UNIT-IV

SHIFTERS: -

Shifters are important elements in many microprocessor designs for arithmetic shifting, logical shifting and rotation functions. Any general-purpose n bit shifter should be able to shift incoming data by up to n-1 places. Here all the shifts should be on end-around basis.

i.e for a 4-bit word, a I bit shift right is equivalent to 3 bit shift left etc.

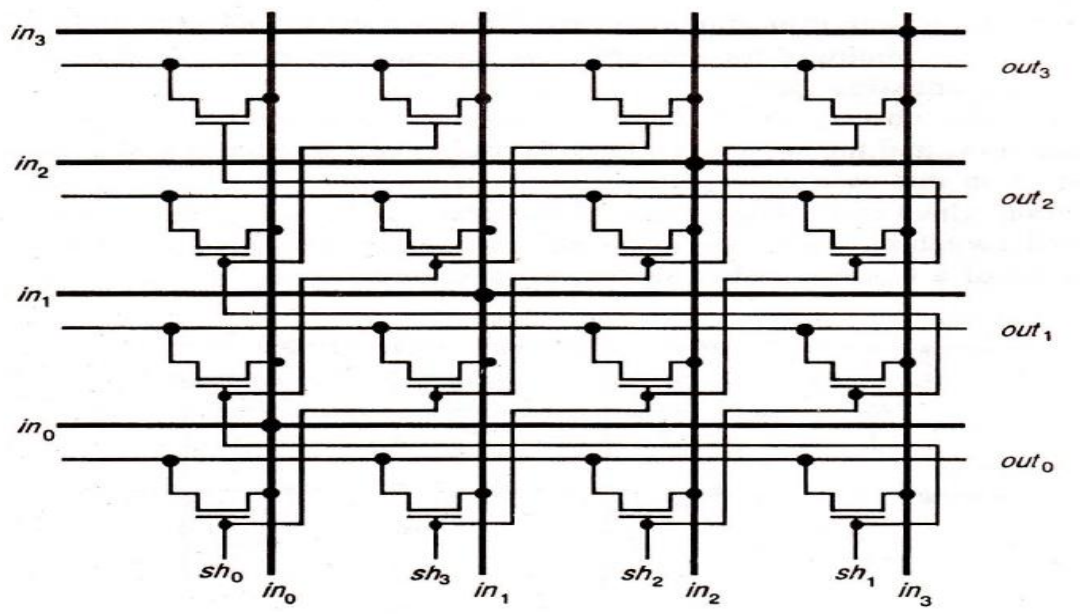
Thus, we can design a shifter that shift left & right by zero, one, two or three places by designing a circuit which will shift right only. To design a shifter the requirements are

1. The input from a 4-line parallel data bus.
2. 4 output lines for the shifted data.
3. Means of transferring input data to output lines with any shift from 0 to 3 bits for 4-bit shift register.

To meet the above requirements, the available switches like MOS pass transistor and transmission gates are used. The solution which meets these requirements is a cross-bar switch where data to flow horizontally and control signals vertically. Here in this approach 16 control signals are there to drive all transistors which increases complexity. A small change in above arrangement is known as barrel shifter. A **barrel shifter** is a digital circuit that can shift a data word by a specified number of bits without the use of any sequential logic, only pure combinational logic. A **barrel shifter** is often used to shift and rotate n-bits in modern microprocessors, typically within a single clock cycle.

Design of 4X4 Barrel Shifter:-

A barrel shifter shifts the data either left or right and any number of bits. The amount of shift is given as separate input. The design of 4 X 4-barrel shifter is shown below:



The circuit is designed by using pass transistors and it is placed in matrix format. The input to the shifter is the value to be shifted in[3:0] and shift amount sh[3:0]. The function to be performed depends on connections of bus and output is in out[3:0]. The above shifter is used to shift left. The interbus switches have their gate inputs connected in a staircase fashion in groups of four and there are 4 shift control inputs. CMOS transmission gates may be used in place of simple pass transistor switches.

Design of ALU:-

An arthematic logic unit (ALU) be able to add two binary numbers (A+B) & must also be able to subtract (A-B). For the logical operations it is able to do AND for two binary numbers (A.B), it is desired to OR (A+B) and to do Ex-Or Function and also detect equality. Subtraction by an adder is an easy operation. To find the difference A-B, it is necessary to compliment B and Add '1' to number to obtain 2's Compliment. Then Add this quantity to A using standard addition process. It is desirable to keep the architecture of ALU as simple as possible by designing the adder to perform logical operations and to perform subtraction.

- For an adder, $S_k = \overline{H_k} C_{k-1} + H_k \overline{C_{k-1}}$

- New carry $C_k = A_k B_k + H_k C_{k-1}$

  where $H_k = Half\ Sum = A_k\overline{B_k} + \overline{A_k}B_k$ and C_{k-1} is carry input.

Consider the sum output,

let $C_{k-1} = 0$, then $S_k = \overline{H_k}.0 + H_k.1 = H_k = A_k\overline{B_k} + \overline{A_k}B_k \rightarrow \boldsymbol{Ex - Or\ Operation}$.

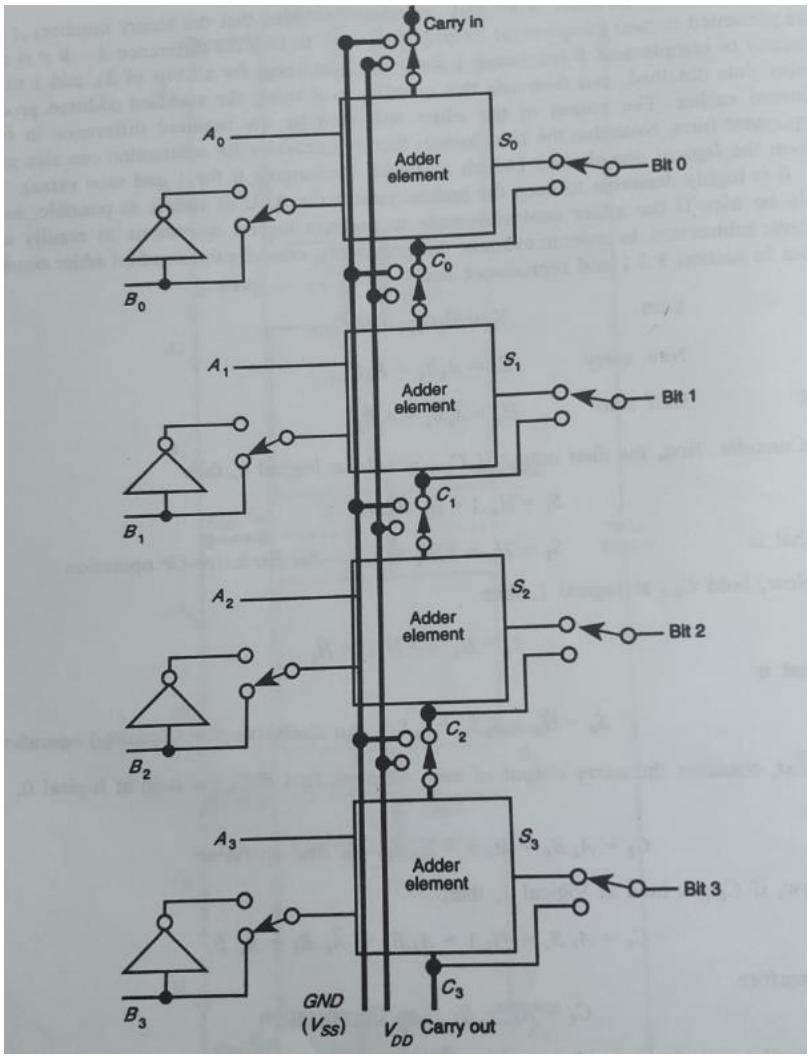
let $C_{k-1} = 1$, then $S_k = \overline{H_k}.1 + H_k.0 = \overline{H_k} = \overline{A_k B_k} + A_k B_k \rightarrow \boldsymbol{Ex - Nor\ Operation}$.

Consider the carry output,

let $C_{k-1} = 0$, then $C_k = A_k B_k + H_k.0 = A_k B_k \rightarrow \boldsymbol{AND\ Operation}$

let $C_{k-1} = 1$, then $C_k = A_k B_k + H_k.1 = A_k B_k + A_k\overline{B_k} + \overline{A_k}B_k = A_k + B_k \rightarrow \boldsymbol{OR\ Operation}$

The suitable switching of carry line between adder elements will give ALU logical operations. The possible arrangement of adder elements for both Arithmetic & Logical functions is



Adders

Adders form the basis for many processing operations from counting to multiplication to filtering. So adder circuits are very important to digital system. Adders can be implemented in various forms to meet different speed and density requirements.

They are

- Half Adder
- Full Adder
- Ripple Carry Adder (RCA)
- Carry Look Ahead Adder.
- Carry Save Adder.
- Carry Select Adder.

For any adder A, B are adder inputs, C is carry input, SUM is sum output & CARRY is carry output. The generate signal G is A.B occurs when a carry output is internally generated within the adder. When the propagate signal P is (A+B) true, then the carry-in C is passed to the carry output (CARRY) when C is true.

Single Bit Adders:-

The truth table for binary full adder is

C	A	B	A.B (G)	A+B (P)	SUM	CARRY
0	0	0	0	0	0	0
0	0	1	0	1	1	0
0	1	0	0	1	1	0
0	1	1	1	1	0	1
1	0	0	0	0	1	0
1	0	1	0	1	0	1
1	1	0	0	1	0	1
1	1	1	1	1	1	1

From the above truth table,

$$sum = \bar{A}B\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + ABC$$

$$= \bar{C}\{\bar{A}B + A\bar{B}\} + C\{\bar{A}\bar{B} + AB\}$$

$$= \bar{C}(A \oplus B) + C\overline{(A \oplus B)}$$

$$sum = A \oplus B \oplus C$$

$$Carry = AB + BC + CA$$

$$= \overline{\overline{AB + C(A + B)}}$$

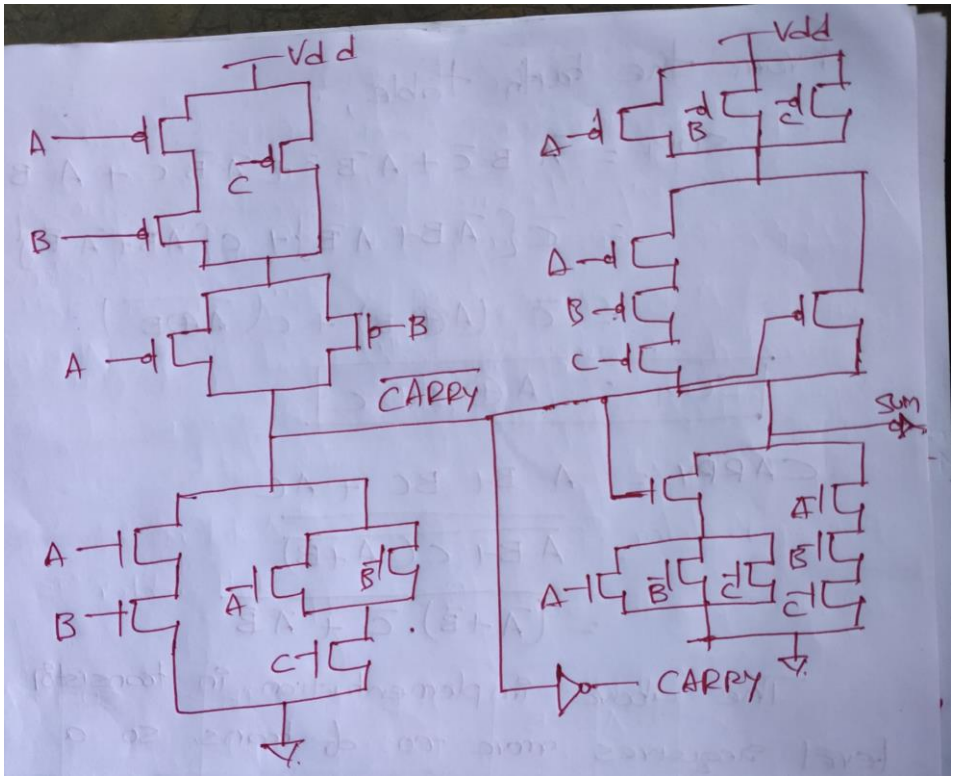
$$= \overline{(\bar{A} + \bar{B}).\bar{C} + \bar{A}\bar{B}}$$

The above implementation in transistor level requires more number of transistors. So a compact design based on the sum is designed to reuse carry term as follows.

$$SUM = ABC + (A + B + C)\overline{CARRY}$$

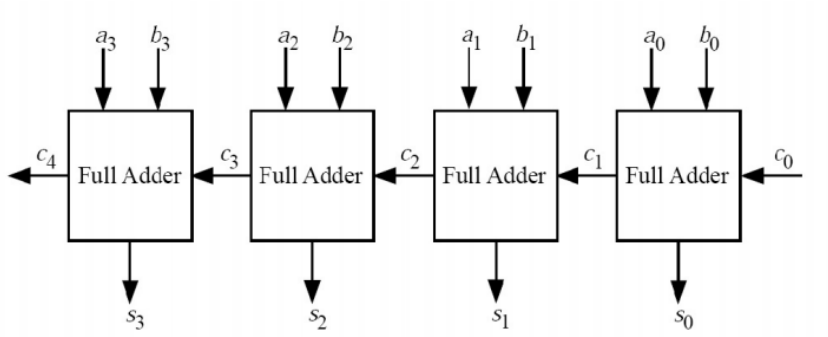
$$= ABC + (A + B + C)\overline{AB + C(A + B)}$$

The schematic of single bit adder is shown below:



RIPPLE CARRY ADDER:-

An n bit adder may be represented by cascading n 1 bit adders. A simple adder is Ripple Carry Adder. A Ripple Carry Adder is a digital circuit that produces the arithmetic sum of two binary numbers. It can be constructed with full adders connected in cascaded, with carry output of each full adder is connected to carry input of next full adder in the chain. The inputs are n bit A, B Values. The carry signal of stage i is fed to carry signal of stage i+1 and the SUM signal forms n bit output. Since the carry output is used in generation of sum in the circuit , SUM will be delayed with respect to carry. The total delay associated with the adder is $T_n = nT_c$.



Carry Look Ahead Adder:-

This adder solves the carry delay problem by calculating carry signals in advance, based on input signals. It is based on fact that carry is generated in two cases:

1) When both bits A_i and B_i are 1

2) When any one of the bits is 1 and carry-in is 1.

The linear growth of adder carry delay with the size of word for an n-bit adder may be improved by calculating the carries to each stage in parallel

The carry $C_{i+1} = G_i + P_i C_i$

where $G_i = A_i . B_i \rightarrow$ Generate Signal

$P_i = A_i \oplus B_i \rightarrow$ Propagate Signal

On expanding, $C_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} G_{i-2} + \cdots \ldots \ldots + P_i \ldots \ldots . P_0 C_0$

$$Sum = C_i \oplus A_i \oplus B_i$$

The size and fan-in of the gates are needed to implement CLA. So, the number of stages of CLA is limited to FOUR. For four stages of CLA,

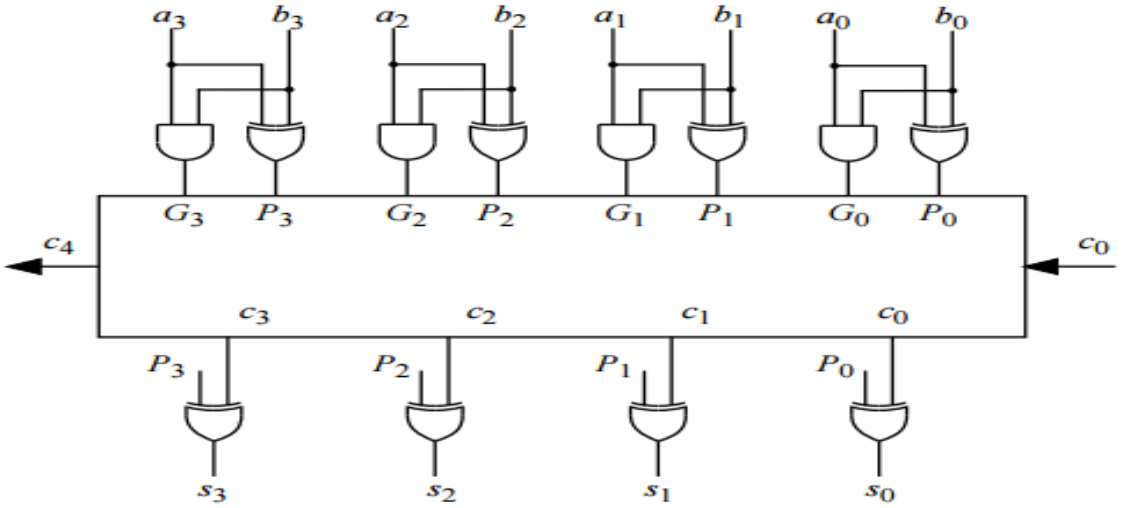
$$C_1 = G_0 + P_0C_0$$

$$C_2 = G_1 + P_1G_0 + P_1P_0C_0$$

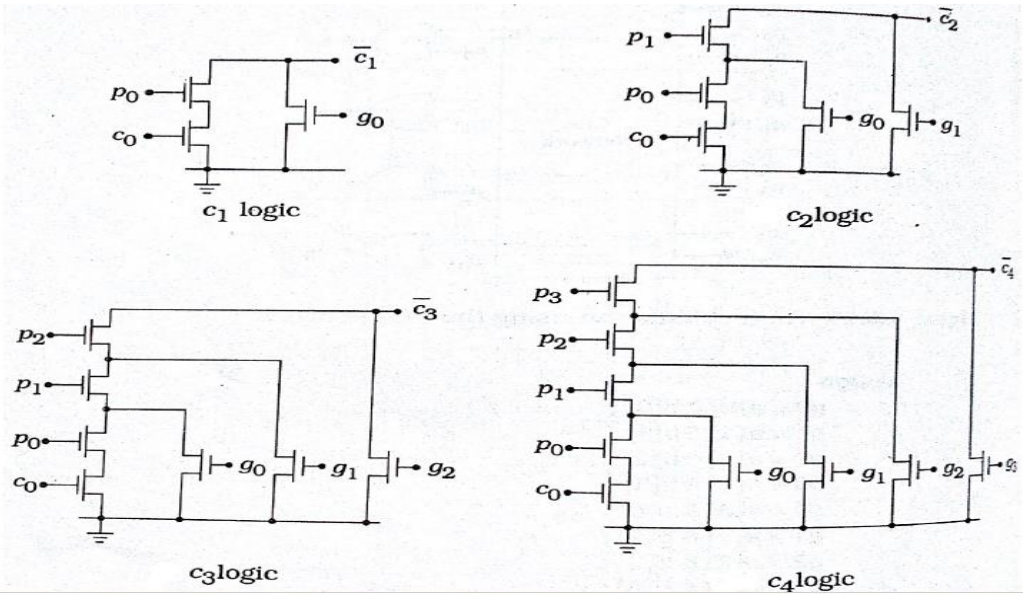
$$C_3 = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0$$

$$C_4 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 + P_3P_2P_1P_0C_0$$

The CLA of 4 stages is shown below:



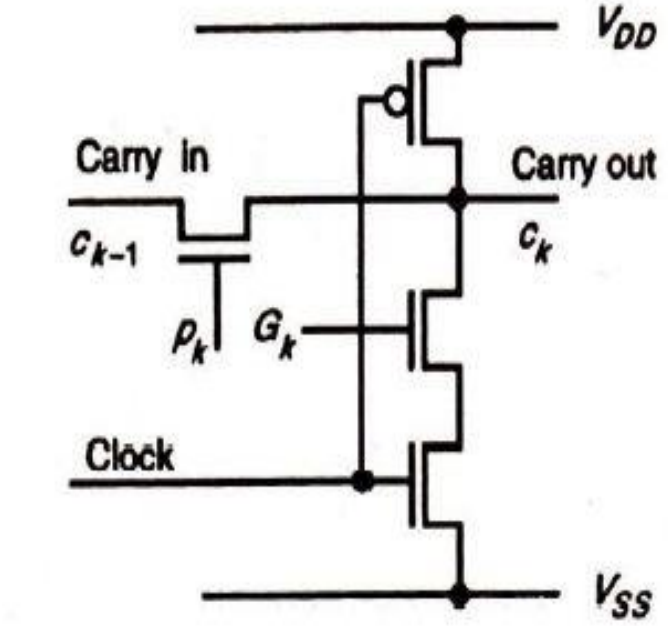
For C_1, C_2, C_3 and C_4 there are different implementation as shown below:



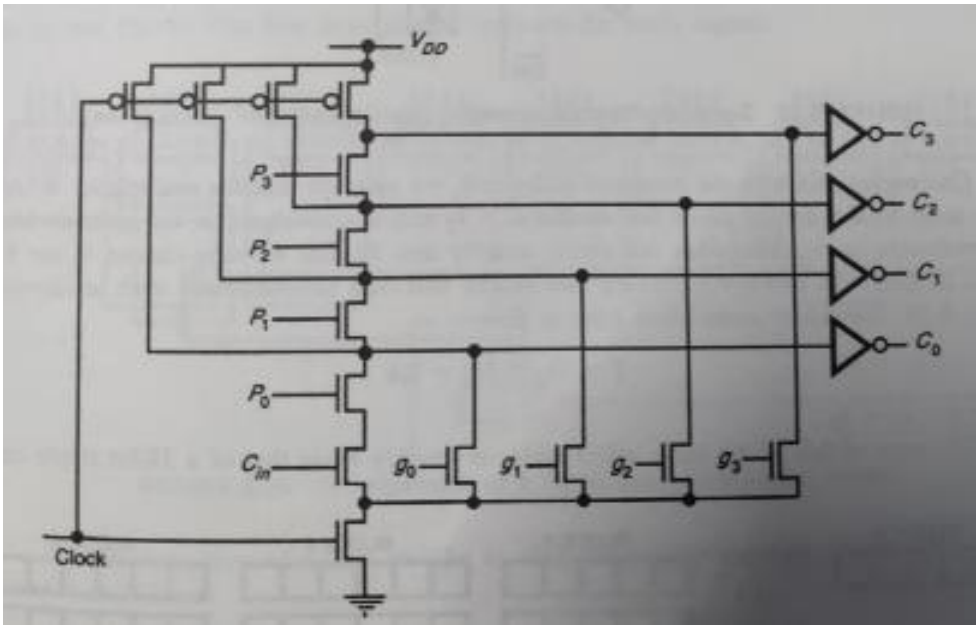
In order to reduce complexity , the dynamic logic technique is used called as '**Multiple output domino logic**' based on MANCHESTER CELL.

Manchester Carry Chain:-

Instead of carry passing through a complete Transmission gate, the carry path is precharged by clock signal and carry path may then be gated by a single n-type pass transistor as shown below:



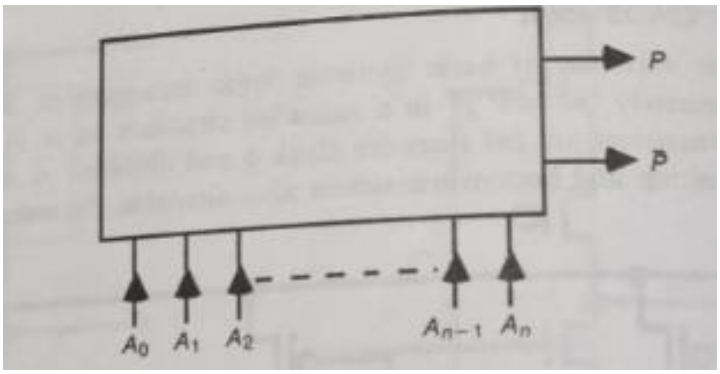
The Manchester cell is very fast, but a large set of such cascaded cell would be slow. Practically an inverter is added with every four cells. The Manchester carry chain for 4 bit CLA is shown below:



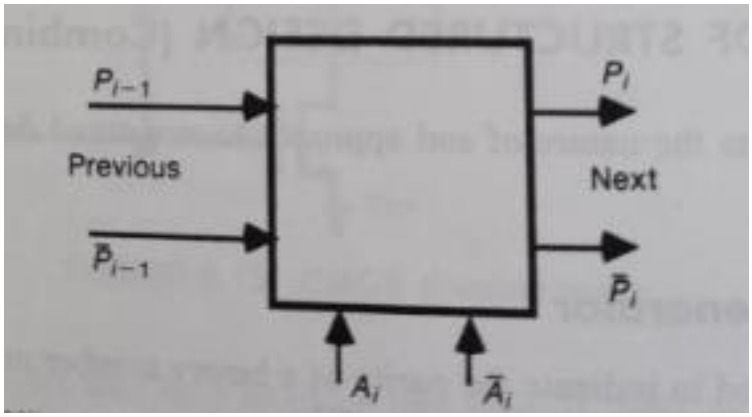
Parity generators:-

A circuit is to be designed to indicate the parity of binary numbers (or) word. The requirement is

$$P = \begin{cases} 1, \text{ Even number of } 1's \text{ at input} \\ 0, \text{ Odd number of } 1's \text{ at input} \end{cases}$$



Since the number of bits is undefined for the input, a general solution on a cascaded bit wise basis is to be found for any number of inputs, so that n can have any value. So, a standard or basic one bit cell from which n-bit parity generator may be formed as shown below.



The parity information is formed from one cell to the next cell and is modified or not by a cell depending on the state of input lines A_i and $\overline{A_i}$.

So, if $A_i = \begin{cases} 1, \text{ Parity is changed} \rightarrow P_i = \overline{P_{i-1}} \\ 0, \text{ Parity is unchanged} \rightarrow P_i = P_{i-1} \end{cases}$

These parity cells are cascaded in structured design approach. The basic parity cell is XOR gate. Parity is very useful tool in information processing in digital signals to indicate any presence of errors in bit information. The parity generating technique is one of the most widely used error detection techniques for the data transmission. In digital systems, when binary data is transmitted and processed , data may be subjected to noise so that such noise can alter 0's (of data bits) to 1's and 1's to 0's.

To indicate any presence of error in bit information, an extra bit is included with the message according to the total number of 1's in set of data called as '**Parity Bit**'. Hence, **parity bit** is added to the word containing data in order to make number of 1's either even or odd. Thus it is used to detect errors , during the transmission of binary data .

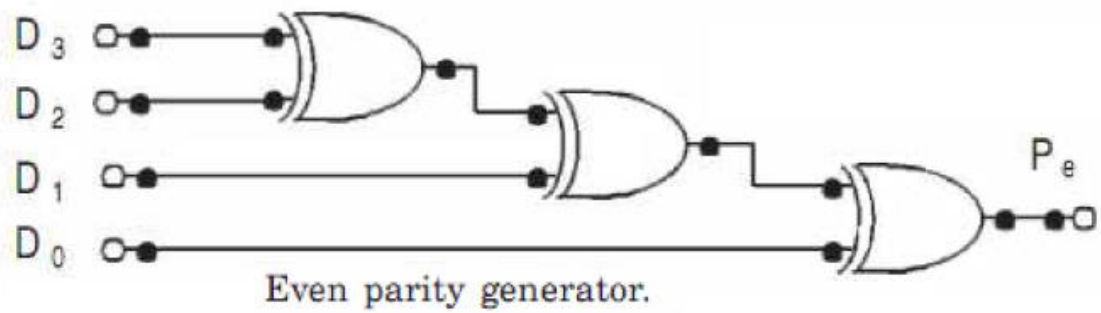
The message containing the data bits along with parity bit is transmitted from transmitter node to receiver node. At the receiving end, the number of 1's in the message is counted and if it doesn't match with the transmitted one, then it means there is an error in the data. A parity generator is a combinational logical circuit that generates the parity bit in the transmitter and the circuit that checks the parity in the receiver is called parity checker. There are two types of Parity: **Even & Odd Parity**.

If the extra bit is '1' for even quantities of '1's and '0' for odd number of 1's it is called as **Odd Parity.**

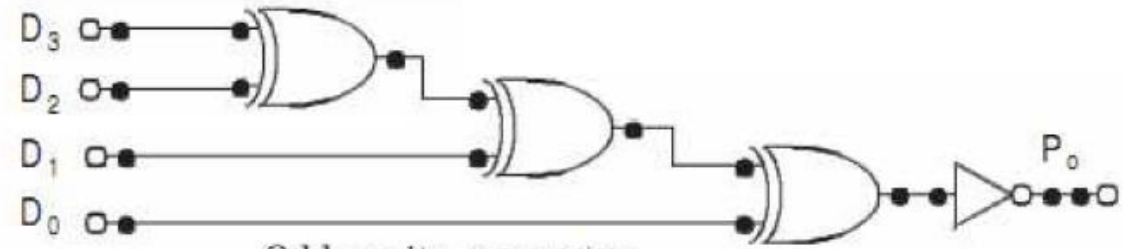
If the extra bit is '0' for even quantities of '1's and '1' for odd number of 1's it is called as **Even Parity.**

If the message bit combination is designated as $D_3D_2D_1D_0$,

then $P_e = D_3D_2D_1D_0$ and $P_0 = \overline{D_3D_2D_1D_0}$



Even parity generator.



Odd parity generator.

Four Bit Message				EVEN PARITY	ODD PARITY
D_3	D_2	D_1	D_0		
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	0	1
0	1	1	0	0	1
0	1	1	1	1	0
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	1	1	0
1	1	1	0	1	0
1	1	1	1	0	1

COMPARATORS:-

Another common and useful combinational logic circuit is digital comparator circuit. Digital or binary comparators are made up from standard AND,NOR and NOT gates that compare the digital systems present at their inputs terminals and produce an output depending upon the condition of those inputs. The input binary numbers need to be compared and determine whether the value of A is greater or smaller or equal to that of B. A comparator is a hardware electronic device that takes two numbers as input in binary form and determines whether one number is greater than, less than or equal to the other number. Comparators are used in central processing units (CPUs) and microcontrollers (MCUs). An XNOR gate is a basic comparator, because its output is "1" only if it's two input bits are equal. There are mainly two types of comparators.

Identity Comparator:-

It is a comparator that have only one output whether they are equal or not.

When $A = B$, the output is "HIGH"

$A \neq B$, the output is "LOW".

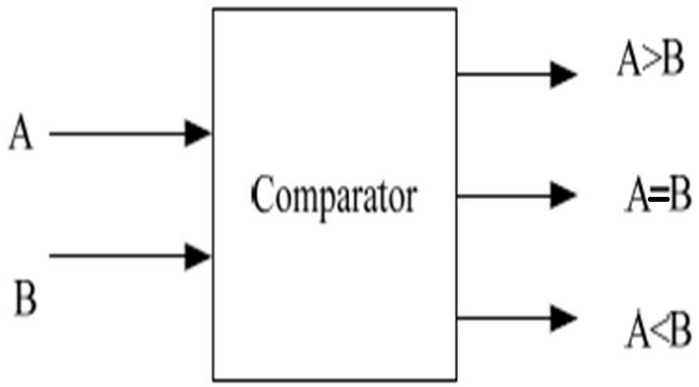
Magnitude Comparator:-

It is a Comparator that has 3 outputs which gives Equality (A=B), Greater Than (A>B) & Less than (A<B) outputs.

The purpose of digital comparator is to compare a set of variables and produce an output condition or flag depending on the result of comparison.

Consider a simple 1 bit comparator. The truth table is given below:

INPUTS		OUTPUTS		
A	B	A>B	A=B	A<B
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

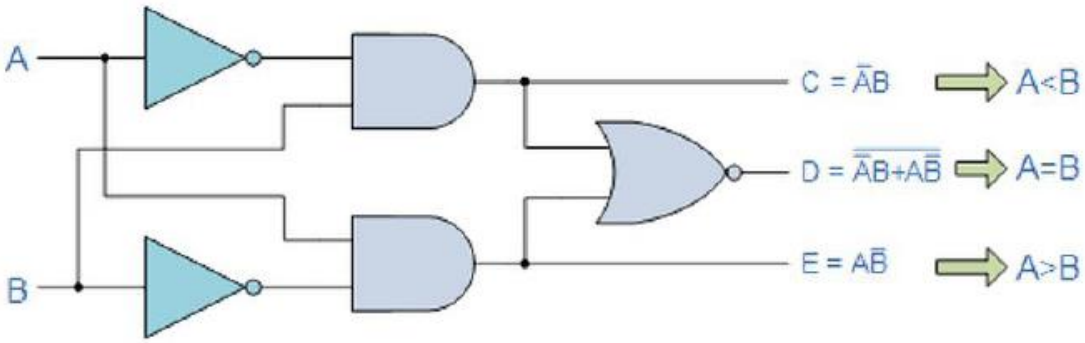


For $A > B$, $OUTPUT\ E = A\bar{B}$

$A = B$, $OUTPUT\ D = \bar{A}\bar{B} + AB$

$A < B$, $OUTPUT\ C = \bar{A}B$

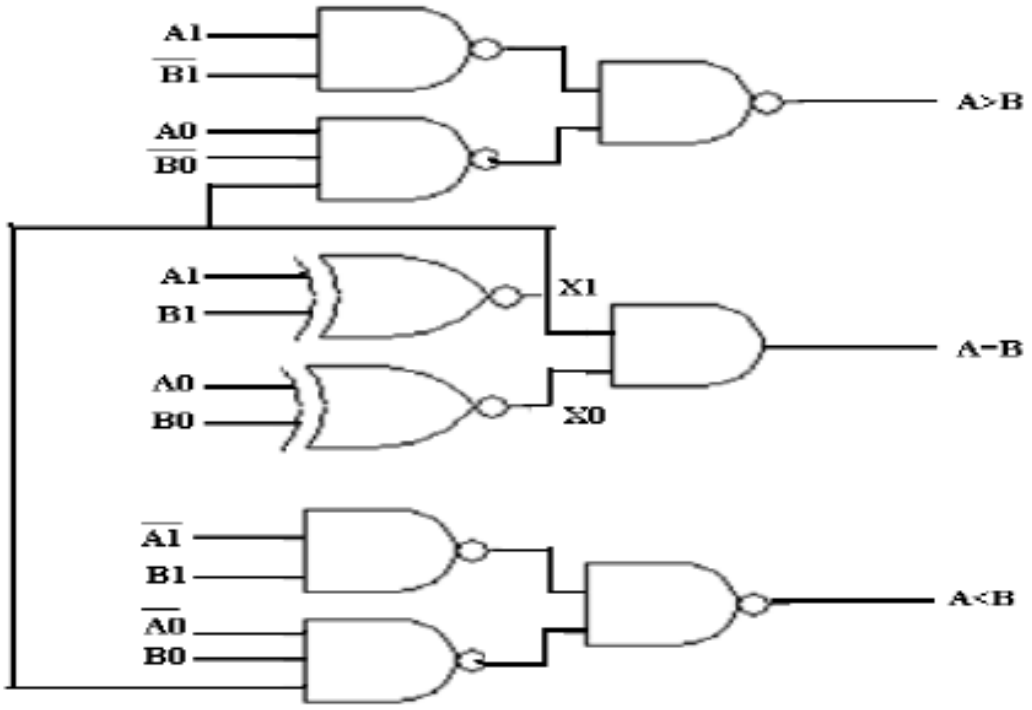
The logic diagram is



2 Bit Comparator

Truth Table of 2-Bit Magnitude Comparator

INPUT				OUTPUT		
A1	A0	B1	B0	A>B	A=B	A<B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0



Logic Diagram of 2-Bit Magnitude Comparator

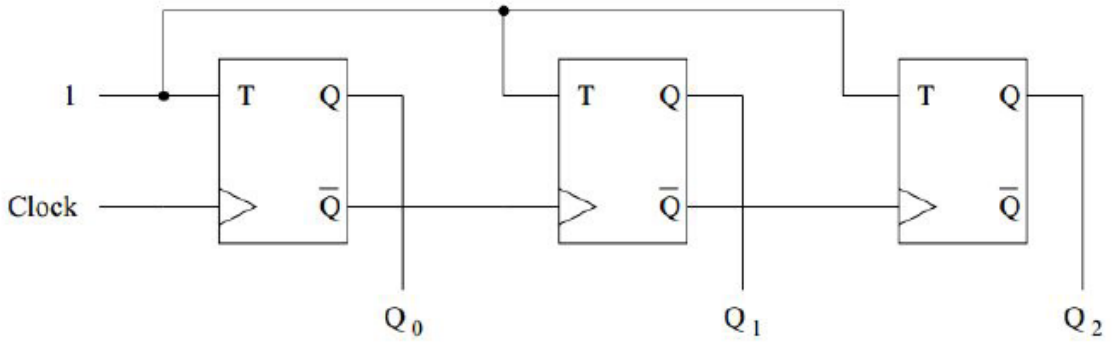
COUNTERS:-

Counters can be implemented using the adders / subtractors circuits and registers (Group of Flip Flop's). A counter is a device which stores (and sometimes it displays) the number of times a particular event or process has occurred, often in relationship to a clock signal. The simplest counter circuit is built by using T- Flip-flop's because the toggle feature is suited for implementation of counting operation. The counters are available in two categories. They are

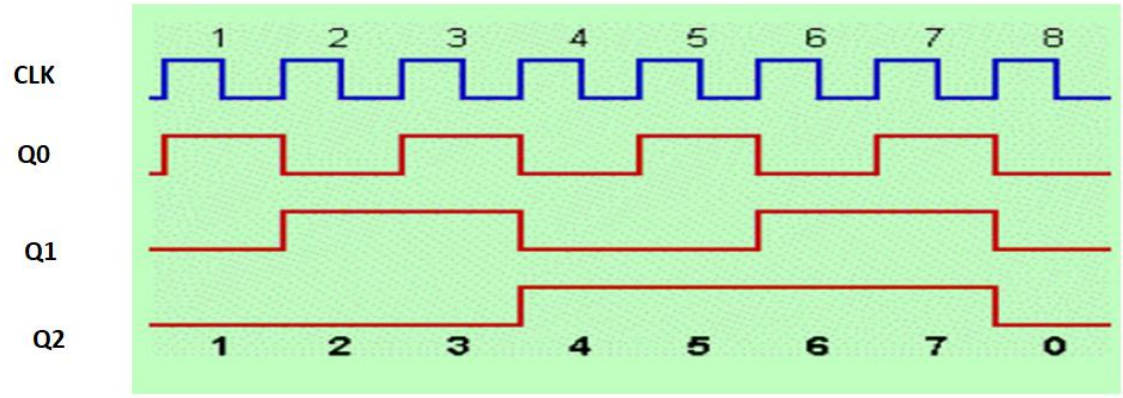
Asynchronous Counters:-

These counters are not clocked by a common pulse (or) clock , so each flip flop changes it's output at different times. The flip flop's in this counters are clocked by output pulse of the previous flip flop. The first flip flop is clocked by external pulse. Asynchronous counter is one in which each flip-flop is triggered by the output of the previous flip-flop, which limits the speed of the operation. The output transition acts as a source for triggering other flip flops. Asynchronous counters are also called ripple-counters.

For Example a 3-Bit Asynchronous Up-Counter is shown below:



Output changes it's state for every positive edge of clock pulse. The exterior clock is connected to the clock input of the first flip-flop only. So, this FF changes the state at the positive edge of every clock pulse. The other two flip flops have their clock inputs driven by $\bar{Q}$ of previous flip flop. Because of the internal propagation delay through a FF, the change of the input clock pulse and a change of the Q output of FF0 can never occur at precisely the same time. So, the FF's cannot be activated concurrently, generating an asynchronous operation.

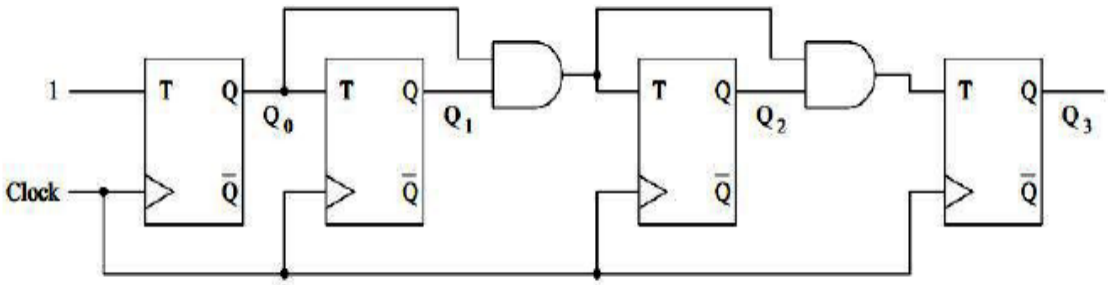


Incrementing for every clock pulse

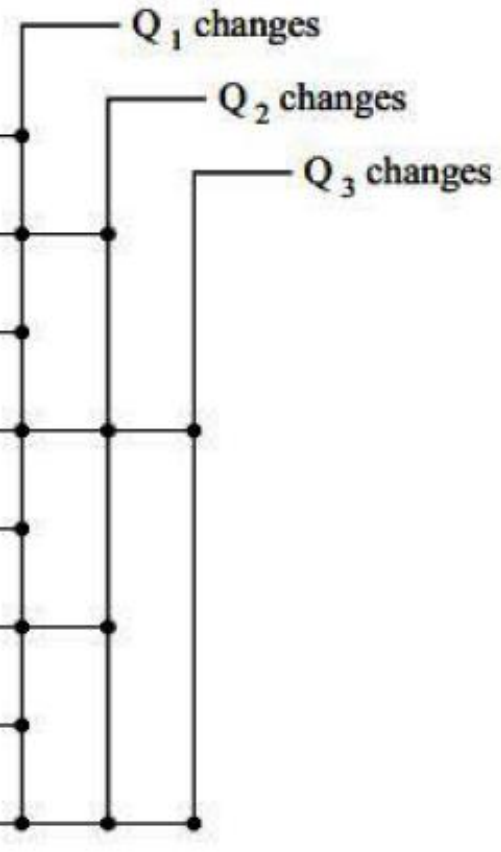
Synchronous Counter:-

These counters have a common clock for all the flip flop's to produce an output which is synchronized with the clock pulse. The CLK inputs of all the FFs are connected together and are activated by the input pulses. So, all the FFs change states instantaneously. These synchronous counters consists of two parts for implementation.

- Memory Element.
- Combinational element.



Clock Cycle	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0



When Q_0 changes from 1 to 0, then there is change in Q_1. When both Q_0 & Q_1 are 1, then there is change in Q_2. The change maybe from 0 to 1 or 1 to 0. When bothQ_2, Q_0 & Q_1 are 1, then there is change in Q_3. The change maybe from 0 to 1 or 1 to 0. It is faster than asynchronous counter. Delay of AND gates does not exceed clock time period. If it exceeds, then there is different values in output. For better operation, clock period should be greater than delay of logic gates. The set-up time of Flip Flop is matched with clock period.

VLSI Design styles

VLSI Design styles are the methodologies how an IC is designed and the way to approach for design the IC. An IC is an assembly of electronic components fabricated as a single unit in which active devices and passive devices and their interconnections are built upon a thin separate of semiconductor material. In VLSI an IC is fabricated in different styles.

The IC's are classified as

1. Full Customs IC's

2.Semi customs IC's

a) Standard cell based IC's

b) Gate array based IC's

c) Structured based IC's

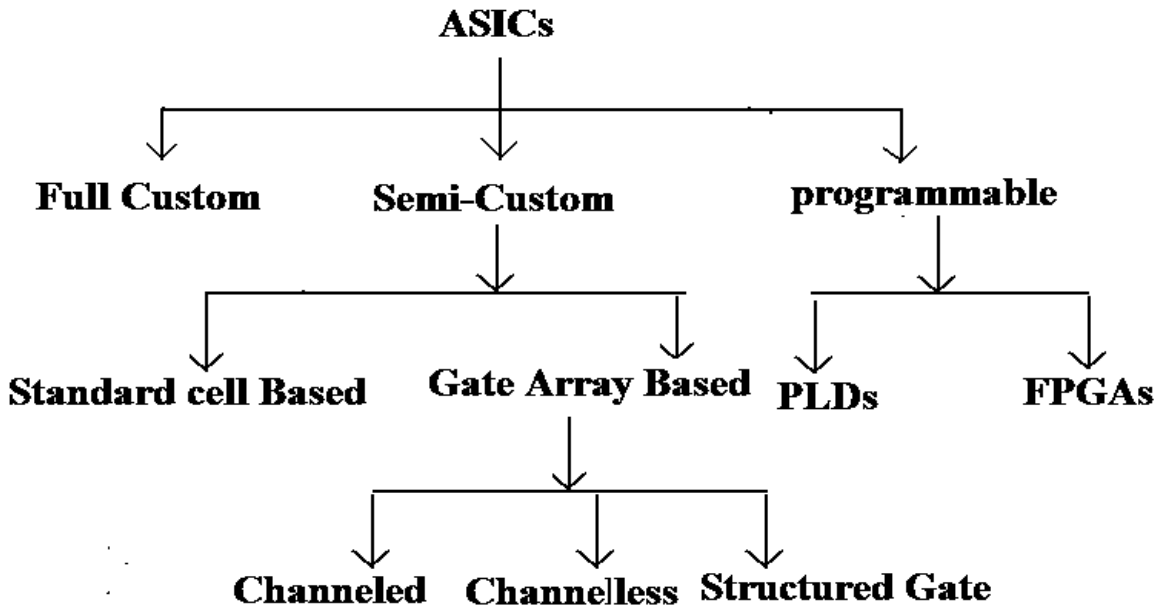
3.Programmable IC's

a) Programmable Logic Devices

b) FPGA's

These IC's are called as ASIC's. ASIC is Application Specific IC. It is non standard IC that is designed for specific use or Application.

Type's of ASIC's:- Based on the design technology ASICs are broadly classified into three types



Full Custom Design

A methodology for designing integrated circuits by specifying the layout of each individual transistor and the interconnections between them. Full customs means that we are doing work started from zero, it means to start your layout from scratch and to build all your design layout by self. A full custom ASIC is one which includes all logic cells that are customized and all mask layers that are customized. A full custom ASIC are the most expensive to manufacture and to design.

These specialized full custom IC's are often intended for specific application. These are used for mass productions. In a full-custom ASIC an engineer designs all of the logic cells, circuits, or layout specifically for one ASIC. It means the designer do not use the pre-tested and precharacterized cells for any part of the design. This is because the existing cell libraries are not fast enough or the logic cells are not small enough and consume large power. A microprocessor is an example of a full-custom IC. Designers spend many hours to prepare the microprocessor chip by manual work or hand. Customizing all of the IC features in this way allows designers to include analog circuits, optimized memory cells, or mechanical structures on an IC. The Full Custom IC technology is used when there are no existing cell libraries (or) circuits must be custom designed.

Advantages:-

1. For Specific application it can be used having a High density, ideal for high volumes

2.It improves Performance optimization.

Disadvantages:-

1. Most expensive to manufacture and to design.

2. The manufacturing lead time.

Semi Custom Design

The ASIC's for which all of the logic cells are predesigned and some of mask layers are customized are called semi custom ASIC's. Semi custom designs are performed at the logic level and it loses some of the flexibility available from a full customs design. Using a predefined cells from a cell library makes the design easier. In order to reduce the cost in most projects various design

approaches have been developed to decrease design time and automate the process of manufacturing.

Advantages:-

1.Short development cycle

2.Less development cost

3. Lowest risk approach

4.Suitable for small quantity of production

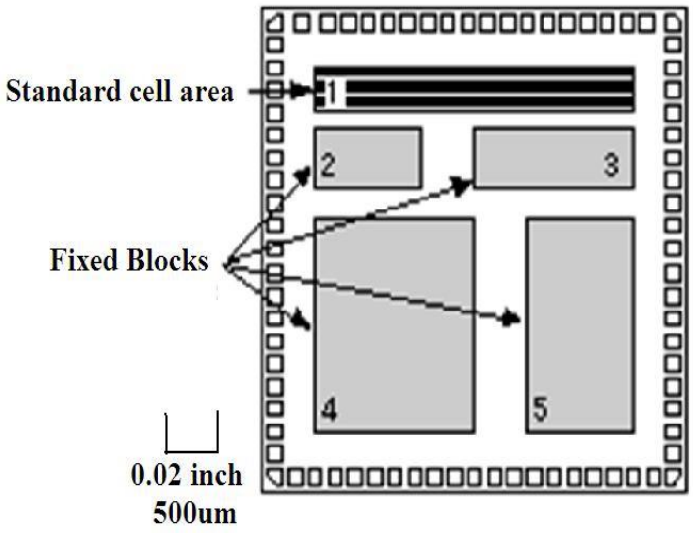
Disadvantages:-

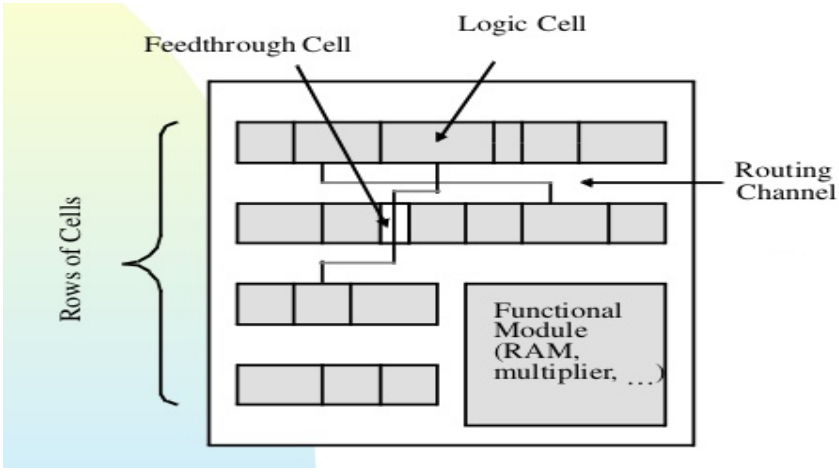
1.Suitable only for Digital IC's.

- There are two types of semi custom ASIC's. They are
  - Standard cell based ASIC's
  - Gate array based ASIC's

Standard Cell Based ASIC's:-

A cell based ASIC uses predesigned logic cells such as AND gates, OR gates, MUX, Flip Flops etc. which are called as standard cells. The standard cell areas called as flexible blocks in a CBIC are built of rows of standard cells. These standard cell areas may be used in combination with microcontrollers etc. called as Mega Cells used for large functions, full custom blocks, functional standard blocks. A Cell Based ASIC die with a standard cell area(Flexible Blocks) with four fixed blocks is shown below:





Routing channel is defined and is left between two rows. Feed through cell is used to connect alternate rows. ASIC designer defines only the placement of standard cells and the interconnect in a CBIC. So the standard cell can be placed anywhere on the silicon, it means all mask layers of a CBIC are customized and are unique to particular customer. CBIC are built of rows of standard cells like a wall built of bricks. Standard cell design are typically organized on the chip, rows of constant height cell. CAD tools automatically converts the design into a chip layout. Standard cell area may be used in combination with larger designed cells as functional module like RAM , Multiplexer. Each standard cell can be optimized individually. During the design of the cell library each & every transistor in standard cell can be chosen to maximize the speed (or) to minimize the area.

Advantage:-

- The designer save time ,money and reduced risk by using a predesigned ,pretested and precharacterized standard cell library.
- Standard cell can be optimized individually.

Dis-Advantage:-

- Time Required to fabricate all layers of IC for a new design
- Expense of designing or buying the standard –cell library.

Gate Array Based ASIC's

In a gate array based ASIC the transistors are predefined on the silicon wafer. The predefined pattern of transistors on a gate array is the **base array** and the smallest element that is replicated to make the base array is called

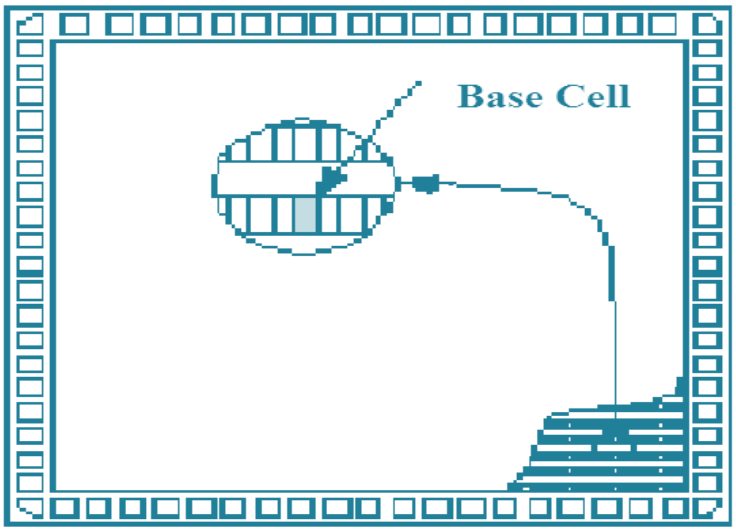
Base Cell. The base cell is also called as primitive cell. The silicon vendors provides master or base wafer to be then personalized according to the interconnection information supplied by the customers. Only the top few layers of metal which define the interconnect between transistors is defined by the designer. The gate array also called masked gate array(MGA) which uses library component that reduce the development time. The designer chooses from a gate array library of predesigned and precharaterized logic cells. The logic cells in a gate array are called macros. The reason is base cell layout is same for each logic cell and only the interconnect is customized.

Gate array design can be classified in to three types:-

1. Channeled Gate Array
2. Channel less Gate array
3. Structured Gate Array

Channeled Gate Array:-

Channel gate array is was the first to be developed. In this gate array, space is left between the rows of transistor based basic cells for wiring. A channeled gate array is similar to CBIC. Both uses the rows of cells separated by channels for interconnect. The difference is that space for interconnect is fixed in height in channeled gate array whereas the space between rows is adjustable in CBIC on layout. In channeled gate array the interconnections are drawn within predefined space between rows of logic cells.

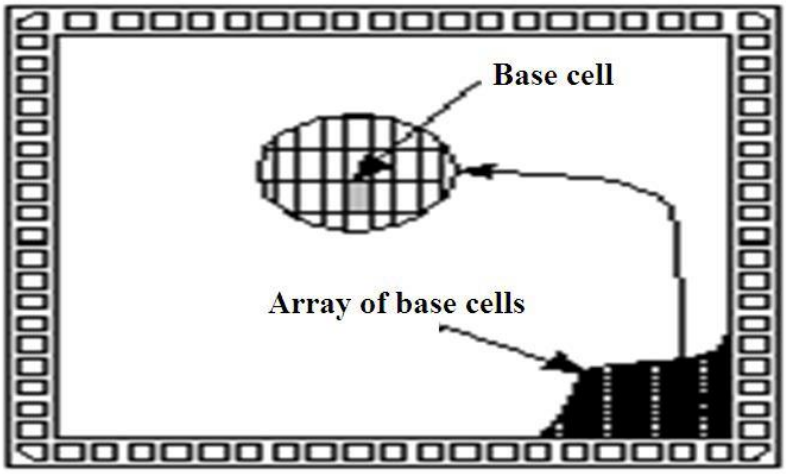


- **Features Of Channeled Gate Array**
- Only interconnection is customized

- The interconnect uses predefined space between rows of the base cells.
- Manufacturing lead time is between two days and two weeks.

Channel Less Gate Array

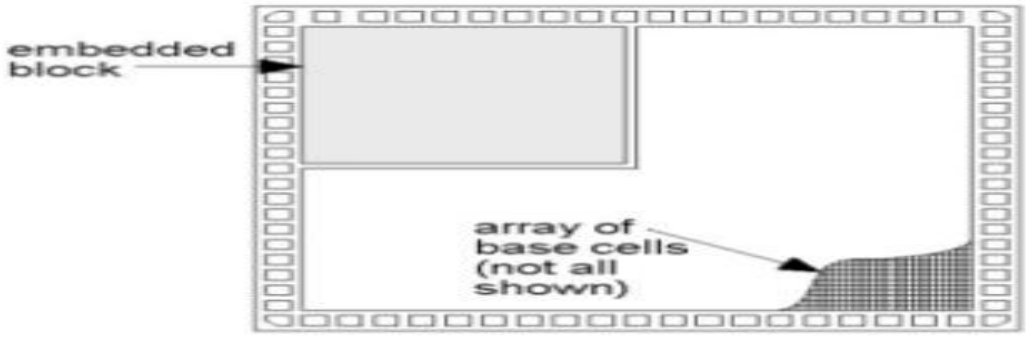
Channel less gate is also known as sea of gate array. This Channel less gate array architecture is widely used now a days. The routing is done by using unused transistors of basic cells. The key difference between a channel less gate array and channeled gate array is that there are no predefined areas set aside for routing between cells on a channel less gate array. The routing is done over the top of devices by connecting metal 1 with transistors of base cell. In a channel less gate there are no connections in the channel and the connections are drawn with the upper metal layers, i.e., on the top of the logic cells.



- **Features Of Channel less Gate Array**
- Only interconnection is customized
- The interconnect uses unused transistors of arrays.
- When we use an area of transistor for routing we do not make any contacts to the devices lying underneath, we simply leave the transistor unused.
- The logic density and amount of logic that can be implemented in a given area is higher for channel less gate arrays.

Structured Gate Array

Structured gate array or Embedded gate array combines the features of CBIC's and MGA's.



Features :-

1.Only the interconnection is customized.

2.Customs block which are same for each design can be embedded.

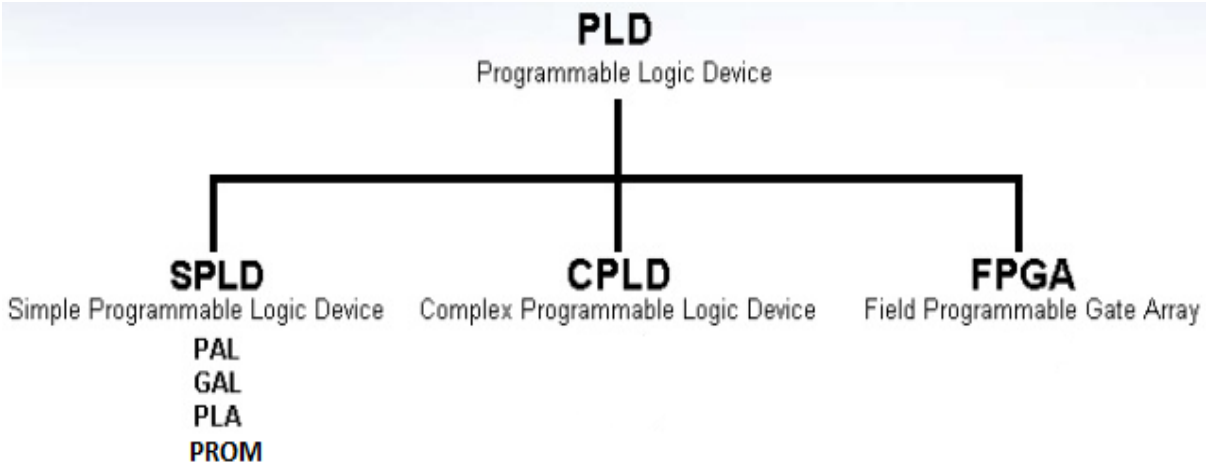
3.This array gives the improved area efficiency and increased performance of CBIC.

4. The disadvantage of an embedded gate array is that the embedded function is fixed.

Programmable Logic Devices

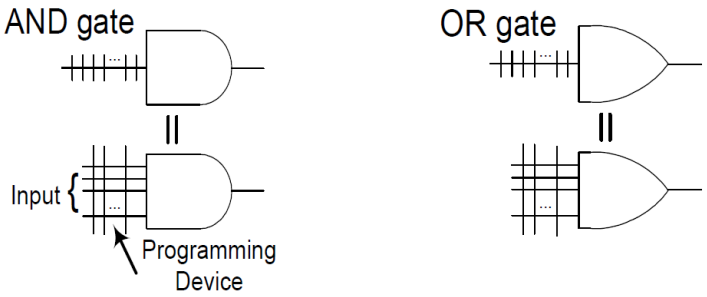
PLD is an electronic component used to build reconfigurable digital circuits. Standard IC's with internal logic gates and interconnects. These gates can be connected to obtain required logic function. The circuit can be erased electrically and reprogrammed with a new design , making them very well suited for academic and prototyping. So an integrated circuit chip that can be configured by the user to implement different digital hardware. The term "Programmable" means changing either hardware or software configuration of an internal logic & interconnects. So it is known as field programmable logic device. The purpose of PLD's is to allow logic circuits and designs to be implemented by user in a single device.

Types of PLD's

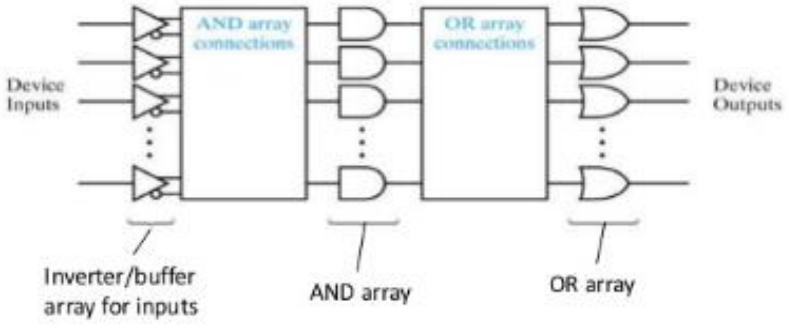


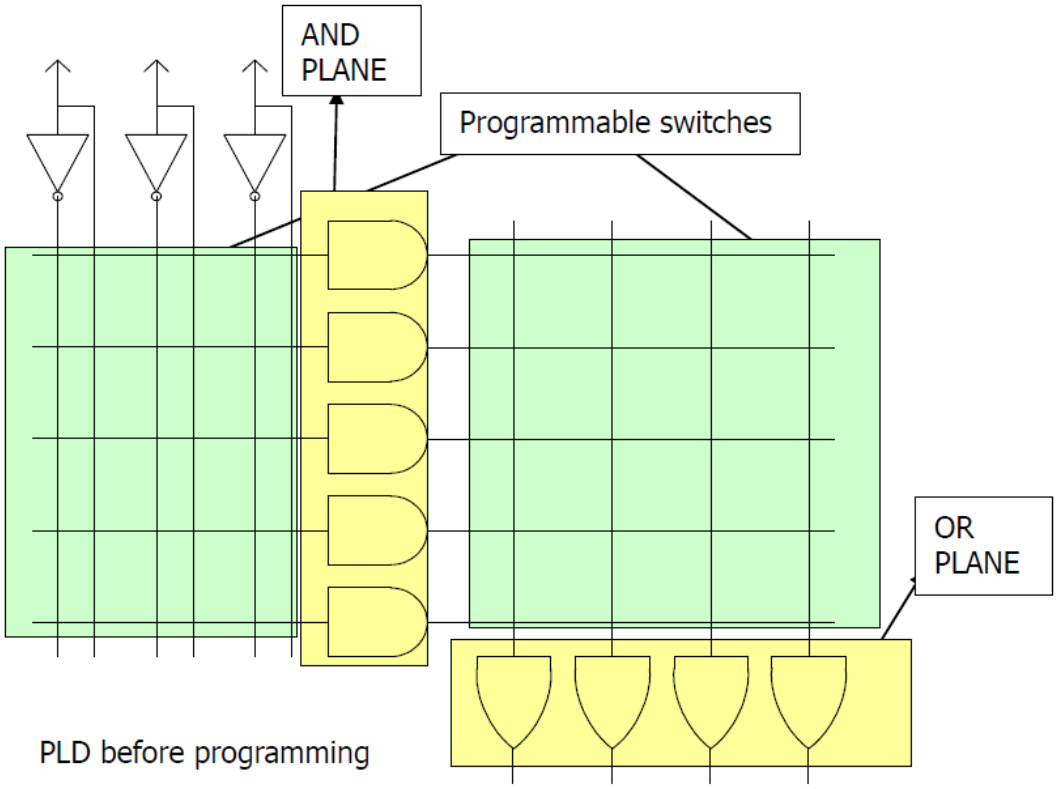
Structure of PLD

Any Boolean function can be expressed in Sum Of Products (SOP). This is implemented in AND-OR Form. So Simple PLD's will have **And Plane & Or plane** along with interconnects (or) programmable switches. The number of inputs and outputs are usually high. To avoid drawing all the inputs and outputs all the inputs to a gate are shown with one line only and the actual inputs are shown as an intersection as shown below:



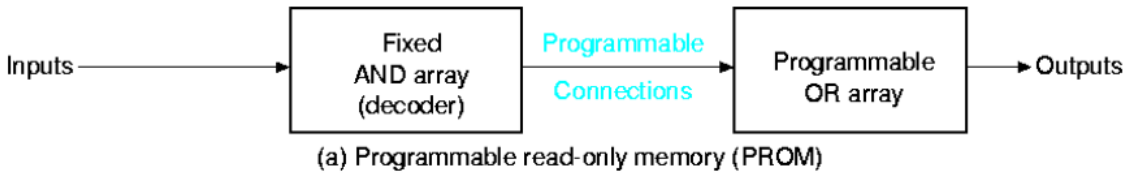
The structure of simple PLD's is shown below:



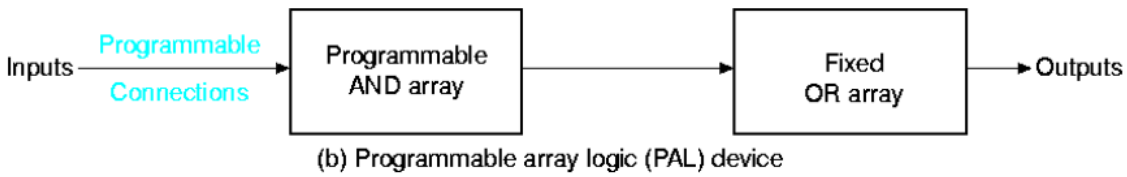


Based on programming of Simple PLd's, there are 3 types.

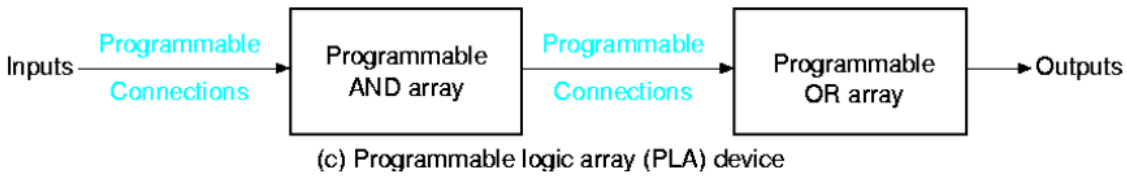
1) PROM- Fixed AND Plane, Programmable OR Plane
2) PAL- Programmable AND Plane, Fixed OR Plane
3) PLA-Programmable OR Plane, Programmable AND Plane.



(a) Programmable read-only memory (PROM)



(b) Programmable array logic (PAL) device



(c) Programmable logic array (PLA) device

PROM-Programmable Read Only Memory:-

It has fixed AND plane & Programmable OR Plane. So ROM has fixed connection at the input and output is programmed. For a ROM, if k bits are input , then 2^k AND plane is available and fixed. In the 2^k, if n outputs are there, then n outputs are programmed. It is denoted by

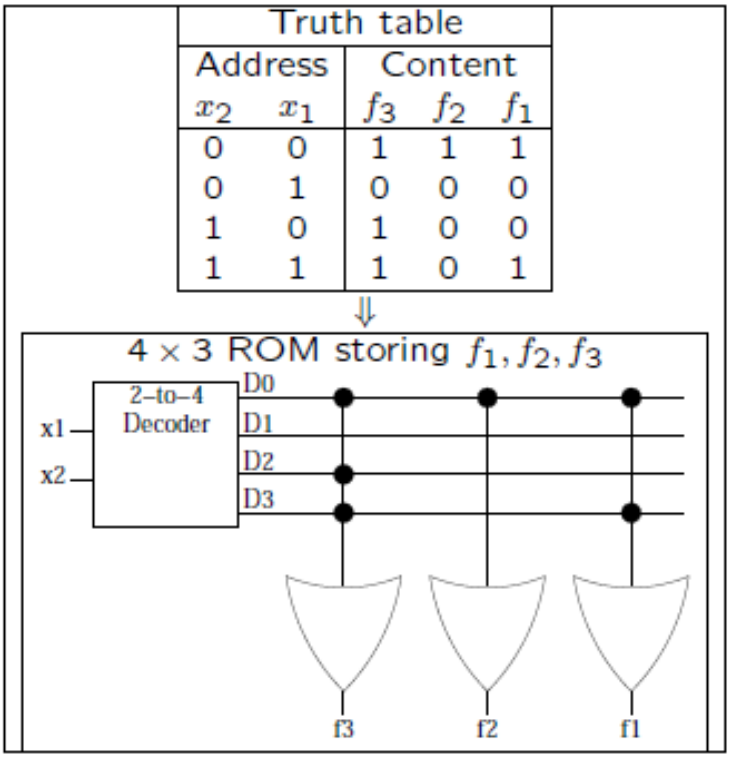
Read-Only Memory

k inputs (address) $\Rightarrow$ $2^k \times n$ ROM $\Rightarrow$ n outputs (data)

A PROM is designed by decoder & programmable OR Plane. A decoder will have fixed and plane. For $2^k \times n$ PROM will have, $k \times 2^k$ decoder to decode input address, n OR gates with 2^k inputs each, Decoder output is connected to all n OR gates through fuses.

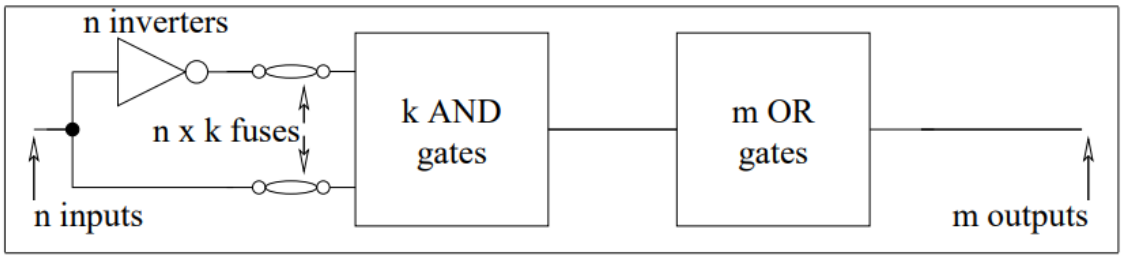
Example: Implement $f_1(x_2, x_1) = \sum m(0, 3)$, $f_2(x_2, x_1) = \overline{x_2 + x_1}$, and $f_3(x_2, x_1) = \prod M(1)$ with a 4×3 ROM

Truth table				
Address		Content		
x_2	x_1	f_3	f_2	f_1
0	0	1	1	1
0	1	0	0	0
1	0	1	0	0
1	1	1	0	1

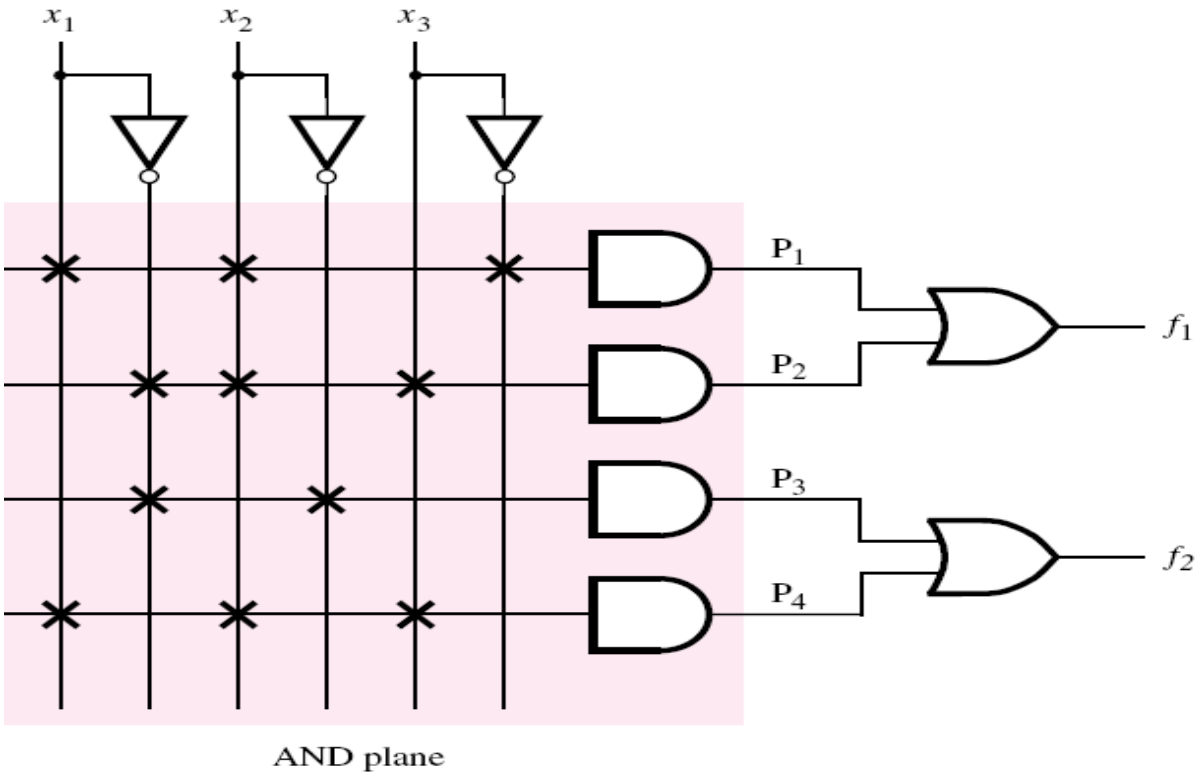


Programmable Array Logic

It has programmable And plane & fixed Or plane. It is more flexible than ROM but more difficult to program. Logic expressions to be stored in PAL is obtained and then minimized and programmed into PAL using programmable interconnects. Only the connection inputs to ANDs are programmable. Easier to program than but not as flexible as PLA. There are feedback connections. Logic expressions for content information to be stored in PAL must be obtained first, then minimized, and finally programmed into the PAL using a PAL program table. PAL program table specifies only product terms of information that will be stored in PAL.



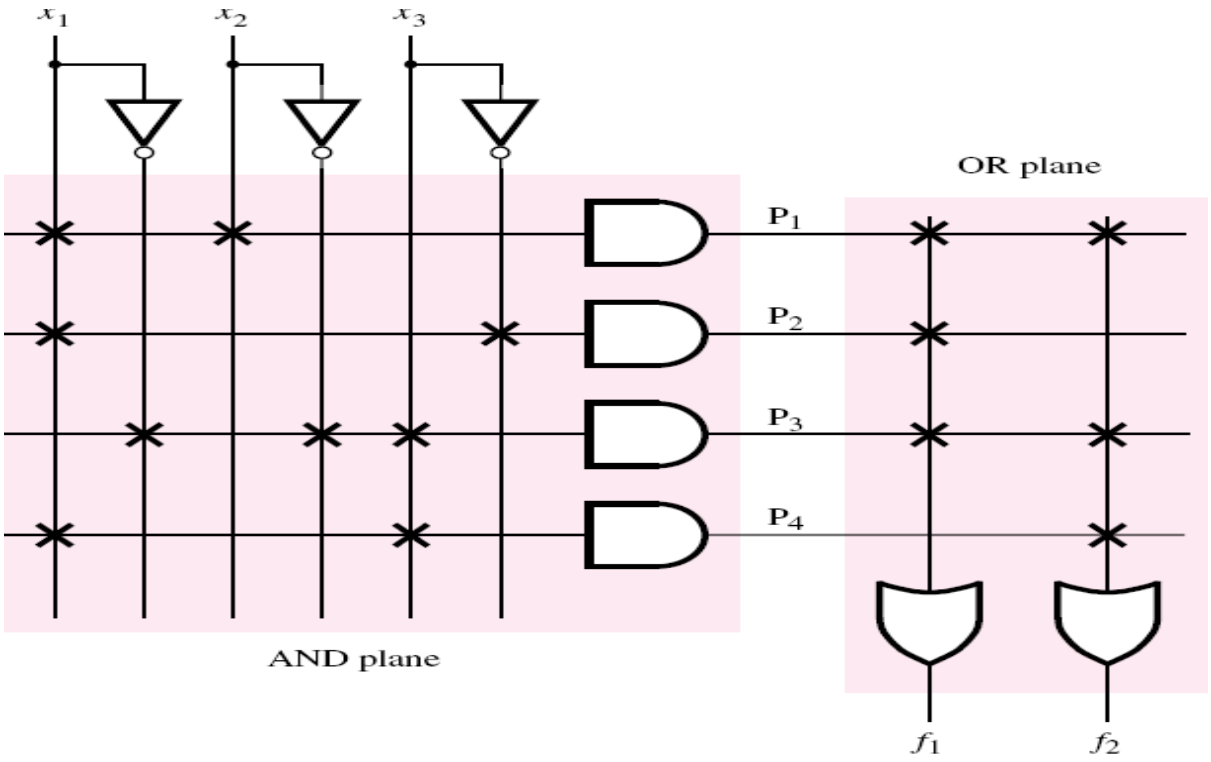
Ex:- Implement $f_1 = x_1x_2\overline{x_3} + \overline{x_1}x_2x_3$ & $f_2 = \overline{x_1x_2} + x_1x_2x_3$



Programmable Logic Array

It has programmable AND plane and programmable OR plane. It is more complex than PAL but more flexible. Uses AND's array instead of decoder to produce product terms of inputs. It has programmable connections before AND's, between AND's and OR's, after OR's. More flexible than ROM but more difficult to program. Logic expressions for content information to be stored in PLA must be obtained, then minimized, and finally programmed into the PLA using a PLA program table. PLA program table specifies product terms and sum terms of information that will be stored in PLA.

- Implement $f_1 = x_1x_2 + x_1\overline{x_3} + \overline{x_1x_2}x_3$ & $f_2 = x_1x_2 + \overline{x_1x_2}x_3 + x_1x_3$

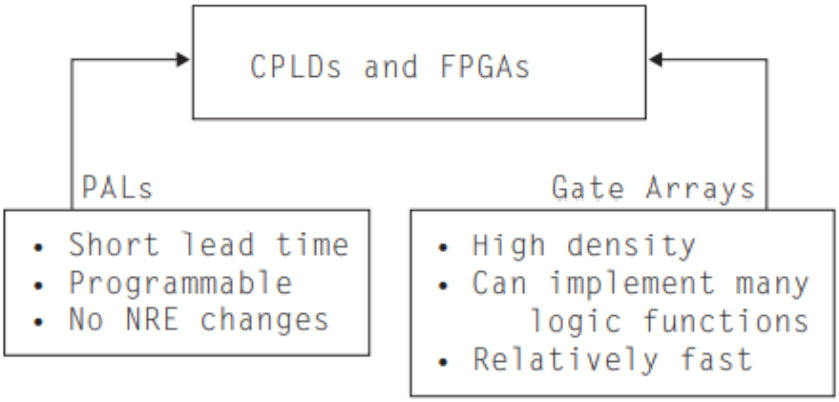


CPLD's & FPGA's

Ideally, hardware designers wanted something that gave them the advantages of an ASIC — circuit density and speed — but with the shorter turnaround time of a programmable device.

The solution came in the form of two new devices — the complex programmable logic device (CPLD) and the field programmable gate array (FPGA).

- CPLDs are as fast as PALs but more complex. FPGAs approach the complexity of gate arrays but are still programmable.
- CPLD architectures and technologies are the same as those for PALs. FPGA architecture is similar to those of gate array ASICs.

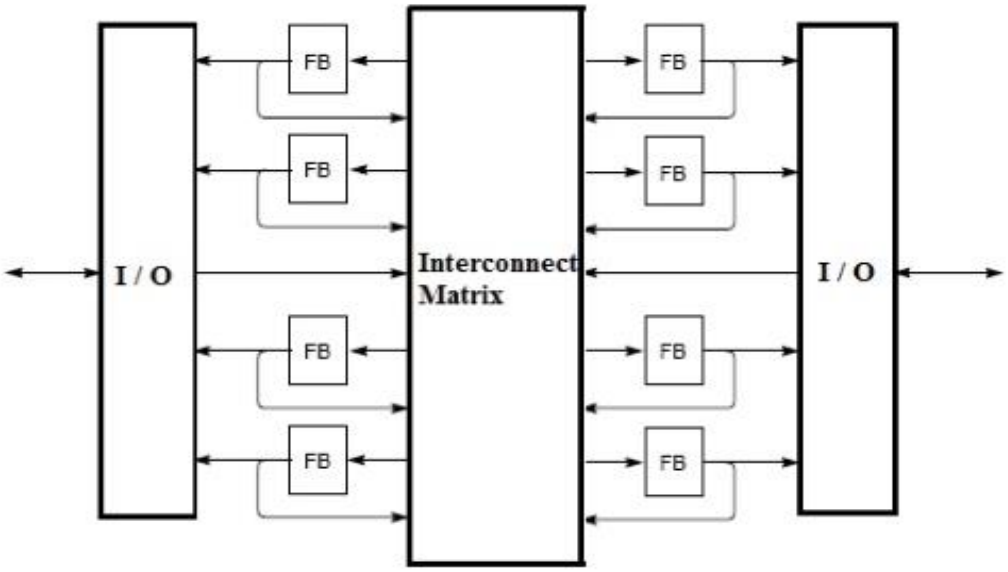


CPLD
(COMPLEX PROGRAMMABLE LOGIC DEVICES)

As the technology around programming devices improved, new devices were developed which combines several PLD's together on a single integrated circuit to form CPLS's. Complex Programmable Logic Devices are exactly the logic devices that are complex and programmable.

There are two main features to understand about CPLDs .One feature is the internal architecture of the device and how this architecture implements various logic functions. The second feature is the semiconductor technology that allows the devices to be programmed and allows various structures in the device to be connected. The concept is to have a few PLD blocks (or) macro cells on a single device with a general-purpose interconnect connected to each other through a cross point switch. A CPLD is a semiconductor device containing programmable blocks called macro cells , which contains the logic implementation of expressions.

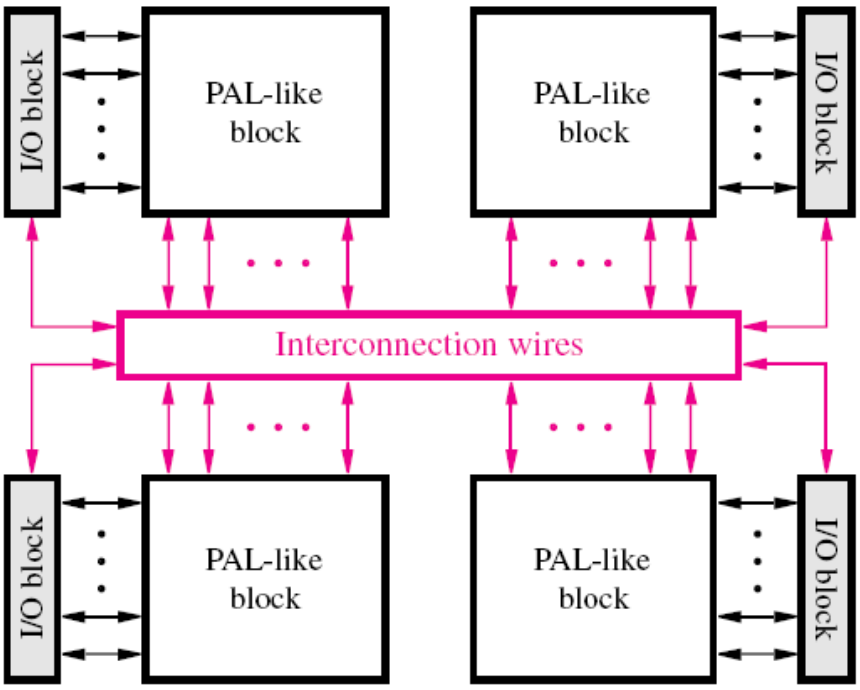
The architecture of CPLD is shown below:



CPLDs are designed to appear just like a large number of PALs in a single chip, connected to each other through a Crosspoint switch.

It consist of function blocks, input/output blocks, and an interconnect matrix. It consists of several blocks each of which is a PLD, which are connected together. I/O's of each of the PLD blocks are connected by a global interconnect array (or) matrix. Each logic block contains 4 to 16 macro cells depending on vendor and architecture. A macro cell on modern CPLD's contain SOP logic function. So a combinational function supports 4 to 16 product terms with wide Fan-In. So, a microcell will have many inputs but the complexity of logic function is limited. CPLD has less flexible internal architecture.

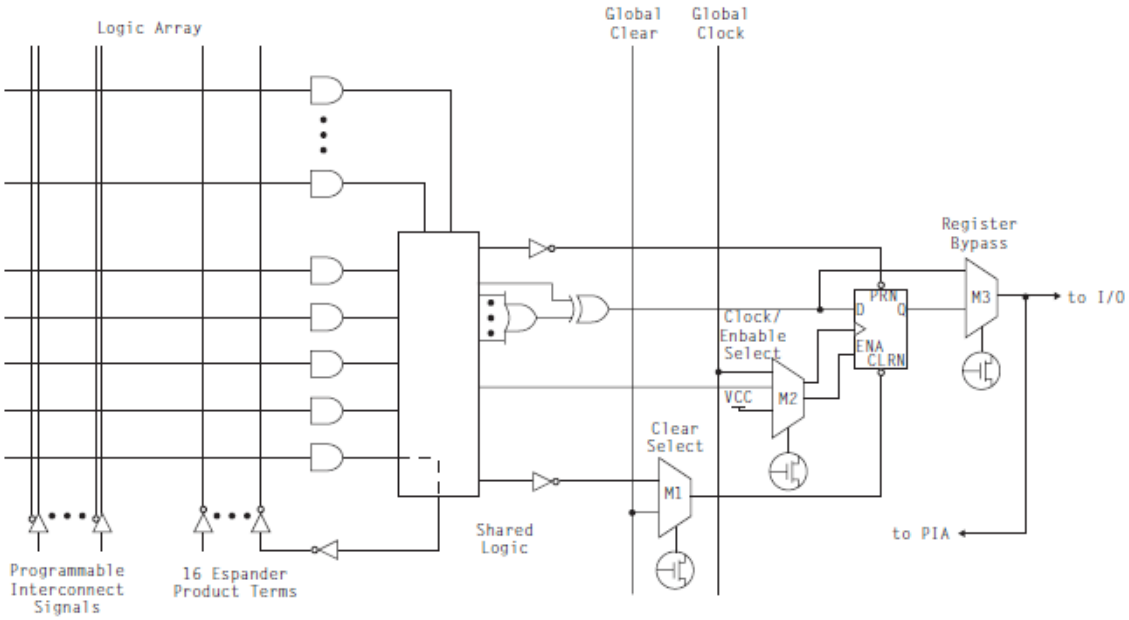
A complex programmable logic device comprises of a group of programmable FB's (functional blocks). The inputs and outputs of these functional blocks are connected together by a GIM (global interconnection matrix). This interconnection matrix is reconfigurable, so that we can modify the contacts between the functional blocks. There will be some input and output blocks that let us to unite CPLD to external world.



Structure of a complex programmable logic device (CPLD).

FUNCTIONAL BLOCKS

Internal structure of Functional Block



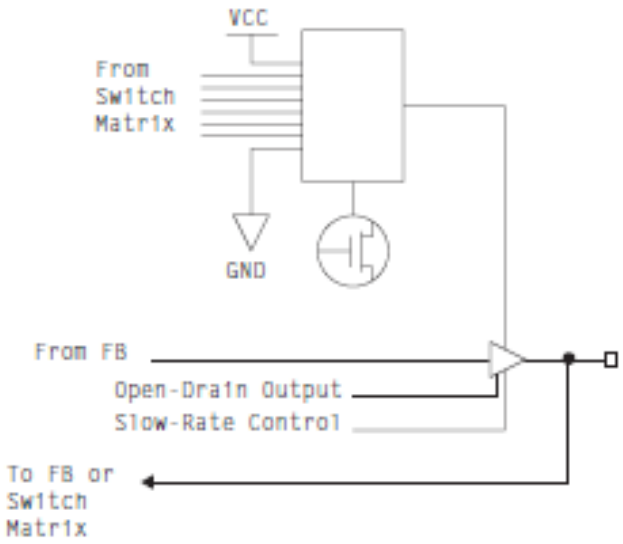
The functional block is similar to the PAL architecture with its wide AND plane and fixed number of OR gates. The AND plane is shown by the crossing wires on the left. The AND plane can accept inputs from the I/O blocks, other function blocks, or feedback from the same function block.

Programming elements at each intersection in the AND plane allow perpendicular traces to be connected or left open, creating "product terms," which are multiple signals ANDed together like in a PAL. The product terms are then ORed together and sent straight out of the block, or through a clocked flip-flop. There are also multiplexers in the Functional Block. Each mux has an FET transistor beneath it, representing a programmable Interconnect element attached to the select line. The mux can be programmed to output one of the inputs.

M1 is the "Clear Select" because it selects the signal that is used to clear the flip-flop. The M2 mux is labeled "Clock/Enable Select" because its two outputs are programmed to control the clock and clock enable input to the flip-flop. The M3 mux is labeled "Register Bypass" because it is programmed to determine whether the output of the functional block is a registered signal (i.e., is the output of a flip-flop) or a combinatorial signal (i.e., is the output of combinatorial logic).

Each functional block would have many OR gates, logic gates, muxes, and flip-flops. The function blocks are designed to be similar to existing PAL architectures, so that the designer can use familiar tools to design them. They may even be able to fit older PAL designs into the CPLD without changing the design.

I/O Block:-



CPLD input/output block

The I/O block is used to drive signals to the pins of the CPLD device at the appropriate voltage levels (e.g., TTL, CMOS, ECL, PECL, or LVDS). The I/O block typically allows each I/O pin to be individually configured for input, output, or bi-directional operation.

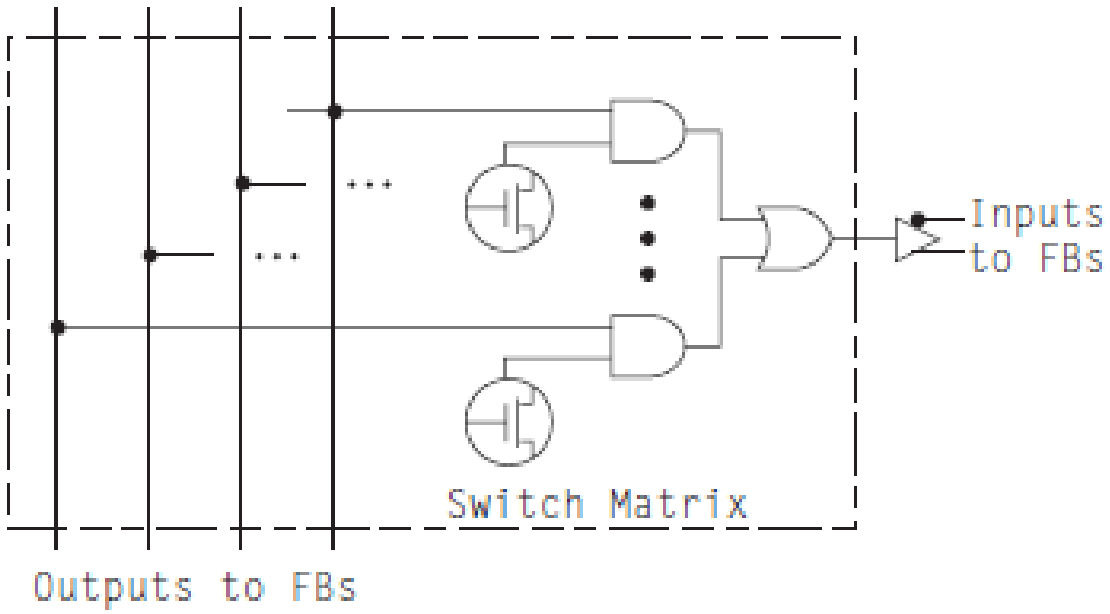
The I/O pins have a tri-state output buffer that can be controlled by global output enable signals or directly connected to ground or VCC. Each output pin can also be configured to be open drain. The outputs can often be programmed to drive different voltage levels, enabling the CPLD to be interfaced to many different devices.

The input signal from the I/O block goes into the switch matrix in order to be routed to the appropriate functional block.

In some architectures, particular inputs have direct paths to particular functional blocks in order to lower the delay on the input, reducing the signal setup time. In most architectures, specific pins of the device connect to specific I/O blocks that can drive global signals like reset and clock. This means that only certain pins of the device can be used to drive these global signals.

Interconnect

CPLD switch matrix



The CPLD interconnect is a very large programmable switch matrix that allows signals from all parts of the device to go to all other parts of the device. The switch matrix takes the outputs of the functional blocks and is programmed

to send those outputs to functional blocks. This way, the designer can route any output signal to any destination. The advantage of the CPLD switch matrix routing scheme is that delays through the chip are deterministic. Designers can determine the delay for any signal by computing the delay through functional blocks, I/O blocks, and the switch matrix. All of these delays are fixed, and delays due to routing the signal along the metal traces are negligible. If the logic for a particular function is complex, it may require several functional blocks, and thus several passes through the switch matrix, to implement. Designers can very easily calculate delays from input pins to output pins of a CPLD by using a few worst-case timing numbers supplied by the CPLD vendor. This contrasts greatly with FPGAs, which have very unpredictable and design-dependent timing due to their routing mechanism.

CPLD Technology and Programmable Elements

Different manufacturers use different technologies to implement the programmable elements of a CPLD. The common technologies are EPROM, EEPROM, and Flash EPROM. The technologies that were used for the simplest programmable devices, PROMs. In functional blocks and I/O blocks, single bits are programmed to turn specific functions on and off.

In the switch matrix, single bits are programmed to control connections between signals using a multiplexer.

When PROM technology is used for these devices, they can be programmed only once. More commonly these days, manufacturers use EPROM, EEPROM, or Flash EPROM, allowing the devices to be erased and reprogrammed. Erasable technology can also allow in-system programmability of the device.

For CPLDs with this capability, a serial interface on the chip is used to send new programming data into the chip after it is soldered into a PC board and while the system is operating. This serial interface is the standard 4-pin Joint Test Action Group (JTAG) interface.

CPLD Design Tools

There are two important design tools available in the market

1.Verilog HDL

2.VHDL(V stands for VHSIC-Very High Speed Integrated Circuit)

- HDL stands for Hardware description Language.

Advantages:-

1.Easy to design.

2.Operation speed is fast.

3. Cost is Low

4. In-system reprogram ability

5. High number of I/O s

6. Flexibility in changing p in & out.

Dis-Advantages:-

- Power consumption is High
- Density of gates Low to medium i.e., up to 10,000 gates.

Application:-

- CPLD can be used for digital designs which perform boot loaders function.
- CPLD is used for loading the configuration data of a field programmable gate array from non-volatile memory.
- Generally, these are used in small design applications like address decoding.
- Complex programmable logic devices are ideal for high performance, critical control applications

FPGA
(FIELD PROGRAMMABLE GATE ARRAY)

FPGA is acronym as Field Programmable Gate Array is an integrated circuits designed to be configured by a customer or designed after manufacture – hence "field programmable". The FPGA configuration is generally specified using a hardware description language to specify how chip will work. FPGA arrived in 1984 as an alternate to programmable logic device & ASICs. FPGA offers all of the features needed to implement most complex designs & supports implementation of relative large logic circuits.

Field Programmable Gate Arrays are given this name because they are structured like a gate array ASIC. Like an ASIC, the FPGA consists of a regular array of logic, an architecture that lends itself to very complex designs.

There are two types of FPGA's

1.SRAM based FPGA's.

2.Antifuse technology-based FPGA's.

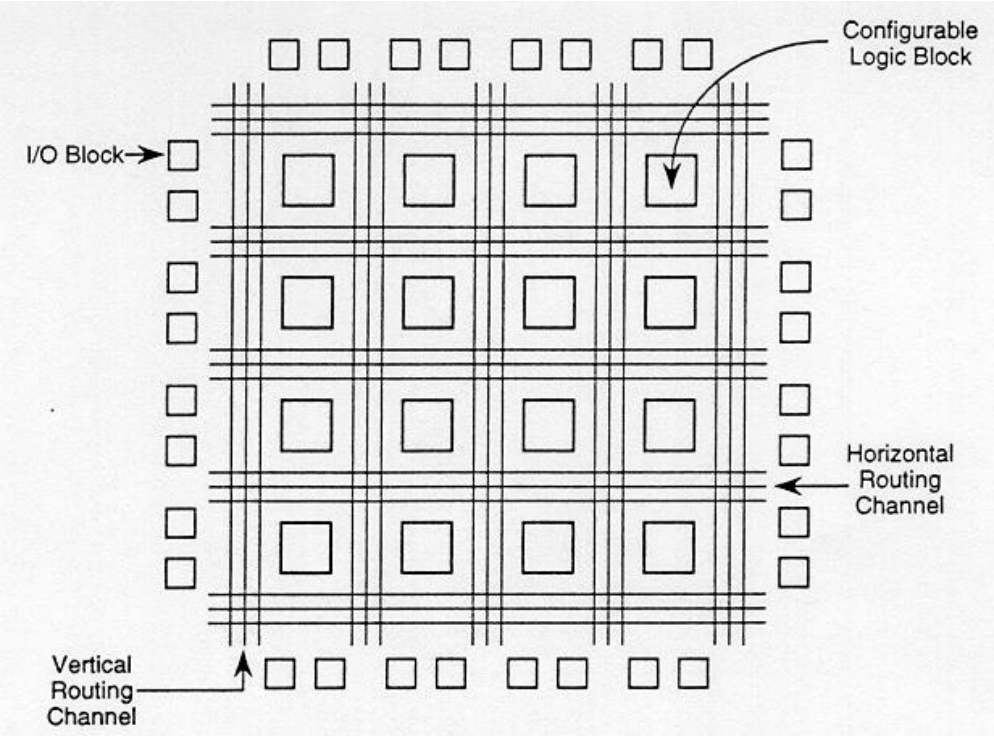
SRAM based FPGAs:-

which is used by the Xilinx and Altera based FPGA's.

Anti-fuse technology based:-

which is used by the Actel logic based technology.

FPGA Architectures:-



Each FPGA vendor has its own FPGA architecture. The architecture consists of configurable logic blocks, configurable I/O blocks, and programmable interconnect to route signals between the logic blocks and I/O blocks. Also, there is clock circuitry for driving the clock signals to each flip-flop in each logic block. Additional logic resources such as ALUs, memory, and decoders may also be available.

The two most common types of programmable elements for an FPGA are static RAM and antifuses. Antifuse technology is a cousin to the programmable fuses in EPROMs.

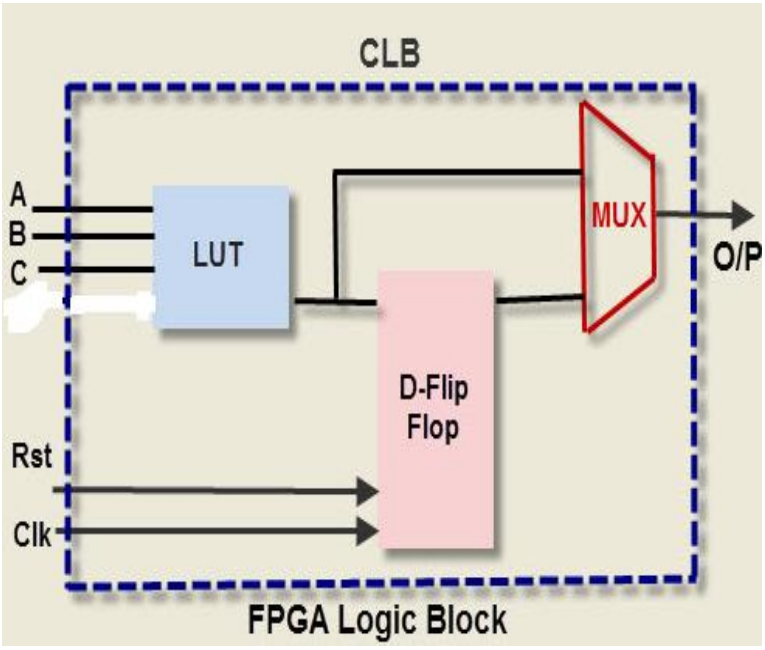
The basic elements of an Field Programmable gate array are

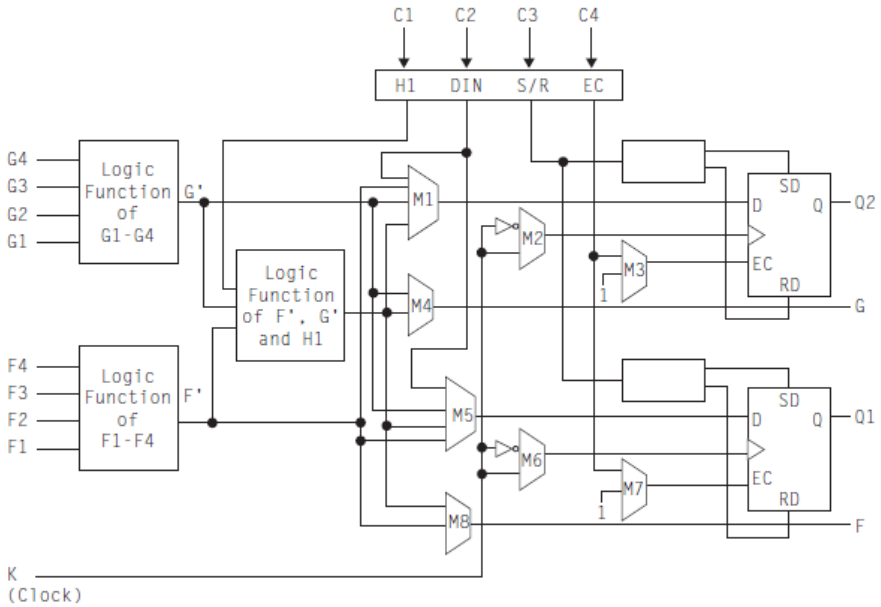
1.Configurable Logic Blocks(CLBs)

2.Configurable I/P & O/P blocks (IOBs)

3. Two layer metal network of vertical and horizontal lines for interconnecting the CLB's.

Configurable Logic Blocks





Configurable logic blocks (CLBs) contain the programmable logic for the FPGA. A typical CLB, containing RAM for creating arbitrary combinatorial logic functions. It also contains flip-flops for clocked storage elements and multiplexers in order to route the logic within the block and to route the logic to and from external resources. These muxes also allow polarity selection, reset input, and clear input selection.

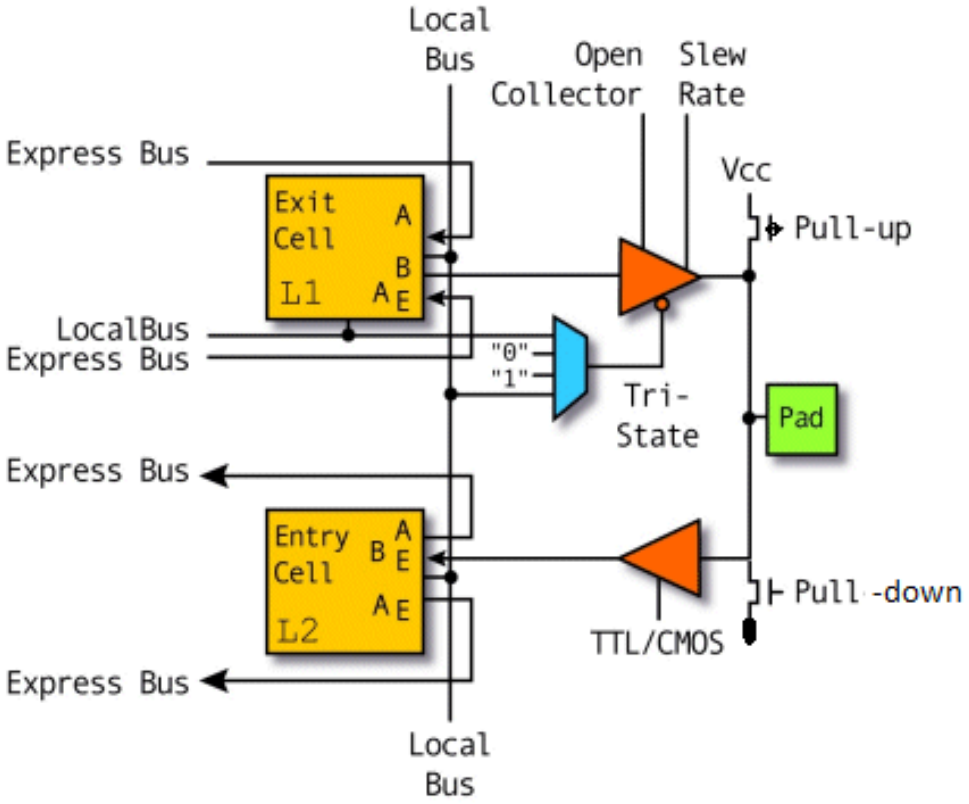
On the left of the CLB are two 4-input memories, also known as 4-input lookup tables or 4-LUTs. 4-input memories can produce any possible 4-input Boolean equation. Four signals labeled C1 through C4 enter at the top of the CLB. These are inputs from other CLBs or I/O blocks on the chip, allowing outputs from other CLBs to be input to this particular CLB.

These interconnect inputs allow designers to partition large logic functions among several CLBs. They also are the basis for connecting CLBs in order to create a large, functioning design. The muxes throughout the CLB are programmed statically. When the FPGA is programmed, the select lines are set high or low and remain in that state. Some muxes allow signal paths through the chip to be programmed.

The clock input to the flip-flops must come only from the global clock signal. The logic outputs do not need to go through the flip-flops. Designers can use a CLB to create simple combinatorial logic. Because of this, multiple CLBs are connected together to implement complex Boolean logic.

This advantage of FPGAs over CPLDs means that designers can implement very complex logic by stringing together several CLBs. The routing delay in an FPGA is a significant amount of the overall delay. So this advantage also results in an overall decrease in the speed of the design.

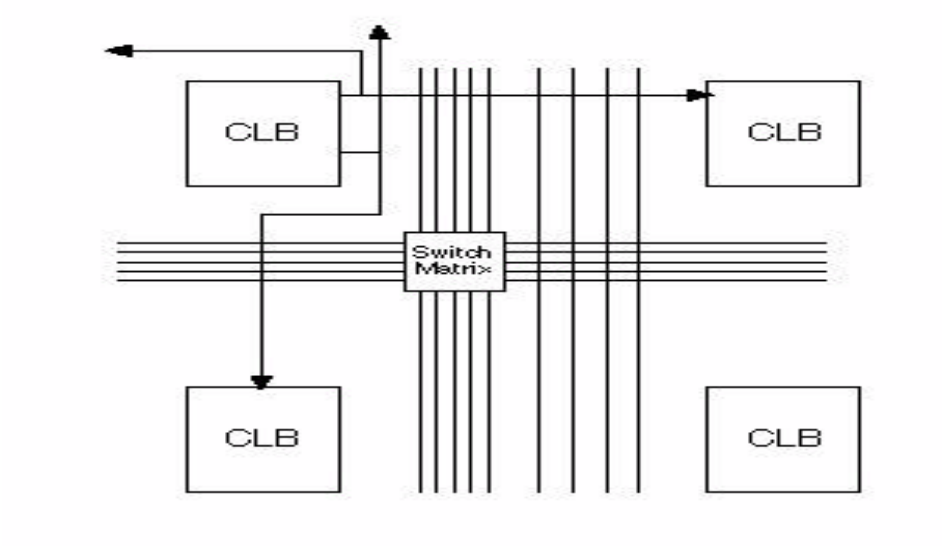
Configurable I/O Blocks



A Configurable input/output (I/O) Block is used to bring signals onto the chip and send them back off again. The output buffer has programmable controls to make the buffer three-state or open collector and to control the slew rate. The input buffer can be programmed for different threshold voltages, typically TTL or CMOS level, in order to interface with TTL or CMOS devices.

There are pull up resistors on the outputs and sometimes pull-down resistors that can be used to terminate signals and buses without requiring discrete resistors external to the chip.

Programmable Interconnect



The interconnect of an FPGA is very different than that of a CPLD, but is rather similar to that of a gate array ASIC.There are long lines that can be used to connect critical CLBs that are physically far from each other on the chip without inducing much delay. These long lines can also be used as buses within the chip.

There are also short lines that are used to connect individual CLBs that are located physically close to each other. Transistors are used to turn on or off connections between different lines.

There are also several programmable switch matrices in the FPGA to connect these long and short lines together in specific, flexible combinations.The routing resources consist of traces that pass by a number of CLBs before reaching switch matrices. These switch matrices allow a signal to be routed from one switch matrix to another to another, eventually connecting CLBs that can be relatively far from each other.

Programming of FPGAs

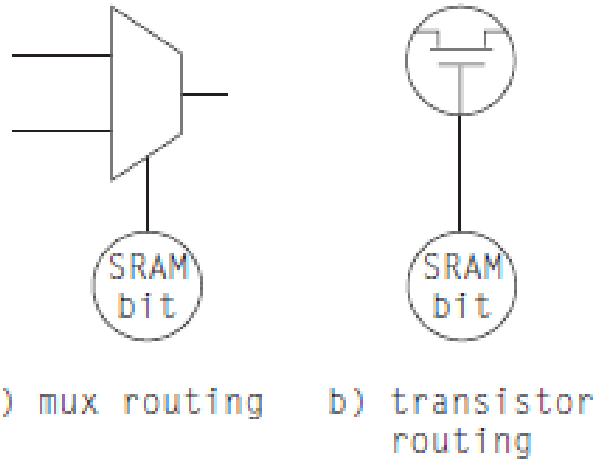
There are 2 methods of programming FPGA.

1) S-Ram Programming.
2) Antifuses.

	SRAM	Antifuse
Volatile	Yes	No
In-system programmable	Yes	No
Speed	Fast	Somewhat faster
Power consumption	Higher	Lower
Density	High	High
IP security	No	Yes
Embedded RAM	Yes	No

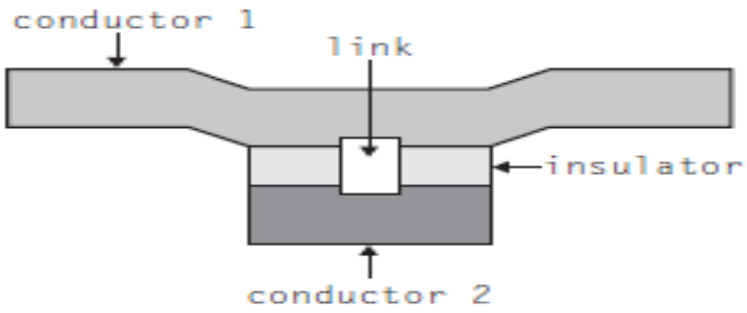
SRAM programming

It involves static RAM bits as the programming elements. These bits can be combined in a single memory and used as a LUT to implement any kind of combinatorial logic. Also, programmers can use individual SRAM bits to control muxes, which select or deselect particular logic within a CLB. For routing, these bits can turn on a transistor that connects two traces in a switch matrix, or they can select the output of a mux that drives an interconnect line.



Anti fuses

A regular fuse normally makes a connection until an excessive amount of current goes through it, generating heat and breaking the connection. With an antifuse, there is a small link between two conductors that are separated by an insulator. When a large voltage is applied across the link, the link melts. As the link melts, the conductor material migrates across the link, creating a conducting path between the two conductors. This process is used to connect traces inside the FPGA.



Advantage:-

1.Low risk and highly flexible

2.High gate density & Suitable for prototyping

3.Reprogrammability of FPGA design. A user can program an FPGA design in a few minutes.

Disadvantage:-

1.Speed is comparatively less.

2.The circuit delay depends on the performance of the design implementation tools.

3.The mapping of the logic design into FPGA Architecture requires sophisticated design implementation tools i.e., CAD.

Comparison Between FPGA and CPLD

S.No	FPGA	CPLD
1	It is more expensive	It is less expensive
2	The Architecture is Gate Array Format	The Architecture is PALs Format
3	Power consumption is medium	Power Consumption is High
4	Speed Performance is application dependent	Speed of the performance is speed
5	FPGA is a volatile memory	CPLD is a Non-Volatile Memory
6	Density of Gates is from 10,000 to 1 Million	Density of Gates is up to 10,000
7	The internal Architecture of FPGA is LUT	The internal Architecture of CPLD is a form of Logic function with Sea of Gates