VLSI DESIGN

ACKNOWLEDGEMENT

The success and final outcome of this Project required a lot of guidance, assistance from many Professors and I am extremely privileged to have got this all along the completion of my project. I owe my deep gratitude to my training company Vyorius, which took keen interest in myproject work and guided all along, till the completion of my project work by providing all the necessary information for developing a practical approach. I am thankful and fortunate enough to get constant encouragement, support and guidance from all Professors of ECE Department. I would like to express my gratitude towards my parents and seniors for their kind co-operation and encouragement which helped me in completion of the Project.

VYORIUS

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY

CERTIFICATE

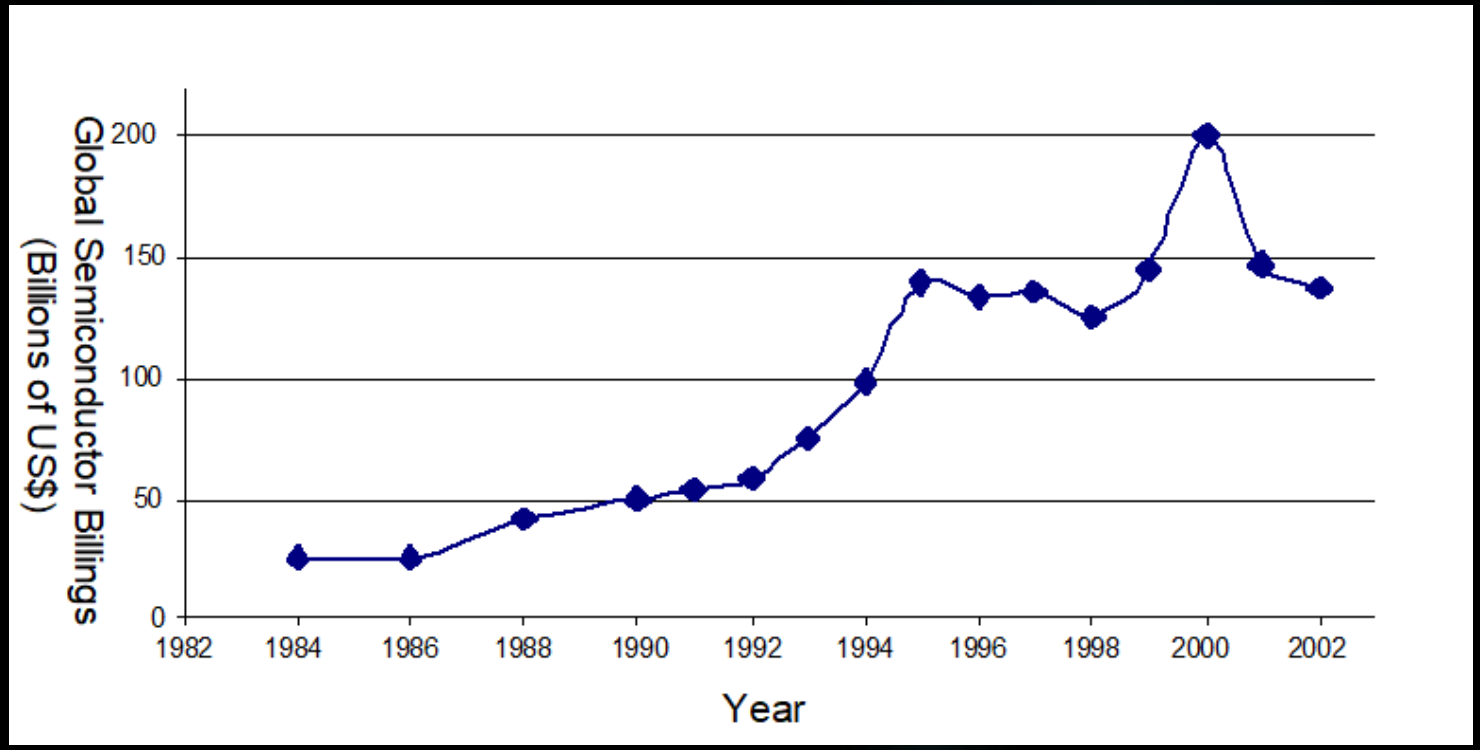
VYORIUS

Introduction

- N Integrated circuits: many transistors on one chip.
- n*Very Large Scale Integration* (VLSI)
- n*Complementary Metal Oxide Semiconductor* (CMOS)
- nFast, cheap, “low-power” transistors circuits

SALES

- 10^{18} transistors manufactured in 2003
- 100 million for every human on the planet

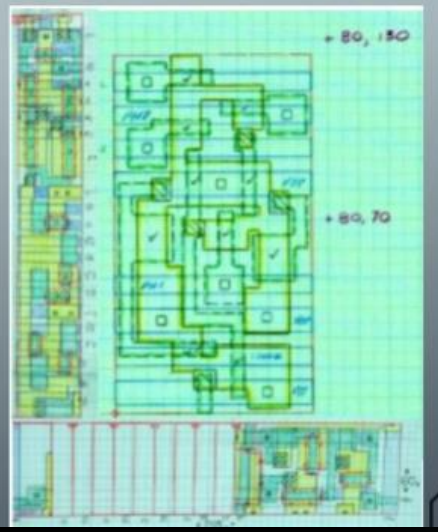


VYORIUS

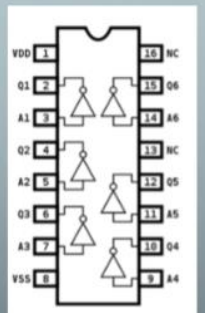
First Transistor

CMOS (Intel 1101 SRAM)

IC Design

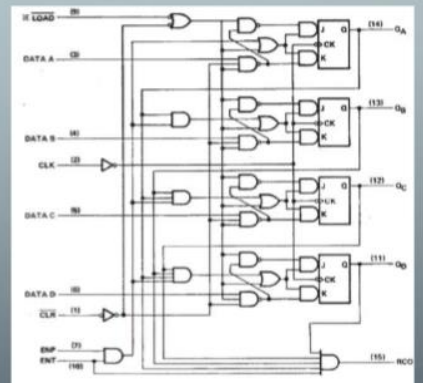


SSI
~10 gates



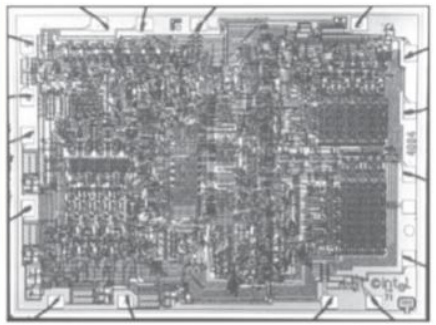
4049 Inverter

MSI
~1000 gates



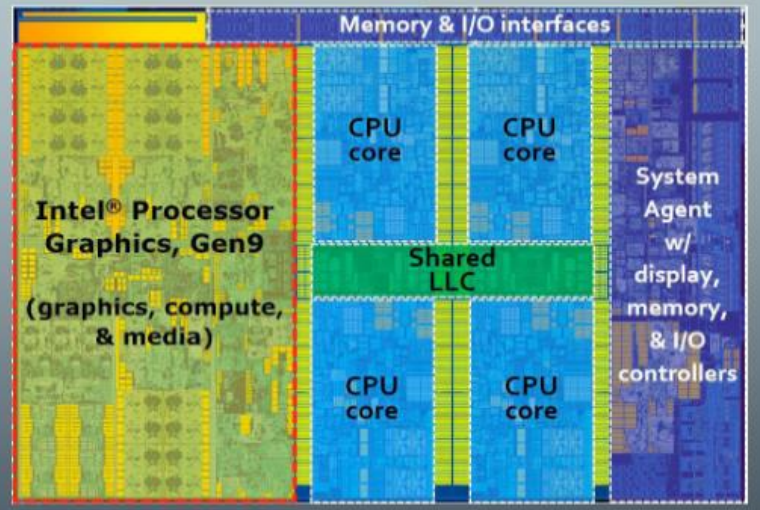
74161 Counter

LSI
~10,000 gates



Intel 4004 4-bit Microprocessor

~million gates



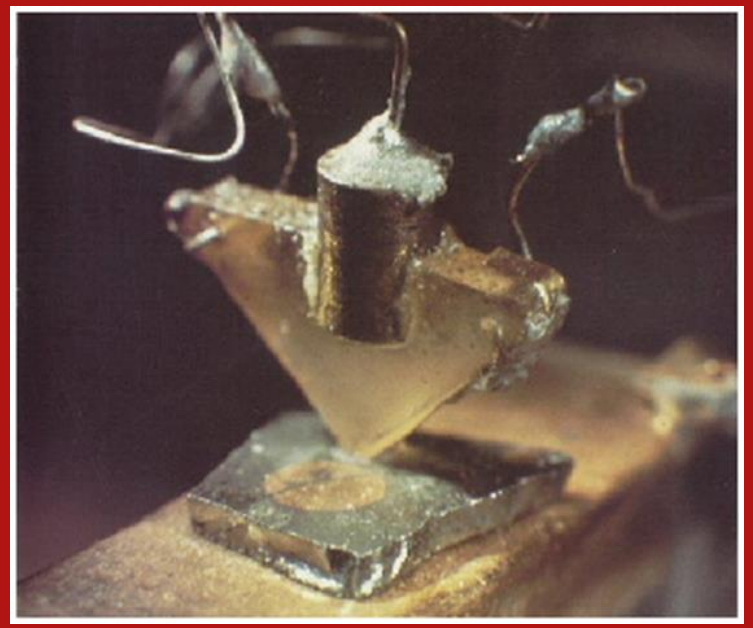
Intel i5

MOSFET scaling (process nodes)

10 µm – 1971
6 µm – 1974
3 µm – 1977
1.5 µm – 1981
1 µm – 1984
800 nm – 1987
600 nm – 1990
350 nm – 1993
250 nm – 1996
180 nm – 1999
130 nm – 2001
90 nm – 2003
65 nm – 2005
45 nm – 2007
32 nm – 2009
22 nm – 2012
14 nm – 2014
10 nm – 2016
7 nm – 2018
5 nm – 2020

Future
3 nm ~ 2022
2 nm ~ 2023

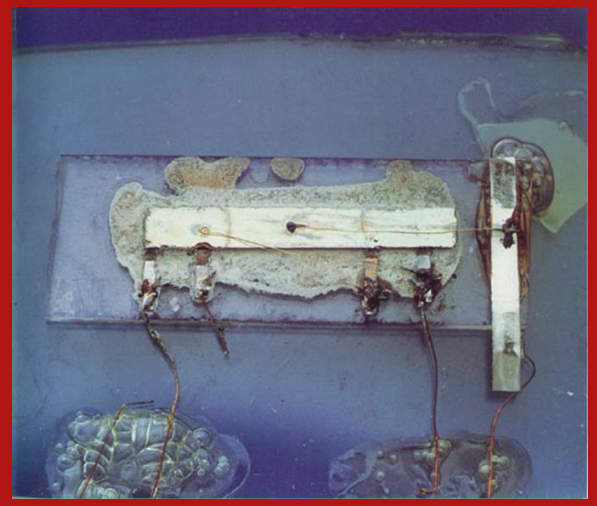
A Brief History Invention of the Transistor



- Vacuum tubes ruled in first half of 20th century Large, expensive, power-hungry, unreliable
- 1947: first point contact transistor (3 terminal devices)
- Shockley, Bardeen and Brattain at Bell Labs

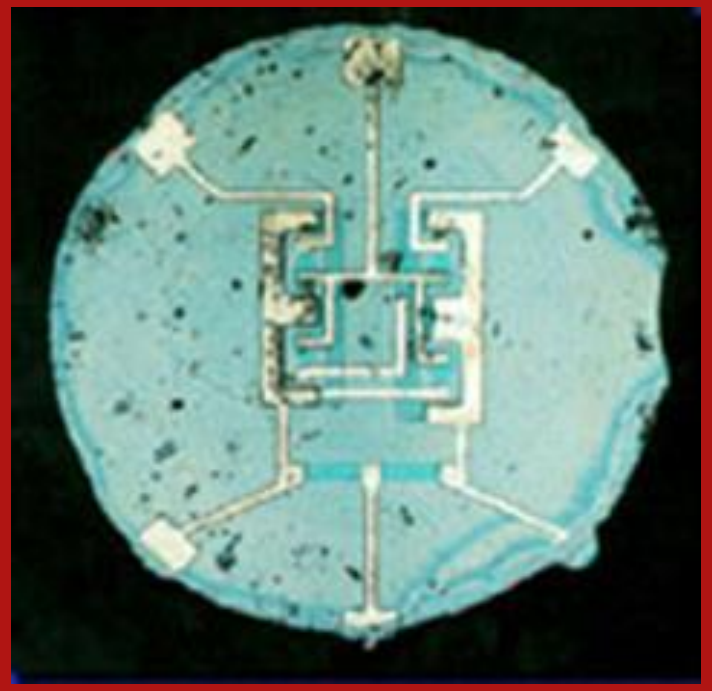
VYORIUS

A Brief History Invention of the Transistor



- 1958: First integrated circuit
- Flip-flop using two transistors
- Built by Jack Kilby (Nobel Laureate) at Texas Instruments
- Robert Noyce (Fairchild) is also considered as a co-inventor

A Brief History Invention of the Transistor



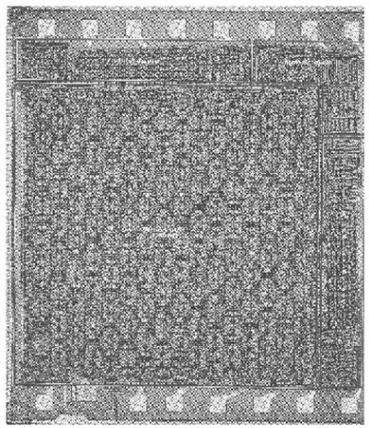
- First Planer IC built in 1961
- 2003
- Intel Pentium 4 microprocessor (55 million transistors)
- 512 Mbit DRAM (> 0.5 billion transistors)
- 53% compound annual growth rate over 45 years
- No other technology has grown so fast so long
- Driven by miniaturization of transistors
- Smaller is cheaper, faster, lower in power!
- Revolutionary effects on society

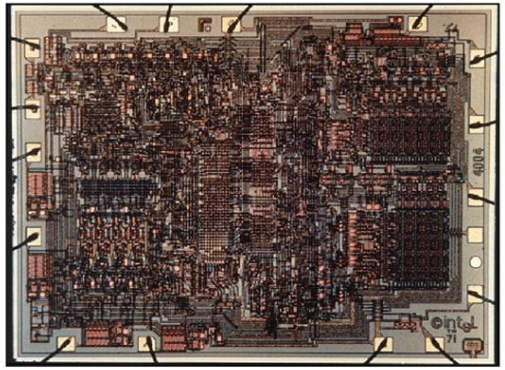
MOS Integrated Circuits

- 1970's processes usually had only nMOS transistors
- Inexpensive, but consume power while idle
- 1980s-present: CMOS processes for low idle power

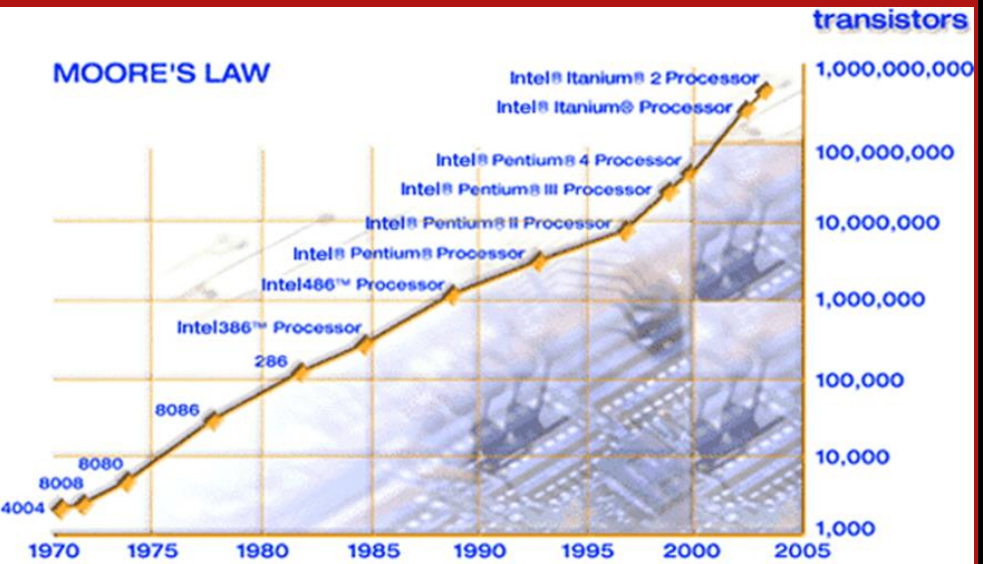
MOS Integrated Circuits

Intel 1101 256-bit SRAM Intel 4004 4-bit μProc





Moore's Law

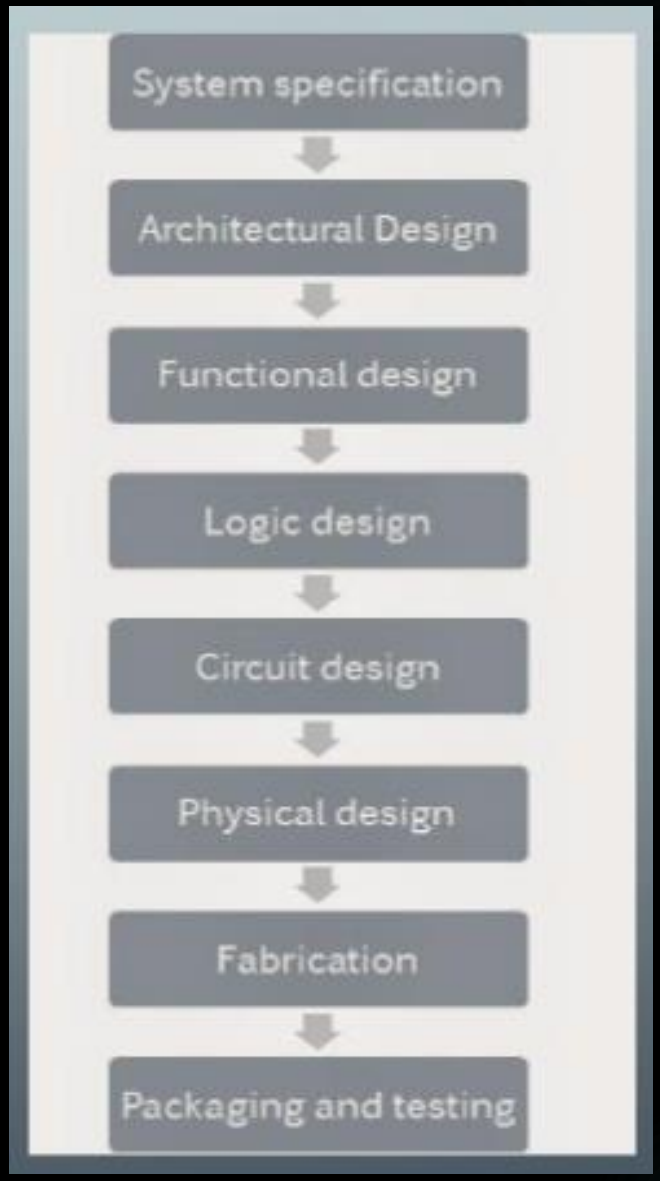


- 1965: Gordon Moore plotted transistor on each chip
- Fit straight line on semilog scale
- Transistor counts have doubled every 26 months

Integration Levels

SSI: 10 gates
MSI: 1000 gates
LSI: 10,000 gates
VLSI: > 10k gates

VLSI Design Flow



VLSI Design Flow

- System Specification

It is the high level representation of the system

Factors to be considered

Performance

Functionality

Physical Dimensions

It is a compromise between market requirements, Technology and economic viability.

End Results: size, speed, power and functionality.

- Architectural Design

Architecture (RISC vs CISC)

No. of ALUs

Floating point units

No. of and structure of pipelines

Size of cache

End Result: Micro-Architectural Specification (MAS)

MAS –Textual description

VLSI Design Flow

- Behavioural or Functional Design

Identification of main functional units and their interconnects

Estimation of area, power and other parameters

Specification of input, output and timing of each unit without specifying its internal structure.

End Result: timing diagram or other relationships, reduces complexity, quick emulation and fast debugging of the system.

- Logic Design

Derivation and Testing of

Control flow

Word widths

Register allocation

Arithmetic operations

Logic operations

Register Transfer Level (RTL) expressed in Hardware

Description Languages (HDL) like Verilog/VHDL.

The Description can be used for Simulation and

Verification.

They consists of Boolean expression and Timing information

Logic Designs can automated using High Level Synthesis (HLS) Tools.

These tools RTL description from a behavioural description of the design

VLSI Design Flow

- Circuit Design
- Developing a circuit representation based on the logic design.
- Boolean expressions are converted into circuit representation with considerations of speed and power requirements of original design.
- Circuit simulation is carried out to check the correctness and timing of each component
- Contains circuit elements (cells, macros, gates, transistors) and interconnection between them
- The circuit diagram representation is called as Netlist.
- Tools that are used to manually enter such description are called schematic capture tools.
- Netlist can also be create automatically from logic (RTL) description by using logic synthesis tools

VLSI Design Flow

- Physical Design
- Circuit representation (Netlist) are converted into geometric representation known as layout.
- Connections between different components are also expressed as geometric patterns.
- Details of the Layout depend on design rules.
- Limitations of fabrication process
- Electrical properties of the fabrication materials.
- Complex process, hence broken down into sub-steps.
- Verification and Validation checks are performed on the Layout.
- Physical Design can be completely or partially automated by Layout Synthesis Tools.
- Layout Synthesis tools have area and performance penalties.
- Manual layout creation is slow but with better area and performance

VLSI Design Flow

- Fabrication
- Layout data is sent in tape for fabrication, hence the release of data is called Tape Out.
- Layout data is converted into photo-lithographic masks for each layer.
- Masks identifies spaces on the wafer, where certain materials need to be deposited, diffused or even removed.
- The fabrication process consists of several steps involving deposition, and diffusion of various materials on the wafer.
- A large wafer is 20 cm (8 inch) in diameter and can be used to produce hundreds of chips, depending of the size of the chip.
- Before the chip is mass produced, a prototype is made and tested.
- Industry is rapidly moving towards a 30 cm (12 inch) wafer allowing even more chips per wafer leading to lower cost per chip.

VLSI Design Flow

- Fabrication
- Layout data is sent in tape for fabrication, hence the release of data is called Tape Out.
- Layout data is converted into photo-lithographic masks for each layer.
- Masks identifies spaces on the wafer, where certain materials need to be deposited, diffused or even removed.
- The fabrication process consists of several steps involving deposition, and diffusion of various materials on the wafer.
- A large wafer is 20 cm (8 inch) in diameter and can be used to produce hundreds of chips, depending of the size of the chip.
- Before the chip is mass produced, a prototype is made and tested.
- Industry is rapidly moving towards a 30 cm (12 inch) wafer allowing even more chips per wafer leading to lower cost per chip.

VLSI Design Flow

- INTRODUCTION TO DIGITAL WORLD
- Digital Systems
- Number Systems
- Logic Gates
- Combination circuits
- Sequential circuits

VLSI Design Flow

- Digital Systems

1. A Digital system is an interconnection of digital modules.
2. Manipulates discrete elements of information that represented internally in the binary form.
3. Examples: Automated Industrial Machinery Microprocessor and Digital computers Signal Processing
4. Characteristics
5. Discrete Elements
6. Signals
7. Binary Digital called a bit (0 or 1)
8. Analog vs Digital Systems
9. Analog Systems process information that varies continuously.
10. Digital Systems use digital circuits that can process digital signals.

VLSI Design Flow

- Advantages of Digital system over Analog system

1. Ease of programmability
2. Reduction in cost of hardware
3. High speed & Reliability
4. Easy to Design
5. Result can be reproduced easily

- Disadvantages of Digital Systems

1. Use more energy than analog circuits
2. They are often fragile
3. Manipulates discrete elements of information by means of a binary code
4. Quantization errorduring analog signal sampling.

ARDUINO

- Number Systems
- Number system is a basis for counting varies items
- Modern computers –Binary
- Human Beings –Decimal
- Binary (base 2)
- Octal(base 8)
- Decimal(base 10)
- Hexadecimal(base 16)

LETS GET STARTED!

Decimal	Binary	Octal	Hexadecimal
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

- Decimal Numbers
  - Radix = 10
  - Digits: 0, 1, 2, 3, 4, 5, 6 , 7, 8 and 9
  - Ex: (1234)10, (12.34)10
- Binary Numbers
  - Radix = 2
  - Digits: 0 and 1
  - Ex: (10101101)2,(1100.1010)2
- Octal Numbers
  - Radix = 8
  - Digits: 0, 1, 2, 3, 4, 5, 6 and 7
  - Ex: (137)8 ,(31.32)8
- Hexadecimal Numbers
  - Radix = 16
  - Digits: 0, 1, 2, 3, 4, 5, 6 , 7, 8 and 9A, B, C, D, E and F
  - Ex: (1ABCD)16

VLSI Design Flow

VYORIUS

Number Base Conversion

- Any to Decimal (Positional power)
- Decimal to Any (using LCM)
- Grouping (Binary to Octal or Hexadecimal)

Exercise:

- Convert (10110101001.10101100)2into Decimal, Octal and hexadecimal system.
- Convert (1024.625)10into Binary, Octal and Hexadecimal systems.
- Convert (726.625)8into Decimal, Binary and Hexadecimal Systems.
- Convert (90AB.1C)16into Decimal, Binary and Octal systems.

Binary Arithmetic

Binary Addition

Rules: 0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10, 1 is the carry bit and 0 is the sum bit.

Ex: Add (1010)2and (1111)2in binary form.

Binary Subtraction

Rules 0 –0 = 0

1 –0 = 1

0 –1 = 1, 1 is the borrow bit and 1 is the subtraction bit.

1 –1 = 0

Ex: Subtract (1010)2from (1111)2

Complements

1's Complement

All 1's into 0's and vice versa

2's Complement

Take 1's complement and add 1

Ex: Subtract (1010)2from (1111)2

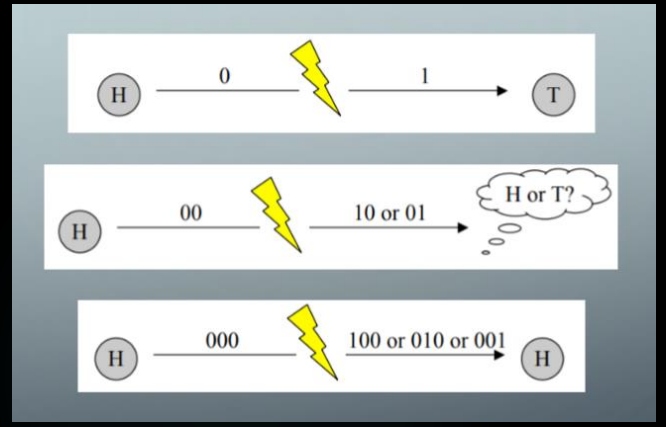
- Even Parity

Binary Code	Even Parity bit	Even Parity Code
000	0	0000
001	1	0011
010	1	0101
011	0	0110
100	1	1001
101	0	1010
110	0	1100
111	1	1111

- Odd Parity

Binary Code	Odd Parity bit	Odd Parity Code
000	1	0001
001	0	0010
010	0	0100
011	1	0111
100	0	1000
101	1	1011
110	1	1101
111	0	1110

- Number Systems
- Transmission of data from one system to another.
- Noise might be introduced
- Error Detection and Correction codes



- Parity Numbers
- Even Parity
- Odd Parity
- Hamming Code
- 7 bit Hamming Code
- Example : (1000)2
- Format:
- b7b6b5b4b3b2b1 = d4d3d2p3d1p2p1Where, p1= d4⊕d2⊕d1p2= d4⊕d3⊕d1p3= d4⊕d3⊕d2
- Check Bits (c3c2c1)Where, c1= b7⊕b5⊕b3⊕b1c2= b7⊕b6⊕b3⊕b2c3= b7⊕b6⊕b5⊕b4

- Further Reading

1. BCD code
2. Excess-3 code
3. Gray code
4. Weighted and Non-weighted

- Digital Signals

1. High Level = +5v
2. Low Level = 0v
3. Digital Logics
4. Positive Logic
5. High = 1/True
6. Low = 0/False
7. Negative Logic
8. High = 0/False
9. Low = 1/True

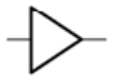
- Basic Identities:

1. A+0 = A, A.0 = 0
2. A+1 = 1, A.1 = 1
3. A+A = A, A.A = A
4. A+$\bar{A}$= 1, A.$\bar{A}$= 0Logic Design

- Properties:

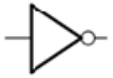
1. Commutative: A+B = B+A, A.B = B.A
2. Associative: A+(B+C) = (A+B)+C, A.(B.C) = (A.B).C
3. Distributive: A.(B+C)=A.B+B.C
4. Complementary property: A.$\bar{A}$= 0, A+$\bar{A}$= 1
5. Idempotency property: A+A=A, A.A=A
6. Absorption property: A+A.B = A, A.(A+B) = A

YES



INPUT	OUTPUT
A	
0	0
1	1

NOT



INPUT	OUTPUT
A	
0	1
1	0

AND

INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

OR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

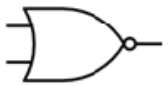
XOR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	0

NAND

INPUT		OUTPUT
A	B	
0	0	1
1	0	1
0	1	1
1	1	0

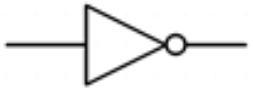
NOR



INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	0

XNOR

INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

Logic Gate	Symbol	Description	Boolean
AND		Output is at logic 1 when, and only when all its inputs are at logic 1,otherwise the output is at logic 0.	$X = A \cdot B$
OR		Output is at logic 1 when one or more are at logic 1.If all inputs are at logic 0,output is at logic 0.	$X = A+B$
NAND		Output is at logic 0 when,and only when all its inputs are at logic 1,otherwise the output is at logic 1	$X = \overline{A \cdot B}$
NOR		Output is at logic 0 when one or more of its inputs are at logic 1.If all the inputs are at logic 0,the output is at logic 1.	$X = \overline{A+B}$
XOR		Output is at logic 1 when one and Only one of its inputs is at logic 1. Otherwise is it logic 0.	$X = A \oplus B$
XNOR		Output is at logic 0 when one and only one of its inputs is at logic1.Otherwise it is logic 1. Similar to XOR but inverted.	$X = \overline{A \oplus B}$
NOT		Output is at logic 0 when its only input is at logic 1, and at logic 1 when its only input is at logic 0.That's why it is called and INVERTER	$X = \overline{A}$

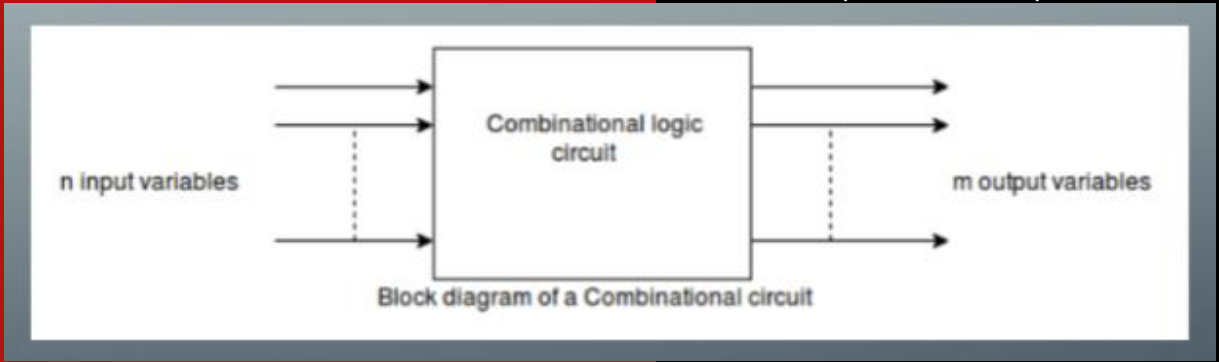
- Universal Gates

1. NAND Gate
2. NOR Gate
3. NOT, AND, OR & XOR Gates using NAND Gate
4. NOT, AND, OR & XOR Gates using NOR Gate

- Further Reading

1. Consensus Theorems
2. Principle of Duality
3. Sum of products (SOP) & Product of Sums (POS)
4. Standard form & Canonical form
5. Karnaugh's Map
6. Quine-Mccluskey Method

- Combination circuits
- A combinational circuit comprises of logic gates whose outputs at any time are determined directly from the present combination of inputs without any regard to previous inputs. Combination circuits

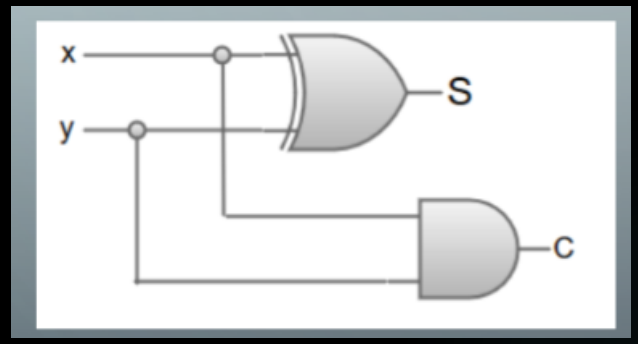


Block diagram of a Combinational circuit

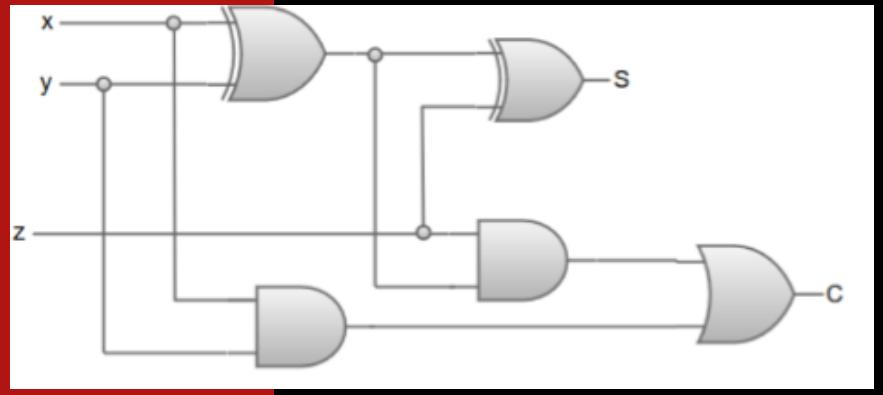
- A simple logic design process involves
- Problem specification
- Truth table derivation
- Derivation of logical expression
- Simplification of logical expression
- Implementation

Inputs		Output	
A	B	S = A⊕B	C= A.B
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

- Adders
- 0 + 0 = 0; 0 + 1 = 1;1 + 0 = 1; 1 + 1 = 10;
- Half-adder

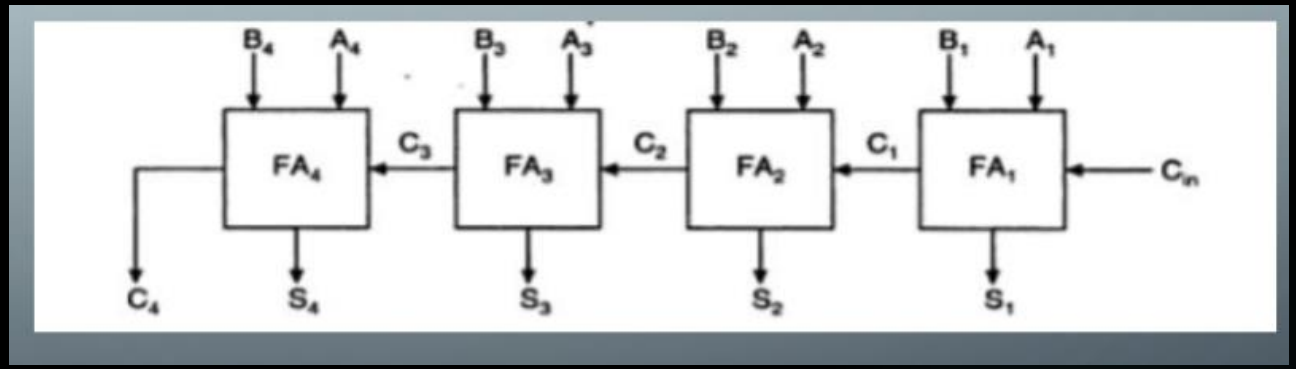


- Full-adder
- Adds two bits and a carry bit

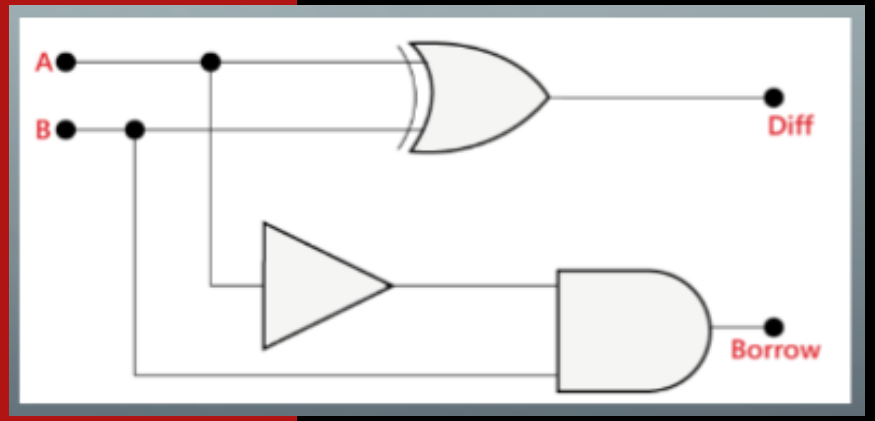


Inputs			Output	
X	Y	Z	S = X⊕Y⊕Z	C= XY+YZ+ZX
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

- Binary Parallel Adder
- Adds two binary numbers in parallel form

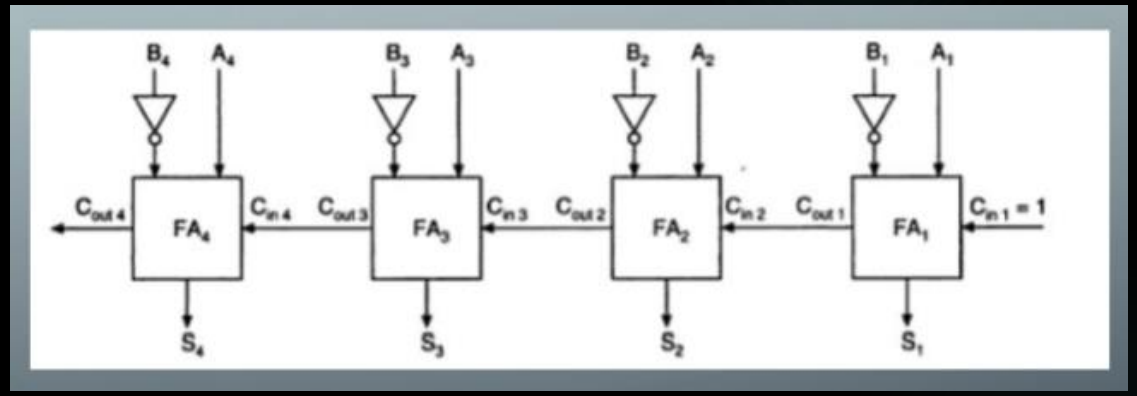


- Subtractor
- Taking complement of the subtrahend and adding it to minuend.
- Half-Subtractor Combination



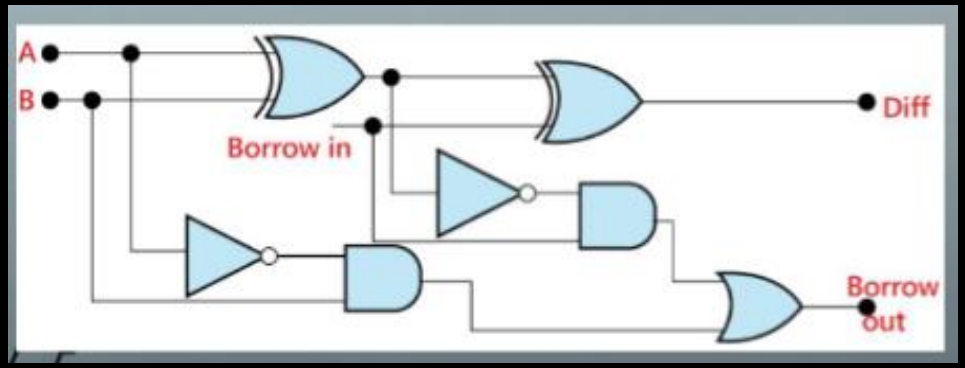
Inputs		Output	
A	B	D = A⊕B	B= $\bar{A}$.B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

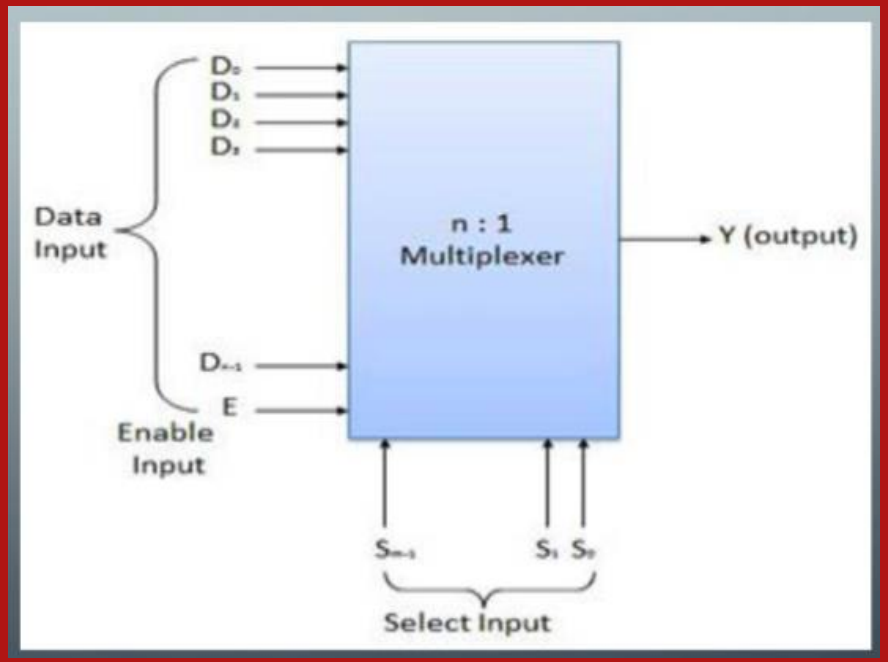
- Binary Parallel subtractor



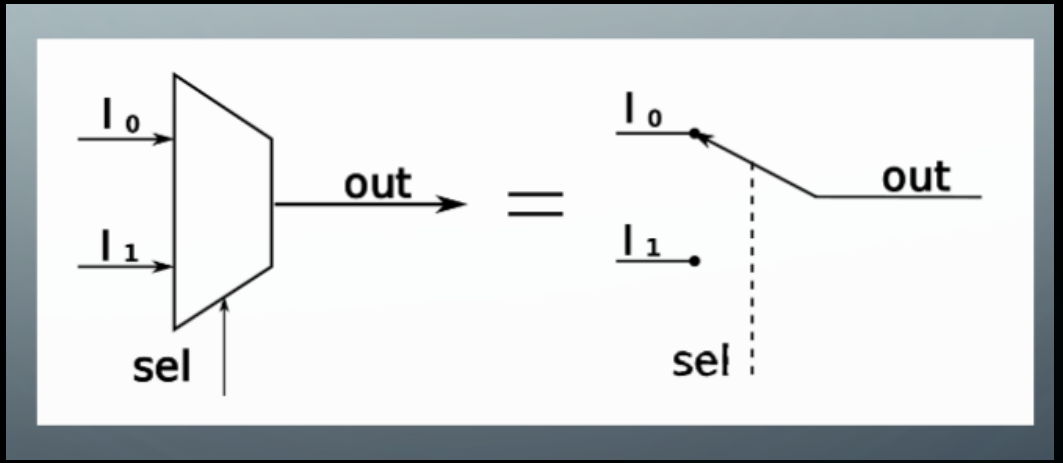
- Full Subtractor

Inputs			Outputs	
A	B	B_i	$D = A \oplus B \oplus B_i$	$B_o = \overline{A}.B + B_i(\overline{A \oplus B})$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

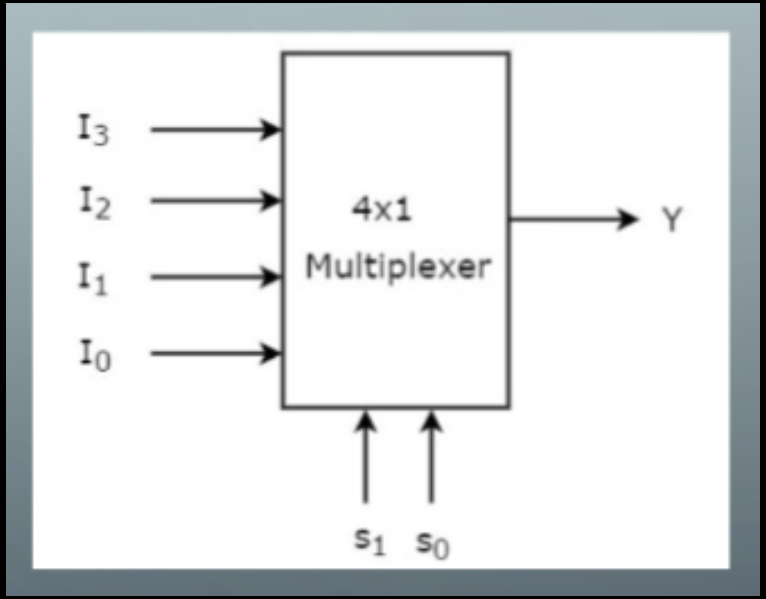




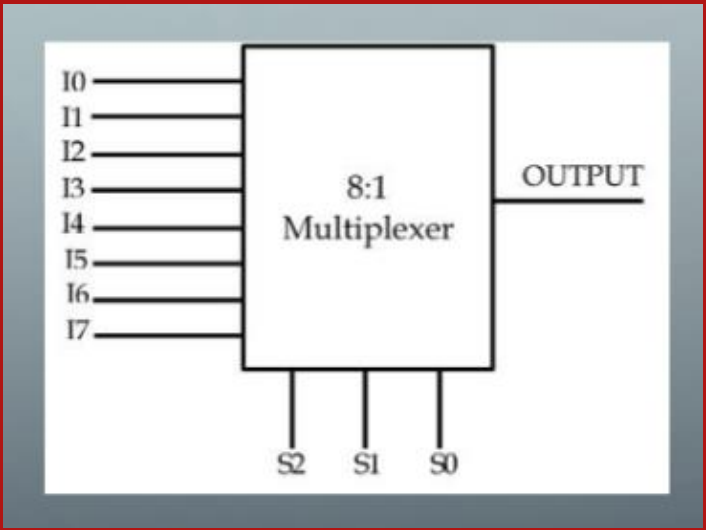
- Multiplexers (MUX)
- 2ndata inputs, ‘n’ selection lines and single output line.



- 4:1 MUX



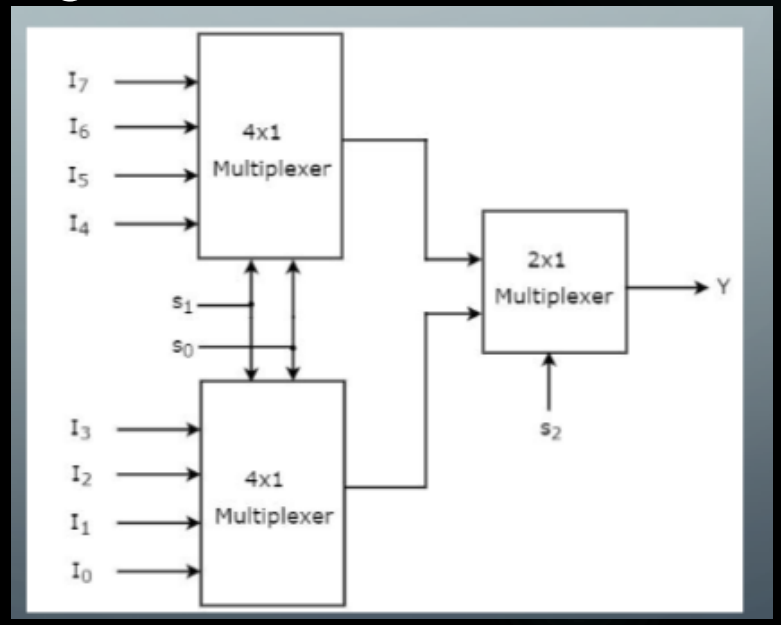
Inputs S_1	Inputs S_0	Output Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

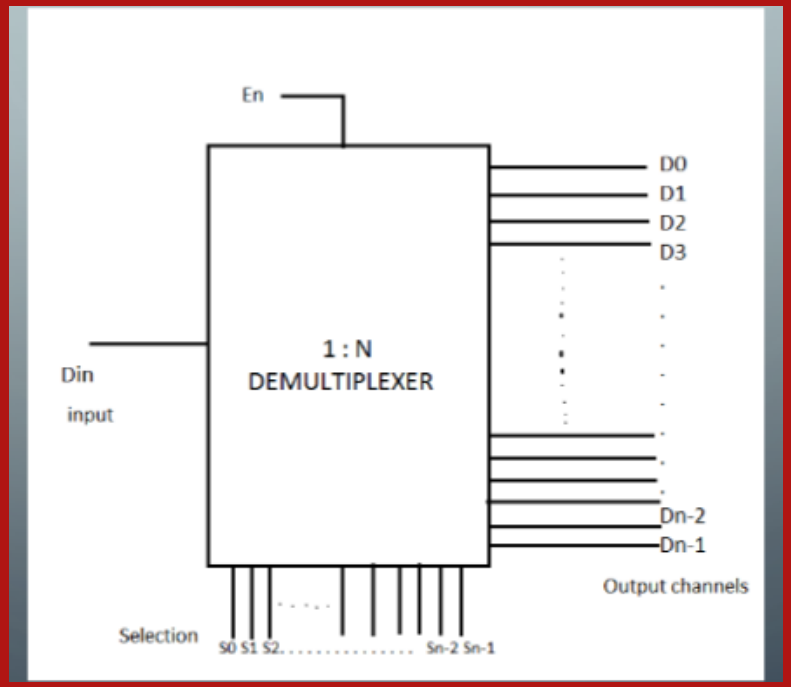


- 8:1 MUX

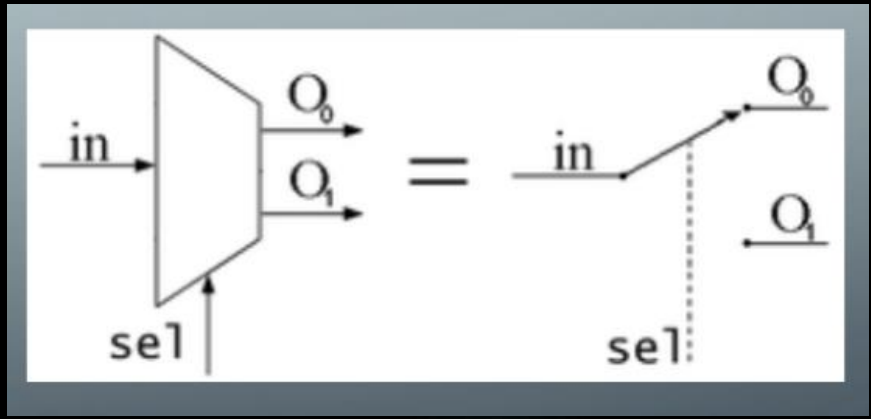
Inputs S_2	Inputs S_1	Inputs S_0	Outputs Y
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

- 8:1 MUX using two 4:1 MUX and one 2:1 MUX



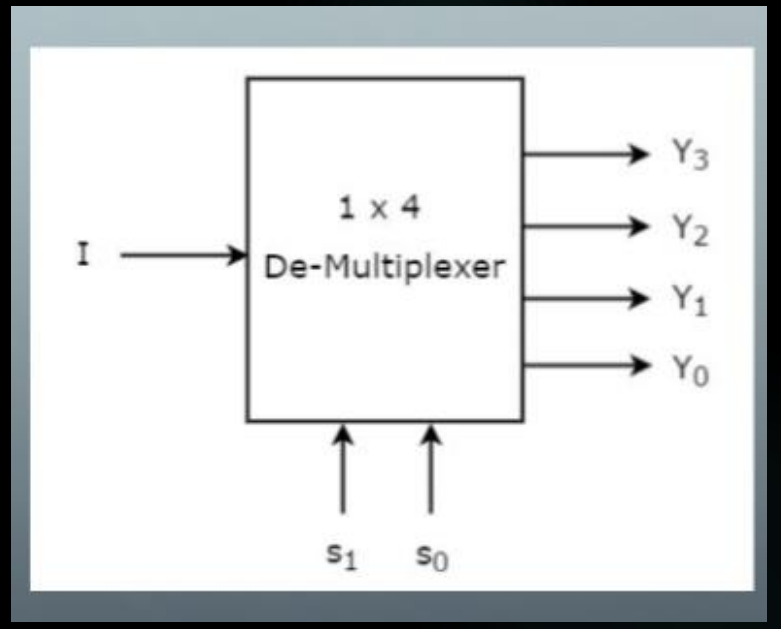


- DeMultiplexers(DeMUX)
- A single input, ‘n’ selection lines and maximum of 2noutputs

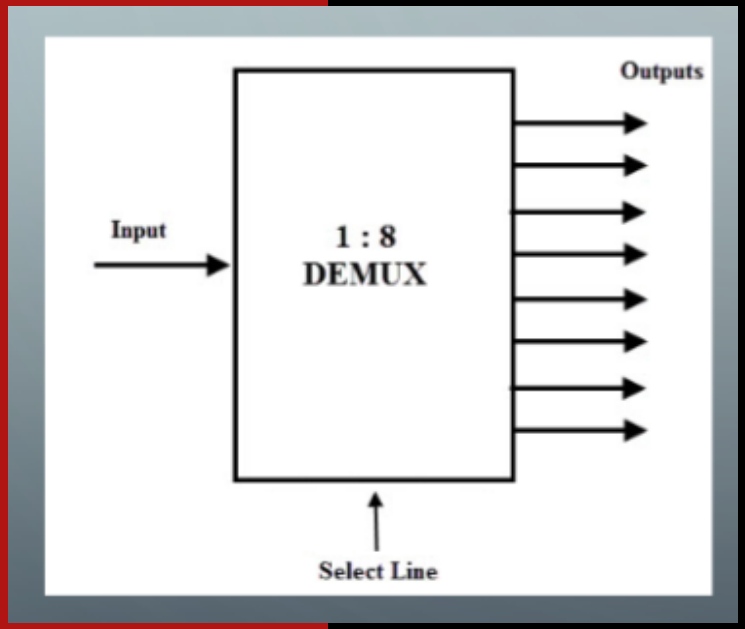


Inputs		Output			
S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	I
0	1	0	0	I	0
1	0	0	I	0	0
1	1	I	0	0	0

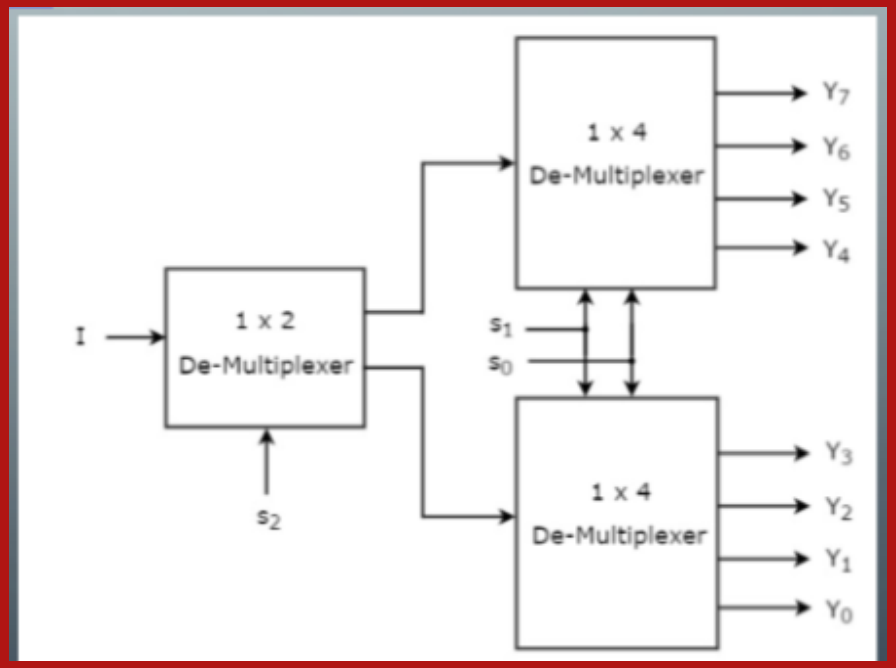
- 1:4 DeMUX



- 1:8 DeMUX

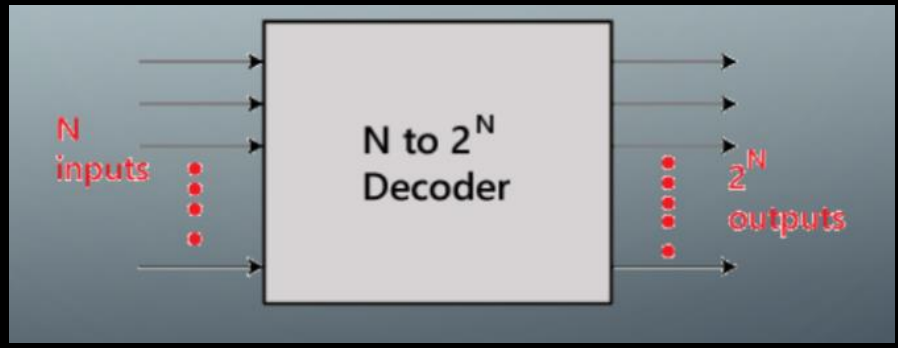


Inputs			Outputs							
S_2	S_1	S_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

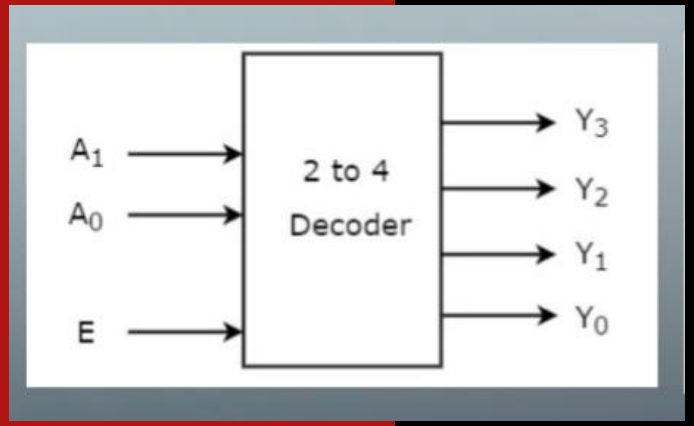


- 1:8 DeMUXusing two 1:4 DeMUXand one 1:2 DeMUX

- Decoder• ‘n’ input lines and maximum of 2noutput lines

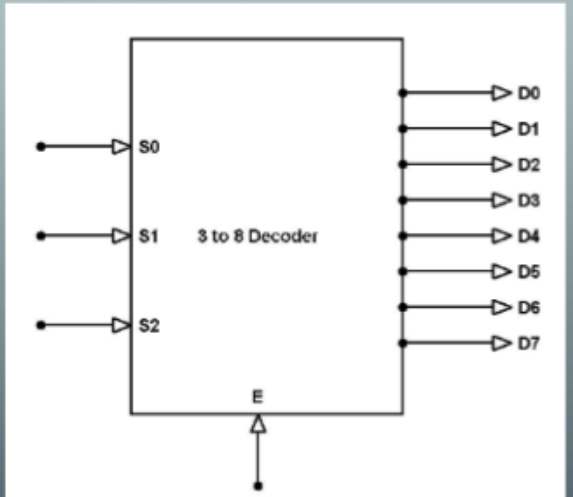


2 to 4 Decoder

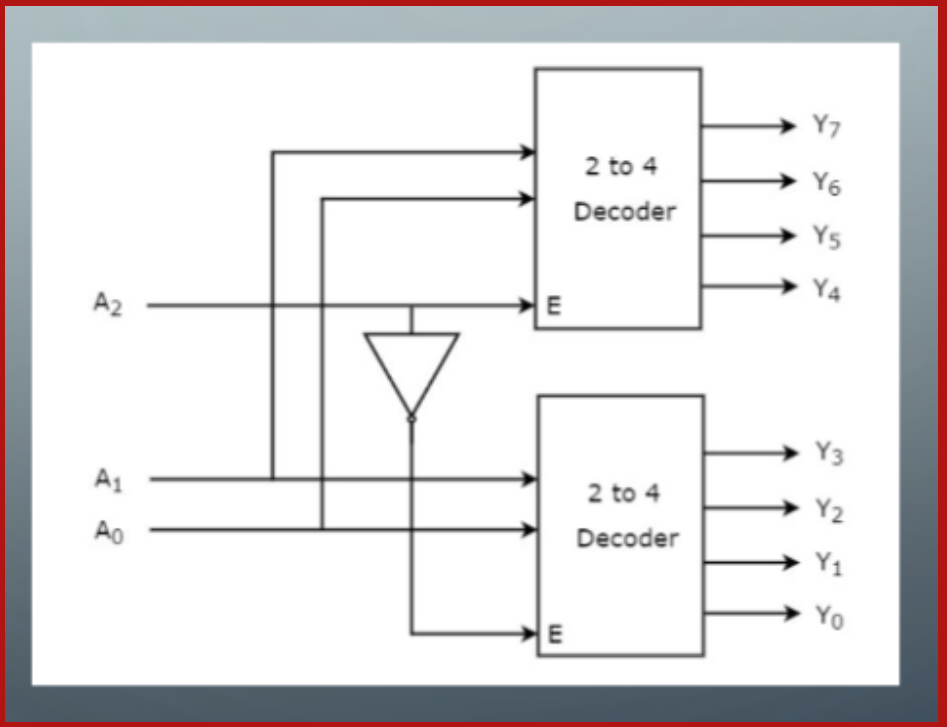


Inputs			Output			
En	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

3 to 8 Decoder

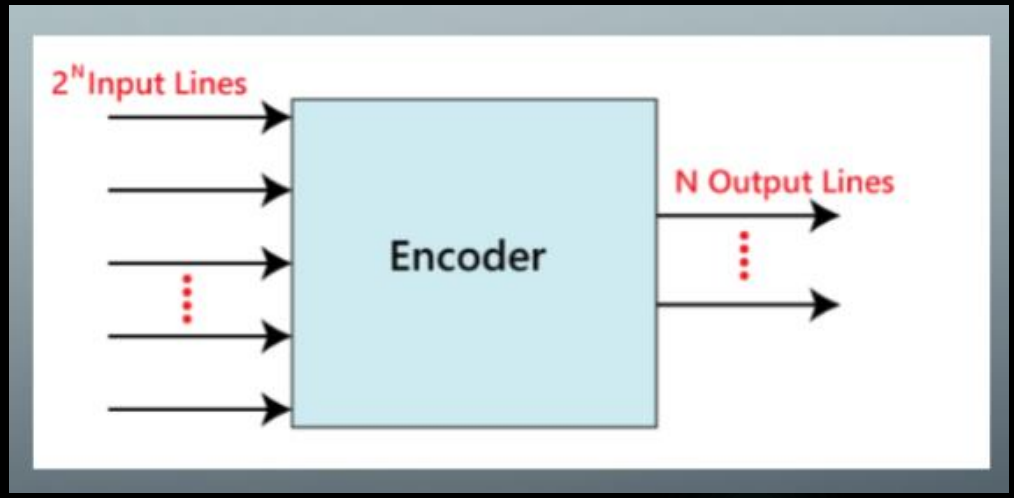


Inputs				Outputs							
E	S_2	S_1	S_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

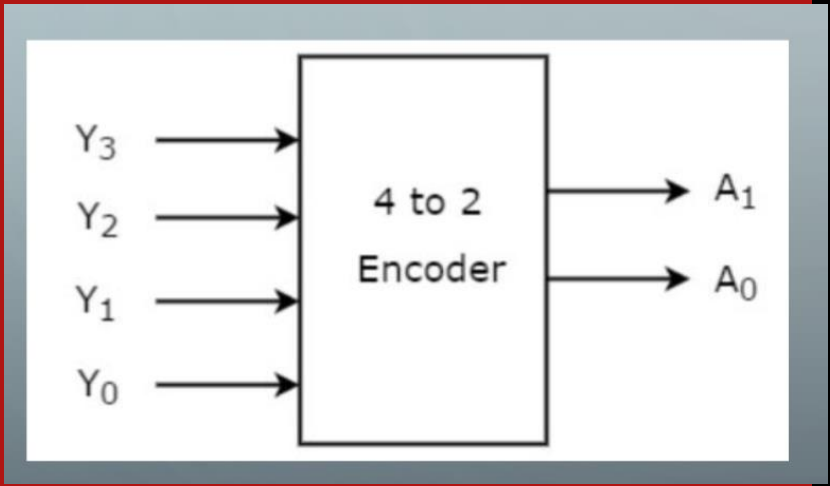


- 3 to 8 Decoder using two 2 to 4 Decoders

- Encoder
- Maximum of 2ninput lines and ‘n’ output lines.

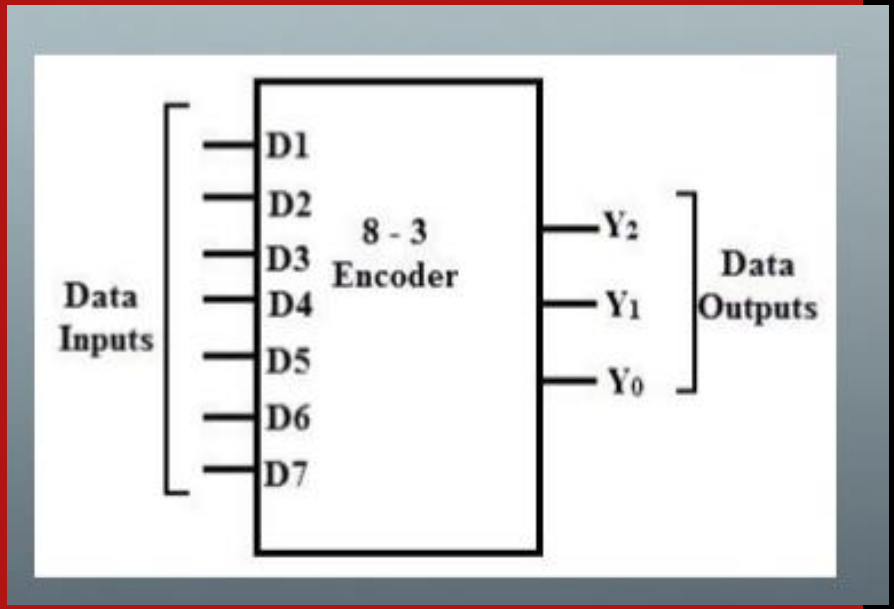


- 4 to 2 Encoder



Inputs				Outputs	
Y_3	Y_2	Y_1	Y_0	A_1	A_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

- 8 to 3 Encoder

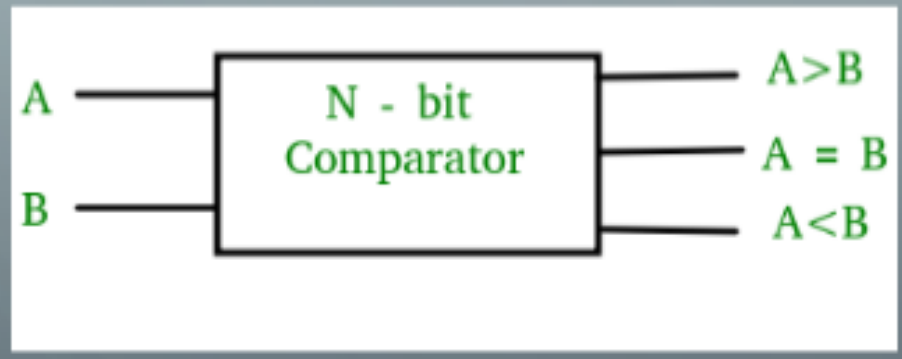


Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

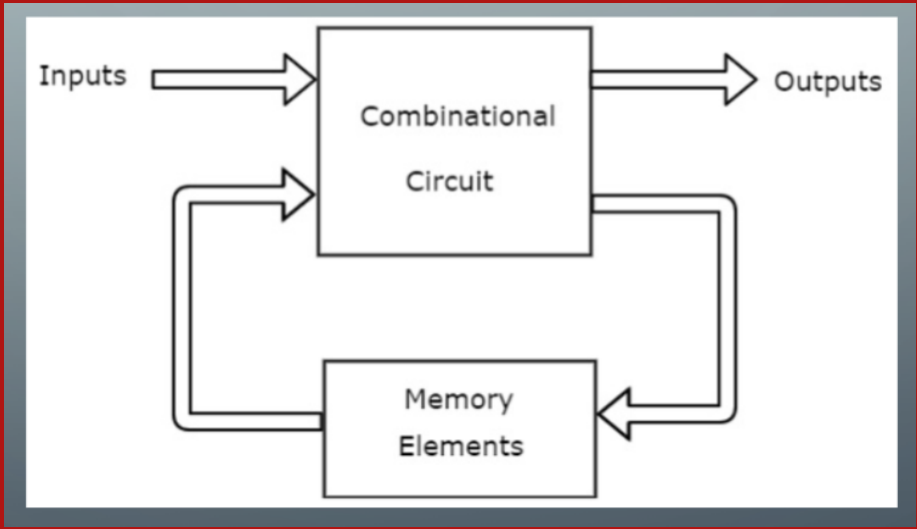
- Priority Encoder

Inputs								Outputs			
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Y_2	Y_1	Y_0	V
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1

- Compares two binary numbers in order to find out whether one binary number is equal, less than or greater than the other binary number.



A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0



- Sequential circuit
- The sequential circuit is a special type of circuit that has a series of inputs and outputs. The outputs of the sequential circuits depend on both the combination of present inputs and previous outputs.

Combinational Circuits	Sequential Circuits
Outputs depend only on present inputs.	Outputs depend on both present inputs and present state.
Feedback path is not present.	Feedback path is present.
Memory elements are not required.	Memory elements are required.
Clock signal is not required.	Clock signal is required.
Easy to design.	Difficult to design.

Types of Sequential Circuits

- Asynchronous sequential circuits

Theclocksignalsarenotusedby theAsynchronoussequentialcircuits.Theasynchronouscircuitisoperatedthroughthepulses.So,thechangesintheinputcanchangethestateofthecircuit.

- Synchronous sequential circuits

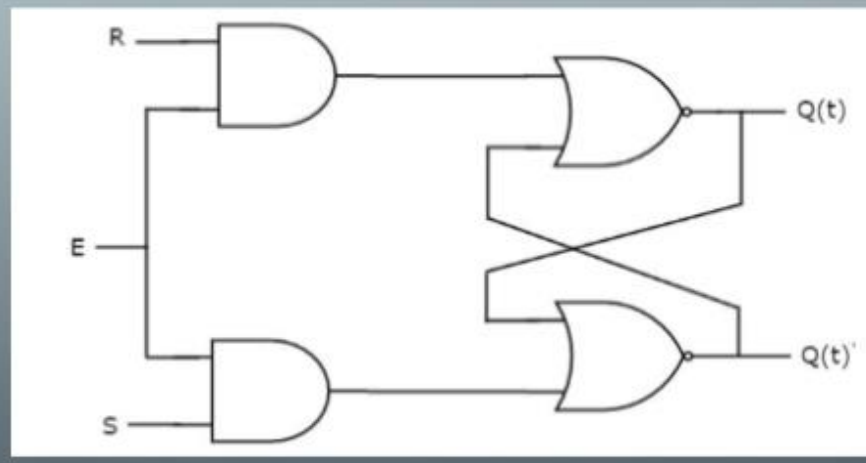
In synchronous sequential circuits, synchronization of the memory element's state is done by the clock signal. The output is stored in either flip-flops or latches(memory devices). Sequential circuits

- Clock signal
  A clock signal is a periodic signal in which ON time and OFF time need not be the same. When ON time and OFF time of the clock signal are the same, a square wave is used to represent the clock signal. Sequential circuits
- Triggering

- Level triggering The logic High and logic Low are the two levels in the clock signal.
- Positive level triggering In a positive level triggering, the signal with Logic High occurs.
- Negative level triggering In negative level triggering, the signal with Logic Low occurs. Sequential circuits
- Edge triggering In clock signal of edge triggering, two types of transitions occur, i.e., transition either from Logic Low to Logic High or Logic High to Logic Low.
- Positive edge triggering The transition from Logic Low to Logic High occurs in the clock signal of positive edge triggering.
- Negative edge triggering The transition from Logic High to Logic low occurs in the clock signal of negative edge triggering.

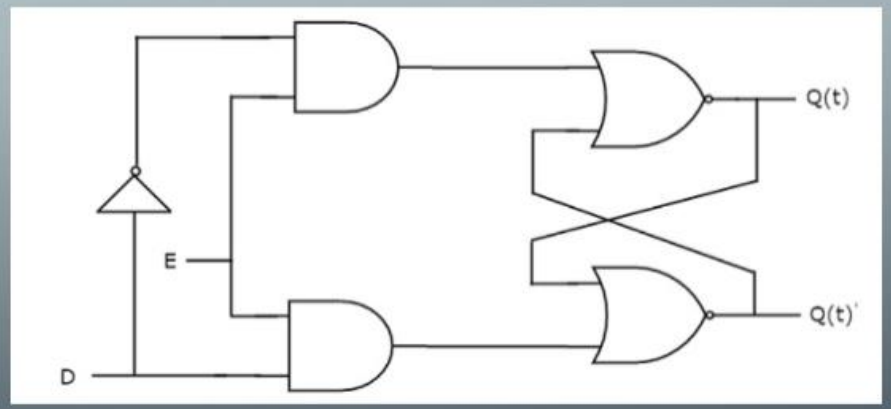
- Memory Elements
- There are two types of memory elements based on the type of triggering that is suitable to operate it.
- Latches
- Latches operate with enable signal, which is level sensitive.
- Flip-flops
- Flip-flops operate with edge signal, which is edge sensitive.
- All the flip-flops can be design using the NAND Gate alone
- Conversion of Flip-Flops
- It is possible to convert every flip-flop into a different flip-flop with changes in the sequential circuit.

- Latches
- They are triggered with level of the signal.
- SR Latch (Set Reset Latch)



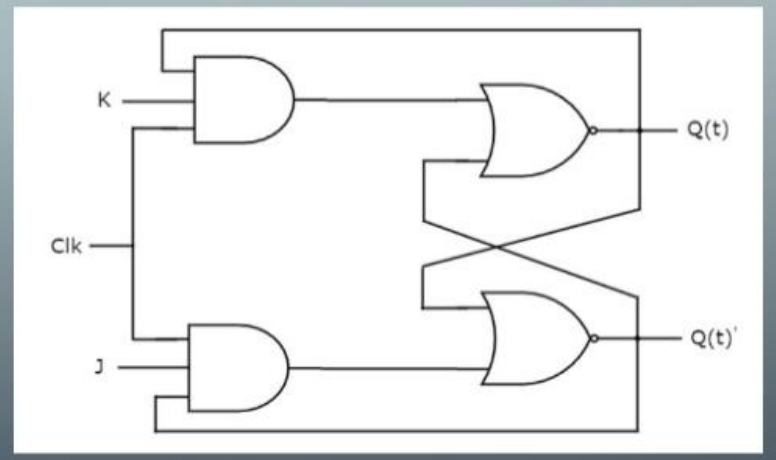
S	R	Qt+
0	0	Qt
0	1	0
1	0	1
1	1	-

- D Latch



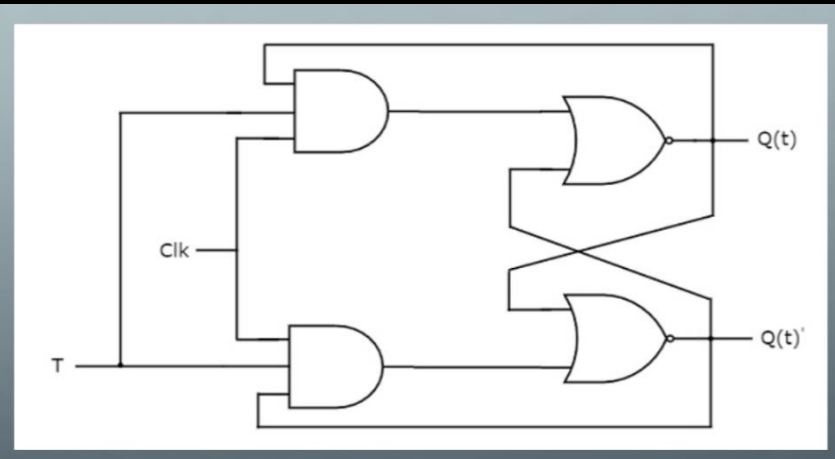
D	Qt+
0	0
1	1

- JK Flip Flop



J	K	Qt+
0	0	Qt
0	1	0
1	0	1
1	1	Qt'

- T Flip Flop



T	Qt+
0	Qt
1	Qt'

VYORIUS

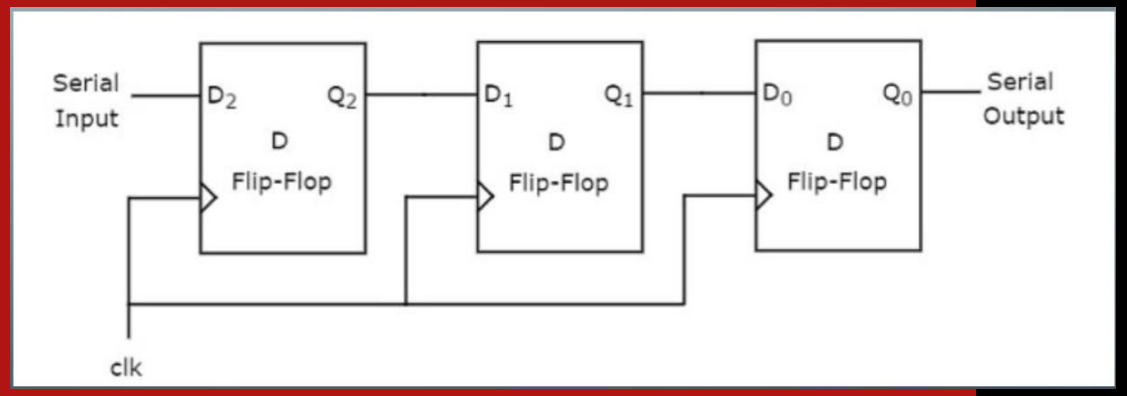
Shift Registers

- Group of flip-flops, which are used to hold store the binary data is known as register.
- If the register is capable of shifting bits either towards right hand side or towards left hand side is known as shift register.

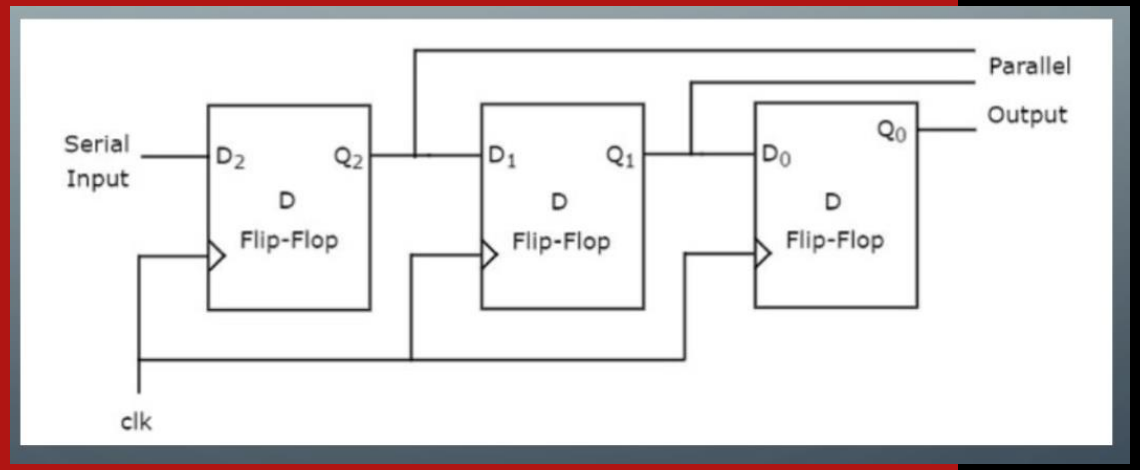
Types:

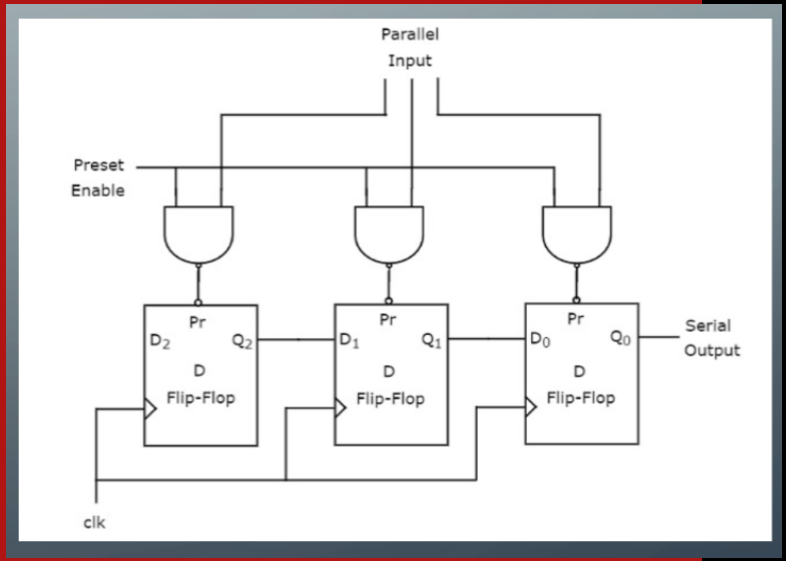
1. Serial In –Serial Out shift register
2. Serial In –Parallel Out shift register
3. Parallel In –Serial Out shift register
4. Parallel In –Parallel Out shift registerSequential circuits

- Serial In –Serial Out shift register The shift register, which allows serial input and produces serial output is known as Serial In –Serial Out SISO shift register

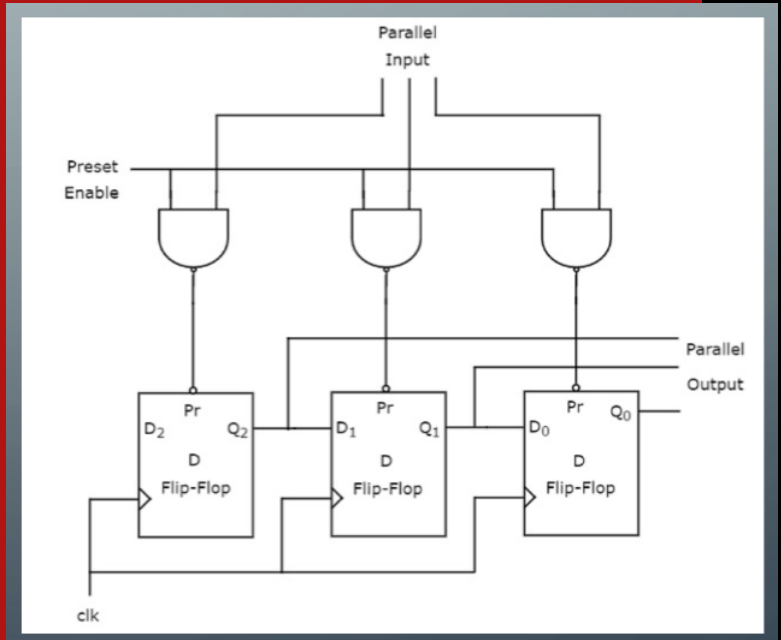


- Serial In -Parallel Out SIPO Shift Register The shift register, which allows serial input and produces parallel output is known as Serial In –Parallel Out SIPO shift register.

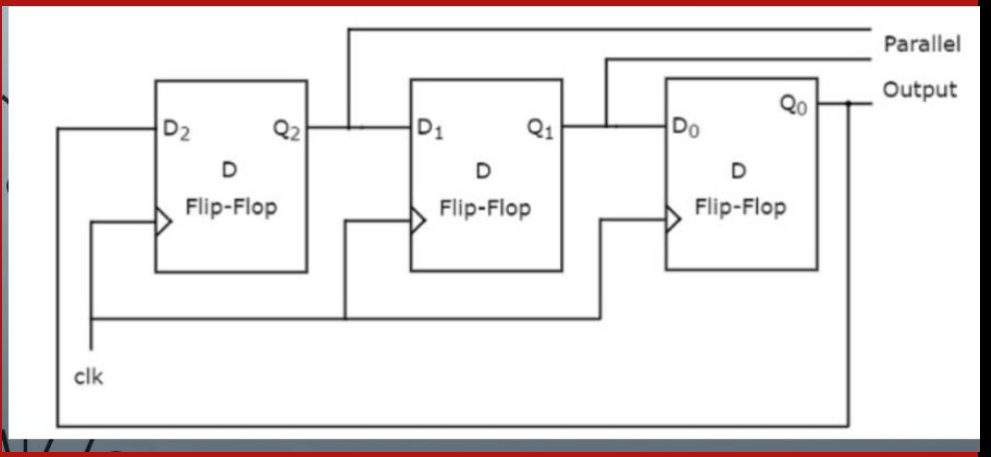




- Serial Out PISO Shift Register The shift register, which allows parallel input and produces serial output is known as Parallel In –Serial Out PISO shift register.



- Parallel Out PIPO Shift Register The shift register, which allows parallel input and produces parallel output is known as Parallel In –Parallel Out PIPO shift register



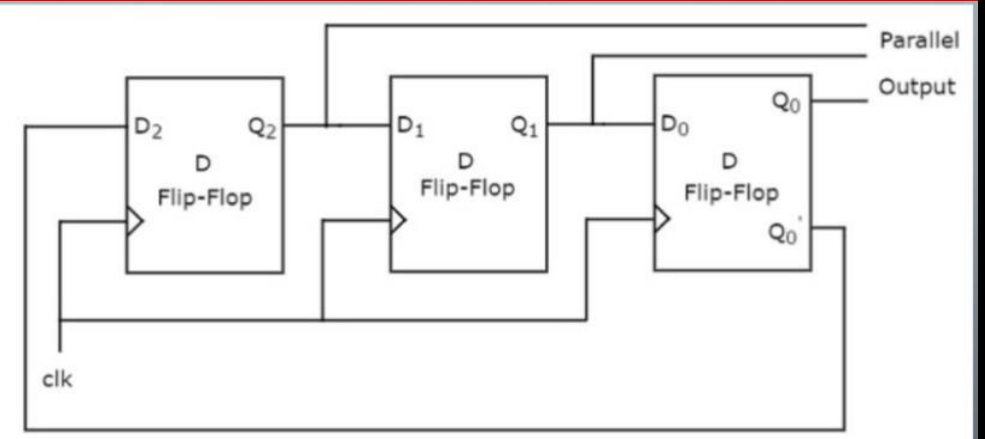
- Shift Registers

  Ring Counter

No of positive edge of Clock	Serial Input = Q_0	Q_2(MSB)	Q_1	Q_0(LSB)
0	-	0	0	1
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1

- Shift Registers

  Johnson Ring Counter



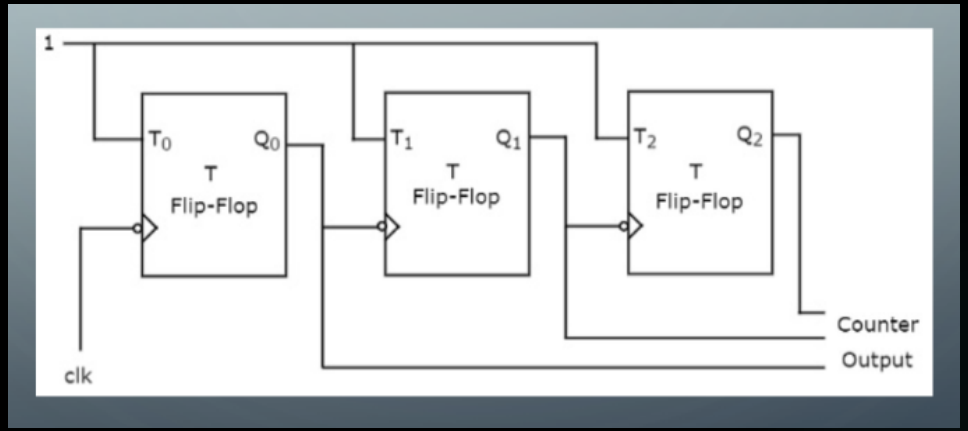
No of positive edge of Clock	Serial Input = Q_0	Q_2(MSB)	Q_1	Q_0(LSB)
0	-	0	0	0
1	1	1	0	0
2	1	1	1	0
3	1	1	1	1
4	0	0	1	1
5	0	0	0	1
6	0	0	0	0

VYORIUS

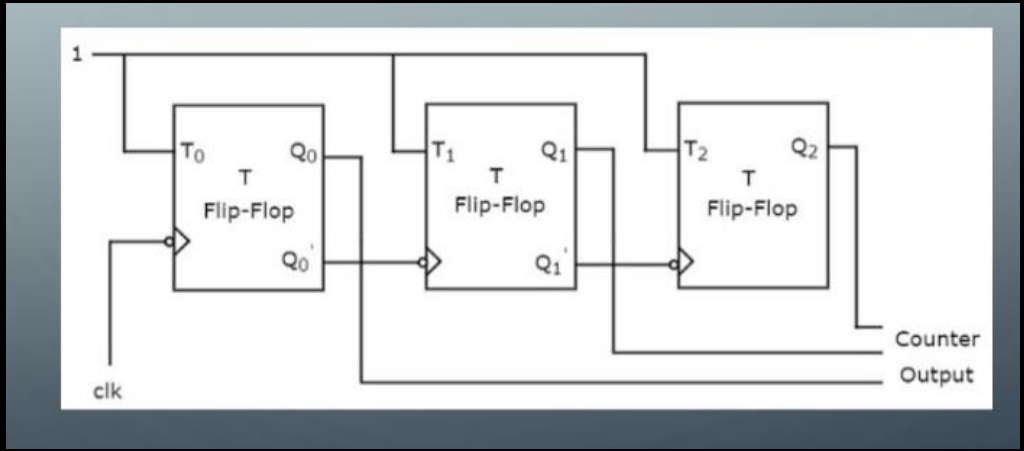
- Counters
- If the counter counts from 0 to $2N-1$, then it is called as binary up counter.
- Similarly, if the counter counts down from $2N-1$ to 0, then is called as binary down counter.
- Types of counters

1. Asynchronous counters
2. Synchronous counters

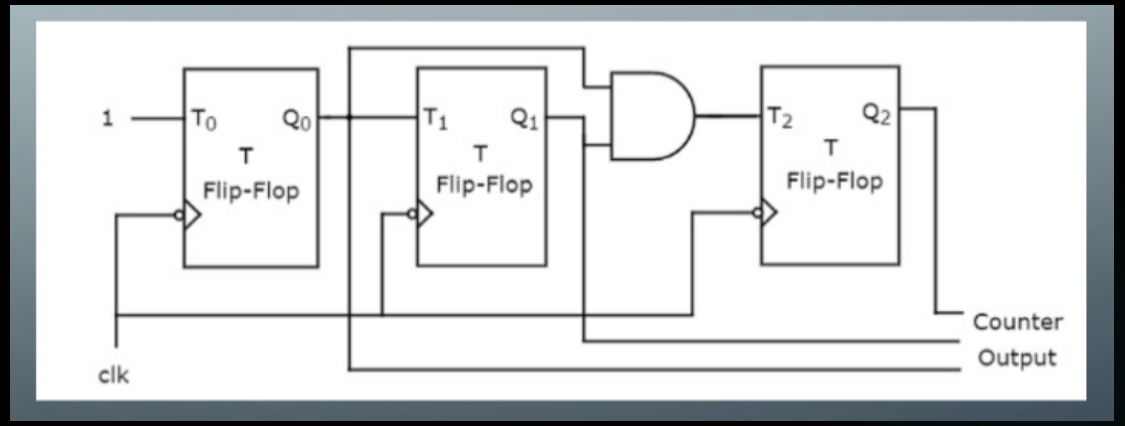
- Asynchronous Counters
- Asynchronous Binary up counter



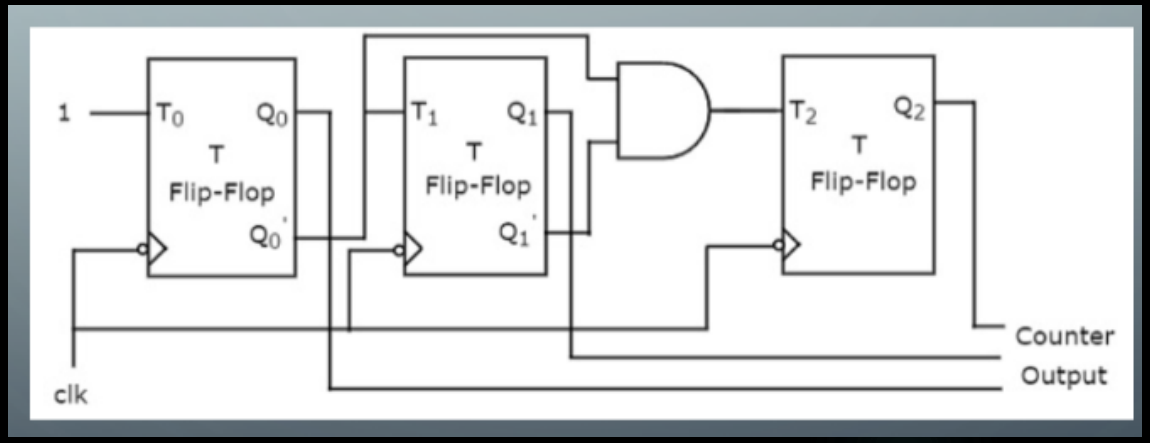
- Asynchronous Binary down counter



- Synchronous Counters
- Synchronous Binary up counter

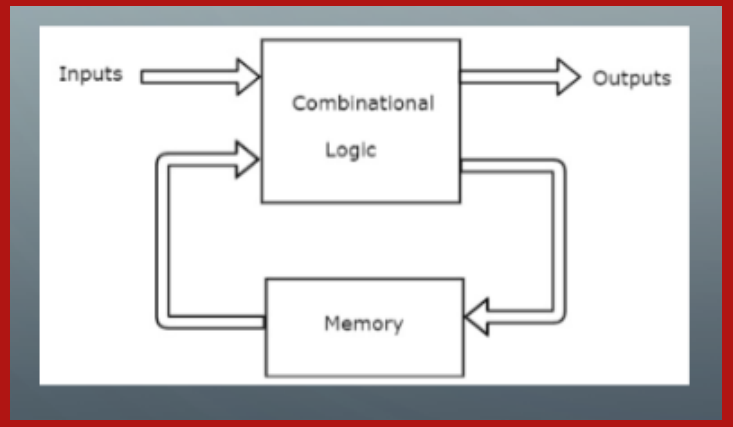


- Synchronous Binary down counter



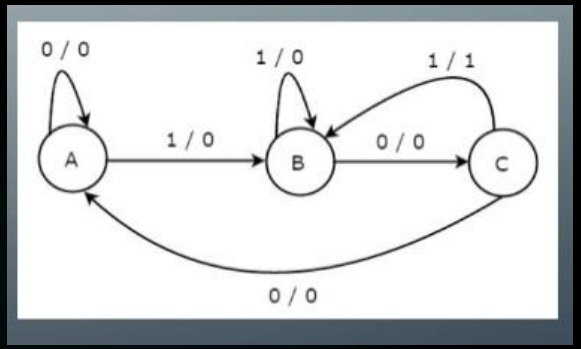
- Finite State Machines
- The behavior of synchronous sequential circuits can be represented in the graphical form and it is known as state diagram.
- Decision making logic and Control of the Digital System.
- There are two types of FSMs.

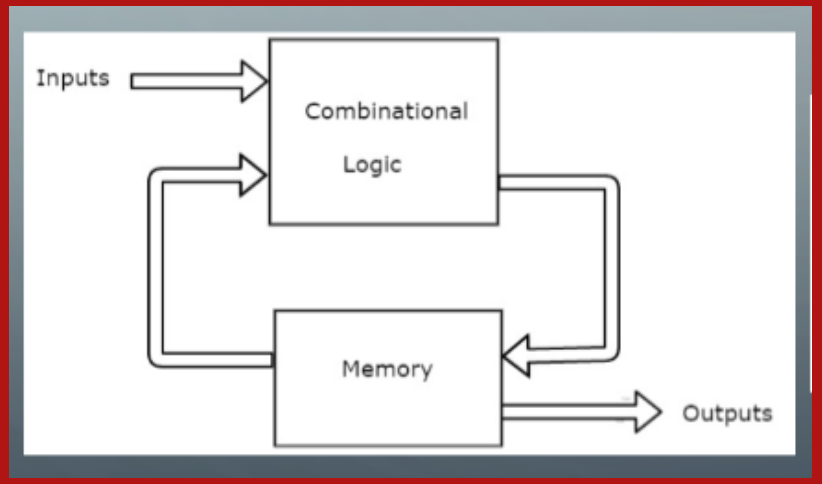
1. Mealy State Machine
2. Moore State Machine



Mealy State Machine

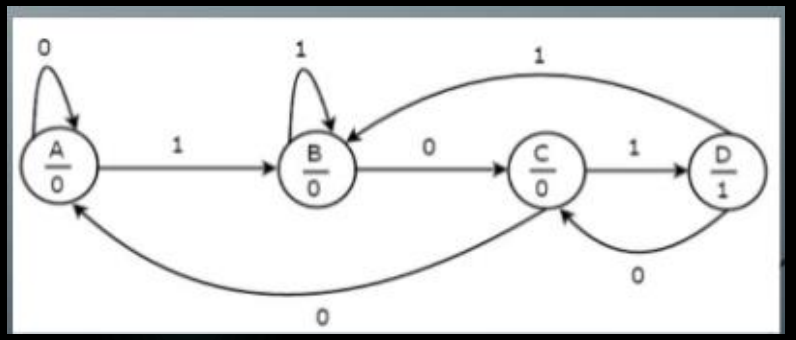
A Finite State Machine is said to be Mealy state machine, if outputs depend on both present inputs & present states.

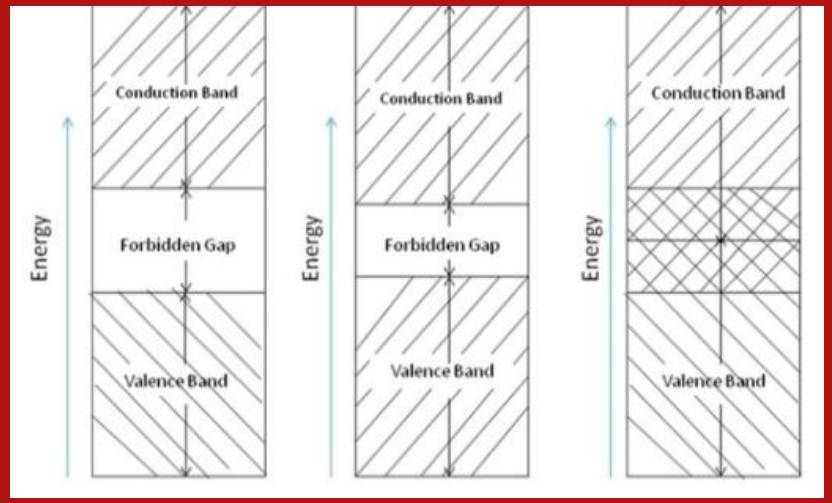


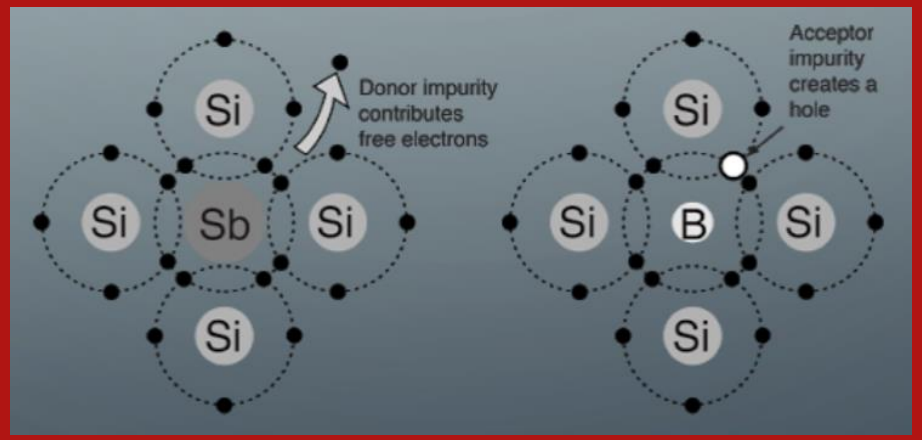


- Moore State Machin

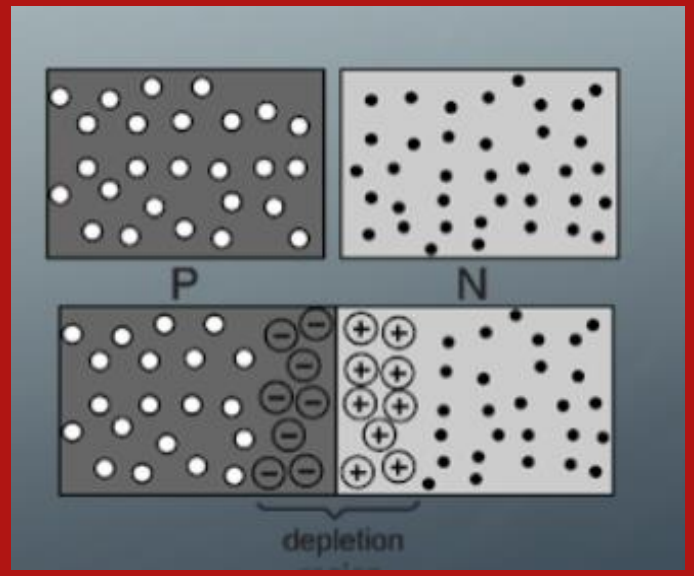
A Finite State Machine is said to be Moore state machine, if outputs depend only on present states.

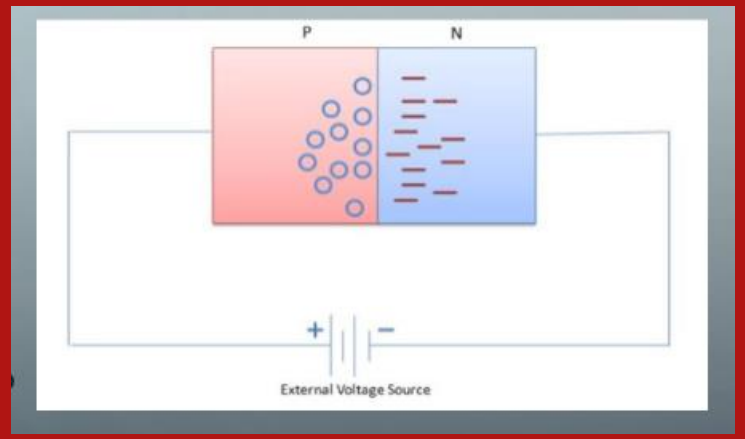






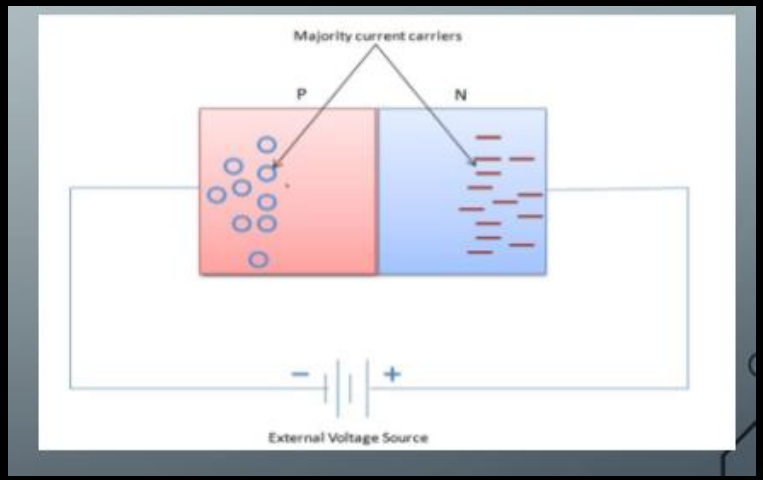
- Semiconductors
- Conductivity in between those of metals and insulators.
- Conductivity can be varied over orders of magnitude by changes in temperature, optical excitation, and impurity content (doping).
- Generally found in column IV and neighboring columns of the periodic table.
- Elemental semiconductors: Si, Ge
- Energy levels of insulators, semiconductors and conductors.
- Valence Band
- Conduction Band
- Forbidden Gap
- Pure Silicon or Germanium are rarely used as semiconductors.
- Adding an impurity to an intrinsic or pure material is called doping and the impurity is called a dopant.
- Extrinsic Semiconductor



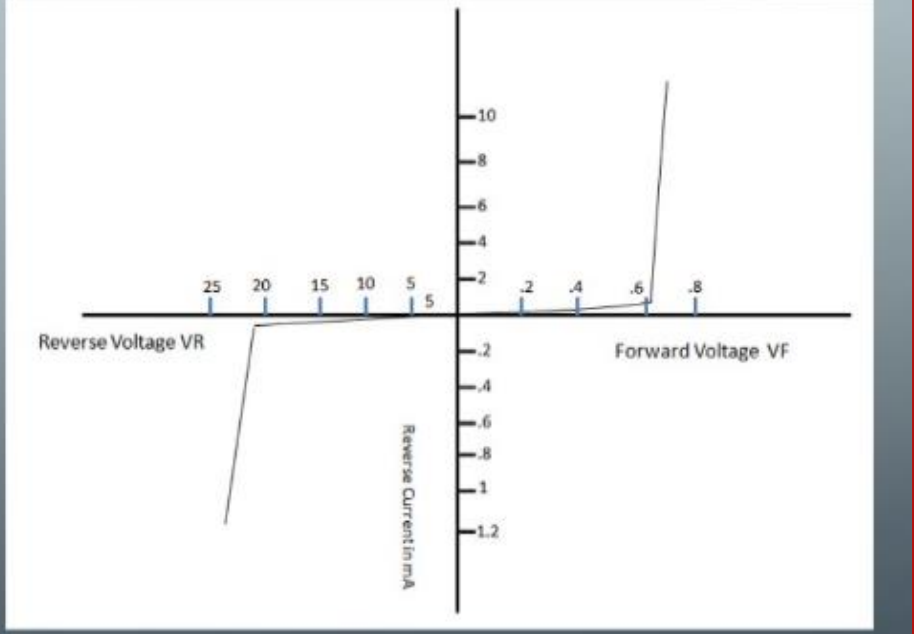


Forward Biasing −An external voltage is added of the same polarity to the barrier potential, which causes an increase in the width of the depletion region.

- P and N materials is generally known as junction diode.
- Diffusion -The electrons move readily across the junction to fill holes.
- The area where these holes and electrons become depleted is generally known by the term depletion region.
- Reverse Biasing −A PN junction is biased in such a way that the application of external voltage action prevents current carriers from entering the depletion region.



Diode Characteristic



Diode Specifications

- Maximum forward current
- Maximum reverse voltage
- Reverse breakdown voltage
- Maximum forward surge current
- Maximum reverse current
- Forward voltage
- Power dissipation
- Reverse recovery time

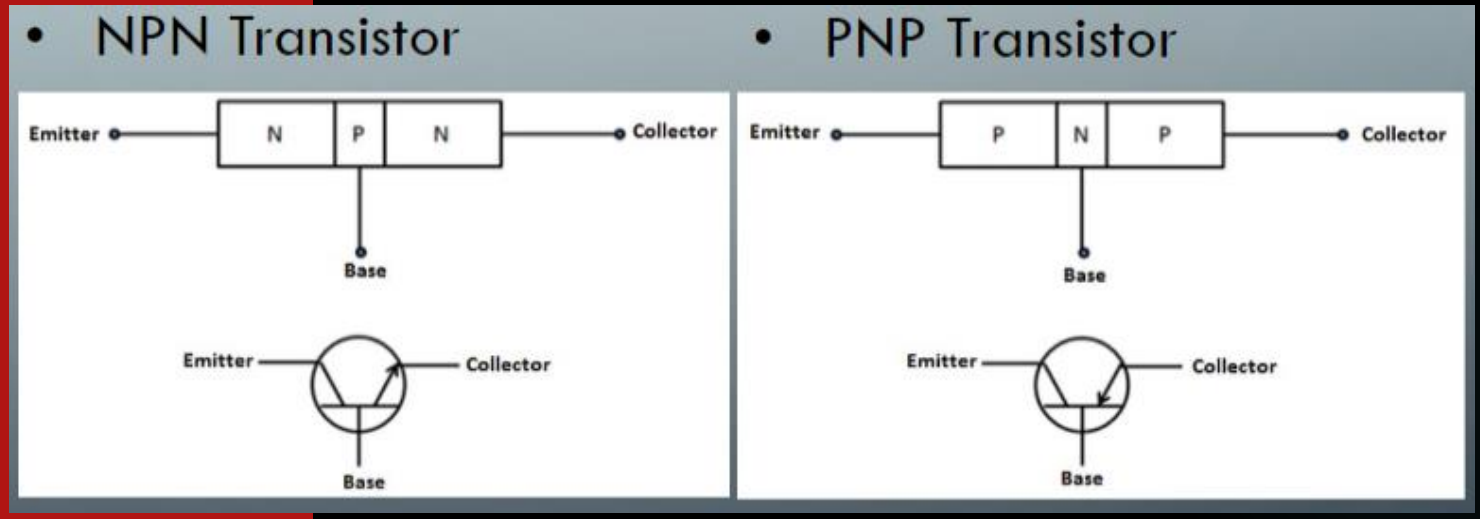
Important Terms

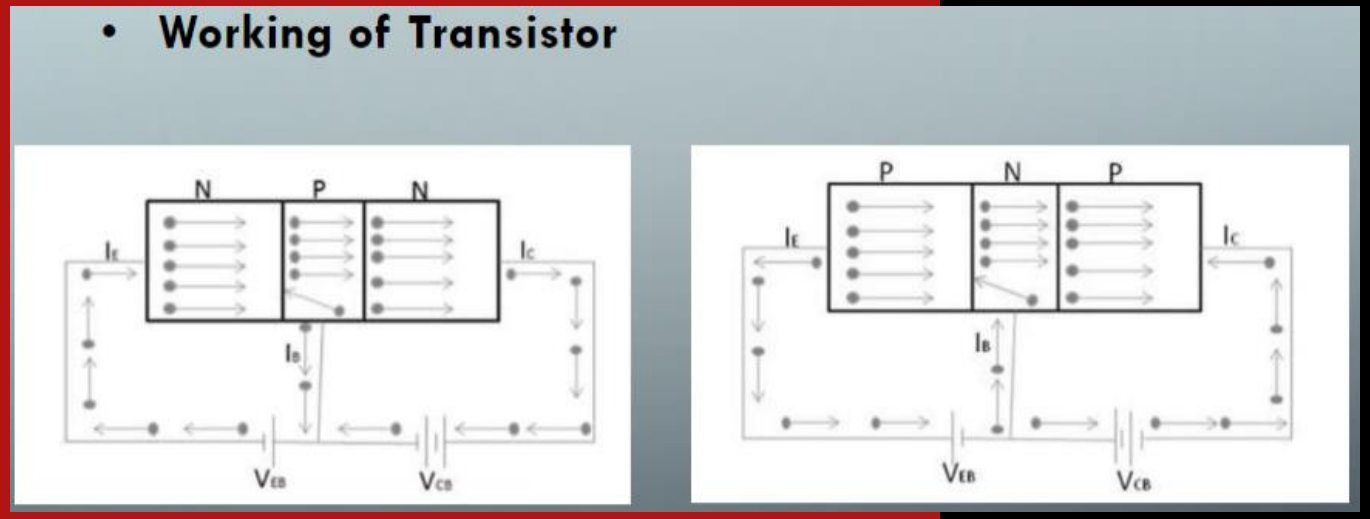
- Breakdown Voltage
- Knee Voltage
- Peak Inverse Voltage
- Maximum Forward Rating
- Maximum Power Rating

Types of Diode

- Light emitting diode (LED)
- Zener Diode
- Photo Diode

Bipolar transistors are mainly formed of two layers of semiconductor material of the opposite type, connected back-to-back.





Manufacturing techniques used in the construction of a transistor

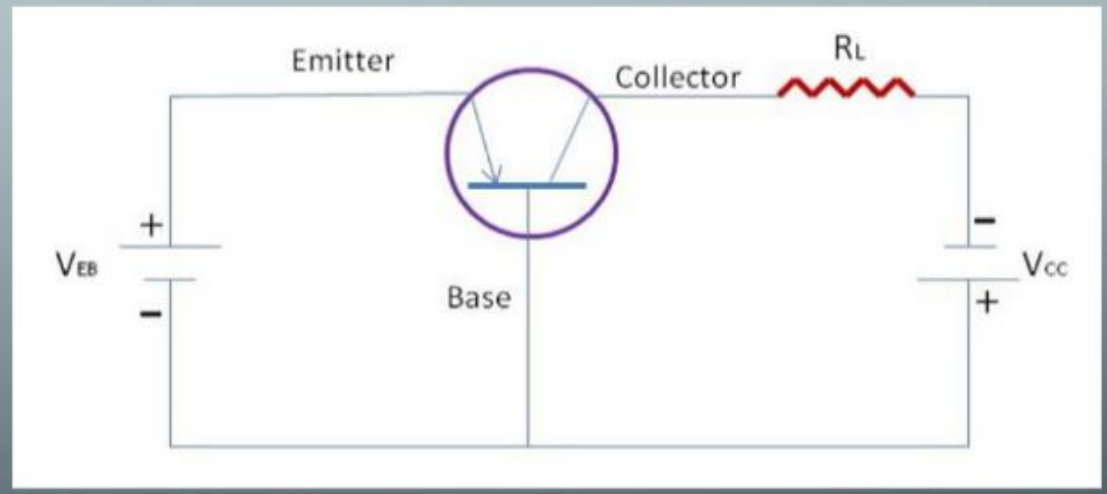
- Diffusion Type
- Grown Type
- Epitaxial Type
- Alloy Type
- Electrochemically Etched Type

- **Transistor Configurations**

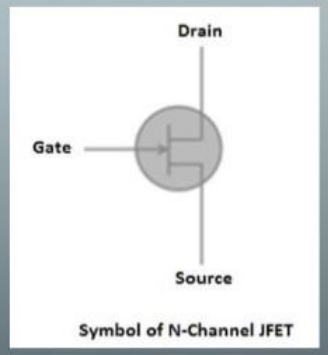
Characteristics	Common Emitter	Common Base	Common Collector
Current Gain	High	No	Considerable
Applications	Audio frequency	High frequency	Impedance matching
Input Resistance	Low	Low	Very high
Output Resistance	High	Very high	Low
Voltage Gain	Approx. 500	Approx. 150	Less than 1

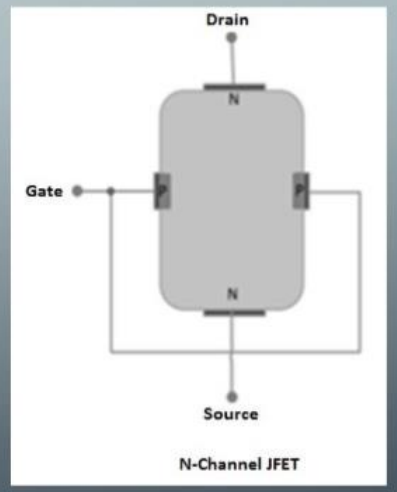
- Current Amplification Factor (α)
- Base Current Amplification Factor (β)

- Transistor as an Amplifier

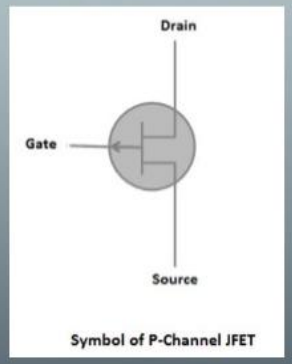


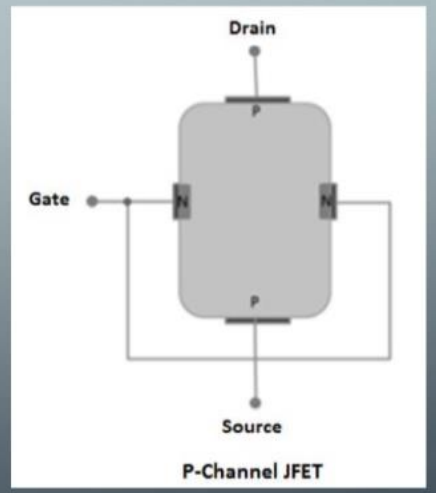
- N-Channel JFETs





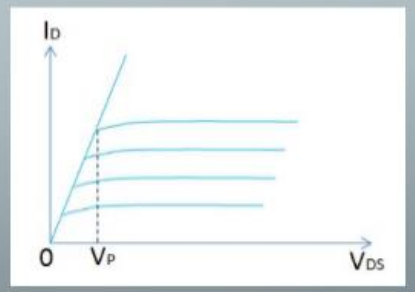
- P-Channel JFETs





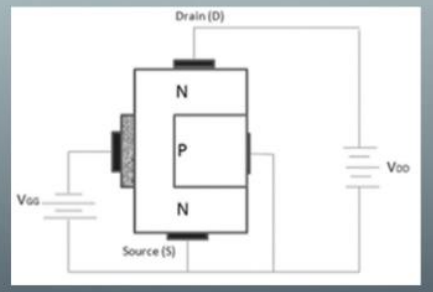
- Field Effect Transistor (FET)
- Junction Field Effect Transistor (JFET) • N –Channel FET
- P –Channel FET
- Metal Oxide Semiconductor FET (MOSFET)
- Depletion Mode
- Enhancement Mode
- Junction Field Effect Transistor (JFET)
- Gate−SidesofNtypebarareheavilydopedtocreateP Njunction.Thesedopedregionsarecalledgate(G).
- Source−Entrypointformajoritycarriersthroughwhicht heyenterintothesemiconductorbar.
- Drain−Exitpointformajoritycarriersthroughwhichtheyl eav ethesemiconductorbar.
- Channel−AreaofNtypematerialthroughwhichmajori tycarrierspassfromthesourcetodrain.

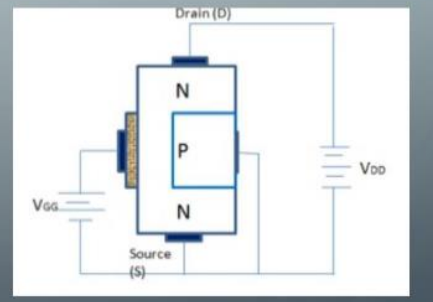
- Output Characteristics of JFET



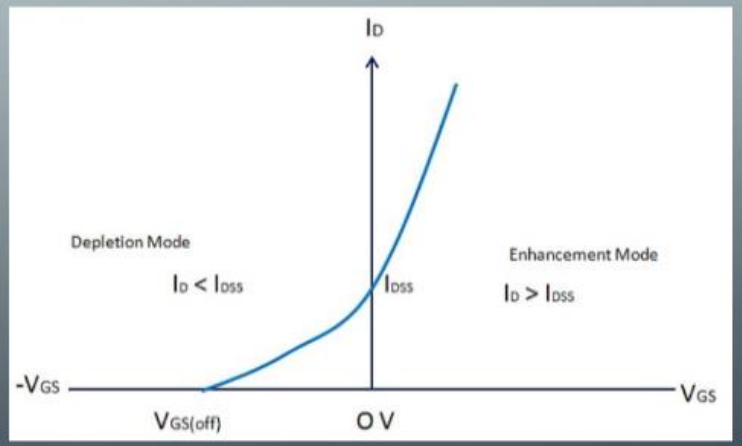
- Parameters of JFET
  - **AC drain resistance**
  - **Transconductance**
  - **Amplification Factor**

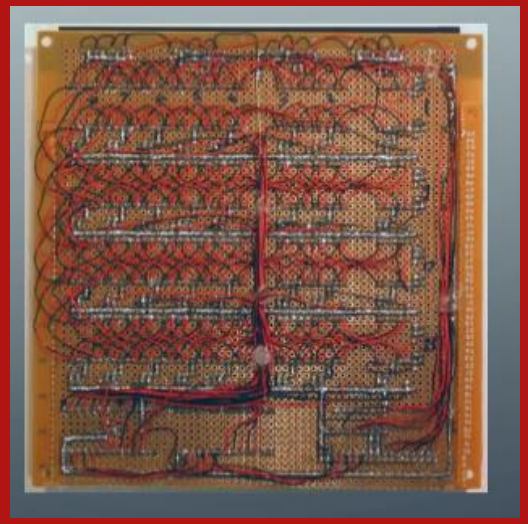
- **Metal-oxide semiconductor field-effect transistors**
  - Depletion Mode
  - Enhancement Mode





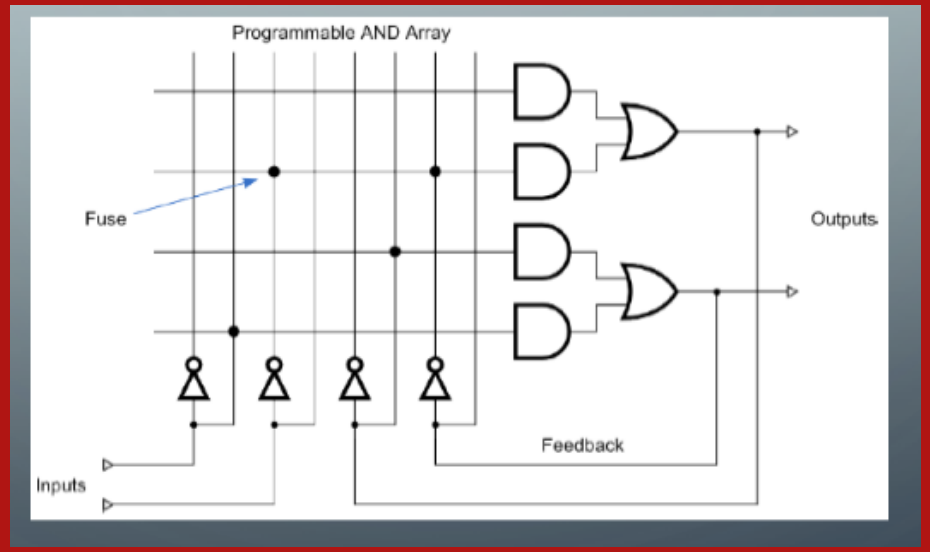
- Transfer Characteristics of MOSFET

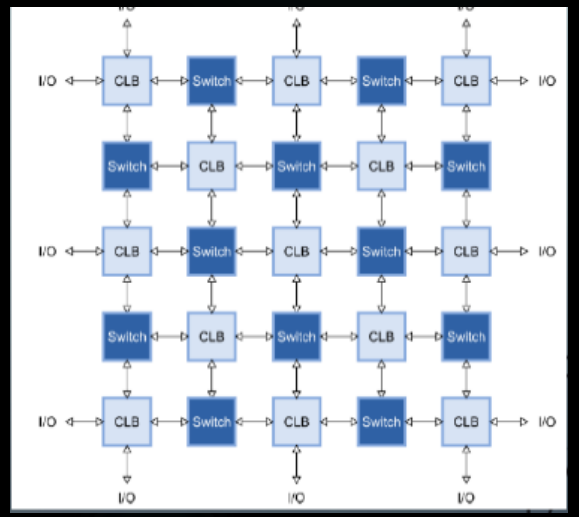


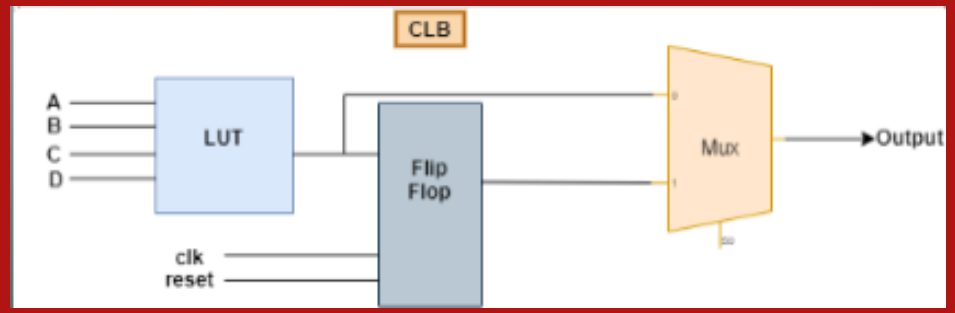


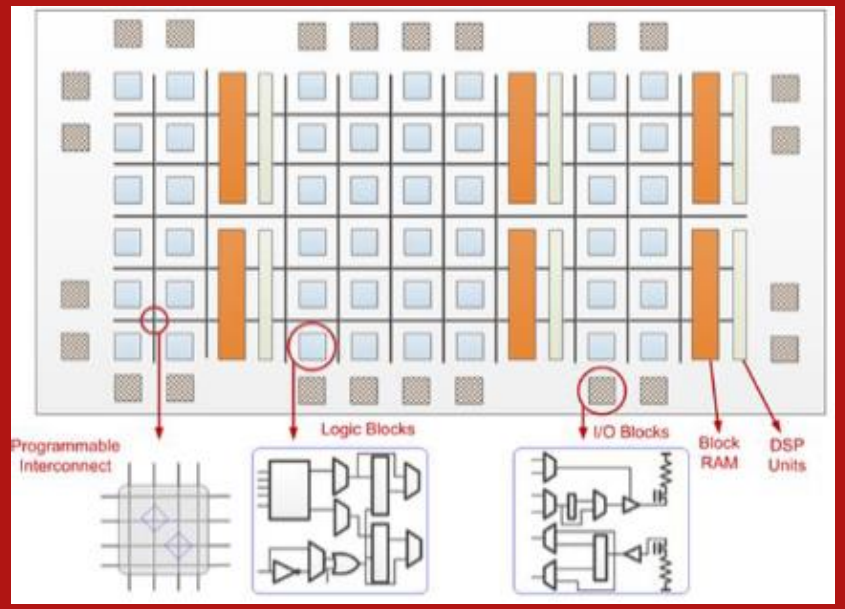
FPGAs

- Systems were built from lots of individual ANDs, ORs, flip-flops, etc.
- Manufacturing of such systems would take a lot of time.
- Placement of an unconnected array of AND-OR gates in a single chip called a programmable logic device (PLD).
- Much larger programmable chips were built; they are called as called complex programmable logic devices (CPLDs) and field-programmable gate arrays (FPGAs)
- A CPLD contains a bunch of PLD blocks, but their inputs and outputs are connected together by a global interconnection matrix.
- A FPGA has a bunch of simple, configurable logic blocks (CLBs) interspersed within a switching matrix that can rearrange the interconnections between the them.







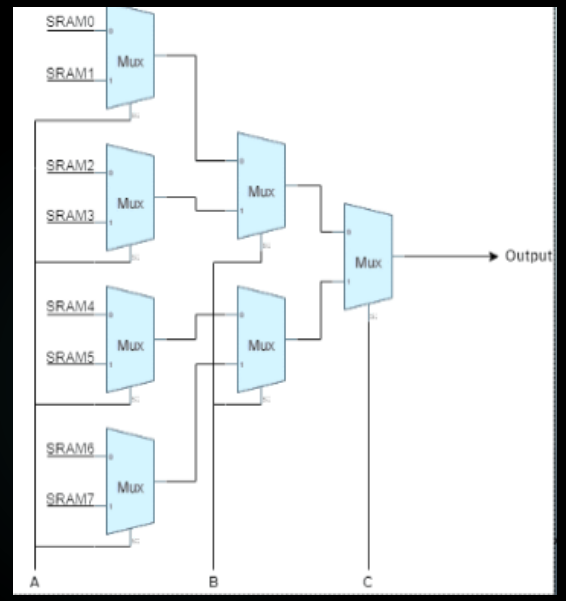


FPGA Architecture•XC2064 –First FPGA

Three Fundamental elements:

1.Configurable Logic Block (CLB)

- Consists of three essential elements:
- Look-Up Tables (LUTs)
- Multiplexers (MUXs)
- Flip-Flops
- Look-Up Table
- Designed to implement any Boolean equations.
- k-LUT–2k SRAM bits and 2k:1 MUXs
- Ex: 3-LUT –8 SRAM bits and 2:1 MUXs



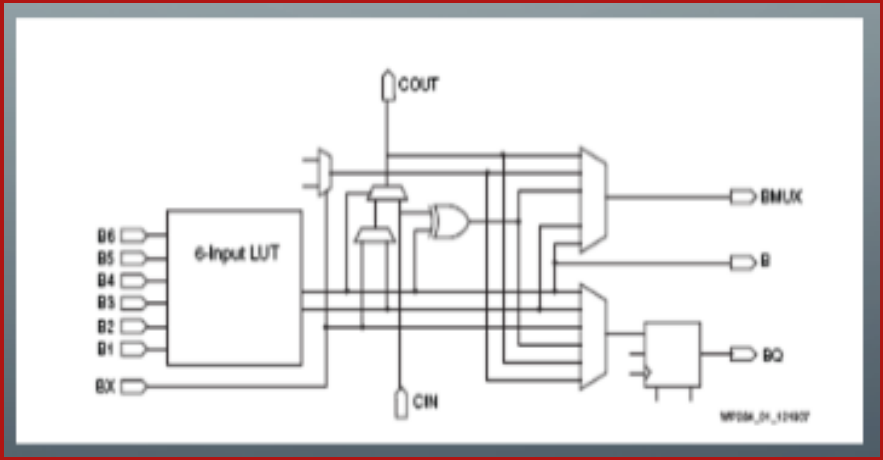
2.Interconnect Architecture

- Programmable switches and wires

3.Input/Output Blocks

- Input from outside of FPGA to the FPGA pin.
- A delay element for the input, having an output for providing a delay to the input signal.
- A multiplexer is providing a delayed input signal.
- A register/latch is providing a register/latch output signal.

- A set/reset line for providing a set/reset signal.
- A decoder for providing the set signal or the reset signal responsive to the set/reset signal.
- An amplifier has an output for providing an amplified signal to a related resource in the FPGA.



- FPGA DSP
- Virtex-5 FPGA is the first FPGA to have DSP (DSP48E2).
- Makes the multiplication and accumulation operation fast.
- DSP tile's generic feature includes a 27-bit pre-adder, 27 x 18 multiplier and a flexible 48-bit ALU.
- FPGA Timing
- Time taken by a signal to propagate from one flip-flop to the next flip-flop (through some combinational logic).
- Combinational logic is not instantaneous.
- Modern FPGA are based on CMOS technology.
- Setup and Hold times.

- Modern FPGA Architecture
- More LUTs in a smaller area with minimum power utilization.
- Xilinx -CLB that contains the two slices; each slice has four 6-input LUTs.
- Clocking Architecture
- Logics are implemented as combinational logic and sequential logics.
- Clock Resources
- Efficient
- Highly precise duty cycle
- Low power
- Jitter-free
- Phase lock loop (PLL)
- Delay lock loop (DLL)

- High-Speed Transceivers
- High-speed communication between FPGA device and external interfaces or devices.
- Standard Protocols: Ethernet, PCI Express and so on.

VYORIUS

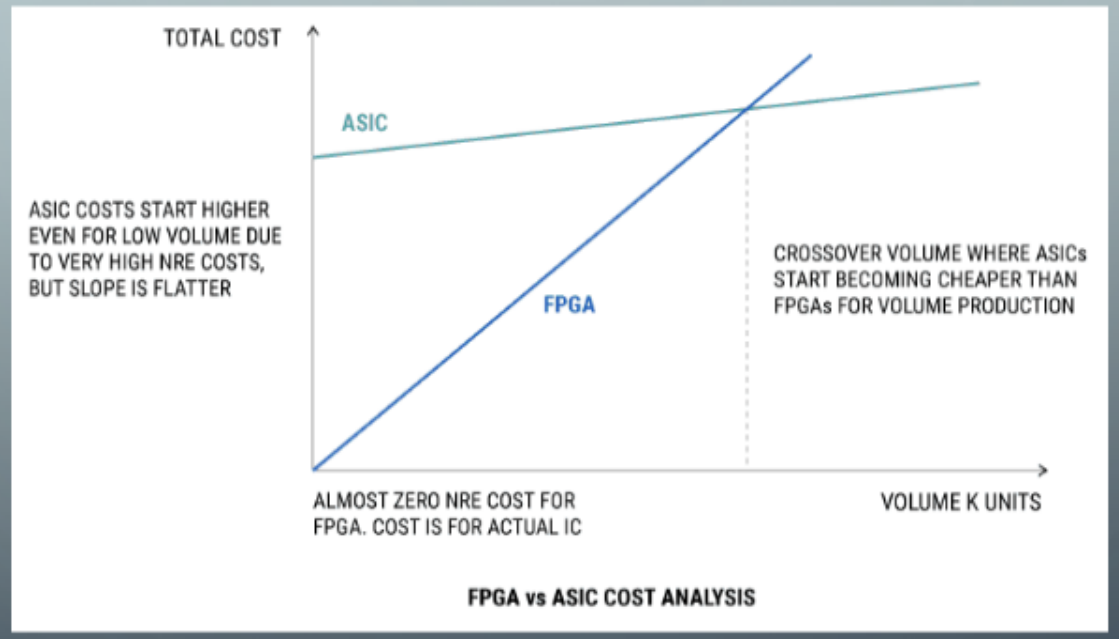
CPLD vs FPGA

No.	CPLD	FPGA
1	Instant-on. CPLDs start working as soon as they are powered up	Since FPGA has to load configuration data from external ROM and setup the fabric before it can start functioning, there is a time delay between power ON and FPGA starts working. The time delay can be as large as several tens of milliseconds.
2	Non-volatile. CPLDs remain programmed, and retain their circuit after powering down.	FPGAs go blank as soon as powered-off. FPGAs uses SRAM based configuration storage. The contents of the memory is lost as soon as power is disconnected.
3	Deterministic Timing Analysis. Since CPLDs are comparatively simpler to FPGAs, and the number of interconnects are less, the timing analysis can be done much more easily.	Size and complexity of FPGA logic can be humongous compared to CPLDs. This opens up the possibility less deterministic signal routing and thus causing complicated timing scenarios. Thankfully implementation tools provided by FPGA vendors have mechanisms to assist achieving deterministic timing. But additional steps by the user is usually necessary to achieve this.
8	No on-die hard IPs available to offload processing from the logic fabric.	Variety of on-die dedicated hardware such as Block RAM, DSP blocks, PLL, DCMs, Memory Controllers, Multi-Gigabit Transceivers etc give immense flexibility. This is not even thinkable with CPLDs.
9	Power down and reprogramming is always required in order to modify design functionality.	FPGAs can change their circuit even while running! (Since it is just a matter of updating LUTs with different content) This is called Partial Reconfiguration, and is very useful when FPGAs need to keep running a design and at the same time update the it with different design as per requirement. This feature is widely used in Accelerated Computing.

VYORIUS

VYORIUS

- Cost Analysis



FPGA vs ASIC COST ANALYSIS

Application Specific Integrated Circuit
Designed for one sole purpose and they function the same their whole operating life.
The logic function of ASIC is specified in a similar way as in the case of FPGAs, using hardware description languages such as Verilog or VHDL

Microcontrollers vs FPGA

No.	Microcontrollers	FPGA
1	Embedded Microcontrollers are similar to computers. They have all the necessary peripherals such as memory, input-output ports and timers.	FPGA is an integrated circuit (IC) containing millions of logic gates and can be made to carry out tasks by programming the logic gates. They need external peripherals such as RAM and ROM for its application.
2	The microcontroller uses a software program to execute commands consecutively, such as C, C++.	The programming connection of the FPGA is on the logic circuit and use programming solutions such as VHDL and Verilog.
3	The processing power of microcontrollers is time-limited and based on its processor cycling power.	FPGAs are space limited; you need to create more logic circuits to achieve your desired scale of coding.
4	Microcontrollers can perform limited tasks because they come with instructions and their circuitry. A programmer has to abide by the restrictions while developing code.	FPGAs are more versatile and flexible due to their nature. They are 'field-programmable'- you can reprogram the FPGA to perform any logic task that can be accommodated within the available logic gates. The logic gates can be rewired as many times as required to change the program and carry out a different task.
5	Microcontrollers read through each line of the program sequentially; that means the commands are also processed in sequence.	FPGAs can process orders simultaneously and can execute numerous lines of codes at a given moment. They are also wired just like an electric circuit so that you can get suitable parallel circuits.
6	However, in microcontrollers, the processor switches from one code to another to achieve some level of parallelism. You will find it easier to write the following codes on microcontrollers than FPGAs.	The parallel processing capability of FPGAs enables you to control interruptions effectively by using Finite State Machines (FSMs). In the case of microcontrollers, you have to account for the time taken by ISR to resolve an interruption.
7	Microcontrollers can execute a program and carry out general tasks. If we want to change the instruction set of the board, then we will need to modify the layout of the silicon IC.	We can rewire an FPGA easily just by reprogramming it. The configuration in an FPGA is loaded on the configurable logic cells when the power is switched on. You don't need to make any changes in the hardware to reprogram the FPGA.
8	Microcontrollers are simple to use and configure and can handle high-speed sequential data. However, they lack some of the features of FPGAs. Microcontrollers are more flexible when it comes to programming but cannot be customized accurately through hardware.	FPGAs are suitable for high-speed processing of parallel data and come with a high degree of customizability. However, they also have the drawbacks of prototype operation and complexity of configuration.
9	For microcontrollers, we can find ready-made packages to perform specific tasks and customize them to suit your needs.	It takes time to create an FPGA function as we need to compile all the codes from scratch and then convert them into machine language.

VYORIUS

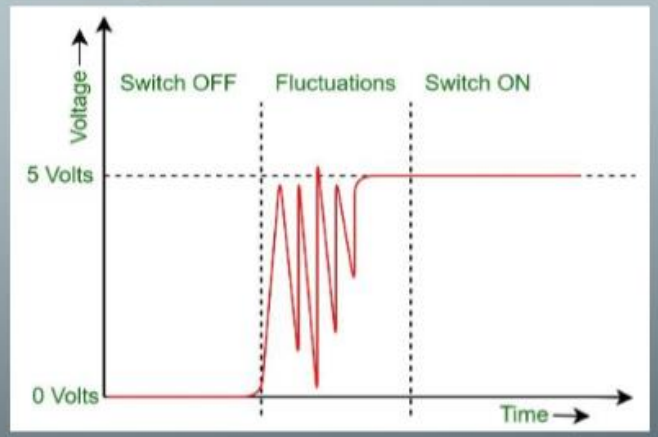
FPGA vs ASIC

No.	FPGA	ASIC
1	Reconfigurable circuit. FPGAs can be reconfigured with a different design. They even have capability to reconfigure a part of chip while remaining areas of chip are still working! This feature is widely used in accelerated computing in data centres.	Permanent circuitry. Once the application specific circuit is taped-out into silicon, it cannot be changed. The circuit will work same for its complete operating life.
2	Design is specified generally using hardware description languages (HDL) such as VHDL or Verilog.	Same as for FPGA. Design is specified using HDL such as Verilog, VHDL etc.
3	Easier entry-barrier. One can get started with FPGA development for as low as USD $30.	Very high entry-barrier in terms of cost, learning curve, liaising with semiconductor foundry etc. Starting ASIC development from scratch can cost well into millions of dollars.
4	Not suited for very high-volume mass production.	Suited for very high-volume mass production.
5	Less energy efficient, requires more power for same function which ASIC can achieve at lower power.	Much more power efficient than FPGAs. Power consumption of ASICs can be very minutely controlled and optimized.
6	Limited in operating frequency compared to ASIC of similar process node. The routing and configurable logic eat up timing margin in FPGAs.	ASIC fabricated using the same process node can run at much higher frequency than FPGAs since its circuit is optimized for its specific function.
7	Analog designs are not possible with FPGAs. Although FPGAs may contain specific analog hardware such as PLLs, ADC etc, they are not much flexible to create for example RF transceivers.	ASICs can have complete analog circuitry, for example WiFi transceiver, on the same die along with microprocessor cores. This is the advantage which FPGAs lack.
8	FPGAs are highly suited for applications such as Radars, Cell Phone Base Stations etc where the current design might need to be upgraded to use better algorithm or to a better design. In these applications, the high-cost of FPGAs is not the deciding factor. Instead, programmability is the deciding factor.	ASICs are definitely not suited for application areas where the design might need to be upgraded frequently or once-in-a-while.
9	Preferred for prototyping and validating a design or concept. Many ASICs are prototyped using FPGAs themselves! Major processor manufacturers themselves use FPGAs to validate their System-on-Chips (SoCs). It is easier to make sure design is working correctly as intended using FPGA prototyping.	It is not recommended to prototype a design using ASICs unless it has been absolutely validated. Once the silicon has been taped out, almost nothing can be done to fix a design bug (exceptions apply).
10	FPGA designers generally do not need to care for back-end design. Everything is handled by synthesis and routing tools which make sure the design works as described in the RTL code and meets timing. So, designers can focus into getting the RTL design done.	ASIC designers need to care for everything from RTL down to reset tree, clock tree, physical layout and routing, process node, manufacturing constraints (DFM), testing constraints (DFT) etc. Generally, each of the mentioned area is handled by different specialist person.

VYORIUS

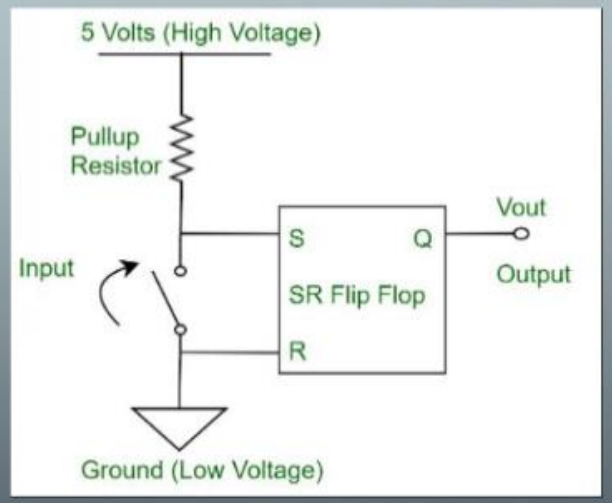
Switch Debounce

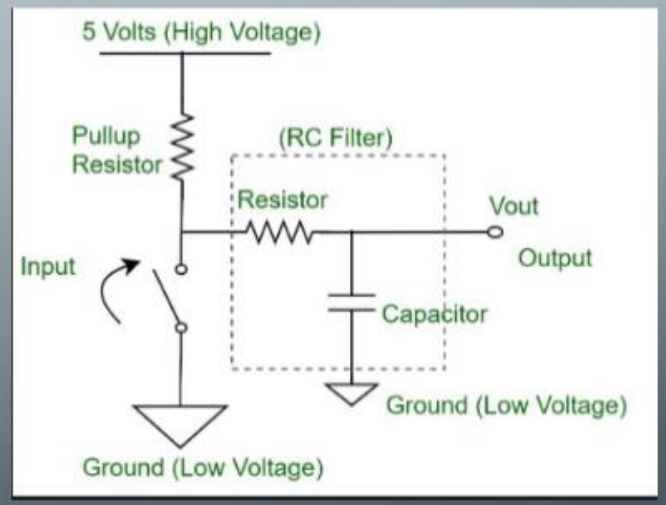
- Switch Debouncing

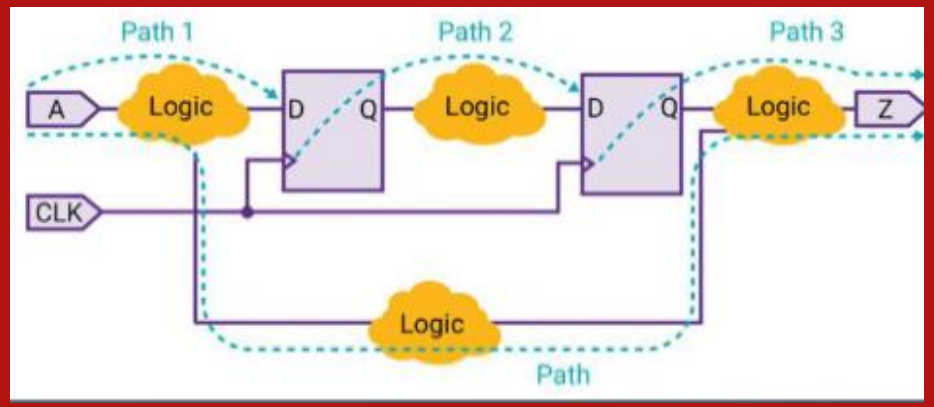


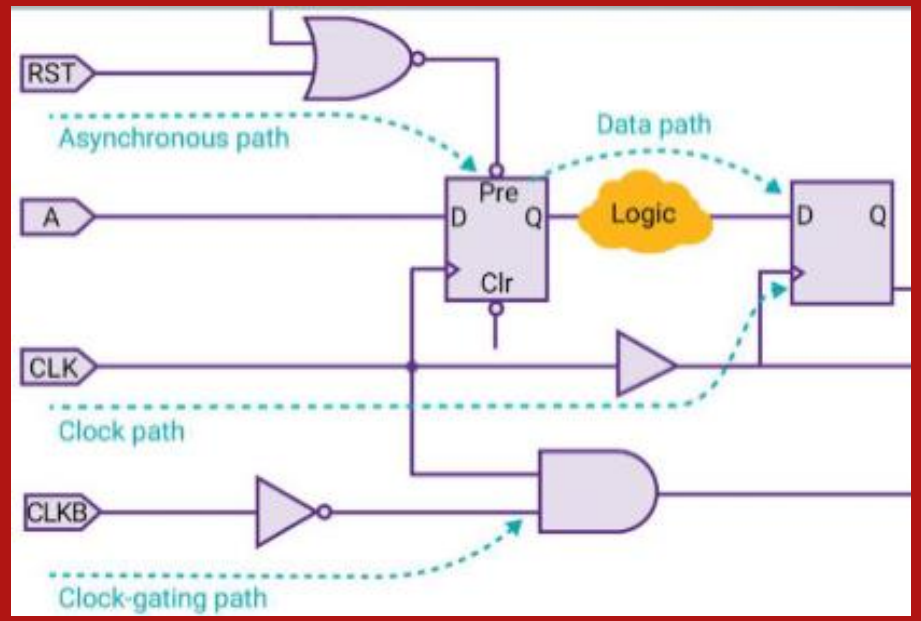
- Software Switch Debouncing

- Hardware Switch Debouncing







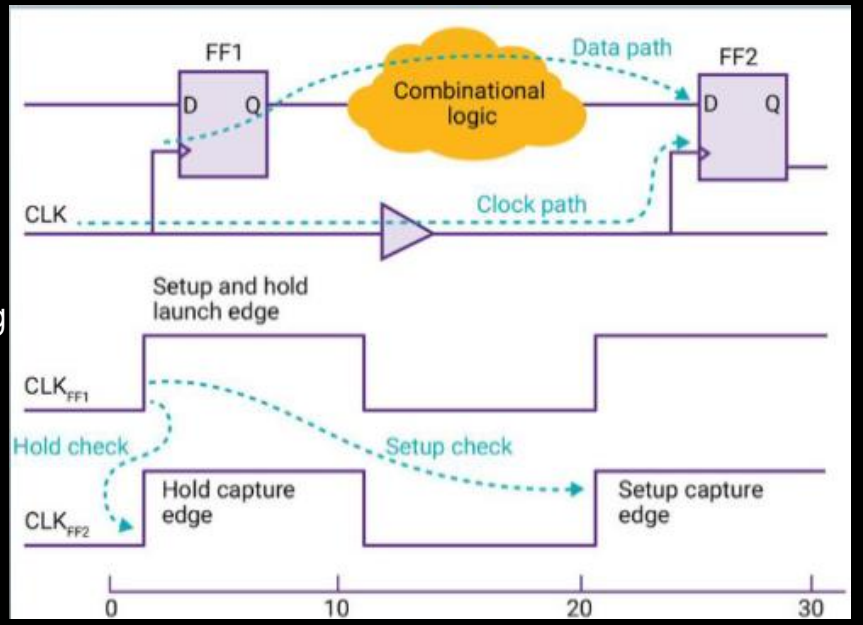


Timing Analysis

- Design is functional by verification methods and to make sure that it will behave correctly after manufacturing by timing analysis.
- Two ways of Timing Analysis
- Static Timing Analysis
- Dynamic Timing Analysis

Timing Analysis

- Static Timing Analysis
- Each timing path consists:
- Start-point
- Combinational logic network
- End-pointTypes of paths for timing analysis
- Clock path
- Clock-gating path
- Asynchronous path



Timing constraints

- A setup constraint specifies how much time is necessary for data to be available at the input of a sequential device before the clock edge that captures the data in the device.
- A hold constraint specifies how much time is necessary for data to be stable at the input of a sequential device after the clock edge that captures the data in the device.

Types of exceptions

- False path
- Multicycle path
- Minimum or maximum delay path

Metastability

- Unstable equilibrium in digital electronics in which the sequential element is not able to resolve the state of the input signal.
- It mostly occurs when data transitions very close to active edge of the clock, hence, violating setup and hold requirements.

PROJECTS

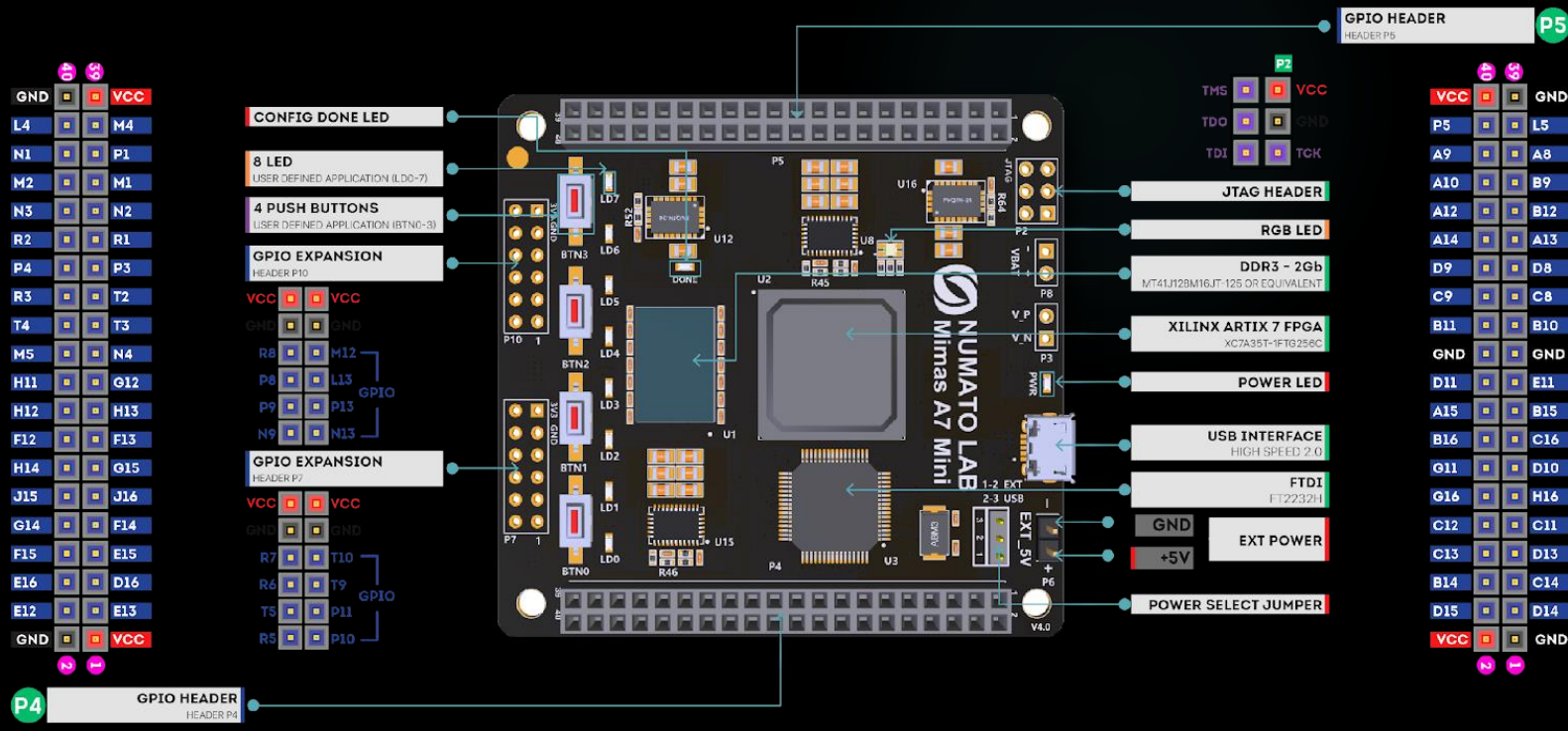
Project topics

1. Basic circuits written in Verilog/VHDL, simulated and implemented on the FPGA
2. UART communication to print a single character
3. FSM Designs: Mealy & Moore Machines and Up Down Counter
4. 16-Bit RISC Processor
5. Digital Design

VYORIUS

PROJECTS

VYORIUS



PROJECTS

VYORIUS

Hardware,

Mimas A7 Mini FPGA Development BoardMimas A7 Mini is an easy-to-use FPGA Development board featuring Artix 7 FPGA (XC7A35T – FTG256C package) with FTDI's FT2232H Dual-Channel USB device. It is an Artix-7 based replacement and upgrades of Mimas Spartan 6 FPGA Board. It is specially designed for the development and integration of FPGA based accelerated features to other designs. The USB 2.0 host interface based on popular FT2232H offers high bandwidth data transfer and board programming without the need for any external programming adapters.

Features,

- Device: Xilinx Artix 7 FPGA (XC7A35T-1FTG256C)
- DDR3: 2Gb DDR3 (MT41J128M16JT-125 or equivalent)
- Built-in programming interface. No expensive JTAG adapters needed for programming the board
- Onboard 128Mb flash memory for FPGA configuration storage and custom user data storage
- High-Speed USB 2.0 interface for On-board flash programming. FT2232H Channel B is dedicated to JTAG Programming. Channel A can be used for custom applications
- 100MHz CMOS oscillator
- 8 LEDs, 1 RGB LED and 4 Push Buttons for user-defined purposes
- FPGA configuration via JTAG and USB
- Maximum IOs for user-defined purposes

FPGA – 70 IOs (35 professionally length matched Differential Pairs) and two 2×6 Expansion Headers

PROJECTS

Applications,

- Product Prototype Development
- Accelerated computing integration
- Development and testing of custom embedded processors
- Communication devices development
- Educational tool for Schools and Universities

VYORIUS

PROJECTS

VYORIUS

Attribute	Value
Dimensions	6 × 4 × 1 in
FPGA	XC7A35T – 1FTG256C
Memory	DDR3 – 2Gb
Configuration Options	JTAG, USB
Host Interface	USB 2.0
Primary Clock Frequency	100MHz
Number Of GPIOs (Max)	70
Number Of Diff Pairs	35

VYORIUS

PROJECTS

Software

Icarus Verilog is a Verilog simulation and synthesis tool. It operates as a compiler, compiling source code written in Verilog (IEEE-1364) into some target format. For batc simulation, the compiler can generate an intermediate form called *vvp assembly*. Thi intermediate form is executed by the ``vvp'' command. For synthesis, the compiler generates netlists in the desired format.

The compiler proper is intended to parse and elaborate design descriptions written to the IEEE standard *IEEE Std 1364-2005*. This is a fairly large and complex standard, so it v take some time to fill all the dark alleys of the standard, but that's the goal.

Icarus Verilog is a work in progress, and since the language standard is not standing s either, it probably always will be. That is as it should be. However, I will make stable releases from time to time, and will endeavour to not retract any features that appea in these stable releases. The quick links above will show the current stable release.

The main porting target is Linux, although it works well on many similar operating systems. Various people have contributed precompiled binaries of stable releases for variety of targets. These releases are ported by volunteers, so what binaries are available depends on who takes the time to do the packaging. *Icarus Verilog* has been ported to That Other Operating System, as a command line tool, and there are installers for users without compilers. You can compile it entirely with free tools, too, although there are precompiled binaries of stable releases.

PROJECTS

Vivado Design Suite is a software suite produced by Xilinx for synthesis and analysis of HDL designs, superseding Xilinx ISE with additional features for system on a chip development and high-level synthesis.[1][5][6][7] Vivado represents a ground-up rewrite and re-thinking of the entire design flow (compared to ISE).[8][9][10] Like the later versions of ISE, Vivado includes the in-built logic simulator ISIM.[11] Vivado also introduces high-level synthesis, with a toolchain that converts C code into programmable logic.[6] Replacing the 15-year-old ISE with Vivado Design Suite took 1000 person-years and cost US$200 million.

VYORIUS

PROJECTS

PuTTY is a free implementation of SSH **(and telnet) for PCs running Microsoft Windows** (it also includes an xterm terminal emulator). You will find PuTTY useful if you want to access an account on a Unix or other multi-user system from a PC (for example your own or one in an internet cafe).

VYORIUS

PROJECTS

VYORIUS

Tenagra is an FPGA System management tool for configuring and communicating with Numato Lab's supported FPGA modules and development platforms. This software is designed to be a single interface for managing the devices and exercising some of the available features. Currently, Tenagra supports configuring the FPGA module/board (programming) and Memory Exerciser that can transfer data between various memories on the device. This includes both external DDR Memory and Block RAM available within the FPGA device. With Tenagra, you can create multiple configuration setups with different bitstreams and settings for each device model so that switching between multiple bitstreams is a breeze. This is especially helpful during development where the device may need to be reprogrammed with various bitstreams repeatedly.

Driver for Mimas A7 Mini Module

NUMATO LAB®

PROJECTS

Applications,

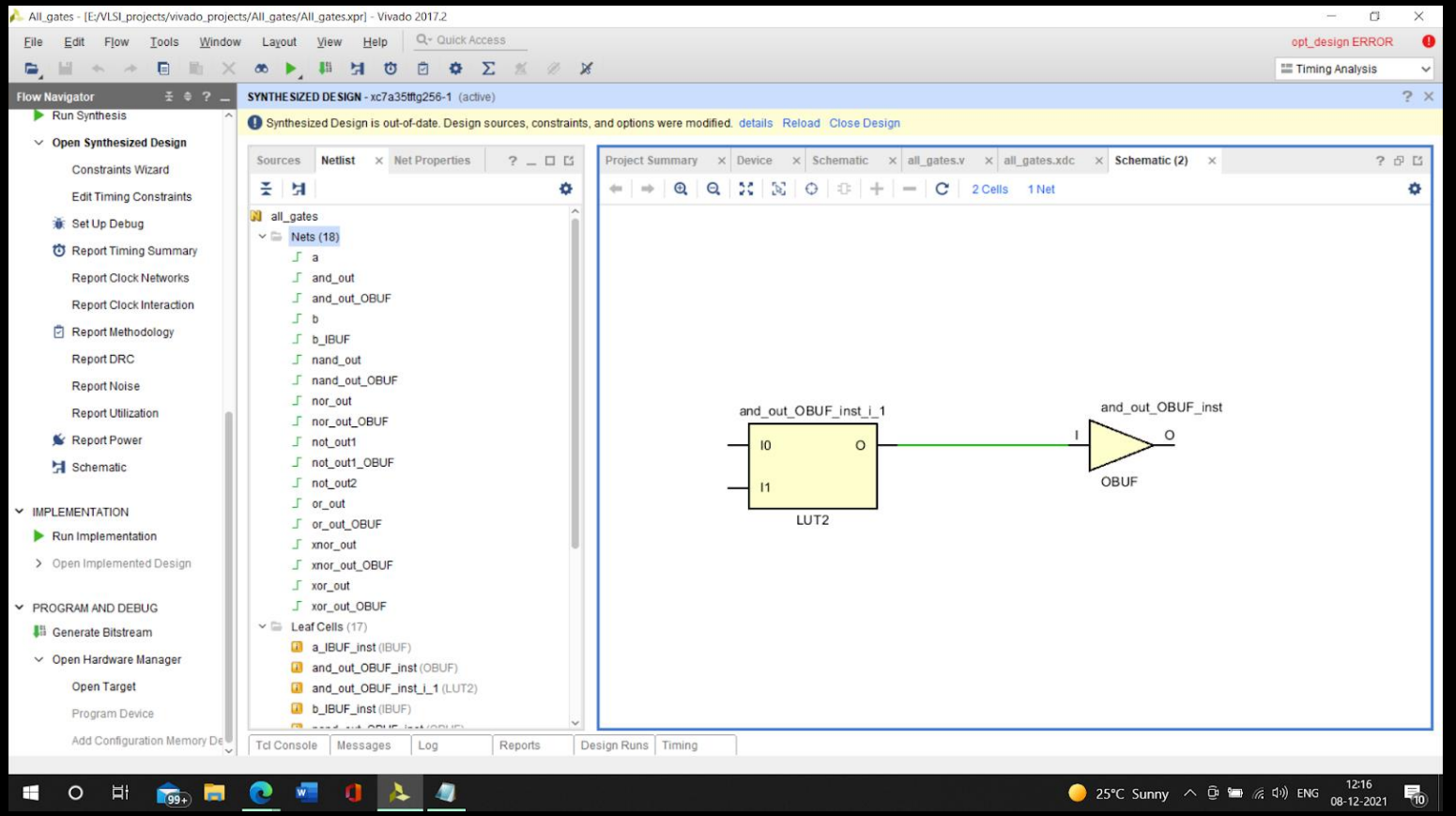
- Product Prototype Development
- Accelerated computing integration
- Development and testing of custom embedded processors
- Communication devices development
- Educational tool for Schools and Universities

VYORIUS

PROJECTS

VYORIUS

Basic circuits written in Verilog/VHDL, simulated and implemented on the FPGA

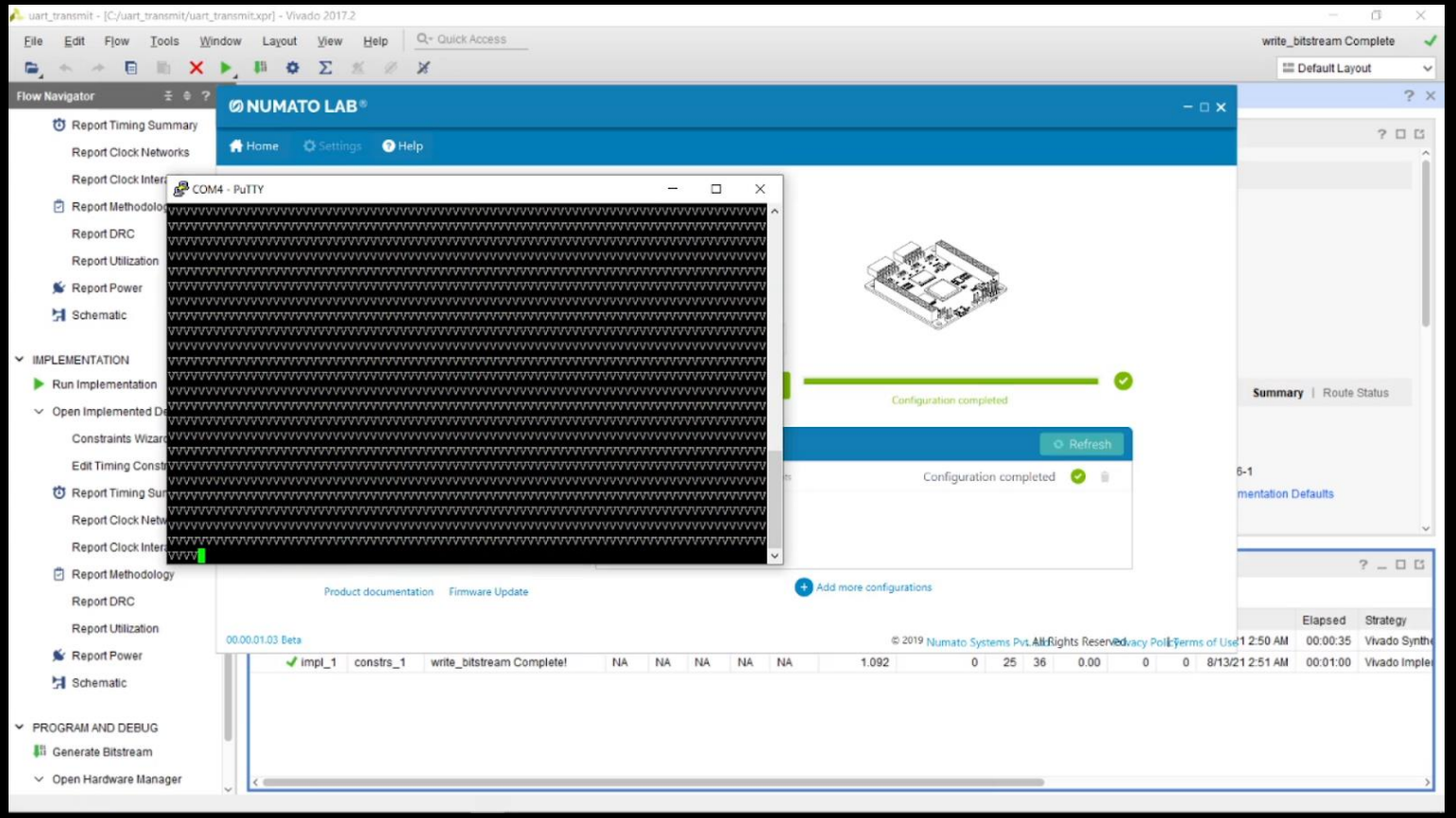


https://drive.google.com/drive/folders/1Elw4VOfRVkyZeTF3mHgWDLpq_YegFnWs?usp=sharing

PROJECTS

VYORIUS

UART communication to print a single character

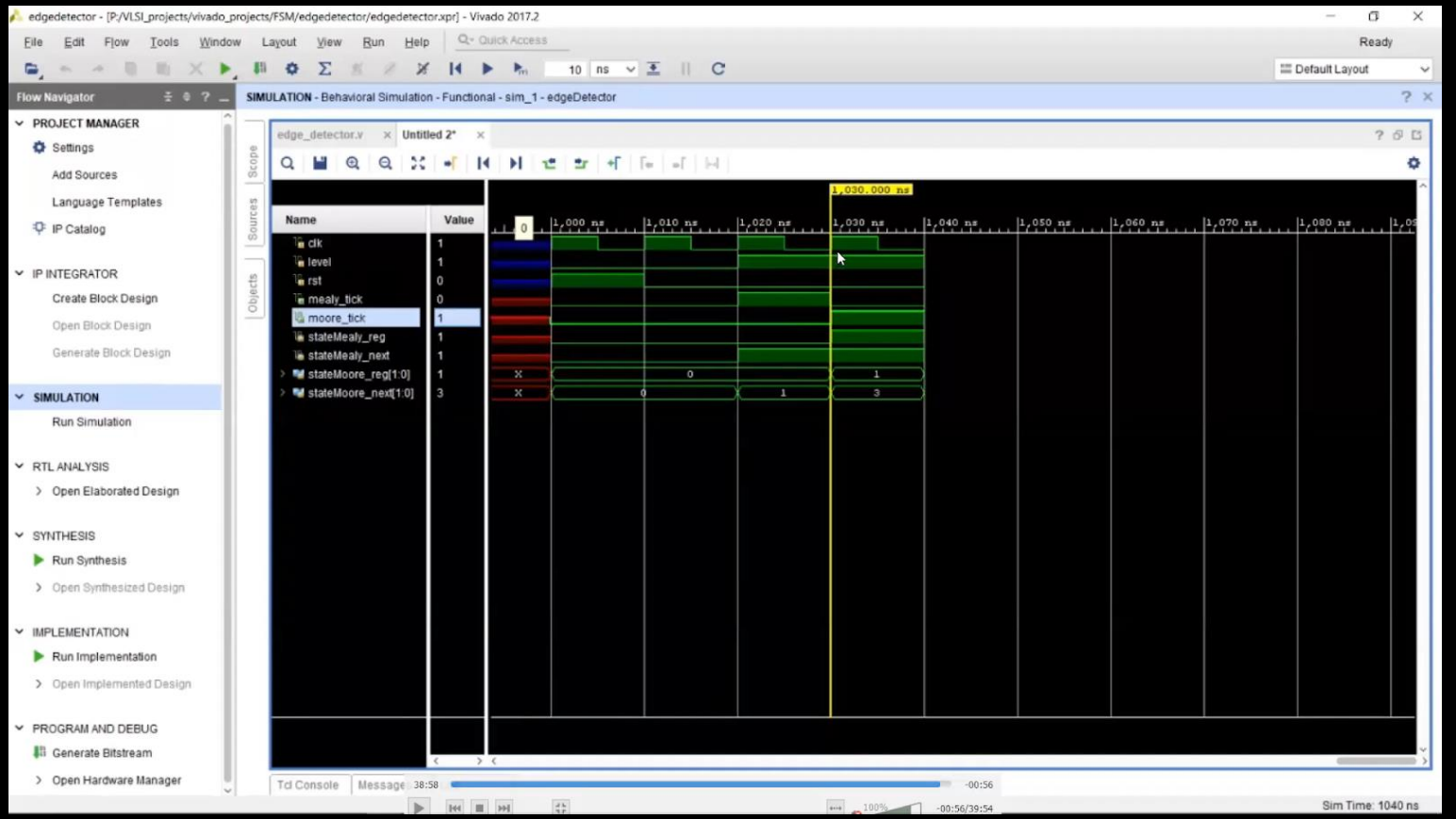


https://drive.google.com/drive/folders/1-eYAfOznCoWypjXwjG1oQEQS2ppcQ64E?usp=sharing

PROJECTS

VYORIUS

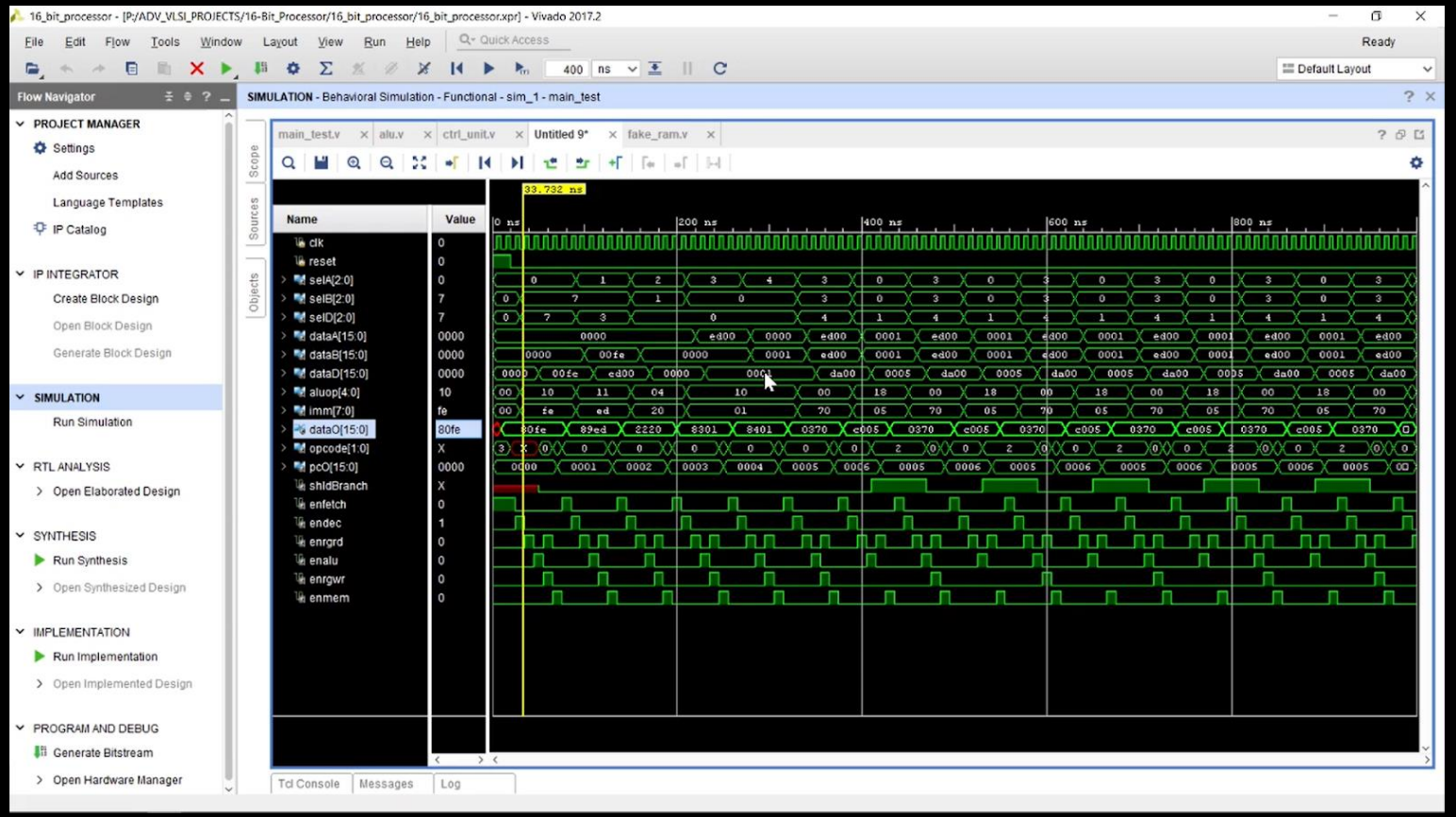
FSM Designs: Mealy & Moore Machines and Up Down Counter



https://drive.google.com/drive/folders/191synVBEn5rxYBt9rDZpA5nwu7pD7N6W?usp=sharing

PROJECTS

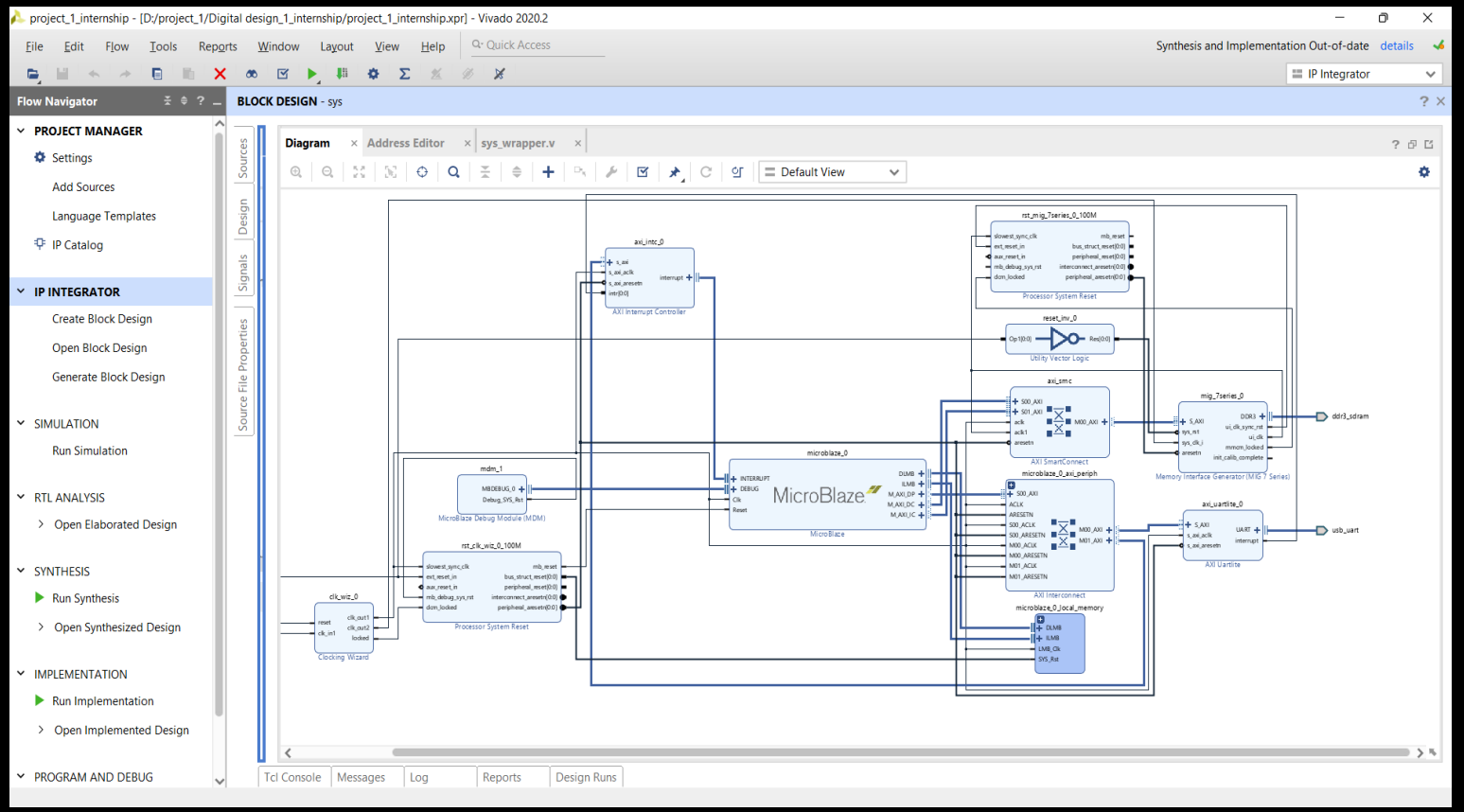
16-Bit RISC Processor



https://drive.google.com/drive/folders/1NG-HMSccV27dC0s1sc-uwBqW2Yt0C77z?usp=sharing

PROJECTS

Digital Design



https://drive.google.com/drive/folders/1j1CQQcNiZkKO7DBlcB5UgeO1-nQJEAlK?usp=sharing

CERTIFICATE

VYORIUS