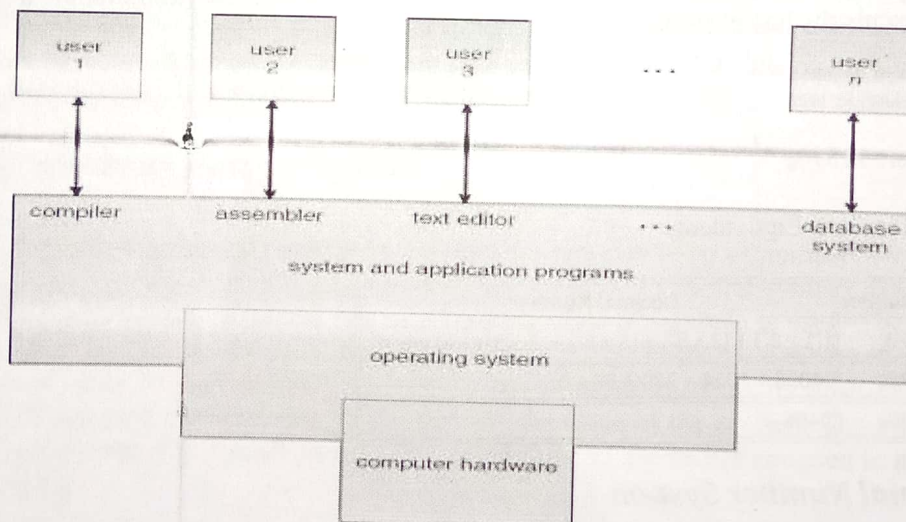
Example

Hexadecimal Number: $19FDE_{16}$

Calculating Decimal Equivalent –

Step	Binary Number	Decimal Number
Step 1	$19FDE_{16}$	$((1 \times 16^4) + (9 \times 16^3) + (F \times 16^2) + (D \times 16^1) + (E \times 16^0))_{10}$
Step 2	$19FDE_{16}$	$((1 \times 16^4) + (9 \times 16^3) + (15 \times 16^2) + (13 \times 16^1) + (14 \times 16^0))_{10}$
Step 3	$19FDE_{16}$	$(65536 + 36864 + 3840 + 208 + 14)_{10}$
Step 4	$19FDE_{16}$	106462_{10}

Introduction to Operating System



Abstract view of the components of a computer system.

The operating system provides an interface between an application program and the computer hardware, so that an application program can interact with the hardware only by obeying rules and procedures programmed into the operating system.

Operating system components:

An operating system is software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

The operating system comprises a set of software packages that can be used to manage interactions with the hardware. The following elements are generally included in this set of software:

- The **kernel**, which represents the operating system's basic functions such as management of memory, processes, files, main inputs/outputs and communication functionalities.
- The **shell**, allowing communication with the operating system via a control language, letting the user control the peripherals without knowing the characteristics of the hardware used, management of physical addresses, etc.
- The **file system**, allowing files to be recorded in a tree structure.

Some popular Operating Systems include Linux, Windows, OS X, VMS, OS/400, AIX, z/OS, etc.

Functions of an operating System:

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

1.2 Algorithm & Flowchart

What is an algorithm?

An algorithm is a part of the plan for the computer program. In fact, an algorithm is an effective procedure for solving a problem in a finite number of steps.

Example

Algorithm: Calling a friend on the telephone

Input: The telephone number of your friend.

Output: None

Steps:

1. Pick up the phone and listen for a dial tone
2. Press each digit of the phone number on the phone
3. If busy, hang up phone, wait 5 minutes, jump to step 2
4. If no one answers, leave a message then hang up
5. Talk to friend
6. Hang up phone

Algorithms may be represented in various ways.

1. Step form
2. Flow chart
3. Pseudo-code

1. Step-form

In this representation, the procedure of solving a problem is stated with written statements. Each statement solves a part of problem & these together complete the solution.

Three basic constructs: sequence, decision and repetition.

i. Sequence:

Sequence means that each step or process in the algorithm is executed in special order.

ii. Decision: (if...then, if...then...else...)

In algorithm outcome of a decision is either true or false, there is no state in between. The outcome a decision is based on some condition that can only result in true or false value. It has the general form:

If proposition then process

e.g.

if today is Friday then collect payment.

The decision can also be stated as:

If *proposition*

Then *process 1*

Else *process 2*

This means that if the proposition is true then execute process 1 else execute process 2. But the first form of the decision **if** *proposition* **then** *process* has a null **else** that is there is no else.

iii. Repeat(repeat and while)

Repetition can be implemented using constructs like the **repeat** loop, **while** loop & **if...then goto...** loop. The repeat loop is used to iterate repeat a process or sequence of processes until some condition become true.

It has the general form:

Repeat

Process 1, process 2 ..., process N

Until *proposition*

e.g.

Repeat

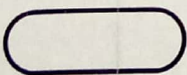
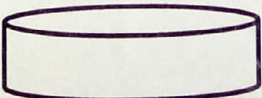
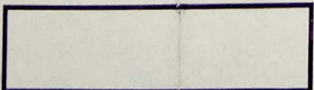
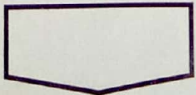
Fill water in tank


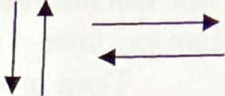
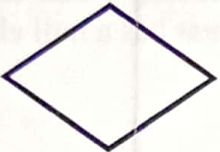


Until *tank is full.*

FLOWCHARTS

A flowchart provides appropriate steps to be followed in order to arrive at the solution to a problem. A flowchart comprises a set of various standard **shaped boxes** that are interconnected by **flow lines**. Flow lines have **arrows** to indicate the direction of the flow of control between the boxes. The activity to be performed is written within the boxes in English. A flowchart is a picture of the separate steps of a process in sequential order.

Standard symbols:

 Start or end of the program / flowchart	 Magnetic disk
 Computational steps or processing function of a program	 Connects remote parts of flowchart not on the same page

 Input entry or output display	 Flow lines
 Decision making and branching	 Magnetic Tape
 Connects remote parts of flowchart on the same page	

Example: Draw a flowchart to add two numbers entered by user.

