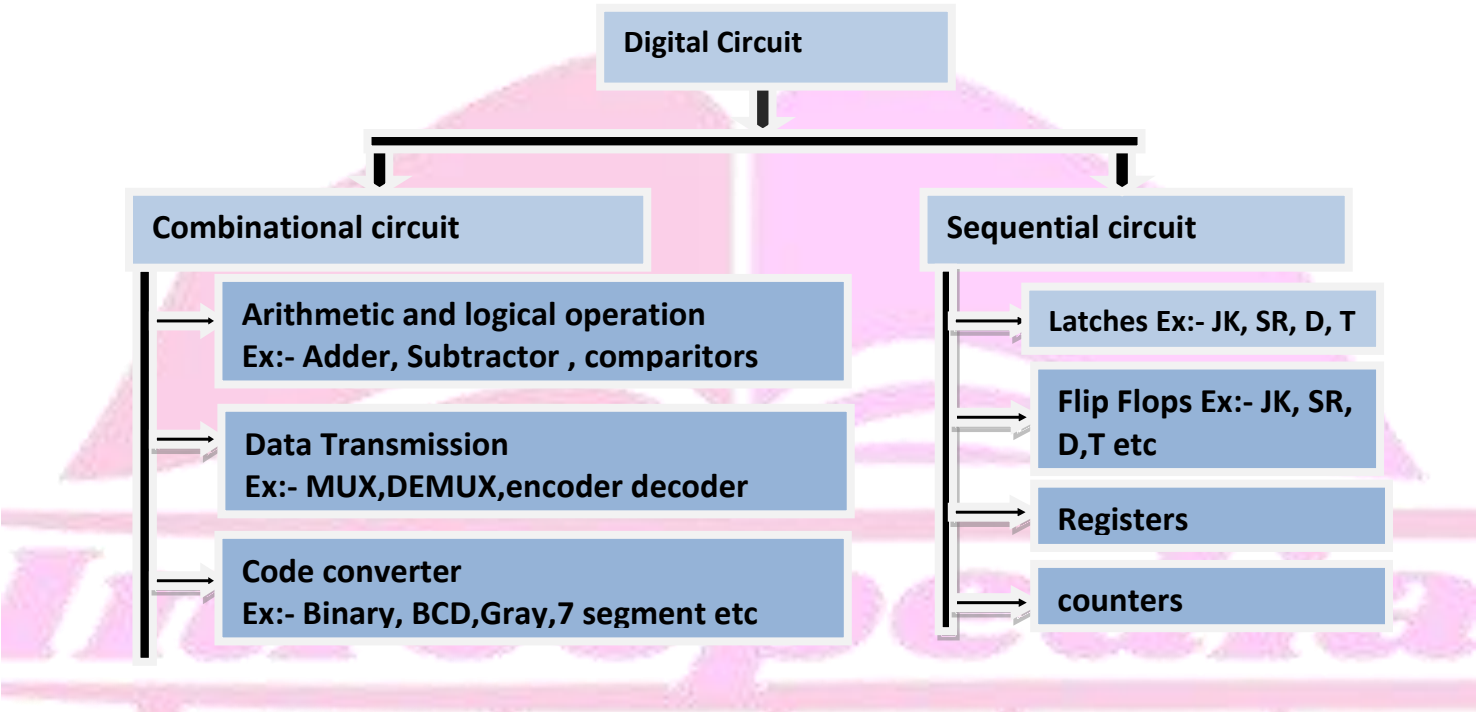
Combinational Circuit

- Digital circuit is made by using logic gates.



There are two types of logic circuits

1. **Combinational Circuits**

- Formed by combination of different logic circuits
- Outputs depend on the current inputs
- Ex- Adder, Multiplexer, De multiplexer, Encoder, Decoder
- No memory allocation required

2. **Sequential Circuits**

- Outputs depend on current and previous inputs
- Requires separating previous, current and future states or tokens.
- Example: Finite State Machines (FSMs), Pipelines, Latch,Flip flop etc.

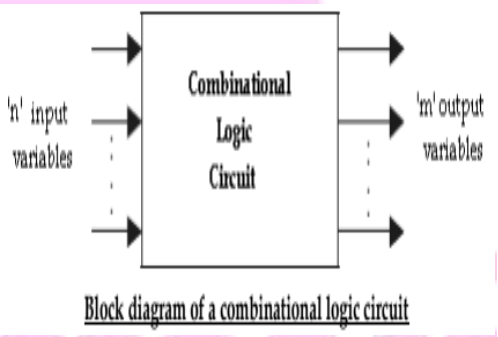
Difference between Combinational and sequential circuits:

SNO	Combinational circuits	Sequential circuits
1	Output of any instance of time depends only upon the present input variables.	Output is generated dependent upon the present input variables and also on the basis of past history of these inputs.
2	Memory unit is not required. i.e. it doesn't allocate any memory to the elements.	Memory unit is required. i.e. it allocates any memory to the elements.
3	Faster	Slower
4	Easy to design	Difficult
5	Implemented using: logic gates	Implemented using flip flops

6	Used to perform arithmetic as well as boolean operations.	used to store data.
7	don't have capability to store any state.	capability to store any state or to retain earlier state.
8	don't have clock, they don't require triggering.	clock dependent they need triggering.

Combinational circuit

- Combination Logic Circuits are made up from basic gates (AND, OR, NOT) or universal gates (NAND, NOR) gates that are "combined" or connected together to produce more complicated switching circuits.
- These logic gates are the building blocks of combinational logic circuits.
- Examples for combinational digital circuits are Half adder, Full adder, Half subtractor, Full subtractor, Code converter, Decoder, Multiplexer, Demultiplexer, Encoder, ROM, etc.
- For n number of input variables to a combinational circuit, 2^n possible combinations of binary input states are possible.
- For each possible combination, there is one and only one possible output combination.
- A combinational logic circuit can be described by m Boolean functions and each output can be expressed in terms of n input variables.



Block diagram of a combinational logic circuit

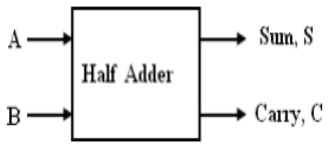
Adders

There are some following types of adder:-

1. Half adder
2. Full adder
3. n-bit parallel adder
4. Look ahead carry adder

Half-Adder

- A half-adder is a combinational circuit that can be used to add two binary bits.
- It has two inputs that represent the two bits to be added and two outputs, with one producing the SUM output and the other producing the CARRY.



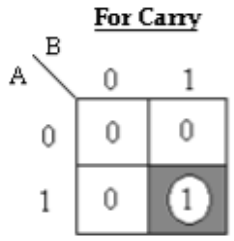
Block schematic of half-adder

The truth table of a half-adder:

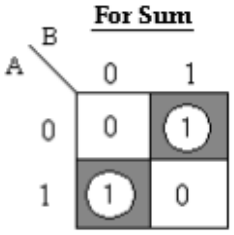
Inputs		Outputs	
A	B	Carry (C)	Sum (S)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Truth table of half-adder

- K-map simplification for carry and sum:
- The Boolean expressions for the SUM and CARRY
  S = A'B+ AB'= A ⊕ B
  C = A . B

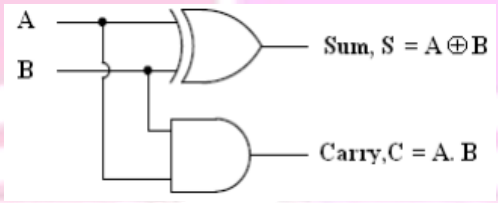


Carry = A.B



Sum = AB'+ A'B
= A⊕B

The logic diagram of the half adder is:

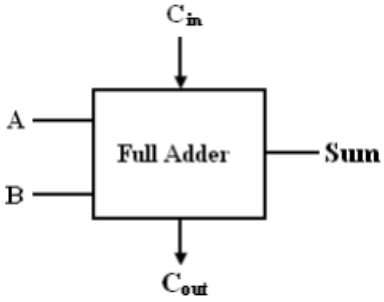


Drawback of half adder:-

Half adder does not handle and do processing on carries.

Full-Adder

- The full adder circuit overcomes the limitation of the half-adder.
- Full Adder is a combinational circuit that performs the addition of three bits (two significant bits and previous carry).
- It consists of three inputs and two outputs, two inputs are the bits to be added, the third input represents the carry form the previous position.
- Full adder circuit adds three bit binary numbers (X,Y,Z) & outputs two numbers of one bit binary numbers, Sum & Carry.



Block schematic of full-adder

The block diagram of full adder:

There are three input variables, eight different input combinations are possible. The truth table is shown below:

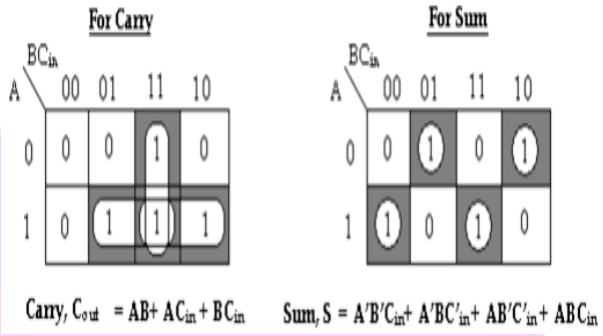
Inputs			Outputs	
A	B	C_{in}	Sum (S)	Carry (C_{out})
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

The Karnaugh map:-

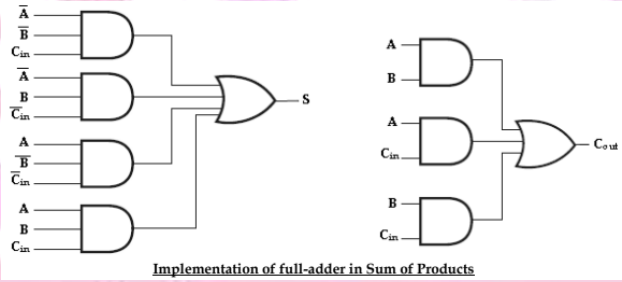
The Boolean expressions for the SUM and CARRY outputs Sum,

S = $A'B'C_{in} + A'BC'_{in} + AB'C'_{in} + ABC_{in} = C_{in} \oplus (A \oplus B)$

$\mathbf{C_{out}} = AB + AC_{in} + BC_{in}$



The logic diagram for the Full adder sum and carry:-

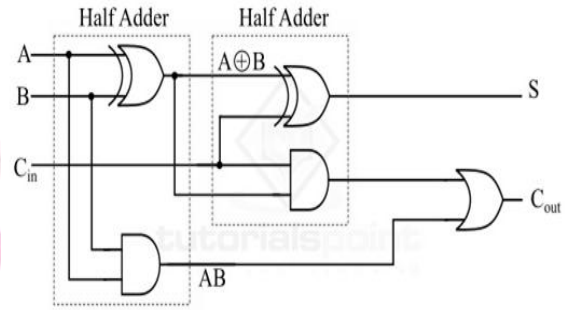


Note:- The logic diagram of the full adder can also be implemented with two half adders and one OR gate.

Full Adder using Half Adder

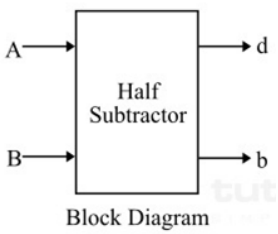
$S = C_{in} \oplus (A \oplus B)$

$C = AB + C_{in}(A \oplus B)$



Half Subtractor

- It is used for the purpose of subtracting two single bit numbers.
- It contains 2 inputs and 2 outputs (difference and borrow).
- It produces the difference between the two binary bits at the input and also produces a borrow output (if any).
- In the subtraction (A-B), A is called as Minuend bit and B is called as Subtrahend bit.



Truth Table of Half Subtractor:-

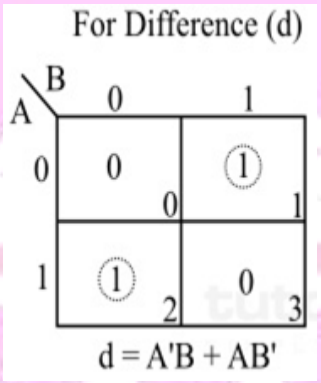
Inputs		Outputs	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

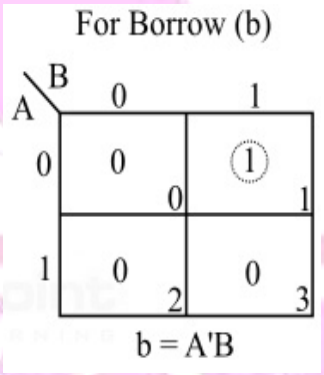
K-Map for Half Subtractor:-

Boolean expression for difference and borrow:-

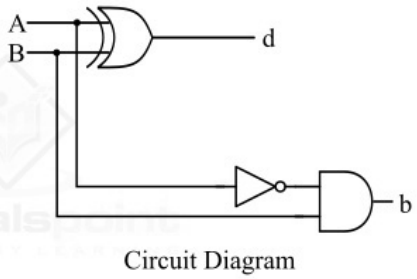
d = A'B+ AB'= A ⊕ B

b = A' . B





Logic circuit diagram of the half subtractor:-



Circuit Diagram

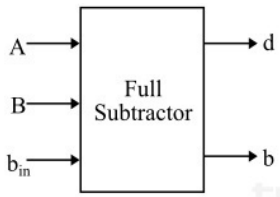
Drawback of half Subtractor

Half subtractors do not take into account "Borrow-in" from the previous circuit. and are not suitable for more complex arithmetic operations.

Full Subtractor

A full-subtractor is a combinational circuit that has three inputs A, B, b_{in} and two outputs d and b.

Where, A is the minuend, B is subtrahend, b_{in} is borrow produced by the previous stage, d is the difference output and b is the borrow output.



Block Diagram

Truth Table for full subtractor:-

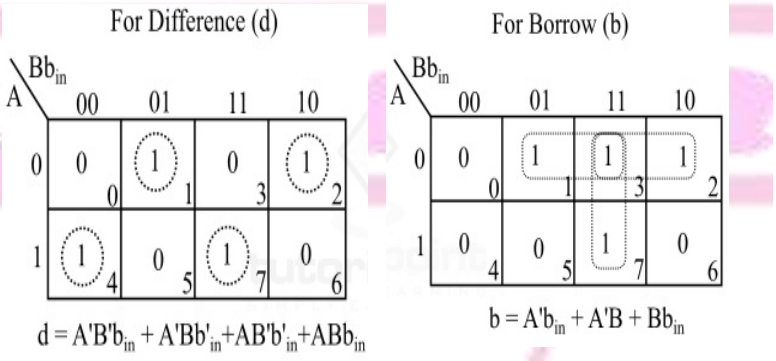
Inputs			Outputs	
A	B	$Borrow_{in}$	Diff	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

K-Map for full Subtractor:-

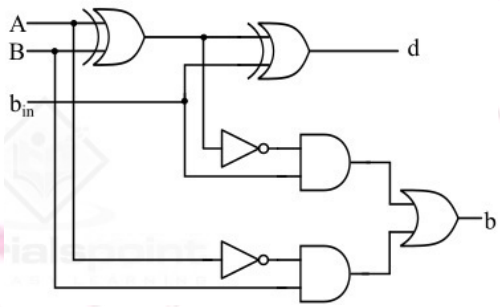
Boolean expression for difference and borrow:-

Difference d=$A \oplus B \oplus b_{in} = A'B'b_{in} + AB'b'_{in} + A'Bb'_{in} + ABb_{in}$

Borrow b=$A'B + (A \oplus B)'b_{in}$

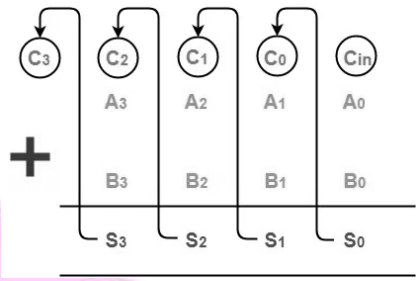


Logic circuit diagram of the full subtractor:-



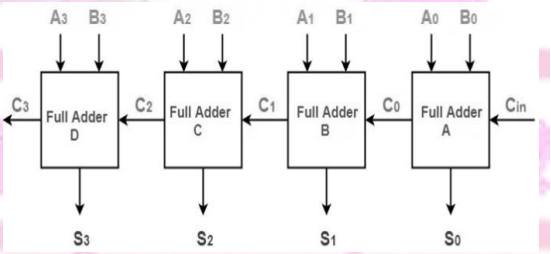
Note:- Full Subtractor can be implemented by using 2 Half Subtractors and an OR gate.

n-bit parallel adder



- It is also called ripple carry adder.
- It is used for the purpose of adding two n-bit binary numbers.
- It requires n full adders in its circuit for adding two n-bit binary numbers.
- Here we will discuss a 4 bit parallel adder to add 4 bit binary number.
- A 4-bit parallel adder consists of four full adders connected in parallel. Each full adder takes in two input bits, along with a carry input from the previous full adder.
- It produces a sum output bit and a carry output bit.

Disadvantages of parallel Adder:-

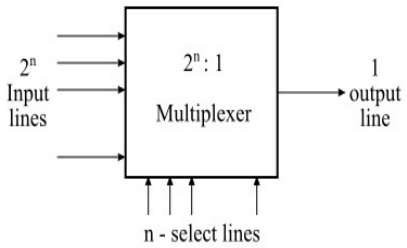


- For n bit addition we need n full adder leads a complex circuit.
- N bit parallel adder does not allow to use all the full adders simultaneously.
- The carry propagation delay (delay associated with the travelling of carry bit) increases as circuit grows.
- After designing a complex circuit only addition is performed not subtraction.

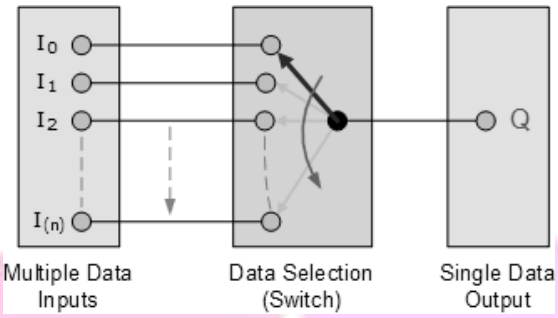
Carry lookahead adder

- In the n bit parallel adder each full adder has to wait for its carry-in from its previous stage full adder.
- Thus, n^{th} full adder has to wait until all (n-1) full adders have completed their operations.
- This causes a delay and makes n-bit parallel adder extremely slow.
- To overcome this disadvantage, we can use Carry Look Ahead Adder.
- Carry Look Ahead Adder is an improved version of the n bit parallel adder.
- It generates the carry-in of each full adder simultaneously without causing any delay.
- The time complexity of carry look ahead adder = Θ (logn).
- Here a carry signal will be generated in two cases:
  - ✓ Input bits A and B are 1
  - ✓ When one of the two bits is 1 and the carry-in is 1.

Multiplexer or MUX



- It accept multiple data inputs and produce single output depending on control or select inputs.
- Here the input combination and selection lines are dependent to each other.
- If there are n selection lines then the input lines will be maximum 2^n.
- And if there are m input lines the $\log_2 m$ selection lines.
- Multiplexers are mainly used to increase amount of the data that can be sent over the network within certain amount of time and bandwidth.
- Multiplexers are data n selector, parallel to serial convertor, many to one circuit etc.

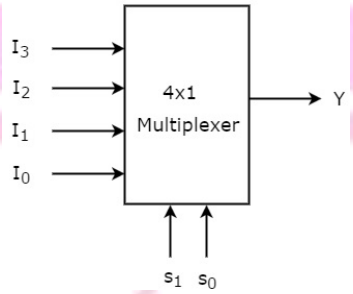


Types of Multiplexer

1. 2-to-1 MUX: Selects 1 out of 2 inputs.
2. 4-to-1 MUX: Selects 1 out of 4 inputs.
3. 8-to-1 MUX: Selects 1 out of 8 inputs.
4. 16-to-1 MUX: Selects 1 out of 16 inputs

4x1 Multiplexer

4x1 Multiplexer has four data inputs I_3, I_2, I_1 & I_0, two selection lines s_1 & s0 and one output Y.
The block diagram of 4x1 Multiplexer:



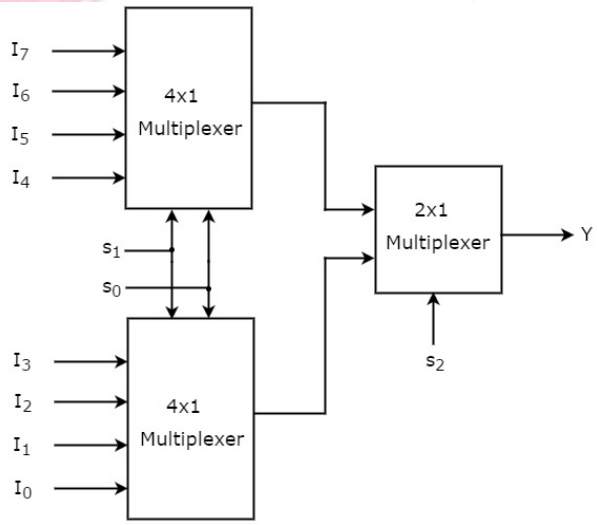
One of these 4 inputs will be connected to the output based on the combination of inputs present at these two selection lines.

Truth table of 4x1 Multiplexer

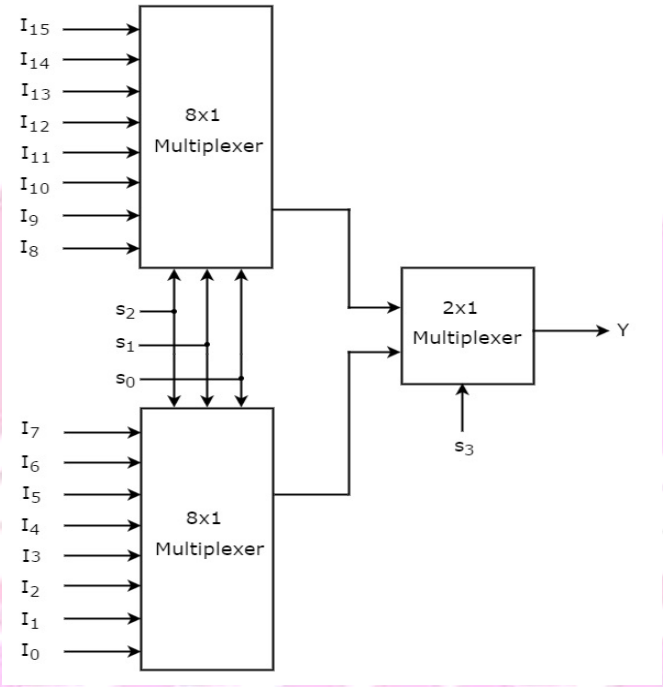
Selection Lines		Output
S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

The Boolean function for output
$Y=S_1'S_0'I_0+S_1'S_0I_1+S_1S_0'I_2+S_1S_0I_3$

The block diagram of 8x1 Multiplexer using 4 x 1 multiplexer



The block diagram of 16x1 Multiplexer using 8 x 1 multiplexer:

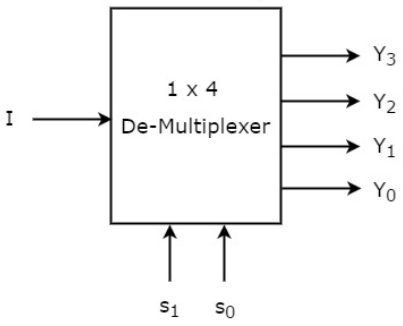


Applications of MUX

- Implementation of Digital Circuits
- Control Unit of CPU
- Parallel to Serial Data Conversion
- Computer Memory
- Data Routing
- Analog to Digital Conversion
- **Signal Selection**: In audio and video systems, MUX is used to select different input signals (e.g., different channels or inputs in a TV).

Demultiplexer or DEMUX

- A De-multiplexer is a combinational circuit that has only 1 input line and maximum 2^N output lines.
- The input will be connected to one of these outputs based on the values of selection lines.



Types of Demultiplexer:

1. 1-to-2 DEMUX: Routes 1 input to 1 of 2 outputs.
2. 1-to-4 DEMUX: Routes 1 input to 1 of 4 outputs.
3. 1-to-8 DEMUX: Routes 1 input to 1 of 8 outputs.
4. 1-to-16 DEMUX: Routes 1 input to 1 of 16 outputs.

The Truth table of 1x4 De-Multiplexer

The Boolean functions for each output as

$Y3=S_1S_0I$

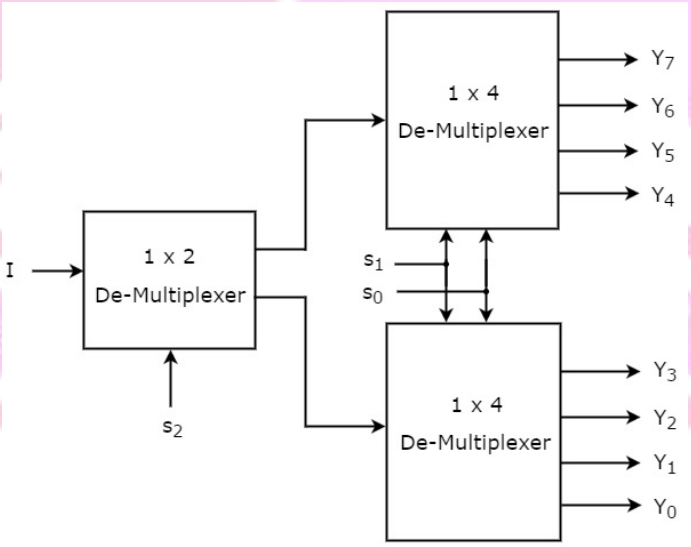
$Y2= S_1S_0' I$

$Y1= S_1' S_0 I$

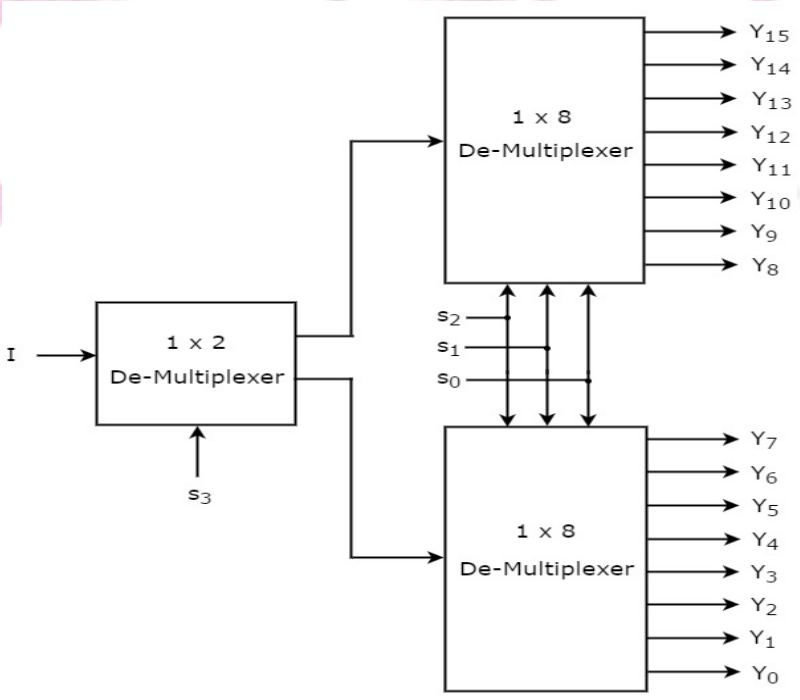
$Y0= S_1' S_0' I$

Selection Inputs		Outputs			
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 X 8 Demultiplexer:



1x16 DeMultiplexer:



Applications of Demultiplexer

- Used in several input and output devices for data routing.
- Used in digital control systems to select one signal from a mutual stream of signals.
- Data transmission in synchronous systems.
- Serial to parallel converters.
- Design automatic test equipment.

Comparison between multiplexer and Demultiplexer

S.No.	Multiplexer	Demultiplexer
1	Selects one of many inputs to pass through to a single output.	Takes a single input and routes it to one of many outputs.
2	Uses control lines to select which input is connected to the output.	Uses control lines to select which output receives the input signal
3	Often used in situations where data needs to be selected or merged from multiple sources.	Used when a single data source needs to be distributed to multiple destinations.

Encoder

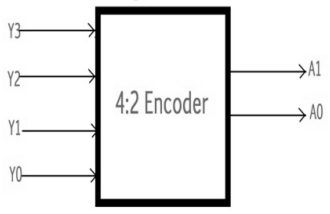
It converts a set of binary inputs into a unique binary code.
It has a maximum of 2^n input lines and 'n' output lines, hence it encodes the information from 2^n inputs into an n-bit code. It will produce a binary code equivalent to the input, which is active High.

Types of Encoders

- **Binary Encoder**: Converts a set of 2^n input lines into an n-bit output code.
- **Priority Encoder**: Similar to a binary encoder but assigns priority to inputs if multiple inputs are active simultaneously.
- **Decimal to BCD Encoder**: Converts decimal input (0-9) into binary-coded decimal (BCD).

4 to 2 Encoder

The 4 to 2 Encoder consists of four inputs Y3, Y2, Y1 & Y0, and two outputs A1 & A0. At any time, only one of these 4 inputs can be '1' in order to get the respective binary code at the output.



The Truth table of 4 to 2 encoders

Logical expression for A1 and A0:

A1 = Y3 + Y2
A0 = Y3 + Y1

INPUTS				OUTPUTS	
Y3	Y2	Y1	Y0	A1	A0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Priority Encoder

- Encoder generate some errors:-
  - ✓ There is an ambiguity, when all outputs of the encoder are equal to zero.
  - ✓ If more than one input is active High, then the encoder produces an output, which may not be the correct code.
- So, to overcome these errors, we should assign priorities to each input of the encoder. Then, the output of the encoder will be the code corresponding to the active high inputs, which have higher priority.
- A 4 to 2 priority encoder has 4 inputs: Y3, Y2, Y1 & Y0, and 2 outputs: A1 & A0.
- Here, the input, Y3 has the highest priority, whereas the input, Y0 has the lowest priority.
- In this case, even if more than one input is '1' at the same time, the output will be the (binary) code corresponding to the input, which is having higher priority.

Application of Encoders

1. **Keyboards:** Encoders convert the keypresses into binary codes for the computer to process.
2. **Multiplexing:** Used in multiplexers to select inputs for processing.
3. **Robotics:** For encoding signals in robotic systems.
4. **Digital Signal Processing (DSP):** In DSP systems, encoders are used to convert analog signals into digital codes for further processing, filtering, and analysis.

Decoder

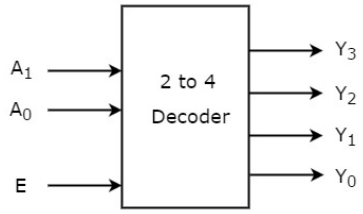
- Decoder is a combinational circuit that has 'n' input lines and maximum of 2^n output lines.
- One of these outputs will be active High based on the combination of inputs present, when the decoder is enabled. Decoder detects a particular code.
- The outputs of the decoder are nothing but the min terms of 'n' input variables lines, when it is enabled.

Types of decoder

- **Binary Decoder:** Converts n-bit binary input into a 2^n output.
- **BCD to 7-Segment Decoder:** Converts BCD input into a format suitable for driving a 7-segment display.
- **Demultiplexer (Demux):** Acts as a decoder with a single input and multiple outputs.

2 to 4 Decoder

Let 2 to 4 Decoder has two inputs A_1 & A_0 and four outputs Y_3, Y_2, Y_1 & Y_0.



One of these four outputs will be '1' for each combination of inputs when enable, E is '1'.
The Truth table of 2 to 4 decoder:

Enable	Inputs		Outputs			
E	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

From Truth table, we can write the Boolean functions for each output as:
Y3=E.A1.A0
Y2=E.A1.A0'
Y1=E.A1'.A0
Y0=E.A1'.A0'
Each output is having one product term. So, there are four product terms in total.

Applications of decoders

- **Memory Addressing:** Used in memory chips to select specific memory locations.
- **Display Systems:** BCD to 7-segment decoders are used in digital displays to convert binary input into readable numeric output.
- **Demultiplexing:** Used in communication systems to route signals to multiple destinations.

Comparison between Encoder and Decoder

S.No.	Encoder	Decoder
1.	Converts multiple inputs into a smaller number of outputs (compression).	Expands a smaller number of inputs into multiple outputs (decompression)
2.	Encodes data for transmission or storage.	Decodes data for further processing or display.
3.	Often used in input devices, communication systems, and data compression.	Used in output devices, data retrieval systems, and signal routing.