LOGIC GATES

Introduction

Boolean functions may be practically implemented by using electronic gates. The following points are important to understand.

- Electronic gates require a power supply.
- Gate **INPUTS** are driven by voltages having two nominal values, e.g. 0V and 5V representing logic 0 and logic 1 respectively.
- The **OUTPUT** of a gate provides two nominal values of voltage only, e.g. 0V and 5V representing logic 0 and logic 1 respectively. In general, there is only one output to a logic gate except in some special cases.
- There is always a time delay between an input being applied and the output responding.

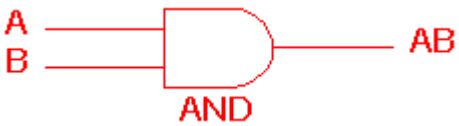
Truth Tables

Truth tables are used to help show the function of a logic gate.

Logic gates

Digital systems are said to be constructed by using logic gates. These gates are the AND, OR, NOT, NAND, NOR, EXOR and EXNOR gates. The basic operations are described below with the aid of truth tables.

AND gate



2 Input AND gate		
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

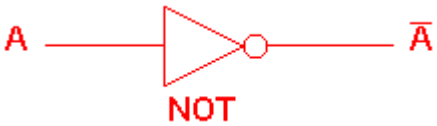
The AND gate is an electronic circuit that gives a **high** output (1) only if **all** its inputs are high. A dot (.) is used to show the AND operation i.e. A.B. Bear in mind that this dot is sometimes omitted i.e. AB

OR gate

2 Input OR gate		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

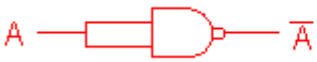
The OR gate is an electronic circuit that gives a high output (1) if **one or more** of its inputs are high. A plus (+) is used to show the OR operation.

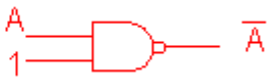
NOT gate



NOT gate	
A	$\overline{A}$
0	1
1	0

The NOT gate is an electronic circuit that produces an inverted version of the input at its output. It is also known as an *inverter*. If the input variable is A, the inverted output is known as NOT A. This is also shown as A', or A with a bar over the top, as shown at the outputs. The diagrams below show two ways that the NAND logic gate can be configured to produce a NOT gate. It can also be done using NOR logic gates in the same way.





NAND gate

2 Input NAND gate		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if **any** of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

NOR gate

2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate. The outputs of all NOR gates are low if **any** of the inputs are high.

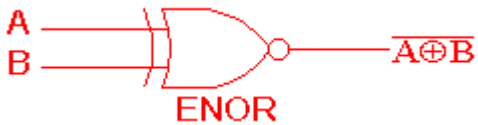
The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

EXOR gate

2 Input EXOR gate		
A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

The '**Exclusive-OR**' gate is a circuit which will give a high output if **either, but not both**, of its two inputs are high. An encircled plus sign (⊕) is used to show the EOR operation.

EXNOR gate



2 Input EXNOR gate		
A	B	$\overline{A⊕B}$
0	0	1
0	1	0
1	0	0
1	1	1

The '**Exclusive-NOR**' gate circuit does the opposite to the EOR gate. It will give a low output if **either, but not both**, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion.

De Morgan Theorem

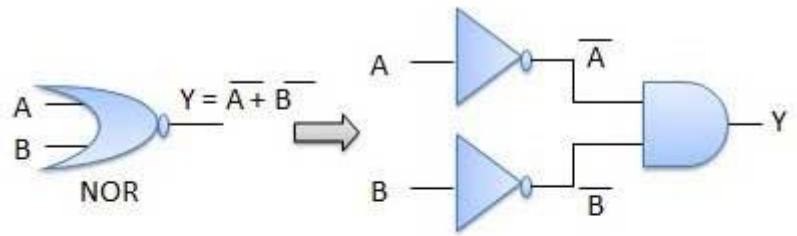
De Morgan has suggested two theorems which are extremely useful in Boolean Algebra. The two theorems are discussed below.

Theorem 1

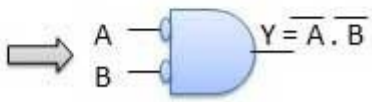
$\overline{A + B} = \overline{A} . \overline{B}$

NOR = Bubbled AND

- The LHS of this theorem represents a NOR gate with inputs A and B, whereas the RHS represents an AND gate with inverted inputs.
- This AND gate is called as **Bubbled AND**.



NOR ☰ Bubbled AND



Bubbled AND

Table showing verification of the De Morgan's second theorem −

A	B	$\overline{A + B}$	$\overline{A}$	$\overline{B}$	$\overline{A} . \overline{B}$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

Theorem 2

$\overline{A.B} = \overline{A} + \overline{B}$

NAND = Bubbled OR

- The left hand side (LHS) of this theorem represents a NAND gate with inputs A and B, whereas the right hand side (RHS) of the theorem represents an OR gate with inverted inputs.
- This OR gate is called as **Bubbled OR**.

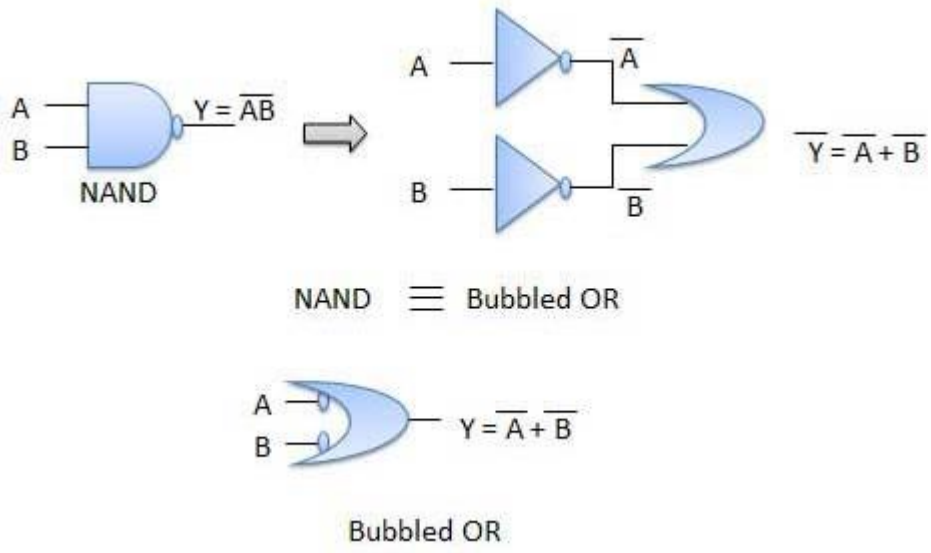


Table showing verification of the De Morgan's first theorem −

A	B	$\overline{AB}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

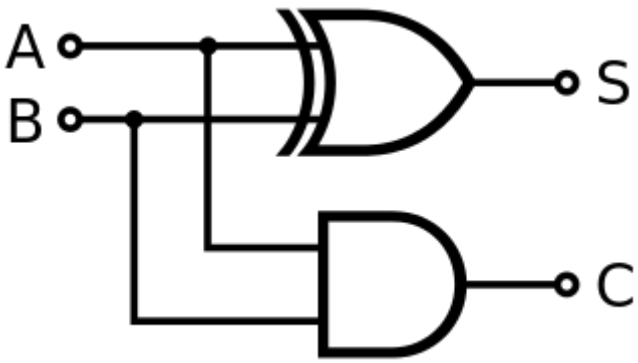
What Does Half Adder Mean?

A half adder is a type of adder, an electronic circuit that performs the addition of numbers. The half adder is able to add two single binary digits and provide the output plus a carry value. It has two inputs, called A and B, and two outputs S (sum) and C (carry). The common representation uses a XOR logic gate and an AND logic gate.

The adder works by combining the operations of basic logic gates, with the simplest form using only a XOR and an AND gate. This can also be converted into a circuit that only has AND, OR and NOT gates. This is especially useful since these three simpler logic gate ICs (integrated circuits) are more common and available than the XOR IC, though this might result in a bigger circuit since three different chips are used instead of just one.

Half Adder truth table:

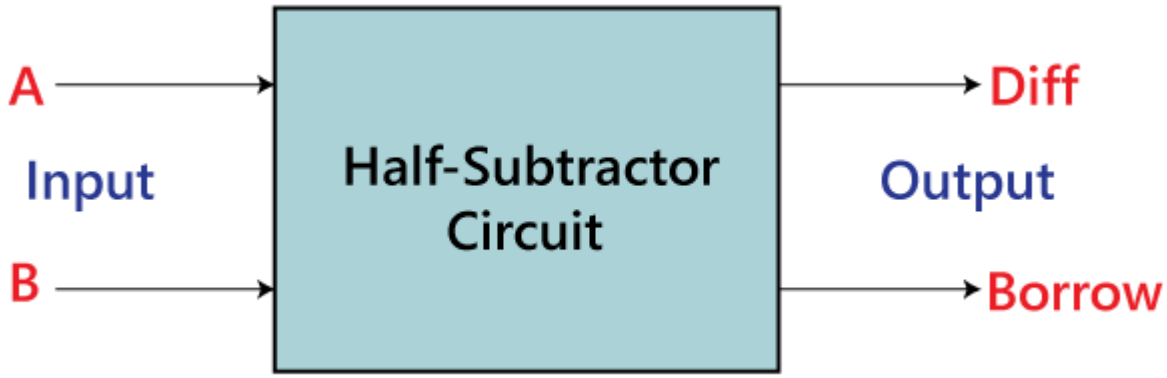
A (input)	B (input)	Sum (output)	Carry (output)
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1



Half Subtractor

The half subtractor is also a building block for subtracting two binary numbers. It has two inputs and two outputs. This circuit is used to subtract two single bit binary numbers A and B. The **'diff'** and **'borrow'** are two output states of the half subtractor.

Block diagram

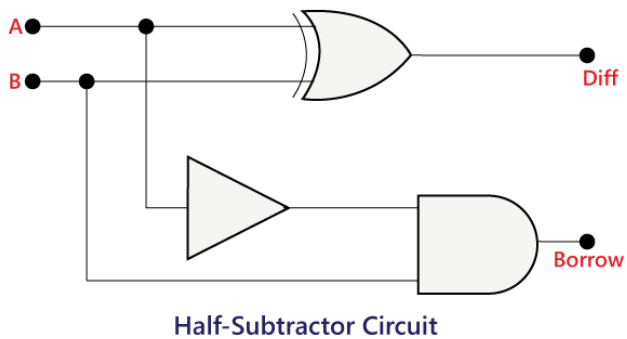


Truth Table

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

Half-Subtractor logical circuit

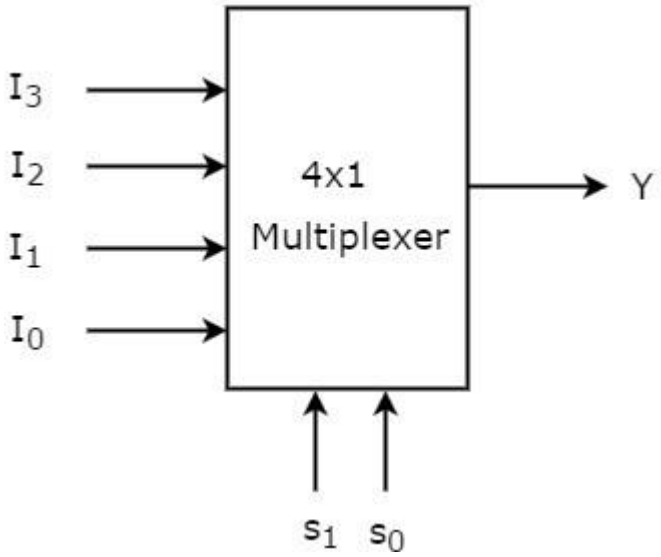
So, the Half Subtractor is designed by combining the 'XOR', 'AND', and 'NOT' gates and provide the Diff and Borrow.



Half-Subtractor Circuit

What is a Multiplexer?

The multiplexer is a device that has multiple inputs and single line output. The select lines determine which input is connected to the output, and also increase the amount of data that can be sent over a network within a certain time. It is also called a data selector.



Multiplexer Types

Multiplexers are classified into four types:

- 2-1 multiplexer (1select line)
- 4-1 multiplexer (2 select lines)
- 8-1 multiplexer(3 select lines)
- 16-1 multiplexer (4 select lines)

(Refer PDF also)

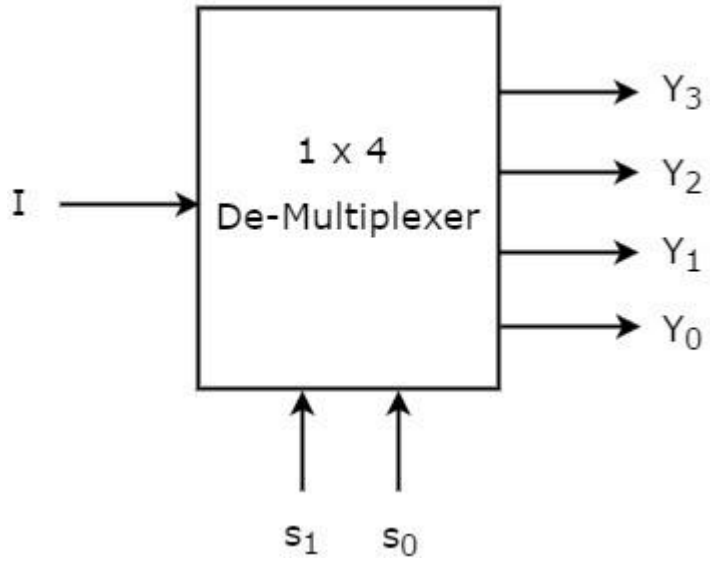
De-Multiplexer

De-Multiplexer is a combinational circuit that performs the reverse operation of Multiplexer. It has single input, 'n' selection lines and maximum of 2^n outputs. The input will be connected to one of these outputs based on the values of selection lines.

Since there are 'n' selection lines, there will be 2^n possible combinations of zeros and ones. So, each combination can select only one output. De-Multiplexer is also calledas **De-Mux**.

1x4 De-Multiplexer

1x4 De-Multiplexer has one input I, two selection lines, s_1 & s_0 and four outputs $Y_3, Y_2, Y_1 \& Y_0$. The **block diagram** of 1x4 De-Multiplexer is shown in the following figure.

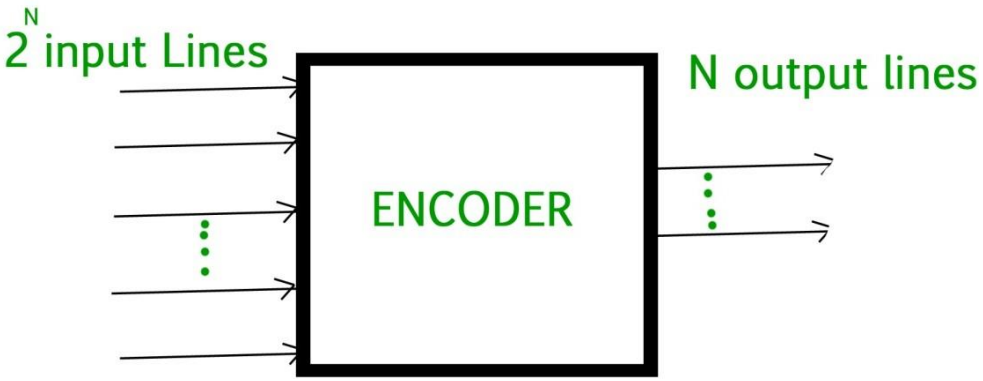


The single input 'I' will be connected to one of the four outputs, Y_3 to Y_0 based on the values of selection lines s_1 & s_0.

(Refer PDF also)

Encoder

An Encoder is a **combinational circuit** that performs the reverse operation of Decoder. It has 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. Therefore, the encoder encodes 2^n input lines with 'n' bits.



4 : 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 figure below shows the logic symbol of 4 to 2 encoder :

The Truth table of 4 to 2 encoder is as follows :

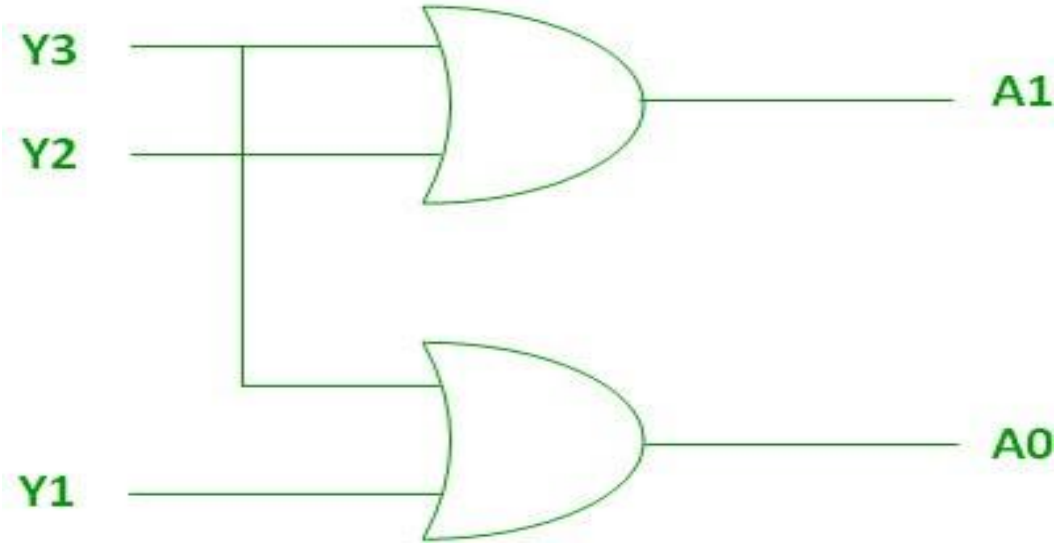
INPUTS				OUTPUTS	
Y3	Y2	Y1	Y0	A1	A0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Logical expression for A1 and A0 :

A1 = Y3 + Y2

A0 = Y3 + Y1

The above two Boolean functions A1 and A0 can be implemented usingtwo input OR gates :



Decimal to BCD Encoder –

The decimal to binary encoder usually consists of **10 input lines** and **4 output lines**. Each input line corresponds to the each decimal digit and 4outputs correspond to the BCD code. This encoder accepts the decoded decimal data as an input and encodes it to the BCD output which is available on the output lines. The figure below shows the logic symbol of decimal to BCD encoder :

The truth table for decimal to BCD encoder is as follows:

INPUTS										OUTPUTS			
Y9	Y8	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	A3	A2	A1	A0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

Logical expression for A3, A2, A1 and A0 :

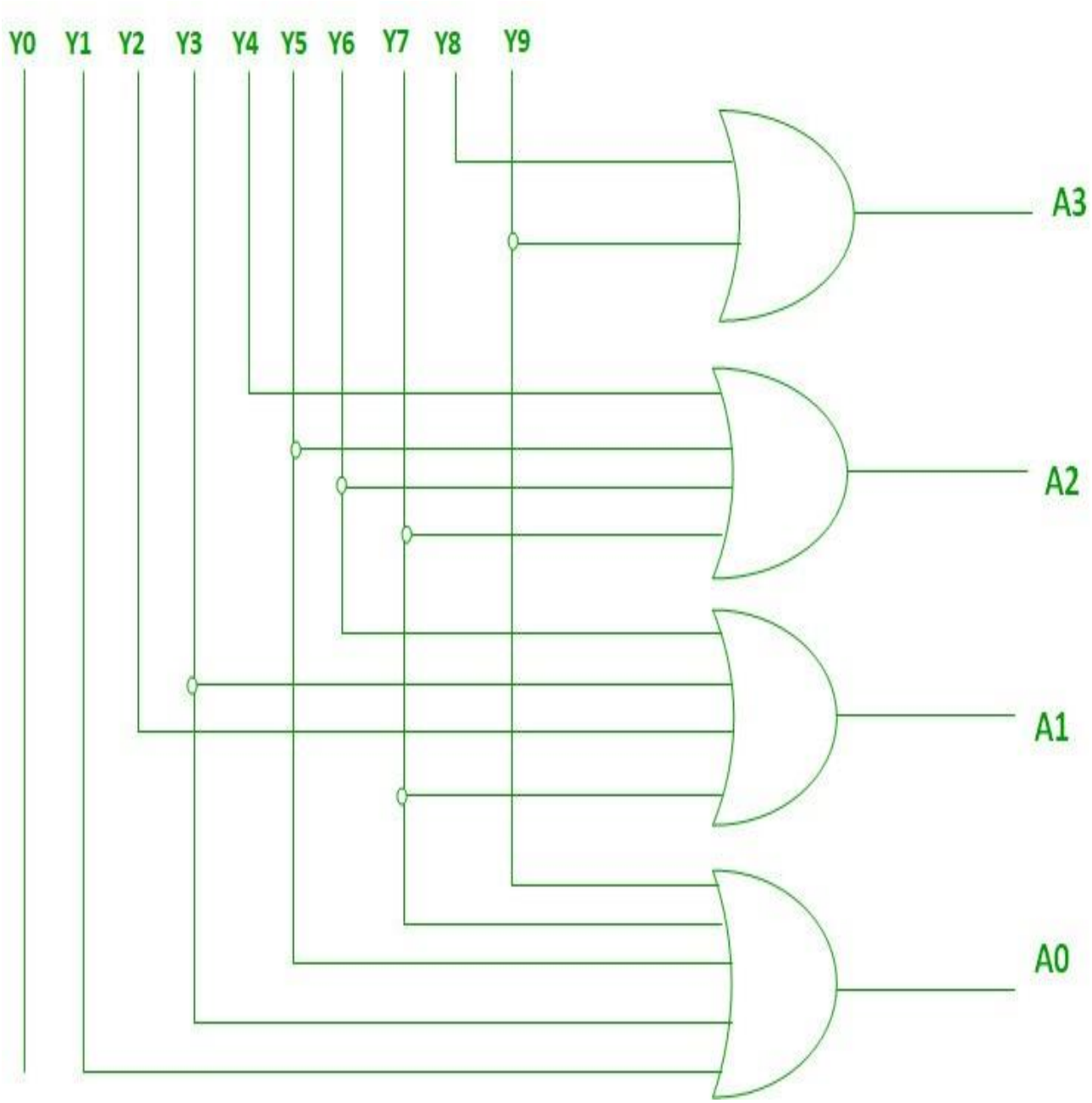
A3 = Y9 + Y8

A2 = Y7 + Y6 + Y5 +Y4

A1 = Y7 + Y6 + Y3 +Y2

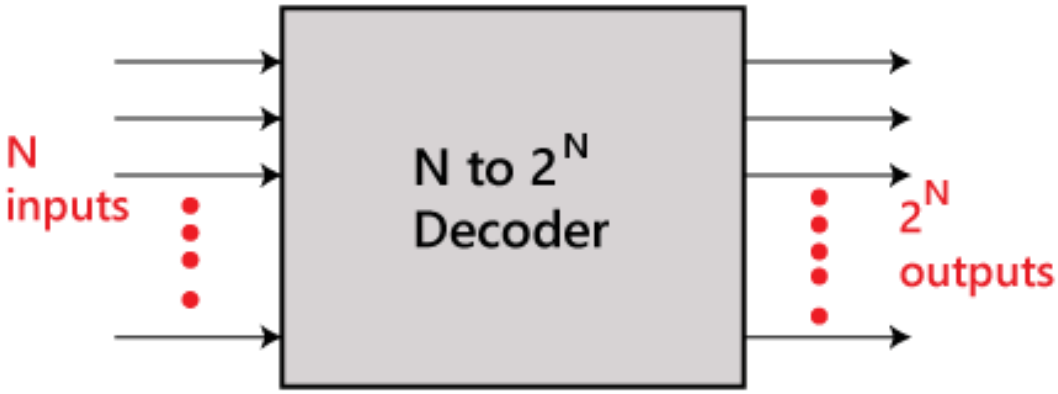
A0 = Y9 + Y7 +Y5 +Y3 + Y1

The above two Boolean functions can be implemented using OR gates :



DECODER

The combinational circuit that change the binary information into 2^N output lines is known as **Decoders.** The binary information is passed in the form of N input lines. The output lines define the 2^N-bit code for the binary information. In simple words, the **Decoder** performs the reverse operation of the **Encoder**. At a time, only one input line is activated for simplicity. The produced 2^N-bit output code is equivalent to the binary information.

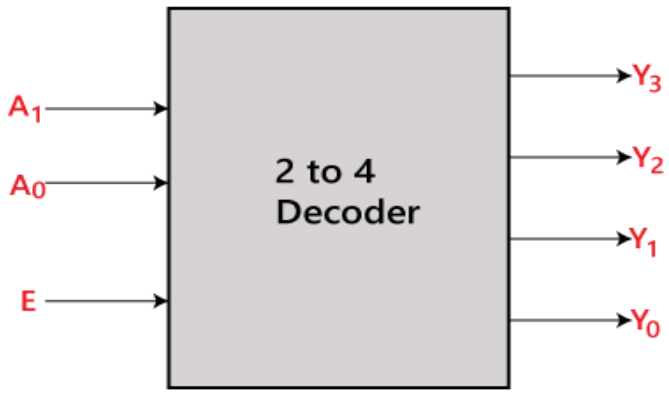


There are various types of decoders which are as follows:

2 to 4 line decoder:

In the 2 to 4 line decoder, there is a total of three inputs, i.e., A_0, andA_1 and E and four outputs, i.e., Y_0, Y_1, Y_2, and Y_3. For each combination of inputs, when the enable 'E' is set to 1, one of these four outputs will be 1.The block diagram and the truth table of the 2 to 4 line decoder are givenbelow.

Block Diagram:



Truth Table:

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

The logical expression of the term Y0, Y0, Y2, and Y3 is as follows:

$Y_3=E.A_1.A_0$

$Y_2=E.A_1.A'$

$Y_1=E.A_1'.A_0$

$Y0=E.A_1'.A_0'$

BCD TO 7 SEGMENT DISPLAY

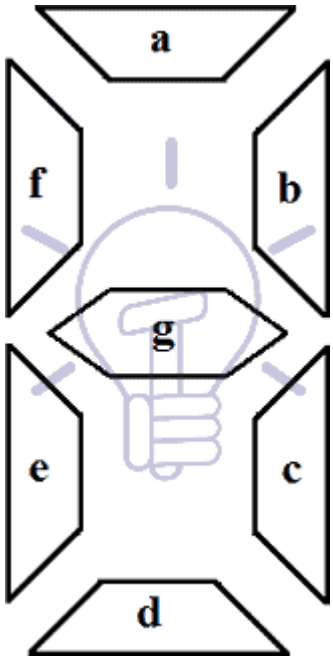
Digits	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

A digital system like a computer can understand and easily read a large number in binary format. However, a human cannot read large binary numbers. To solve this problem we need to display it as a decimal digit using 7-segment display.

7-Segment Display

It is a digital device that can be used for displaying decimal number,alphabets, and characters.

7-Segment display contains **7 LED segments** arranged in a shape given in figure above.



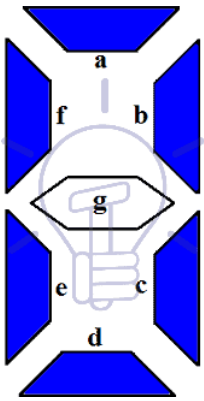
Generally,there are 8 input pins in a 7-Segment display. 7 input pins for each of the 7 LEDs and one pin for the common terminal.

Working of 7-Segment Display (LED & LCD) Circuit

7 LED segments of the display and their pins are "**a", "b", "c", "d", "e", "f"** & **"g"** as shown in the figure given below. Each of the pins will illuminate the specific segment only.

We assume common cathode LED segment as our example.

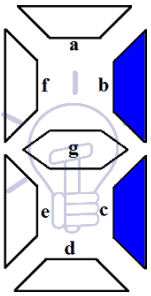
Suppose we want to display digit '0', in order to display 0, we need to turn on "a", "b", "c", "d", "e", "f". & turn-off the "g". which would look like the figure given below.



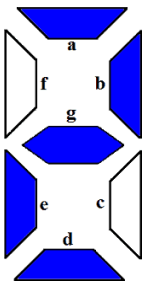
7-Segment Display Segments for all Numbers

Display combination of decimal numbers is given below.

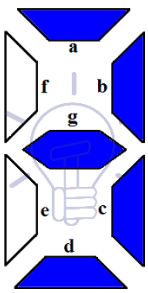
Digit 1: to display the digit 1 we need to turn on the segments b, c. and turn off the LED segments a, d, e, f, and g. This configuration will result in the display as shown in the figure below.



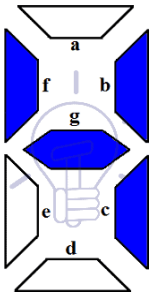
Digit 2: to display the digit 2 we need to turn on the segments a, b, d, e, g. and turn off the LED segments c, f. This configuration will result in thedisplay as shown in the figure below.



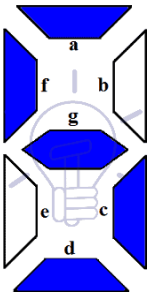
Digit 3: to display the digit 3 we need to turn on the segments a, b, c, d, g. and turn off the LED segments e, f. This configuration will result in thedisplay as shown in the figure below.



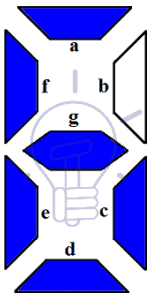
Digit 4: to display the digit 4 we need to turn on the segments b, c, f, g. and turn off the LED segments a, d, e. This configuration will result in the display as shown in the figure below.



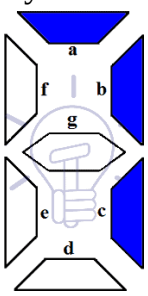
Digit 5: to display the digit 5 we need to turn on the segments a, c, d, f, g.and turn off the LED segments b, e. This configuration will result in the display as shown in the figure below.



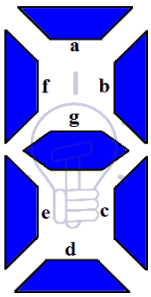
Digit 6: to display the digit 6 we need to turn on the segments a, c, d, e, f, g. and turn off the LED segments b. This configuration will result in thedisplay as shown in the figure below.



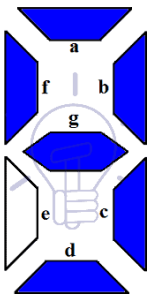
Digit 7: to display the digit 7 we need to turn on the segments a, b, c. andturn off the LED segments d, e, f, g. This configuration will result in the display as shown in the figure below.



Digit 8: to display the digit 8 we need to turn on the segments a, b, c, d,e, g only. This configuration will result in the display as shown in the figure below.



Digit 9: to display the digit 2 we need to turn on the segments a, b, c, d, f,
g. and turn off the LED segments e. This configuration will result in thedisplay as shown in the figure below.



To display these digits using binary numbers we need to decode these binary numbers into the combination used for each pattern or display using Decoder.

Truth Table

Assume common cathode 7-Segment display. Suppose the binary input **ABCD** to the decoder and output a, b, c, d, e, f, & g for the display.

	INPUT				OUTPUT						
Digits	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

FLIP FLOPS

The RS Flip Flop is considered as one of the most basic sequential logiccircuits. The Flip Flop is a one-bit memory bi-stable device.

It has two inputs, one is called **"SET"** which will set the device

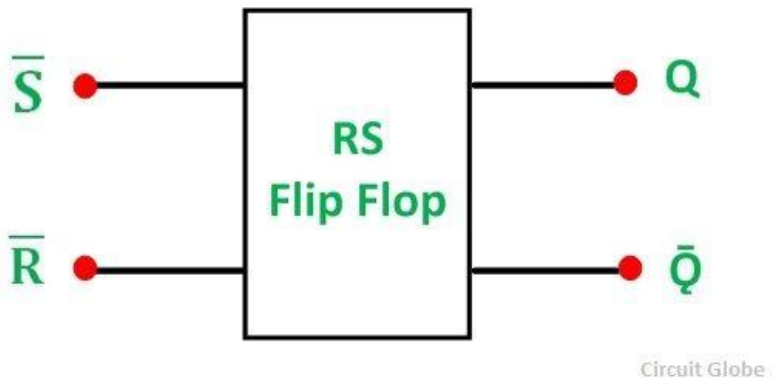
(output = 1) and is labeled S and another is known as **"RESET"** which willreset the device

(output = 0) labeled as R. The RS stands for **SET/RESET.**

The flip-flop is reset back to its original state with the help of RESET inputand the output is Q that will be either at logic level "1" or logic"0". It
depends upon the set/reset condition of the flip-flop. Flip flop word means that it can be **"FLIPPED"** into one logic state or **"FLOPPED"** backinto another.

The basic NAND gate RS Flip Flop circuit is used to store the data and thusprovides feedback from both of its outputs again back to its inputs. The RS Flip Flop actually has three inputs, SET, RESET and its current output Qrelating to its current state.

The symbol of the RS Flip-Flop is shown below:



The basic NAND gate RS flip flop circuit is used to store the data and thus provides feedback from both of its outputs again back to its inputs. The RS flip flop actually has three inputs, SET, RESET and clock pulse.

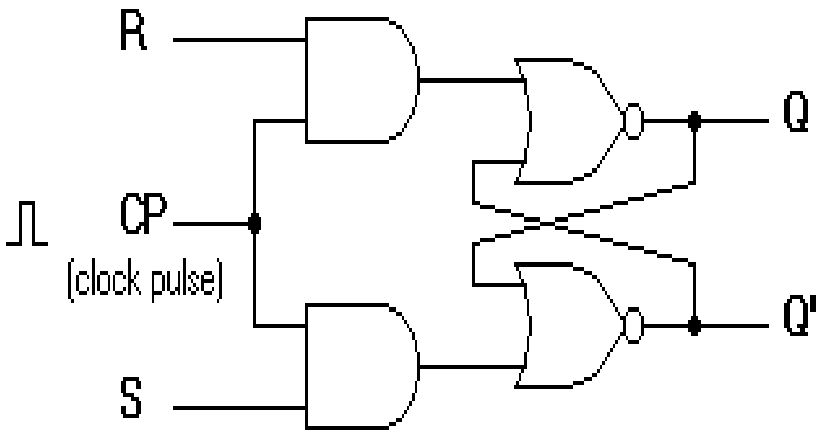


Figure-1:R-S flip flop circuit diagram

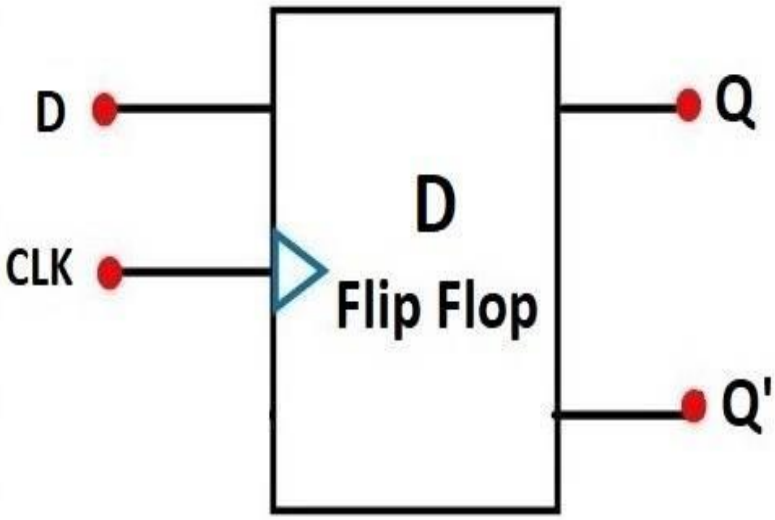
INPUTS			OUTPUT	STATE
CLK	S	R	Q	
X	0	0	No Change	Previous
↑	0	1	0	Reset
↑	1	0	1	Set
↑	1	1	-	Forbidden

D Flip-Flop

The D flip-flop is a two-input flip-flop. The inputs are the data (D) inputand a clock (CLK) input. The clock is a timing pulse generated by the
equipment to control operations. The D flip-flop is used to store data at apredetermined time and hold it until it is needed. This circuit is sometimes called a delay flip-flop. In other words, the data input is delayed up to one clock pulse before it is seen in the output. The simplest form of a D flip-
flop is shown in the figure below

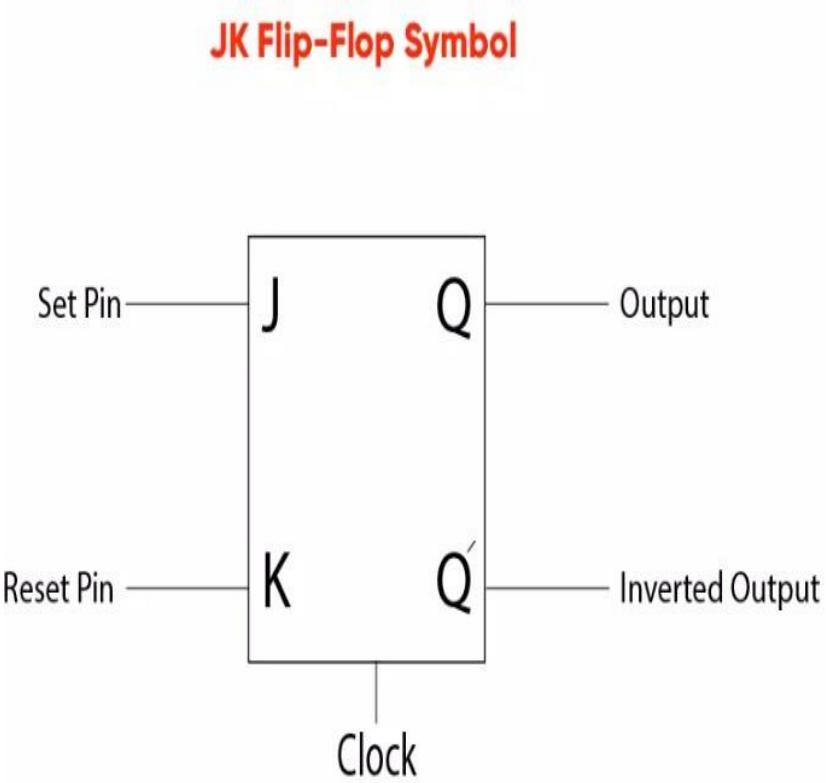
Truth Table of DFlip Flop

D	CLK	Q	Q`
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0



J-K Flip-Flop

The J-K flip-flop is the most widely used flip-flop because of its versatility.When properly used it may perform the function of an R-S or D flip-flop. The standard symbol for the J-K flip-flop is shown in the figure below.



JK Flip-Flop Logic Table

C	J	K	Q	Q`
HIGH	0	0	Latch	Latch
HIGH	0	1	0	1
HIGH	1	0	1	0
HIGH	1	1	Toggle	Toggle
LOW	0	0	Latch	Latch
LOW	0	1	Latch	Latch
LOW	1	0	Latch	Latch
LOW	1	1	Latch	Latch

This simple **JK flip Flop** is the most widely used of all the flip-flop designsand is considered to be a universal flip-flop circuit. The two inputs labelled“J” and “K” are not shortened abbreviated letters of other words, such as “S” for Set and “R” for Reset, but are themselves autonomous letters chosen by its inventor Jack Kilby to distinguish the flip-flop design from other types.

The sequential operation of the JK flip flop is exactly the same as for the previous SR flip-flop with the same “Set” and “Reset” inputs. The differencethis time is that the “JK flip flop” has no invalid or forbidden input states of the SR Latch even when S and R are both at logic “1”.

The **JK flip flop** is basically a gated SR flip-flop with the addition of a clockinput circuitry that prevents the illegal or invalid output condition that can occur when both inputs S and R are equal to logic level “1”. Due to this additional clocked input, a JK flip-flop has four possible input combinations,“logic 1”, “logic 0”, “no change” and “toggle”. The symbol for a JK flip flop issimilar to that of an *SR Bistable Latch* as seen in the previous tutorial except for the addition of a clock input.

COUNTERS

A **Counter** is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal. Counters are used in digital electronics for counting purpose, they can count specific event happening in the circuit. For example, in UP counter a counter increases count for every rising edge of clock. Not only counting, a counter can follow the certain sequence based on our design like any random sequence 0,1,3,2… .They can also be designed with the help of flip flops. They are used as frequency dividers where the frequency of given pulse waveform is divided. Counters are sequential circuit that count the number of pulses can be either in binary code or BCD form. The main properties of a counter are timing , sequencing , and counting. Counter works in two modes: Up counter & Down counter.