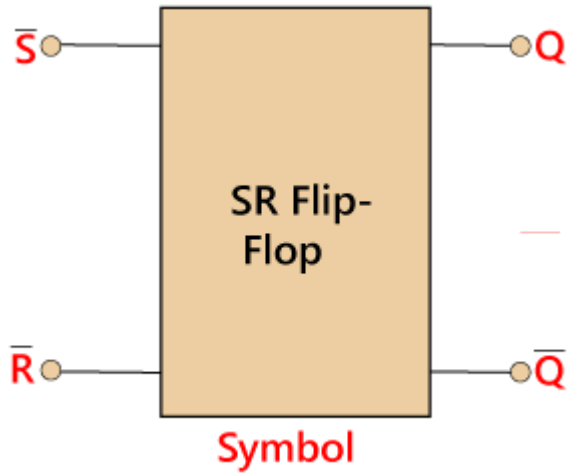
SR Flip Flop

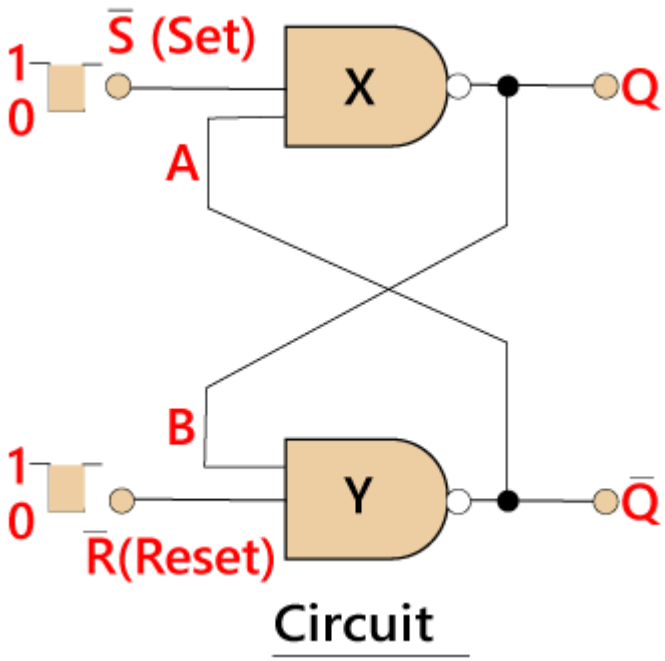
The SR flip flop is a 1-bit memory bistable device having two inputs, i.e., SET and RESET. The SET input 'S' set the device or produce the output 1, and the RESET input 'R' reset the device or produce the output 0. The SET and RESET inputs are labeled as **S** and **R**, respectively.

The SR flip flop stands for "Set-Reset" flip flop. The reset input is used to get back the flip flop to its original state from the current state with an output 'Q'.

Block Diagram:



Circuit Diagram:



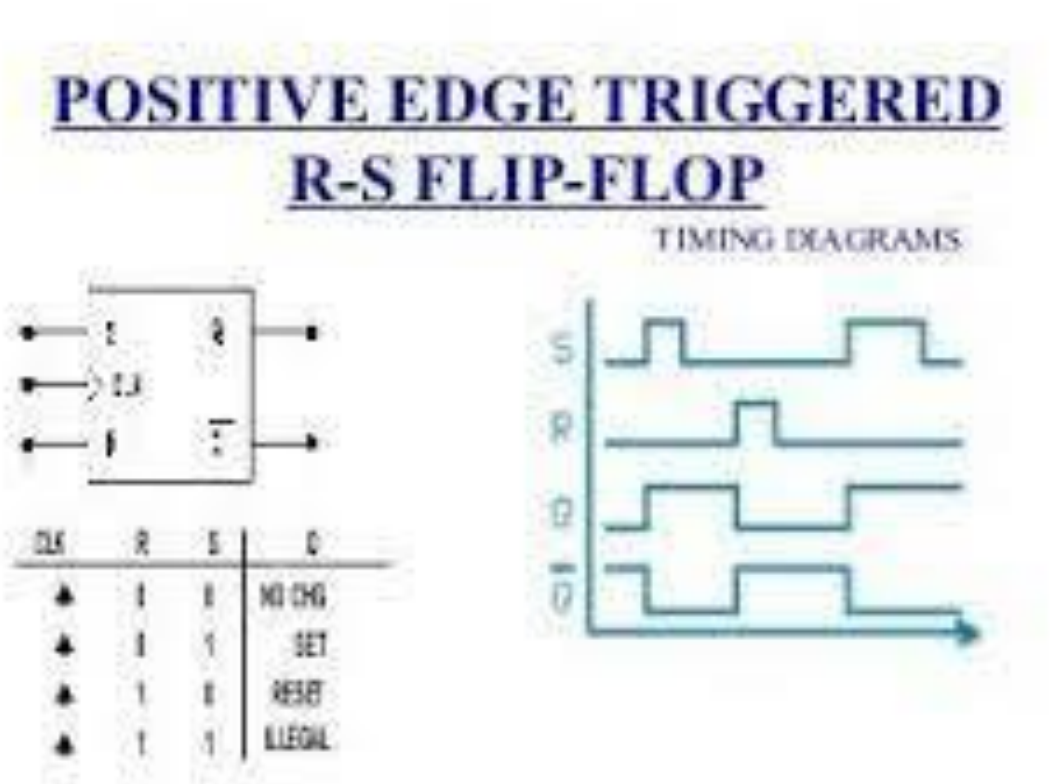
S	R	Q	Q’	STATE
0	0	0	0	NO CHANGE
0	1	0	1	RESET
1	0	1	0	SET
0	1	X	X	FORBIDDEN

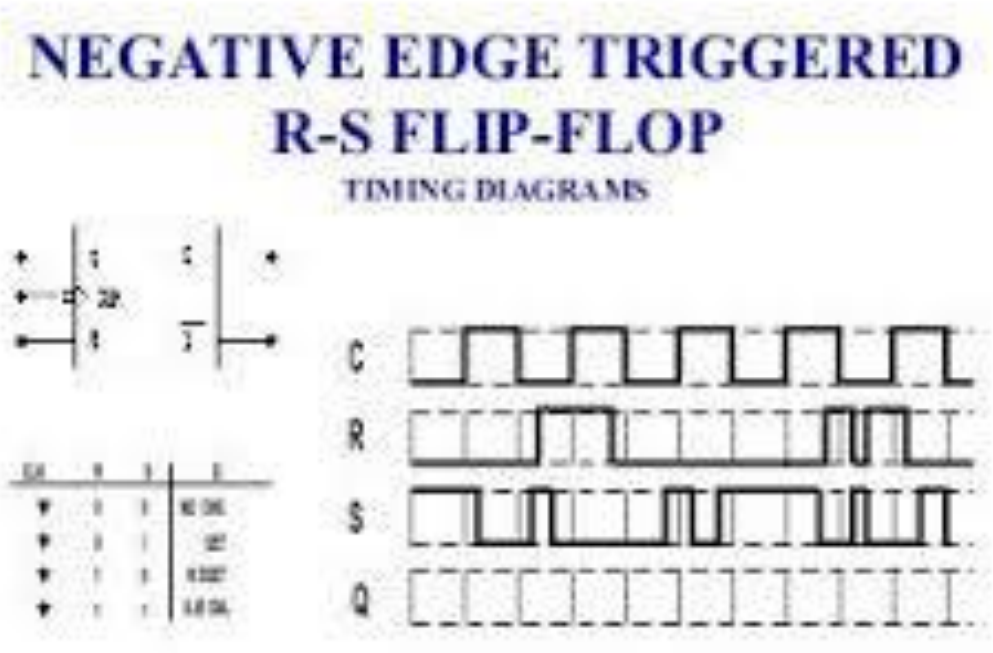
When set 'S' and reset 'R' inputs are set to 1, the outputs Q and Q' will be either 1 or 0. These outputs depend on the input state S or R before the input condition exist. So, when the inputs are 1, the states of the outputs remain unchanged.

Edge Triggering

In a sequential circuit, if the output changes when the signal transits from a high level to a low level or from a low level to a high level, we call it edge triggering.

The edge that changes the voltage from low level to the high level is called rising edge (positive edge). And, the edge that changes the voltage from high level to the low level is called falling edge (negative edge).



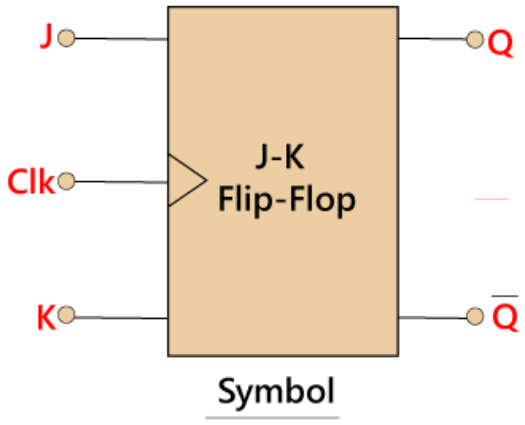


JK Flip Flop

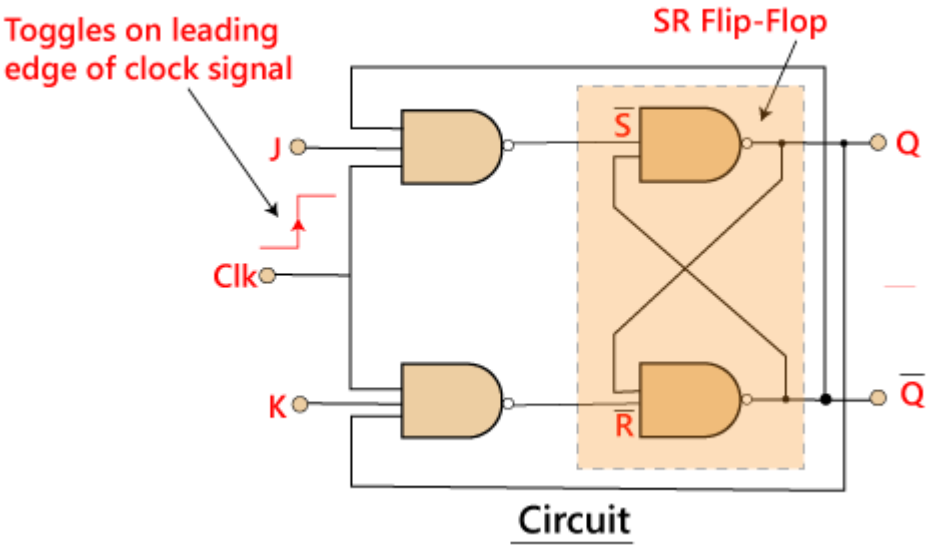
The JK flip flop is one of the most used flip flops in digital circuits. The JK flip flop is a universal flip flop having two inputs 'J' and 'K'.

The JK flip flop work in the same way as the SR flip flop work. The JK flip flop has 'J' and 'K' flip flop instead of 'S' and 'R'. The only difference between JK and SR is that when both inputs of SR flip flop is set to 1, the circuit produces the invalid states as outputs, but in case of JK flip flop, there are no invalid states even if both 'J' and 'K' flip flops are set to 1.

Block Diagram:

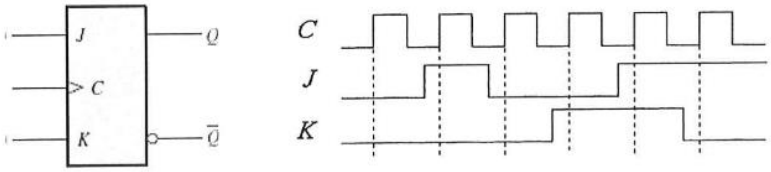


Circuit Diagram:



Positive Edge Triggered

The Outputs are changed at the raising edge of the clock which is called positive trigger



Truth Table For A Positive Edge-Triggered J-K Flip-Flop

Inputs			Outputs		Condition Comments
J	K	CLK	Q	$\overline{Q}$	
0	0	↑	Q_0	$\overline{Q}_0$	No Change
0	1	↑	0	1	Latch RESET
1	0	↑	1	0	Latch SET
1	1	↑	$\overline{Q}_0$	Q_0	TOGGLE

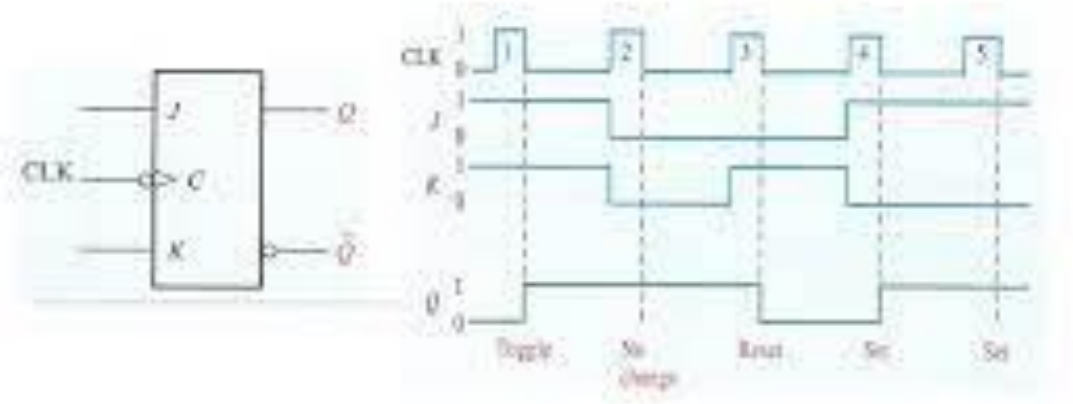
↑ = Clock transition LOW to HIGH

Q_0 = output level prior to clock transition

Fig. 1(g)

EXAMPLE (Negative Edge Triggered)

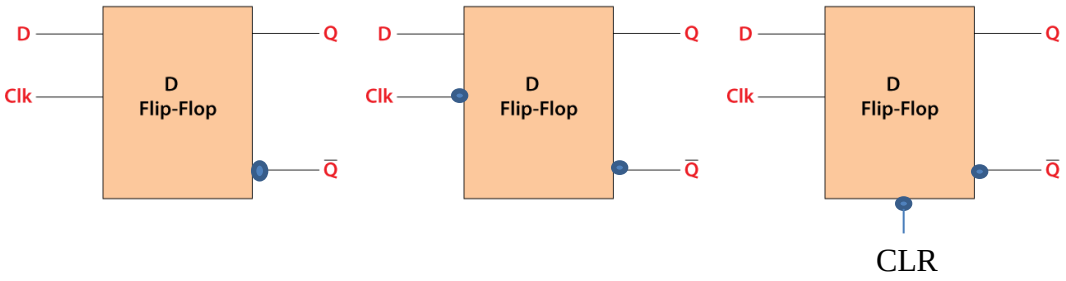
The Outputs are changed at the negative edge of the Clock.



The JK flip flop suffers from the **"race"** problem. This problem occurs when the state of the output Q is changed before the clock input's timing pulse has time to go **"Off"**. We have to keep short timing plus period (T) for avoiding this period

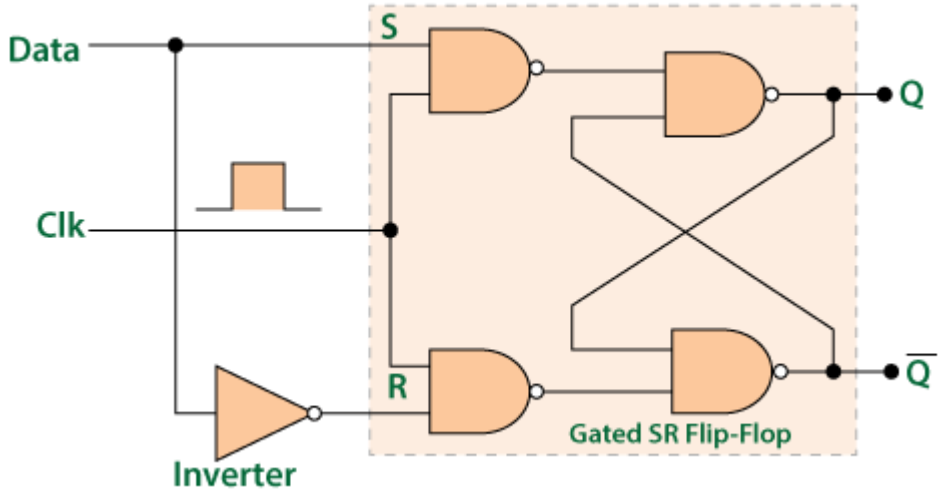
Edge Triggered D Flip Flop

Block Diagram



D Flip Flop Positive Triggered Negative Trigger

Circuit Diagram



The "CLOCK" or "ENABLE" input is used to avoid this for isolating the data input from the flip flop's latching circuitry. When the clock input is set to true, the D input condition is only copied to the output Q.

Truth Table

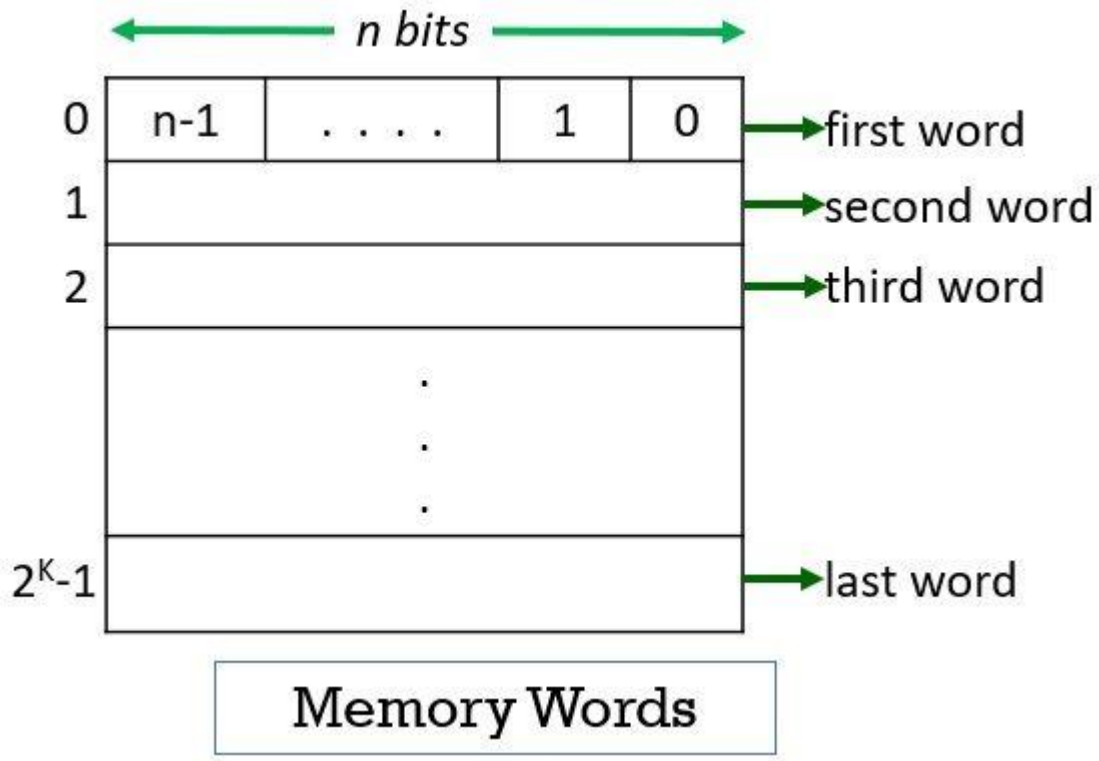
Clock	D	Q	Q'	Description
↓ » 0	X	Q	Q'	Memory no change
↑ » 1	0	0	1	Reset Q » 0
↑ » 1	1	1	0	Set Q » 1

Memory Locations and addresses

- Memory locations and addresses determine how the computer's memory is organized so that the user can efficiently store or retrieve information from the computer.

- The computer's memory is made of a silicon chip which has millions of storage cell, where each storage cell is capable to store a *bit* of information which value is either 0 or 1.

- The group of n bit is termed as ***word*** where n is termed as the *word length*.



Byte Addressability

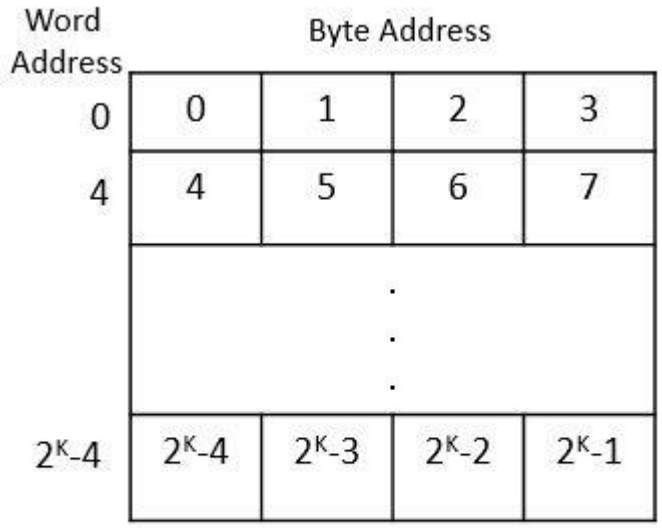
- 8 bits together form a byte and this is the fix for every memory. But the word length varies from memory to memory and it ranges from 16 to 64 bit.

- Most modern computers assign successive addresses to successive byte locations in memory. This assignment of addresses to individual byte locations is termed byte addressability and memory is referred to as byte-addressable memory.

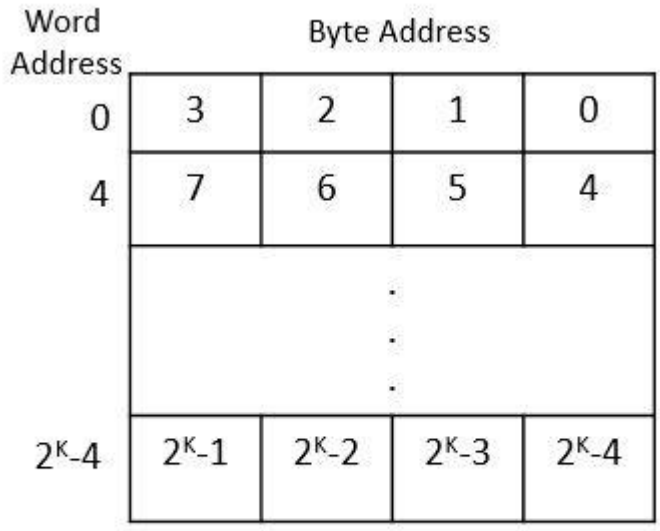
- If we have a machine with a word length of 32 bit then the successive words are located at the addresses 0, 4, 8, 12… where each word would have 4 bytes of information

Big-Endian and Little-Endian Assignments in Byte Addresses

The big-endian and little-endian are two methods of assigning byte addresses across the words in the memory.



Big-Endian Assignment



Little-Endian Assignment

Word Alignment

In a machine with word length 32-bit, the word boundaries occur at the bytes addresses 0, 4, 8… It is said that the word has *aligned addresses* if they begin with the byte address that is multiple of the number of bytes present in that word.

Accessing numbers and characters

Numbers -> Occupies one word and accessed from memory by specifying its word address

Individual characters -> Accessed by their address

Memory Operations

The memory unit supports two basic operations: read and write. The read operation reads previously stored data and the write operation stores a new value in memory

The memory unit supports two basic operations: read and write. The read operation reads previously stored data and the write operation stores a new value in memory

Steps in a Typical Read Cycle:

1. Place the address of the location to be read on the address bus
2. Activate the memory read control signal
3. Wait for the memory to retrieve the data and place them on the data bus
4. Read the data from the data bus
5. Drop the memory read control signal to terminate the read cycle

Steps in a Typical Write Cycle:

1. Place the address of the location to be written on the address bus
2. Place the data to be written on the data bus
3. Activate the memory write control signal on the control bus
4. Wait for the memory to store the data at the location
5. Drop the memory write signal to terminate the write cycle.

Assembly Languages

- An assembly language is a type of programming language that translates high-level languages into machine language.
- It is a necessary bridge between software programs and their underlying hardware platforms.
- Assembly language relies on language syntax, labels, operators, and directives to convert code into usable machine instruction.
- Assembly language may pass through single-pass or multi-pass assemblers, each with specific uses and benefits.

Pros

Execution may be more simple compared to other languages

Execution is usually faster compared to other languages

Allows for direct control over hardware

Code may remain smaller compared to other languages

Cons

Programming may be more challenging to pick up

Syntax of assembly languages is difficult

Not portable between machines

Assembler

- Assembly language must be translated into machine language using an assembler. There are two primary types of assemblers.
- A single-pass assembler scans a program one time and makes an equivalent binary program.
- A multi-pass assembler means the assembler uses more than one pass.

Assembler Directives

Assembler directives supply data to the program and control the assembly process.

- It is used by the assembler
- Assemble code and data into specified sections
- Reserve space in memory for uninitialized variables
- Control the appearance of listings
- Initialize memory
- Assemble conditional blocks

There are 4 fields

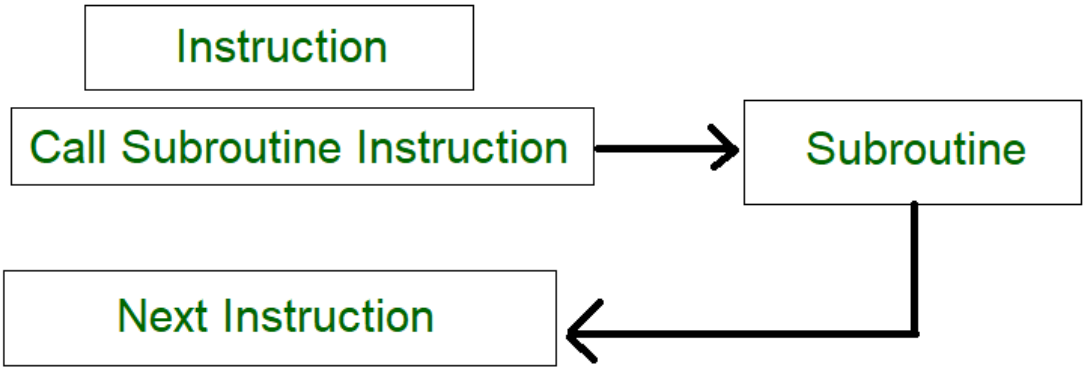
Label: operation operands comment

	100	**Load R2,N**		***ORIGIN***	***100***
	104	Add R1,R1,R2			
LOOP	108	Load R2,NUM1	**LOOP:**	***LD***	***R2,N***
				ADD	***R1,R2,R1***
				LD	***R2,NUM1***
		ADD R1,R1,R2		***ADD***	***R1,R2,R1***
				ST	***R1,SUM***
		Branch_if [R1] = 600 LOOP		***ORIGIN***	***200***
SUM	200	Store R1,SUM	**SUM:**	***RESERVE***	***4***
N	204		**N:**	***DATAWORD***	***1***
NUM1		1	**NUMn:**	***RESERVE***	***600***
		1		***END***	
NUMn	804	600			

Subroutines

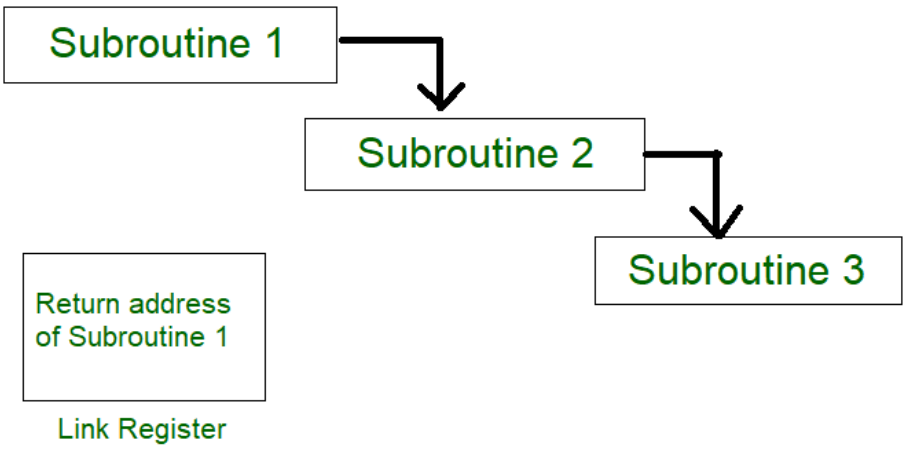
- A set of instructions that are used repeatedly in a program can be referred to as Subroutine. Only one copy of this Instruction is stored in the memory.
- When a Subroutine is required it can be called many times during the Execution of a particular program.
- A call Subroutine Instruction calls the Subroutine.
- The content of the PC saved by the call Instruction to make a correct return to the calling program.

- The address of the next instruction available in the program counter (the return address) is stored in a temporary location so the subroutine knows where to return.
- The control is transferred to the beginning of the subroutine. The last instruction of every subroutine commonly called return from a subroutine transfers the return address from the temporary location into the program counter.



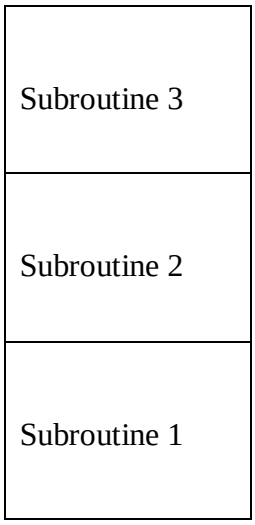
Subroutine Nesting

Subroutine nesting is a common Programming practice In which one Subroutine calls another Subroutine.



Link register stores the return address of Subroutine 1 this will be *overwritten* by the return address of Subroutine 2. As the last Subroutine called is the first one to be returned. So **stack data structure** is the most efficient way to store the return addresses of the Subroutines.

Stack Memory Stores the return addresses of the Subroutines.

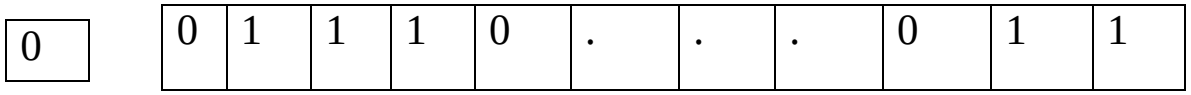


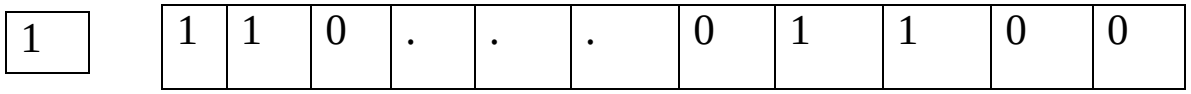
Additional Instructions

Logical Shift instructions

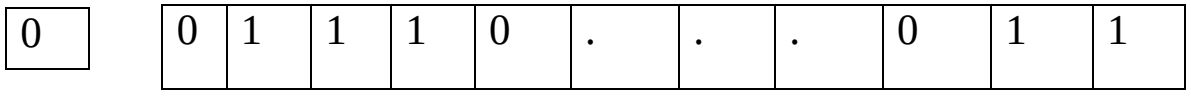
logical shift is a bitwise operation that shifts all the bits of its operand. The two base variants are the **logical left shift** and the **logical right shift**. Number of bit positions a given value shall be shifted, such as *shift left by n* or *shift right by n*

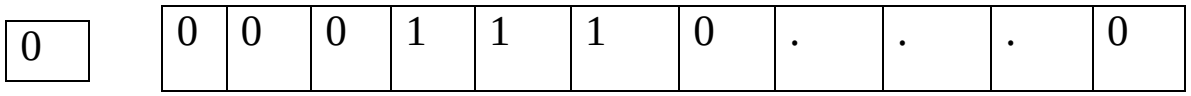
LshiftL R3,R3,#2





LshiftR R3,R3,#2

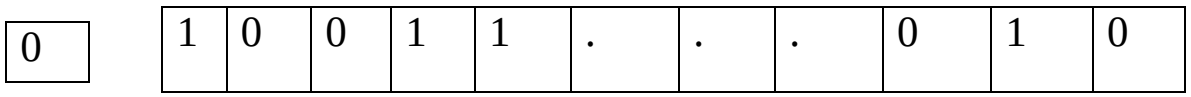


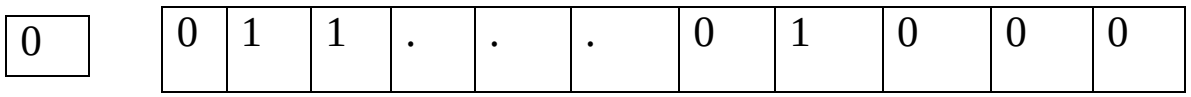


Arithmetic Instructions

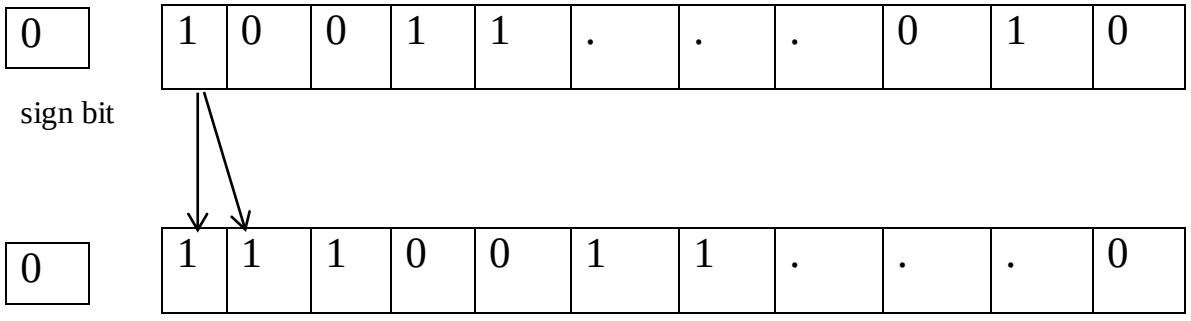
Arithmetic shift is a shift operator. Used to perform multiplication or division of signed integers by powers of two. It moves all the bits to the given number of bit positions.

AshiftL R3,R3,#2





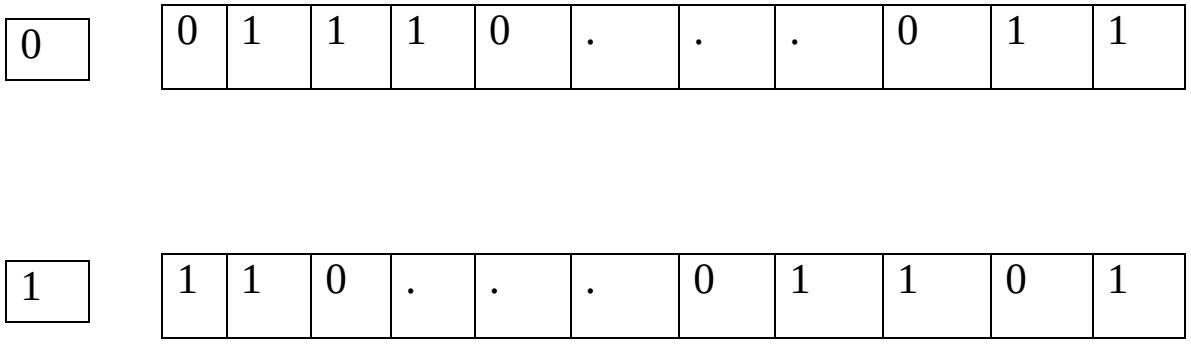
AshiftR R3,R3,#2



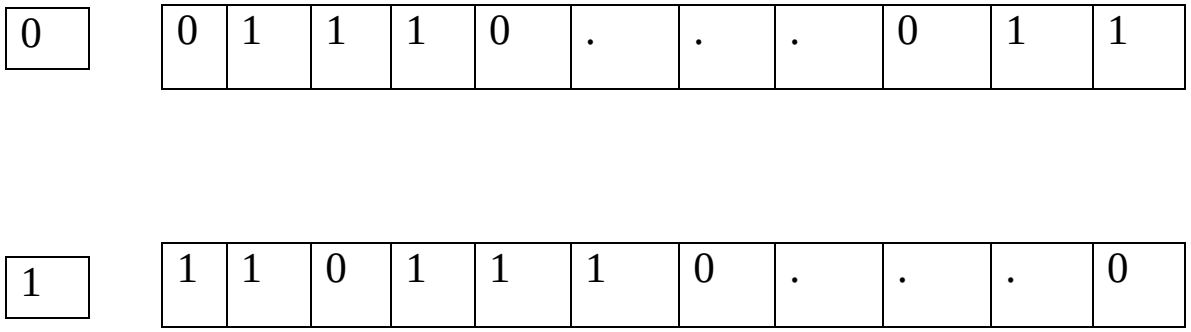
Rotate Instructions

It operates the content of the accumulator and the result is also stored in the accumulator. The Rotate instruction is used to rotate the bits of accumulator.

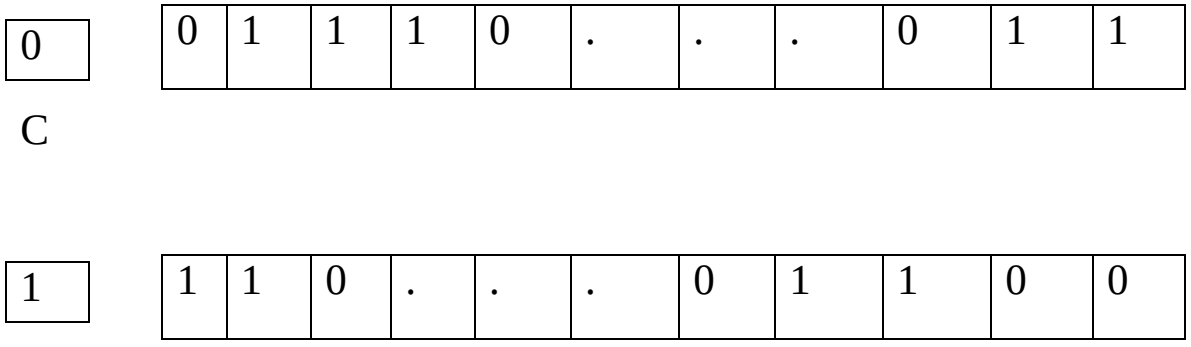
RotateL R3,R3,#2



RotateR R3,R3,#2



RotateLC R3,R3,#2 (Carry)



RotateRC R3,R3,#2

0	0	1	1	1	0	.	.	.	0	1	1

C

1	1	0	0	1	1	1	0	.	.	.	0

Applications of ROTATE Instructions: The ROTATE instructions are primarily used in arithmetic multiply and divide operations and for serial data transfer

Pipeline

Basic concepts

- Pipeline used to increase the system efficiency.

  IR doesn't wait until the next instruction to be fetched. when executing the current instruction, the next instruction to be processed will be fetched.

- It allows storing and executing instructions in an orderly process. It is also known as **pipeline processing**.

- It arranges the hardware such that more than one operation can be performed at the same time.
- It improves the hardware efficiency by introducing faster circuits.
- It increases the overall instruction throughput.

  Throughput = Number of instructions / Total time taken to complete the instructions

- It reduces the cycle time of the processor

  Non-overlapped execution

Cycle	1	2	3	4	5	6	7	8
S1	I1				I2			
S2		I1				I2		
S3			I1				I2	
S4				I1				I2

Total time = 8 clock cycle

Overlapped execution

Cycle	1	2	3	4	5	6	7	8
S1	I1	I2						
S2		I1	I2					
S3			I1	I2				
S4				I1	I2			

Total time = 5 clock cycle

- 5 Stages of pipeline
    1. Instruction fetch
    2. Instruction Decode
    3. Instruction Execute
    4. Memory Access
    5. Write back

Types of pipeline

1. Arithmetic pipeline (used for floating point operations)
2. Instruction pipeline (Used for fetching, decoding and executing the instructions)

Pipeline organization

Pipelining is a technique where multiple instructions are overlapped during execution. Pipeline is divided into stages and these stages are connected with one another to form a pipe like structure.Pipeline has two ends, Input and output. Between these two ends, there are multiple segments such that the output of one stage is connected to the input of the next stage and each stage performs a specific operation. Instructions enter from one end and exit from other end. Interface registers are used to hold the intermediate output between two stages. Interface registers and all the stages in the pipeline are controlled by a common clock.

Pipeline issues

Whenever a pipeline has to stall due to some reason it is called pipeline hazards.
Below we have discussed four pipelining hazards.

1. **Timing variations**
   It states that different instructions have different processing time.
2. **Data hazards**

When several instructions are in partial execution because of pipelining execution, they may reference same data then the problem arises.

3. **Interrupts**

   Pipeline sets an unwanted instruction into instruction stream then the interrupts occur during the instruction being executed

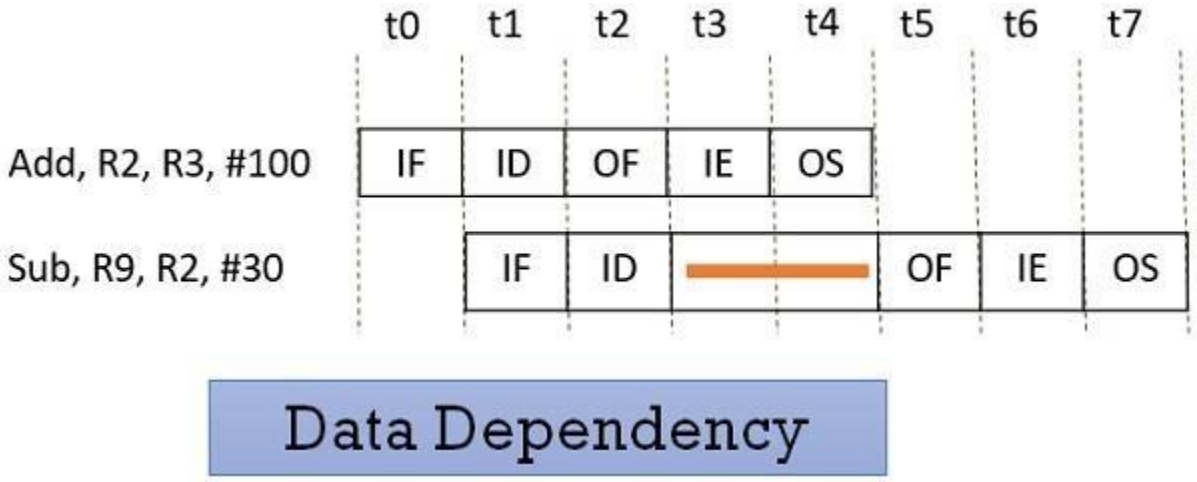
4. **Resource Limitation**

   If two instruction request for accessing the same resource in clock cycle, then one of the instruction has to stall and the other instruction has the access to use the resource

5. **Data dependency**

   It arises when an instruction depends upon the result of a previous instruction butthis result is not yet available.

Consider the following two instructions and their pipeline execution:



The result of the **Add** instruction is stored in the register **R2** and the final result is stored at the end of the execution ofthe instruction which will happen at the clock cycle **t4**.

But the **Sub** instruction need the value of the register **R2** at the cycle **t3**. So the Sub instruction has to **stall** two clock cycles. If it doesn't stall it will generate an incorrect result. Thus depending of one instruction on other instruction for data is **data dependency**.

6. **Memory Delay**

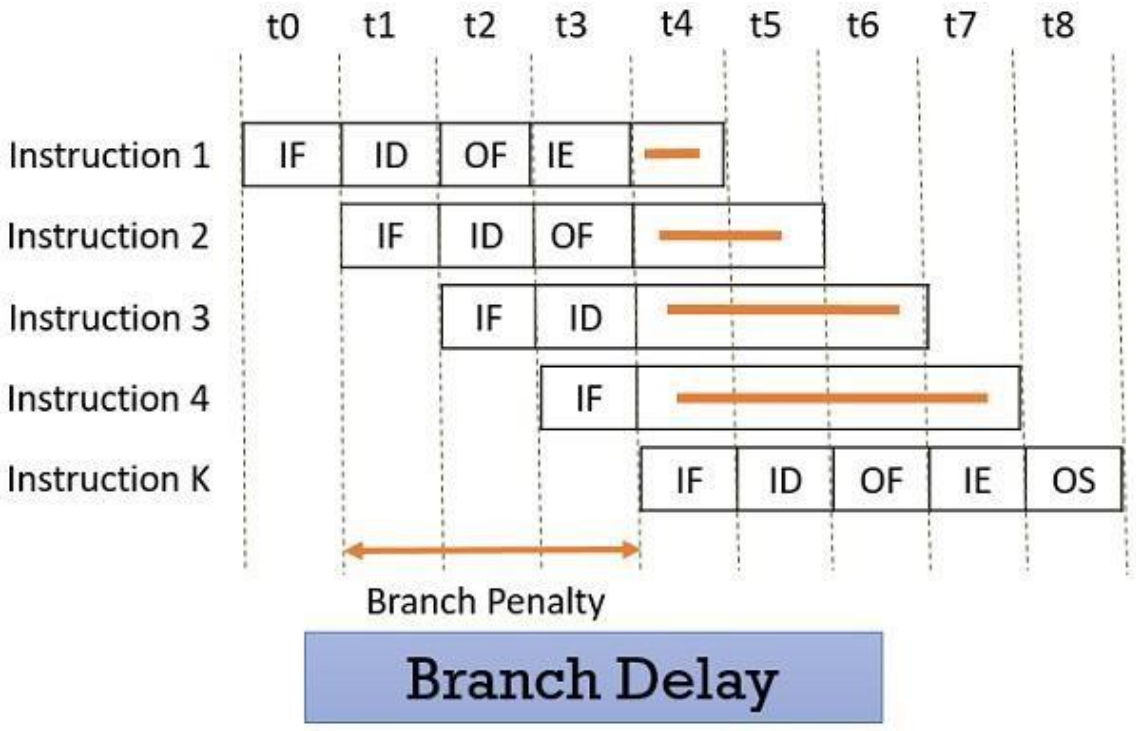
   When an instruction or data is required, it is first searched in the cache memory if not found then it is a **cache miss**. The data is further searched in the memory which may take ten or more cycles. So, for that number of cycle the pipeline has to stall and this is a **memory delay** hazard. The cache miss, also results in the delay of all the subsequent instructions.

7. **Branch Delay**

   Suppose the four instructions are pipelined I_1, I_2, I_3, I_4 in a sequence. The instruction I_1 is a branch instruction and its target instruction is I_k. Now,

processingstarts and instruction I_1 is fetched, decoded and the target address is computed at the 4^{th} stage in cycle t3.

But till then the instructions I_2, I_3, I_4 are fetched in cycle 1, 2 & 3 before the targetbranch address is computed. As I_1 is found to be a branch instruction, the instructions I_2, I_3, I_4 has to be discarded because the instruction I_k has to be processed next to I_1. So, this delay of three cycles 1, 2, 3 is a **branch delay**.



Semiconductor memory

A device for storing digital information that is fabricated by using **integrated circuit** technology is known as ***semiconductor memory***.

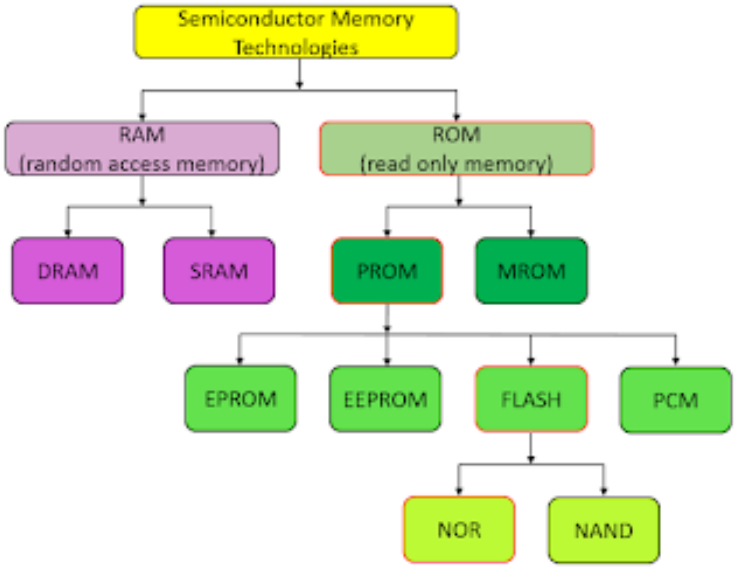
Semiconductor memory is the main memory element of a microcomputer-based system and is used to ***store program and data***.

The semiconductor memory is directly accessible by the microprocessor.

Types of semiconductor memory

Electronic semiconductor memory technology can be split into two main type

RAM - ***Random Access Memory***
ROM - ***Read Only Memory***



Random Access Memory (RAM)

Random access memory is a form of semiconductor memory technology that is used for reading and writing data in any order. It is a Volatile memory. The data stored in the memory is not permanent.

DRAM

It uses capacitor to store each bit of data and the level of charge on each capacitor determines whether that bit is a logical 1 or 0. However these capacitors do not hold their charge indefinitely, and therefore the data needs to be refreshed periodically. As a result of this dynamic refreshing it is called Dynamic RAM

SRAM

Static Random Access Memory. In this the data does not need to be refreshed dynamically. These semiconductor devices are able to support faster read and write times than DRAM and in addition its cycle time is much shorter

SDRAM (Synchronous Dynamic RAM)

It is synchronized to the clock of the processor and is capable of keeping two sets of memory addresses open simultaneously. By transferring data alternately from one set of addresses, and then the other, SDRAM cuts down on the delays associated with non-synchronous RAM, which must close one address bank before opening the next.

MRAM

This is ***Magneto-resistive RAM***, or Magnetic RAM. It is a non-volatile RAM memory technology that uses magnetic charges to store data instead of electric charges.

Read Only Memory (ROM)

A ***ROM*** is a form of semiconductor memory technology used where the data is written once and then not changed. In view of this it is used where data needs to be stored permanently, even when the power is removed. BIOS of a computer will be stored in ROM.

PROM

Programmable Read Only Memory. It is a semiconductor memory which can only have data written to it once, the data written to it is permanent. These memories are bought in a blank format and they are programmed using a special PROM programmer. Typically a PROM will consist of an array of fuseable links.

The PROM stores its data as a charge on a capacitor. There is a charge storage capacitor for each cell and this can be read repeatedly as required. However it is found that after many years the charge may leak away and the data may be lost.

EPROM

Erasable Programmable Read Only Memory. This form of semiconductor memory can be programmed and then erased at a later time. This is normally achieved by exposing the silicon to ultraviolet light. There is a circular window in the package of the EPROM to enable the light to reach the silicon of the chip.

EEPROM

Electrically Erasable Programmable Read Only Memory. Data can be written to it and it can be erased using an electrical voltage. This is typically applied to an erase pin on the chip.

Flash memory

Flash memory is a development of EEPROM technology. Data can be written to it and it can be erased, although only in blocks, but data can be read on an individual cell basis. To erase and re-program areas of the chip, programming voltages at levels that are available within electronic equipment are used.

Applications : Memory cards for digital cameras, mobile phones

PCM

This type of semiconductor memory is known as ***Phase change Random Access Memory***, P-RAM or just Phase Change memory, PCM. There are high resistance and low resistance state. It is possible to detect the state of an individual cell and hence use this for data storage.

Direct Memory Access

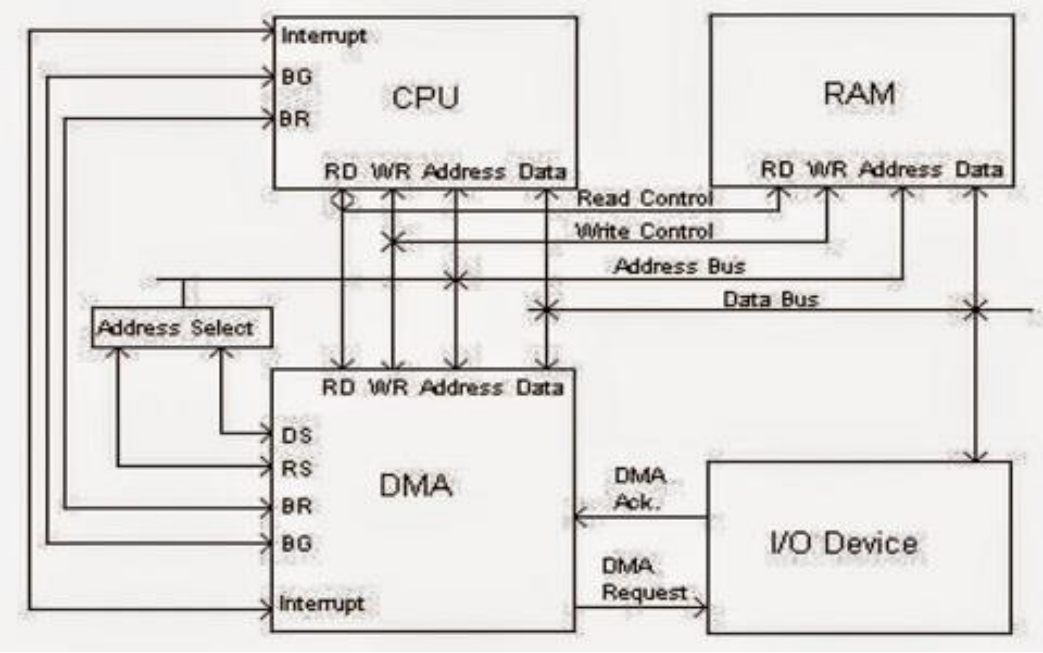
Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations.

The process is managed by a chip known as a DMA controller (DMAC).

Working of DMA Controller

DMA controller has to share the bus with the processor to make the data transfer. The device that holds the bus at a given time is called bus master. When a transfer from I/O device to the memory or vice versa has to be made, the processor stops the execution of the current program, increments the PC, moves data over stack then sends a DMA select signal to DMA controller over the address bus.

If the DMA controller is free, it requests the control of bus from the processor by raising the bus request signal. Processor grants the bus to the controller by raising the bus grant signal, now DMA controller is the bus master. The processor initiates the DMA controller by sending the memory addresses, number of blocks of data to be transferred and direction of data transfer. After assigning the data transfer task to the DMA controller, instead of waiting ideally till completion of data transfer, the processor resumes the execution of the program.



When an I/O device wants to initiate the transfer then it sends a DMA request signal to the DMA controller, for which the controller acknowledges if it is free. Then the controller requests the processor for the bus, raising the bus request signal. After receiving the bus grant signal it transfers the data from the device.

DMA controller now has the full control of buses and can interact directly with memory and I/O devices independent of CPU. It makes the data transfer according to the control instructions

received by the processor. After completion of data transfer, it disables the bus request signal and CPU disables the bus grant signal thereby moving control of buses to the CPU.

DMA has 3 registers

1.**Address register** : Holds the starting address of the memory

2. **Word count register** : Holds the word count value

3. **Control register** : It specifies whether read or write operation has performed

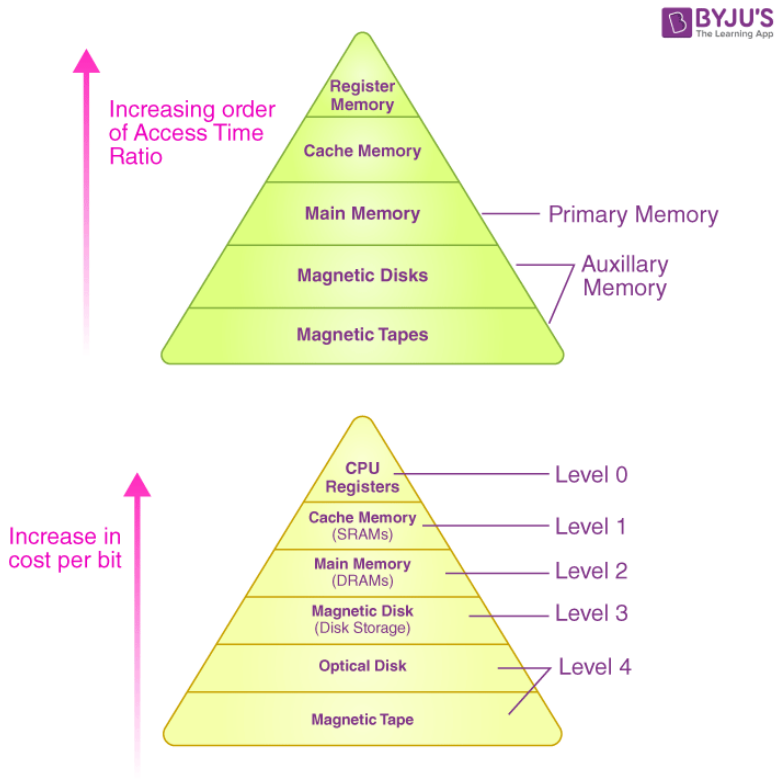
The DMA transfers the data in

a) **Burst Mode**: In this mode, all the word blocks are transferred in single bus.

b) **Cycle Stealing Mode**: In this mode, Words are transferred one by one. The DMA steals the bus from the CPU for transferring each word.

Memory Hierarchy

Memory Hierarchy, in Computer System Design, is an enhancement that helps in organising the memory so that it can actually minimise the access time. The computer memory can be divided into 5 major hierarchies that are based on use as well as speed



Internal Memory or Primary Memory

It consists of CPU registers, Cache Memory, and Main Memory. It is accessible directly by the processor.

1. Registers

The register is usually an SRAM or static RAM in the computer processor that is used to hold the data word that is typically 64 bits or 128 bits.

2. Cache Memory

The cache basically holds a chunk of information that is used frequently from the main memory.

3. Main Memory

In a computer, the main memory is nothing but the CPU's memory unit that communicates directly. It's the primary storage unit of a computer system. The main memory is very fast and a very large memory that is used for storing the information

External or Secondary Memory

It consists of Magnetic Tape, Optical Disk, Magnetic Disk, i.e. it includes peripheral storage devices that are accessible by the system's processor via I/O Module.

4. Magnetic Disks

In a computer, the magnetic disks are circular plates that's fabricated with plastic or metal with a magnetised material. It is used to save the data

5. Magnetic Tape

Magnetic tape refers to a normal magnetic recording designed with a slender magnetizable overlay that covers an extended, thin strip of plastic film. It is used mainly to back up huge chunks of data.

Characteristics of Memory Hierarchy

1. Capacity

It refers to the total volume of data that a system's memory can store.

2. Access Time

It refers to the time interval present between the request for read/write and the data availability

3. Performance

Because of this memory hierarchy design, the system's performance increased. One of the primary ways to increase the performance of a system is minimising how much a memory hierarchy has to be done to manipulate data.

4. Cost per bit

The cost per bit increases as one moves from the bottom to the top in the Memory Hierarchy, i.e. External Memory is cheaper than Internal Memory.