Dept of ECE,BMSCE

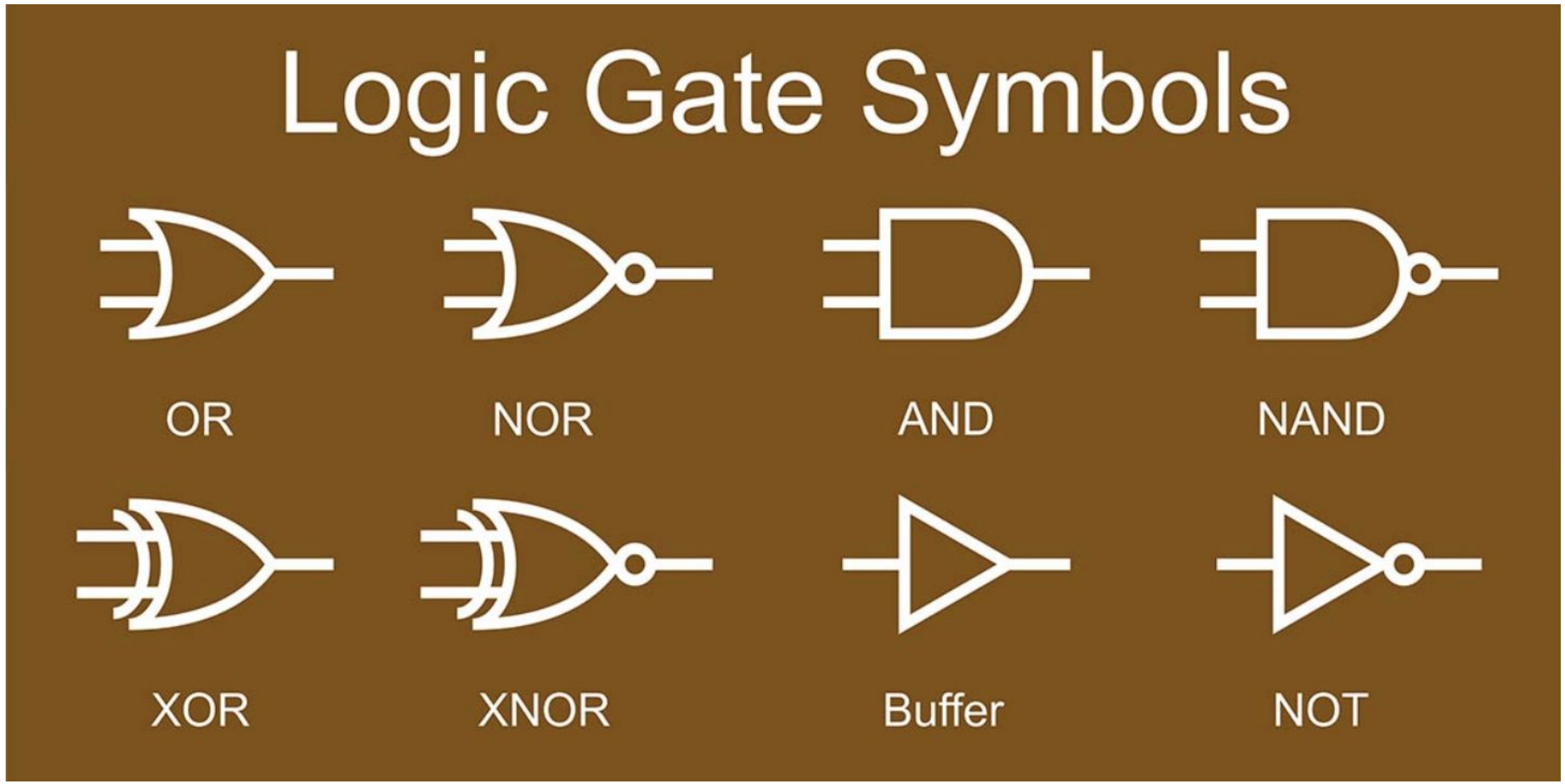
Course: Basic Electronics and Communication

Unit II

UNIT 2

Logic Gates

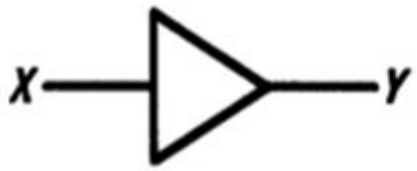
- Logic Systems consists of Gates and Flip-flops
- Logic gates are electronic circuits designed to produce the basic logic functions such as AND, OR, etc.
- Flip-flops are memory devices capable of storing logic constants.
- The Interconnection of Gates and Flip-flops results in Logic Networks.


Logic Gate Symbols
OR
NOR
AND
NAND
XOR
XNOR
Buffer
NOT

Buffers:

- Buffers do not affect the logical state of a digital signal.
- A logic 1 input results in a logic 1 output whereas a logic 0 input results in a logic 0 output.
- Buffers are used to provide extra current drive at the output.
- The Boolean expression for the output : Y = X.

Logic Diagram



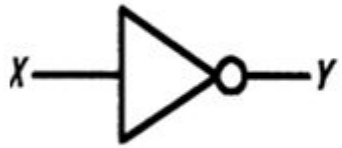
Truth Table

X	Y
0	0
1	1

Inverters

- Inverters are used to complement the logical state.
- A logic 1 input results in a logic 0 output and vice versa.
- The Boolean expression for the output, Y, of an Inverter with an input, X, is: $Y = \overline{X}$

Logic Diagram



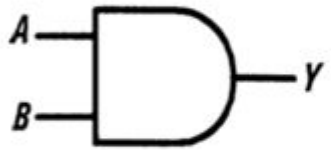
Truth Table

X	Y
0	1
1	0

AND Gate:

- AND gates will only produce a logic 1 output when all inputs are simultaneously at logic 1.
- Other input combination results in a logic 0 output.
- The Boolean expression for the output, $Y = A \cdot B$

Logic Diagram



Truth Table

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate

- OR gates will produce a logic 1 output whenever any one, or more, inputs are at logic 1.
- OR gate will only produce a logic 0 output whenever all of its inputs are at logic 0.
- The Boolean expression for the output, Y = A + B

Logic Diagram

Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

NAND Gate:

- NAND (i.e. NOT-AND) gates will only produce a logic 0 output when all inputs are simultaneously at logic 1.
- Other input combination will produce a logic 1 output.
- A NAND gate, is an AND gate with its output inverted. The circle shown at the output denotes this inversion.
- The Boolean expression for the output: $Y = \overline{A \cdot B}$

Logic Diagram

Truth Table

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

- NOR (i.e. NOT-OR) gates will only produce a logic 1 output when all inputs are simultaneously at logic 0.
- Other input combination will produce a logic 0 output.
- A NOR gate, is an OR gate with its output inverted. A circle is used to indicate inversion.
- The Boolean expression for the output. $Y = \overline{A + B}$

Logic Diagram

Truth Table

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

EX-OR Gate

- A two input Exclusive-OR gates will produce a logic 1 output, whenever either one of the inputs is at logic 1 and the other is at logic 0.
- Exclusive-OR gates produce a logic 0 output whenever both inputs have the same logical state (i.e. when both are at logic 0 or both are at logic 1).
- The Boolean expression for the output, Y, of an exclusive-OR gate with inputs A and B is: $Y = A \cdot \overline{B} + B \cdot \overline{A}$

Logic Diagram

Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

EX-NOR Gate

- A two input Exclusive-NOR gates will produce a logic 1 output, whenever both of its inputs are same otherwise it produces logic 0 output.
- The Boolean expression for the output, Q, of an exclusive-NOR gate with inputs A and B is: $Q=(A\cdot B)+(\bar{A}\cdot\bar{B})$

Logic Diagram

Truth Table

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1

Adder

- An adder is a device that will add together two bits and give the result as the output.
- There are two kinds of **adders - half adders and full adders**.
- A half adder just adds two bits together and gives a two-bit output.
- A full adder adds two inputs and a carried input from another adder, and also gives a two-bit output.

Half adder

The circuit adds two binary variables, yields a carry but does not accept carry from another circuit(adder).

The truth table of the half adder with its circuit is shown below.

Truth table:

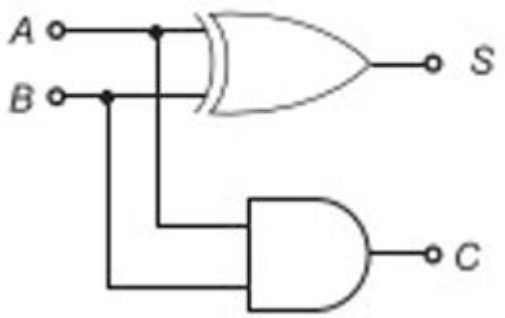
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

S = Sum; C = Carry

$$S = \overline{A}B + A\overline{B} = A \oplus B$$

$$C = AB$$

Logic Diagram



Full adder

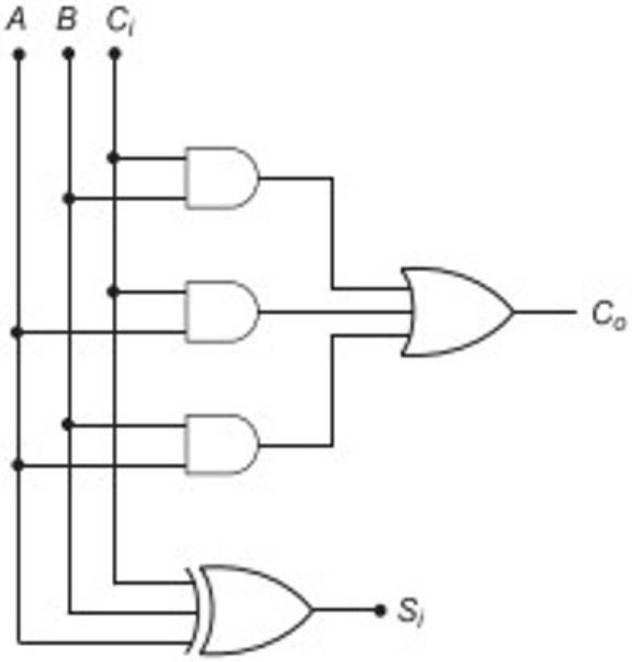
- Full Adder is the **adder which adds three inputs and produces two outputs**.
- The first two inputs are A and B and the third input is an input carry, C_i.
- The output carry is given as C_o and the normal output as S which is SUM.

Truth Table of Full adder

Row No.	A	B	C_i	S	C_o
1	0	0	0	0	0
2	0	0	1	1	0
3	0	1	0	1	0
4	0	1	1	0	1
5	1	0	0	1	0
6	1	0	1	0	1
7	1	1	0	0	1
8	1	1	1	1	1

Full adder …

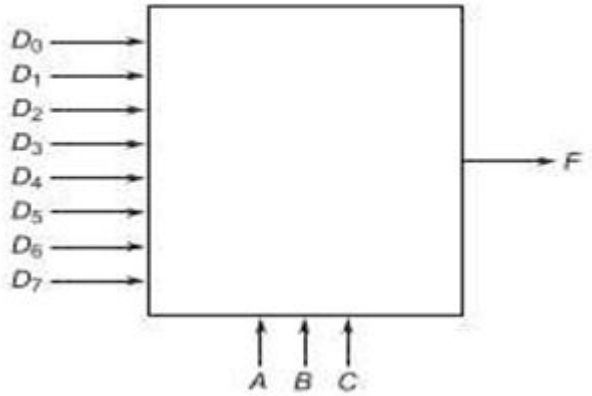
- It is observed from the truth table that C_0=1 for rows which have two 1's otherwise it is 0.
- Its Boolean Function is C_0=AB+BC_i+C_iA
- It can be implemented by three AND and one OR gates. S=1 for rows with one 1 and three 1's., i.e odd number of 1's. its implemented by a three input XOR.



Multiplexer

Multiplexer is to select one signal from a group of 2^n inputs, and map on to a single output line.

- Example: 2:1, 4:1, 8:1
- Block diagram of an 8:1 Multiplexer is shown below.
- Lines D0,D1,….. D7 are the data input lines and F is the output line, Lines A, B and C are called select lines.

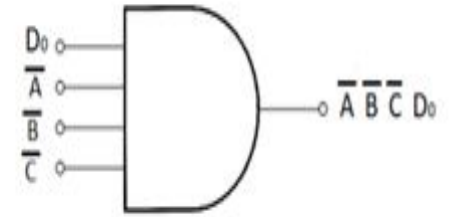


Multiplexer Implementation

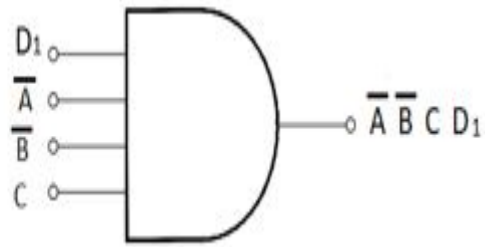
Select Lines			O/P	
A	B	C	F	
0	0	0	D_0	$\bar{A}\,\bar{B}\,\bar{C}\,D_0$
0	0	1	D_1	$\bar{A}\,\bar{B}\,C\,D_1$
0	1	0	D_2	$\bar{A}\,B\,\bar{C}\,D_2$
0	1	1	D_3	$\bar{A}\,B\,C\,D_3$
1	0	0	D_4	$A\,\bar{B}\,\bar{C}\,D_4$
1	0	1	D_5	$A\,\bar{B}\,C\,D_5$
1	1	0	D_6	$A\,B\,\bar{C}\,D_6$
1	1	1	D_7	$A\,B\,C\,D_7$

Select Lines			O/P
A	B	C	F
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

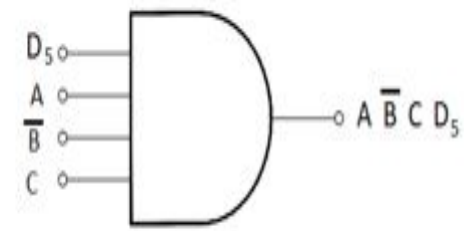
- A=0 B=0 C=0



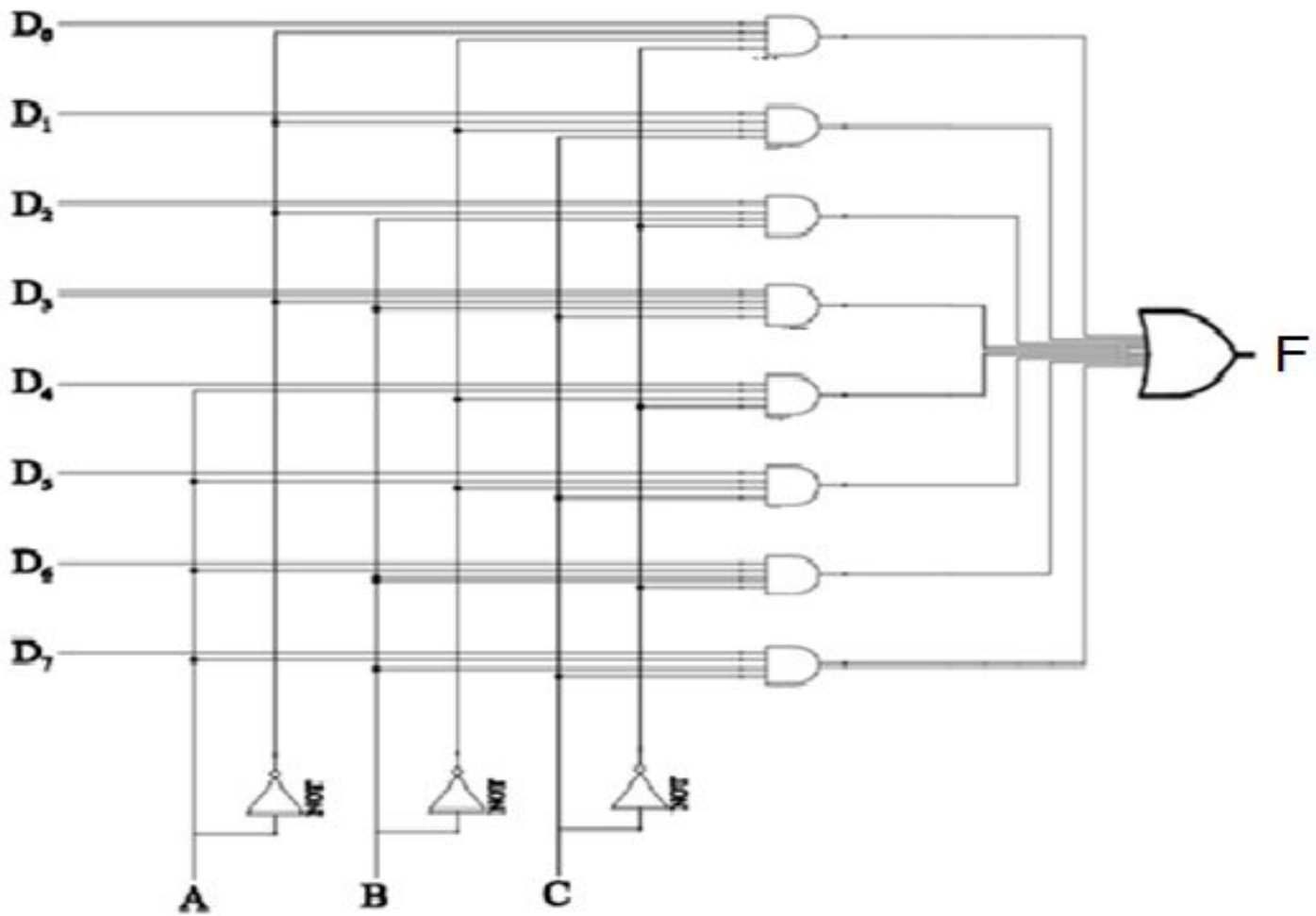
- A=0 B=0 C=1



- A=1 B=0 C=1



Multiplexer logic diagram

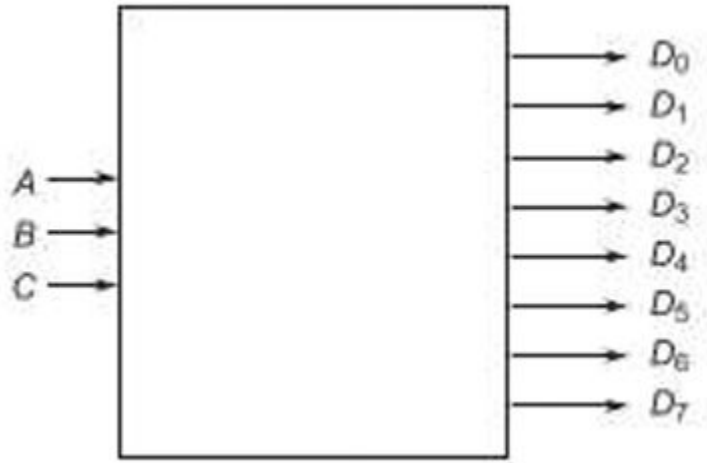


Multiplexer Application

- To choose one of the several Inputs
- 2^n :1 multiplexer can be used to Implement Boolean function with n variables by associating each input line with a row of truth table.

Decoder

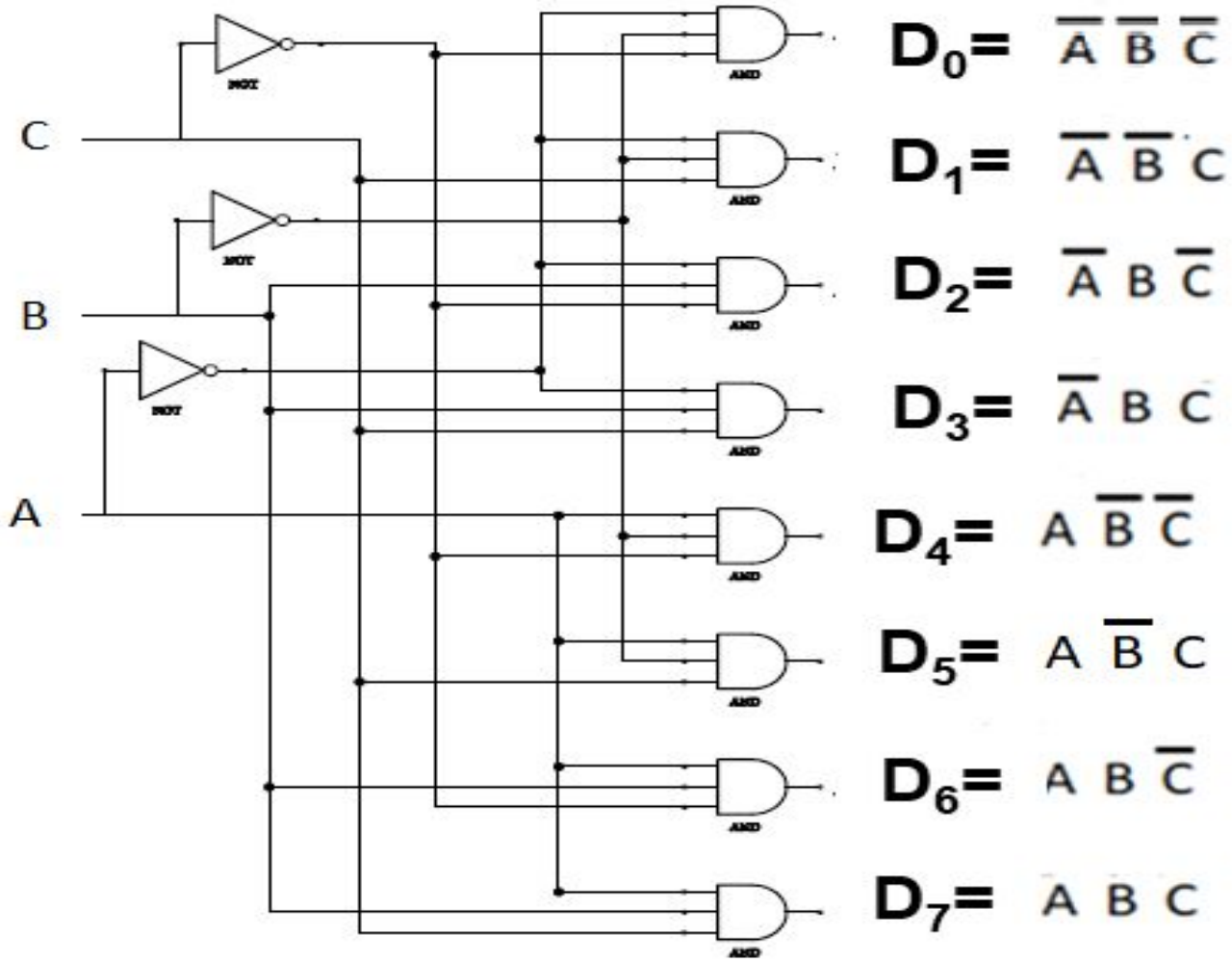
- It decodes an n-bit binary number, producing a signal on one of 2^n output lines.
- If input A=1, B=0, C=1 , the output D_5 is 1 and all other outputs are 0.



Truth table of 3-to-8 decoder

INPUTS			OUTPUTS							
A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Decoder logic diagram

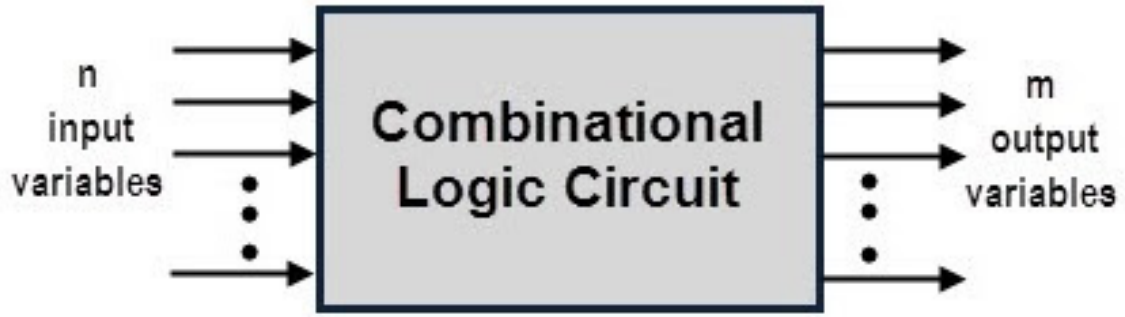


Decoder application

- A decoder with an Enable line can be used as a De-Multiplexer, which directs a single data line to one of the 2^n output lines
- Decodes memory address for read or write operation of RAM

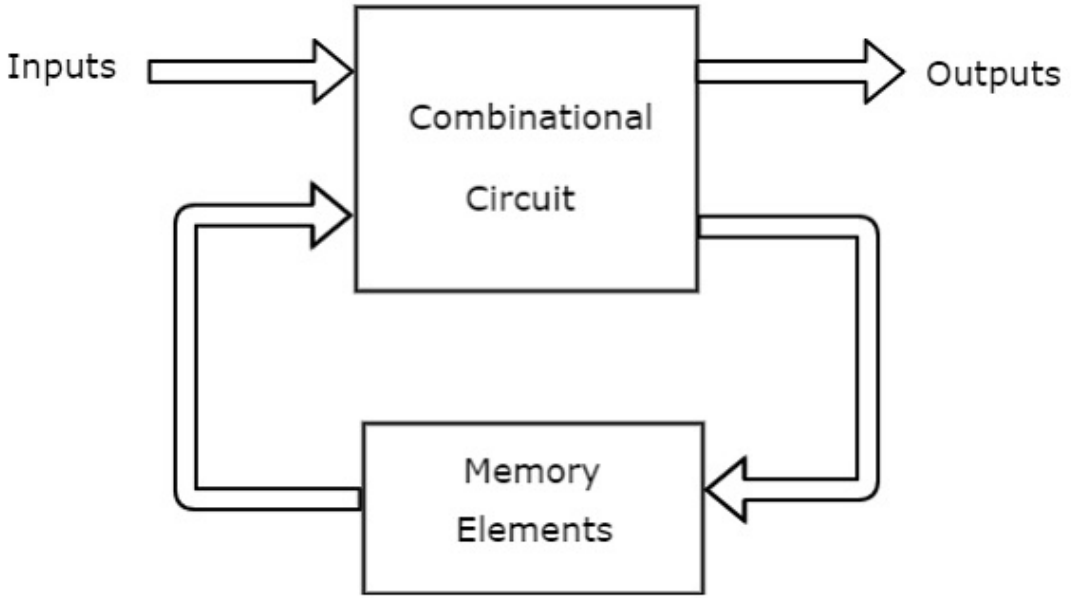
COMBINATIONAL LOGIC CIRCUITS

- A combinational logic circuit is defined as a circuit in which the outputs at any instant are dependent only upon the inputs present at that instant.
- Examples – Adders, subtractors, decoders, multiplexers.



SEQUENTIAL LOGIC CIRCUITS

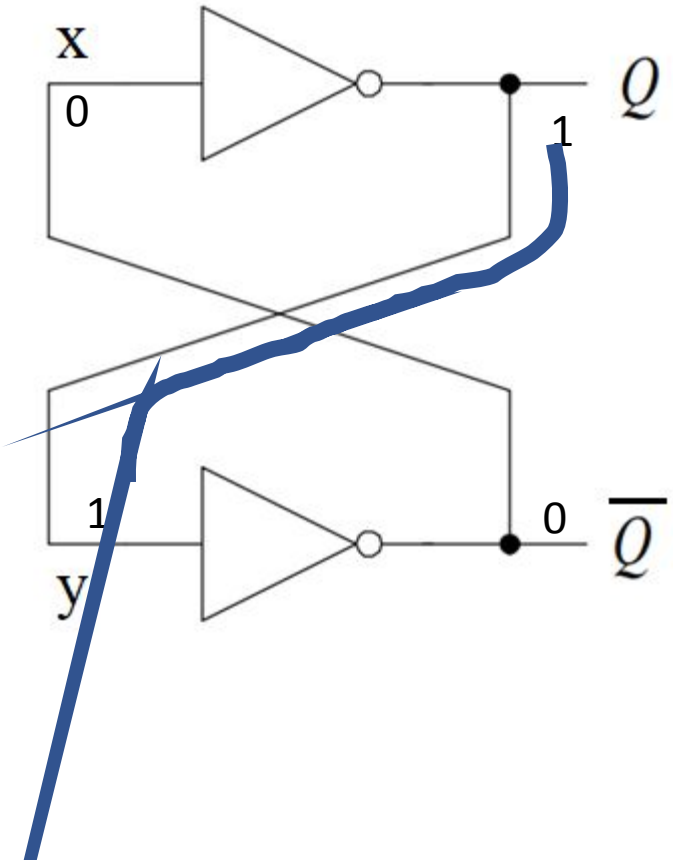
- A sequential circuits is defined as a circuit in which the outputs at any instant are dependent not only upon the present inputs, but also upon the past history or sequence of inputs.
- In order to preserve past history of inputs, sequential circuits are said to have *memory*.

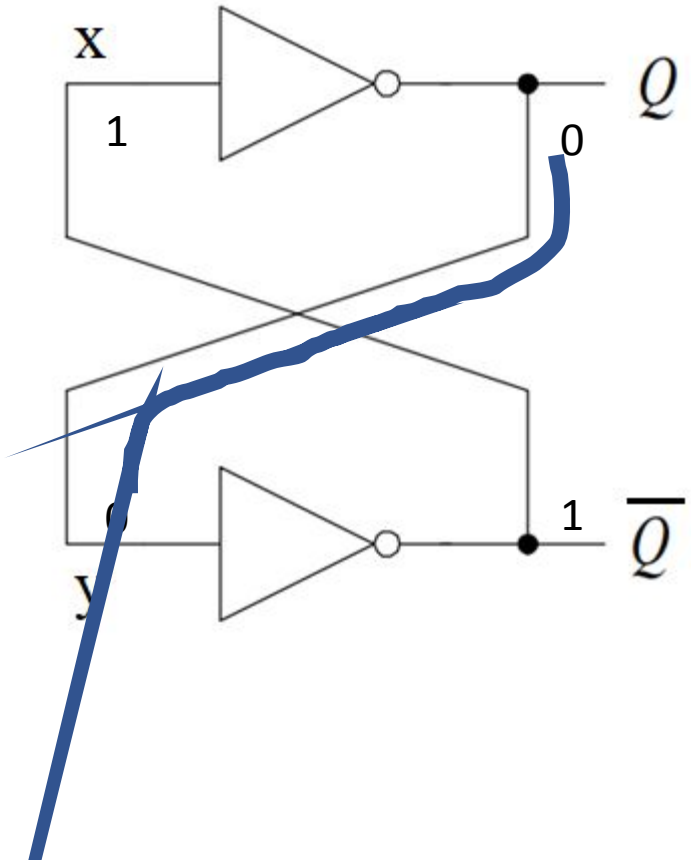


Examples include counters, registers etc.

BISTABLES

- All memory elements consist of a basic bistable element.
- The basic bistable element has two stable conditions or states.





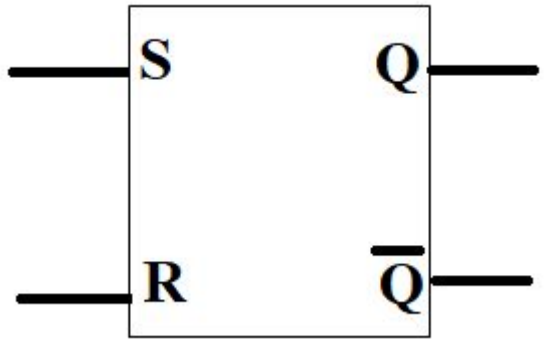
BISTABLES

- As a result of having two stable conditions, the basic bistable element is used to store binary symbols.
- In the case of positive logic, when the output line Q is 1, the element is said to be storing 1.
- When the output line Q is 0, the element is said to be storing 0.
- The two outputs are complementary.

When Q=0, $\overline{Q} = 1$

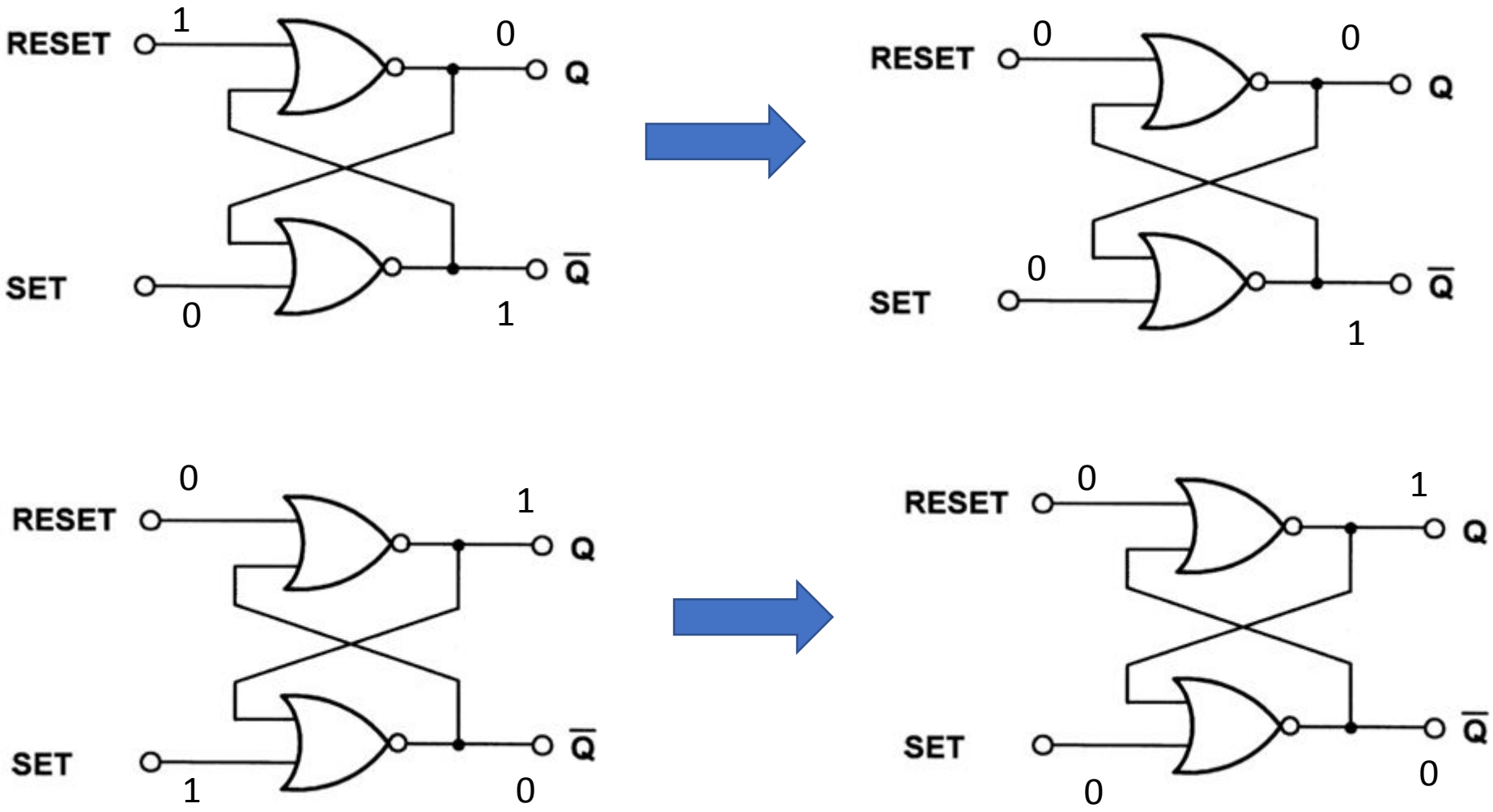
When Q=1, $\overline{Q}$=0

R-S BISTABLES

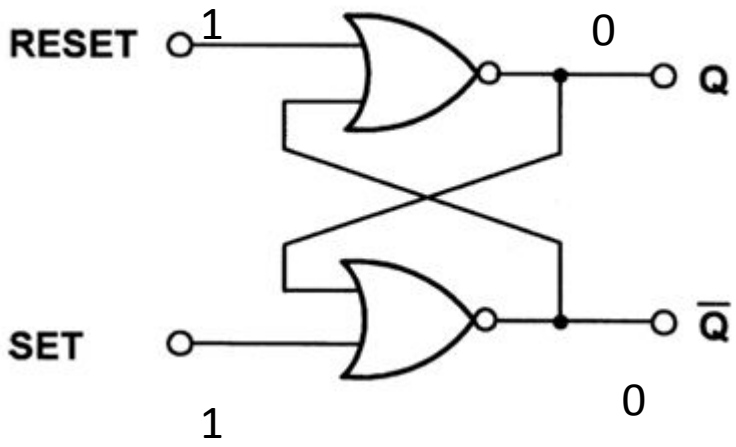


- The simplest form of bistable is the R-S bistable.
- This device has
  - two inputs SET and RESET
  - complementary outputs Q and $\overline{Q}$.
- A logic 1 applied to the SET input will cause the Q output to become (or remain at) logic 1
- A logic1 applied to the RESET input will cause the Q output to become (or remain at) logic 0.
- The bistable will remain in its present state until the application of another input.

OPERATION OF AN R-S BISTABLE

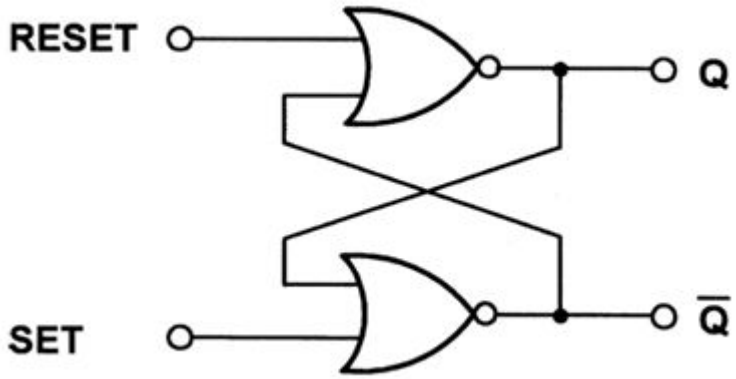


OPERATION OF AN R-S BISTABLE



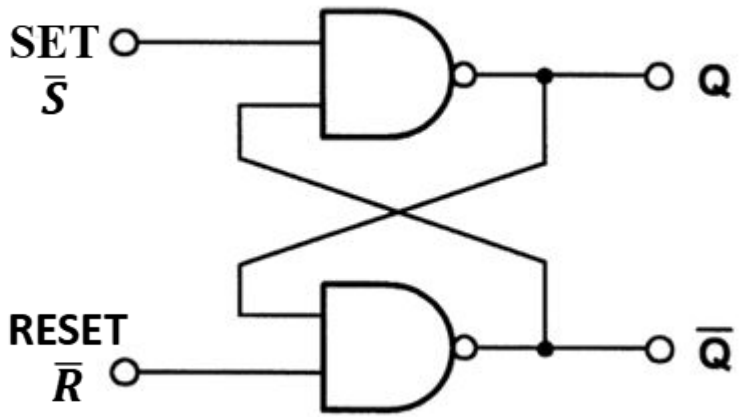
- The above analysis yields contradictory results
- Hence, R=1, S=1 is an invalid and forbidden state
- R-S bistable cannot be used when both the input are at logic 1

TRUTH TABLE OF R-S BISTABLES



INPUTS		OUTPUTS		COMMENTS
S	R	Q_{N+1}	$\overline{Q_{N+1}}$	
0	0	Q	$\overline{Q}$	Memory
0	1	0	1	Reset
1	0	1	0	Set
1	1	?	?	Forbidden

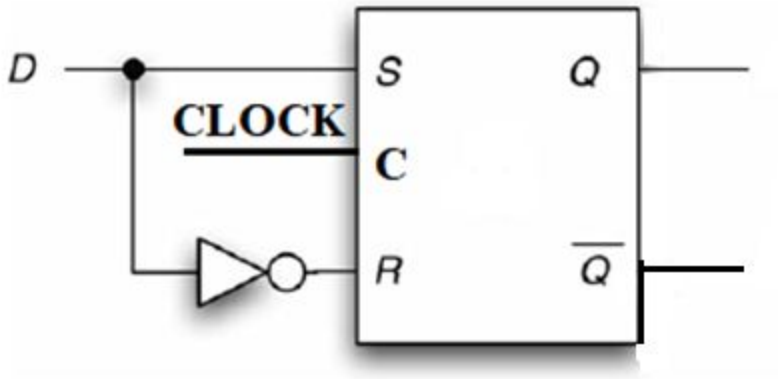
$\overline{S}$-$\overline{R}$ BISTABLE USING NAND GATES



INPUTS		OUTPUTS		COMMENTS
$\overline{S}$	$\overline{R}$	Q_{N+1}	$\overline{Q_{N+1}}$	
0	0	?	?	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q	$\overline{Q}$	Memory

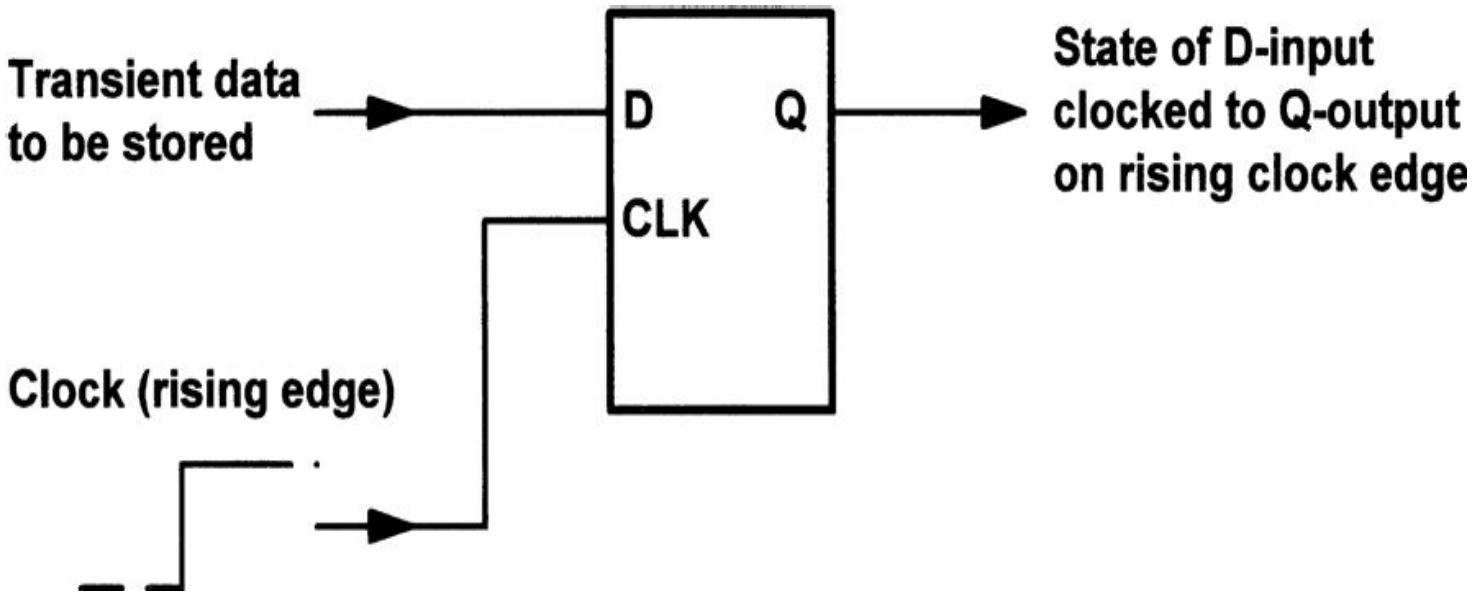
D-TYPE BISTABLES

- The D-type bistable has two inputs: D (standing for ‘data’ or ‘delay’) and CLOCK (CLK).
- D-type bistable is a modified Set-Reset bistable with the addition of an inverter to prevent the S and R inputs from being at the same logic level.



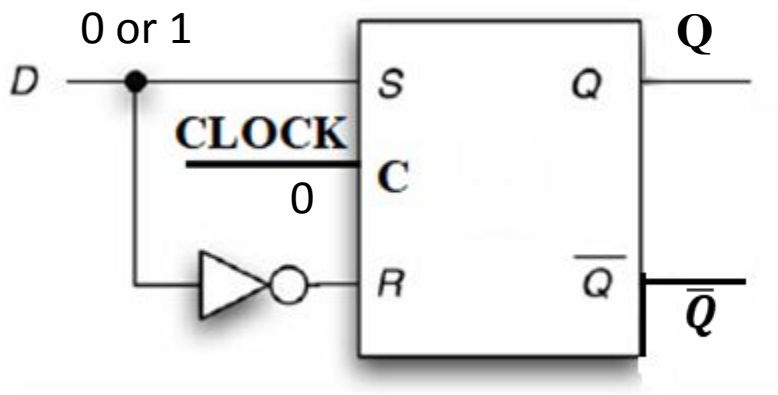
WORKING OF A D-TYPE BISTABLE

- The D-type bistable consists of
  - Input D that determines its next state and control.
  - Input clock that determines when the D input is effective.

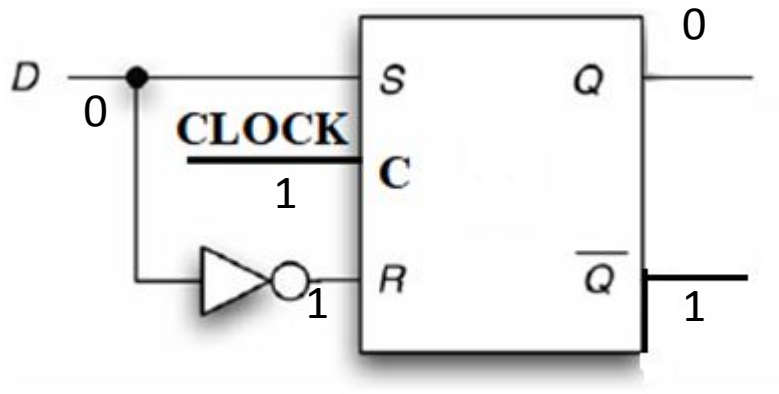


WORKING OF A D-BISTABLE

When clock=0



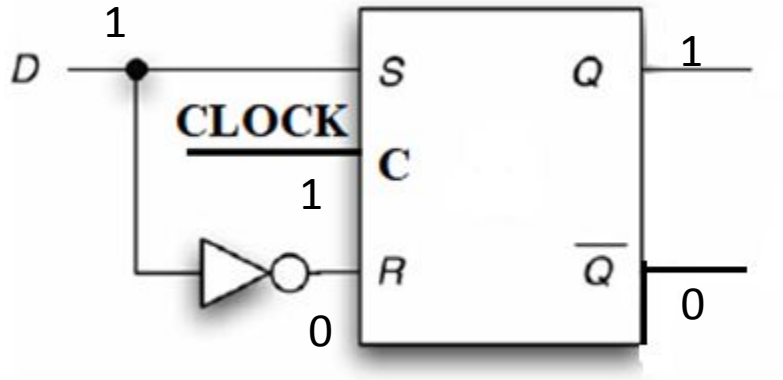
When clock=1, D=0



INPUTS		OUTPUTS		COMMENTS
S	R			
0	0	Q	**Q**	Memory
0	1	0	1	Reset
1	0	1	0	Set
1	1	?	?	Forbidden

WORKING OF A D-TYPE BISTABLE

When clock=1, D=1

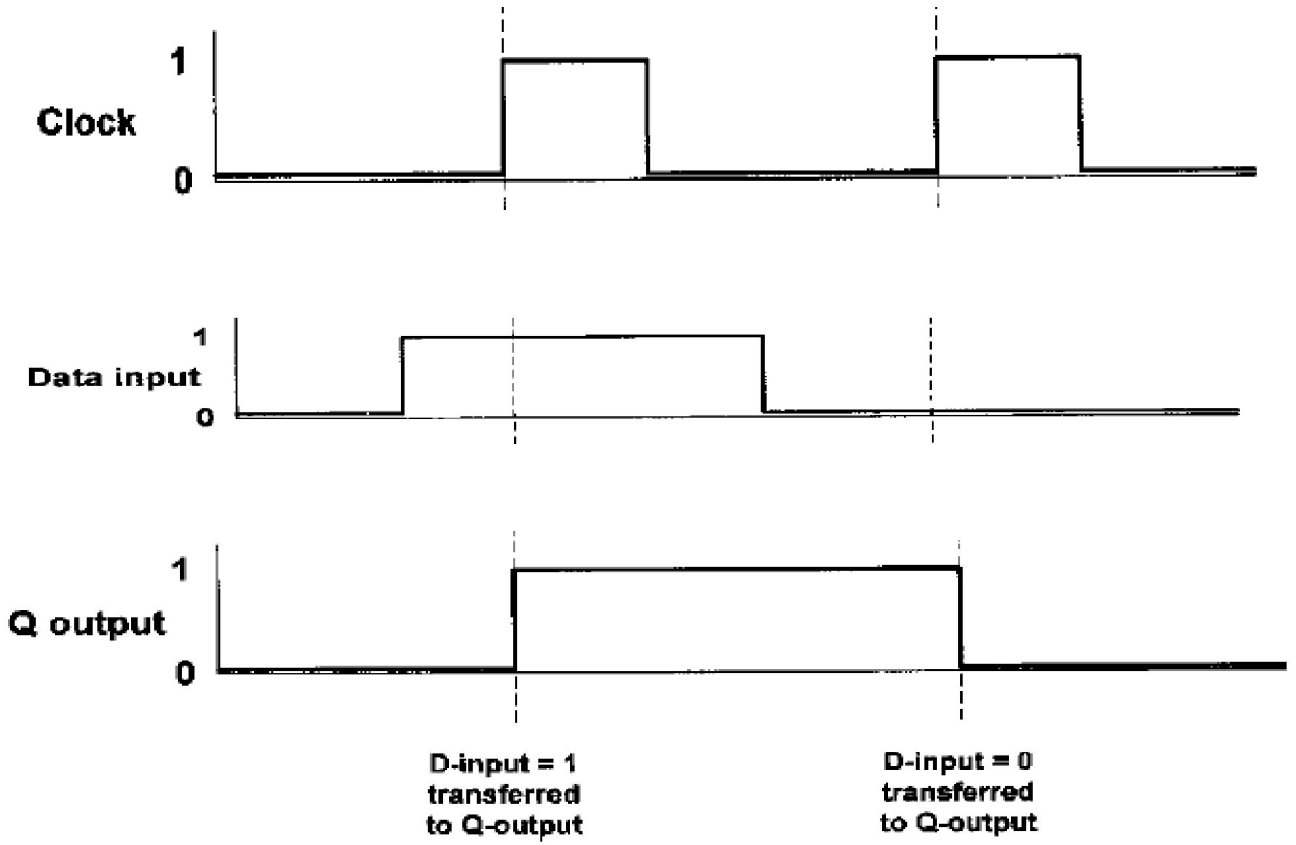


INPUTS		OUTPUTS		COMMENTS
S	R			
0	0	Q	**Q**	Memory
0	1	0	1	Reset
1	0	1	0	Set
1	1	?	?	Forbidden

TRUTH TABLE OF D-TYPE BISTABLE

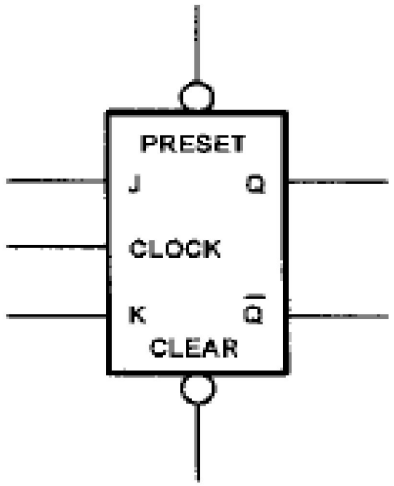
INPUTS		OUTPUTS		COMMENTS
CLOCK	D			
0	X	Q	$\overline{\mathbf{Q}}$	Memory
1	0	0	1	Reset
1	1	1	0	Set

TIMING DIAGRAM FOR A D-TYPE BISTABLE



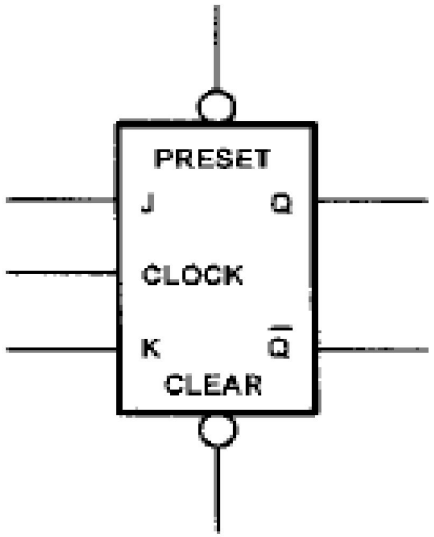
J-K BISTABLES

- J-K bistables have
  - two clocked inputs (J and K),
  - two direct inputs (PRESET and CLEAR),
  - a CLOCK input,
  - outputs (Q and $\overline{Q}$).



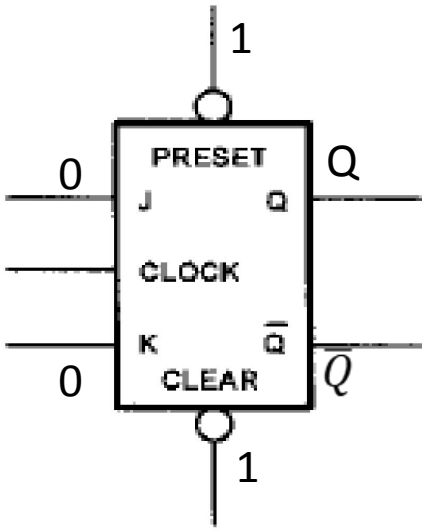
- A 0 on the PRESET input will set the Q output to 1 whereas a 0 on the CLEAR input will set the Q output to 0

INPUT AND OUTPUT STATES (PRESET AND CLEAR INPUTS)



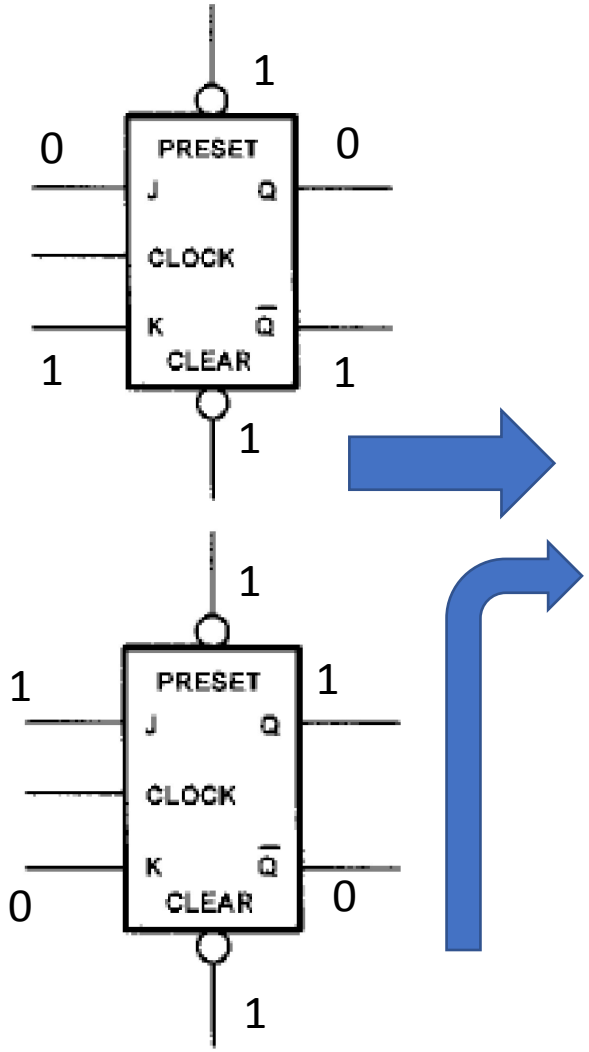
INPUTS		OUTPUT	COMMENTS
CLEAR	PRESET		
0	0	?	Indeterminate
0	1	0	Q is reset
1	0	1	Q is set
1	1	Clocked operation	Enables clocked operation

INPUT AND OUTPUT STATES (CLOCKED OPERATION)



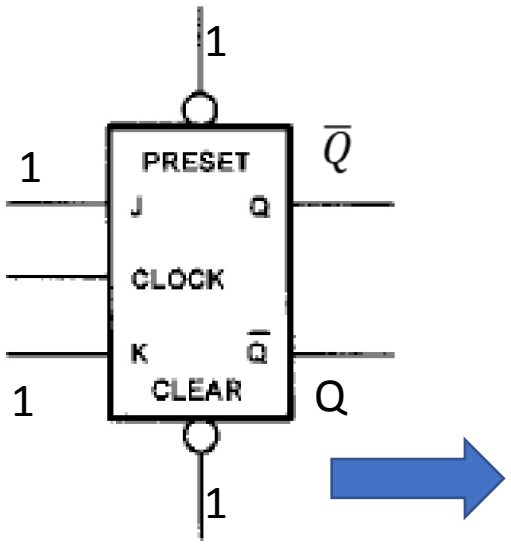
INPUTS		OUTPUT	COMMENTS
J	K		
0	0	Q	No change in state of the Q output
0	1	0	Q is reset
1	0	1	Q is set
1	1		Q output changes to the opposite state

CLOCKED OPERATION OF A J-K BISTABLE



INPUTS		OUTPUT	COMMENTS
J	K		
0	0	Q	No change in state of the Q output
0	1	0	Q is reset
1	0	1	Q is set
1	1		Q output changes to the opposite state

CLOCKED OPERATION OF J-K BISTABLE

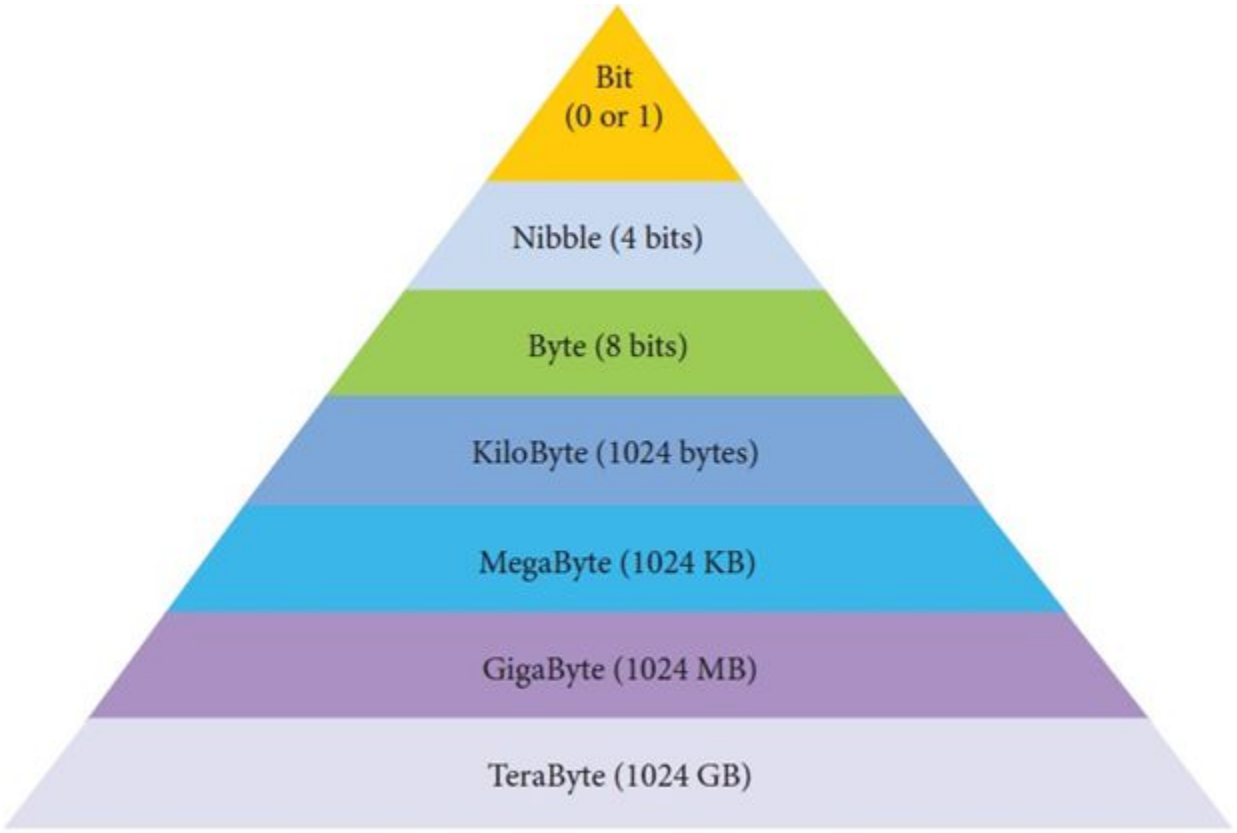


INPUTS		OUTPUT	COMMENTS
J	K		
0	0	Q	No change in state of the Q output
0	1	0	Q is reset
1	0	1	Q is set
1	1		Q output changes to the opposite state

J-K Bistables can be configured in various ways

- Binary Dividers
- Binary Counters
- Shift Registers

Data Representation and Microcontroller System

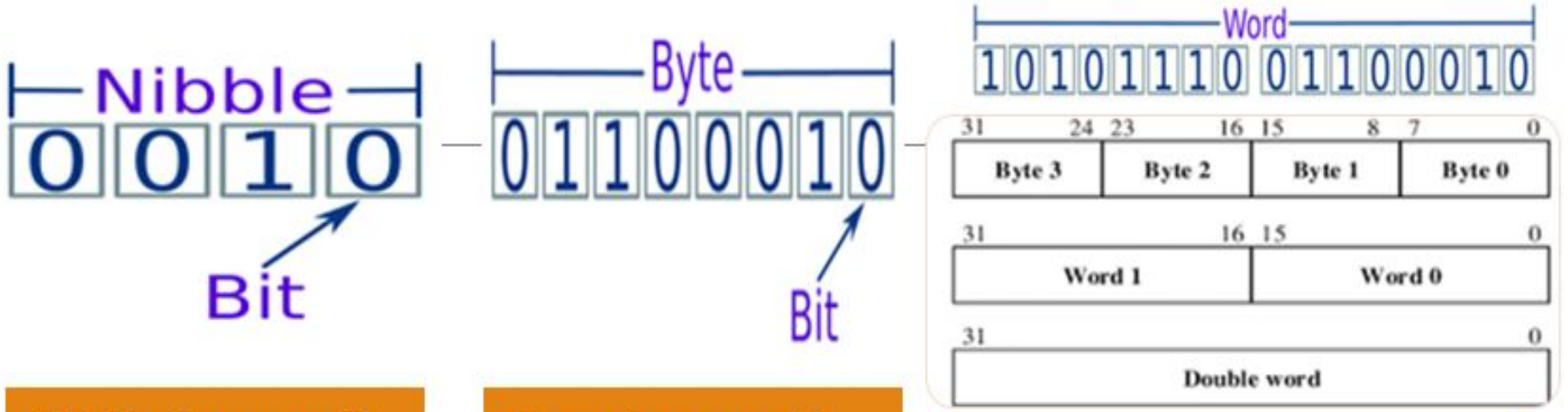


Data Representation

Data Representation

- **Binary(Base 2)**
  - Large binary numbers are inconvenient to handle
  - Hence convert to hexadecimal
- **Hexadecimal(Base 16)**
  - Easy to comprehend
  - Offers advantage over decimal (base 10)
  - Conversion from binary to hexadecimal and vice versa is easy
  - A single hexadecimal character(range is 0 to F) represents 4 binary digits

Nibble, byte, word, double word



Nibble-A group of 4 bits or single hex character

Byte- A group of 8 bits

Represented by 2 hex characters

WORD-A group of 16 bits

Represented by four hexadecimal characters

Double word –A group of 32 bits

Represented using 8 hex characters

groups of four bits (nibble)-

- Each hexadecimal number is

=100 in decimal

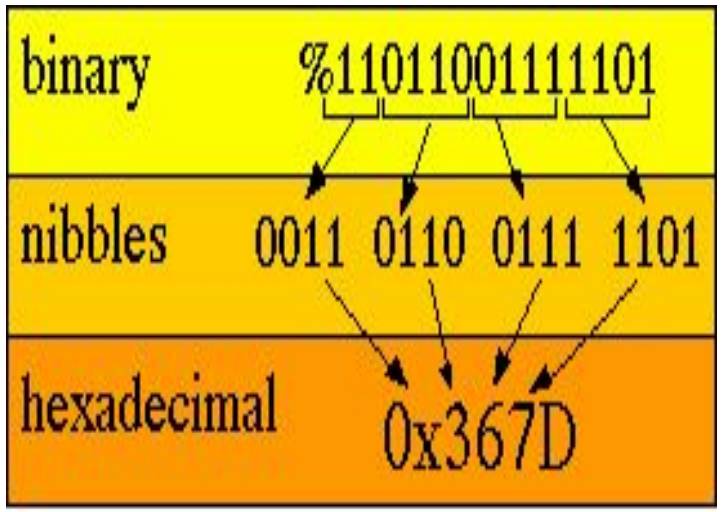
$7FH =

add H to the end of the number

- Ex. A3 in hex = 1010 0011 in binary
- 11101000

Binary, Denary, Hex Table

Binary	Denary	Hexadecimal
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F



Data Types

- One byte of data can be stored at each address of the memory space in a microprocessor
- For a 16 bit address bus, total memory locations =65,536
- Within a byte, bits are numbered 0 (Least Significant Bit) to 7(Most Significant Bit)
- For 16 bit words, 15 is the MSB bit
- Negative(signed) numbers are represented using 2's complement notation.
- Here MSB indicates the sign of the number(1=negative, 0= positive)
- Only bits 0 to 6 represent the magnitude of the number
- Ex: signed 8 bit number 10000001 = -1(denary)

Range of Integer Data values

Data type	Bits	Range of values
Unsigned byte	8	0 to 255
Signed byte	8	-128 to +127
Unsigned word	16	0 to 65,535
Signed word	16	-32,768 to +32,767

Data Storage

- Semiconductor ROM
  - Provides storage for the program code
  - Stores any permanent data
  - These are non-volatile
  - They remain intact when power supply is off

- Semiconductor RAM
  - Provides storage for transient data, variables used by the program
  - Part of RAM temporarily stores data while doing normal tasks

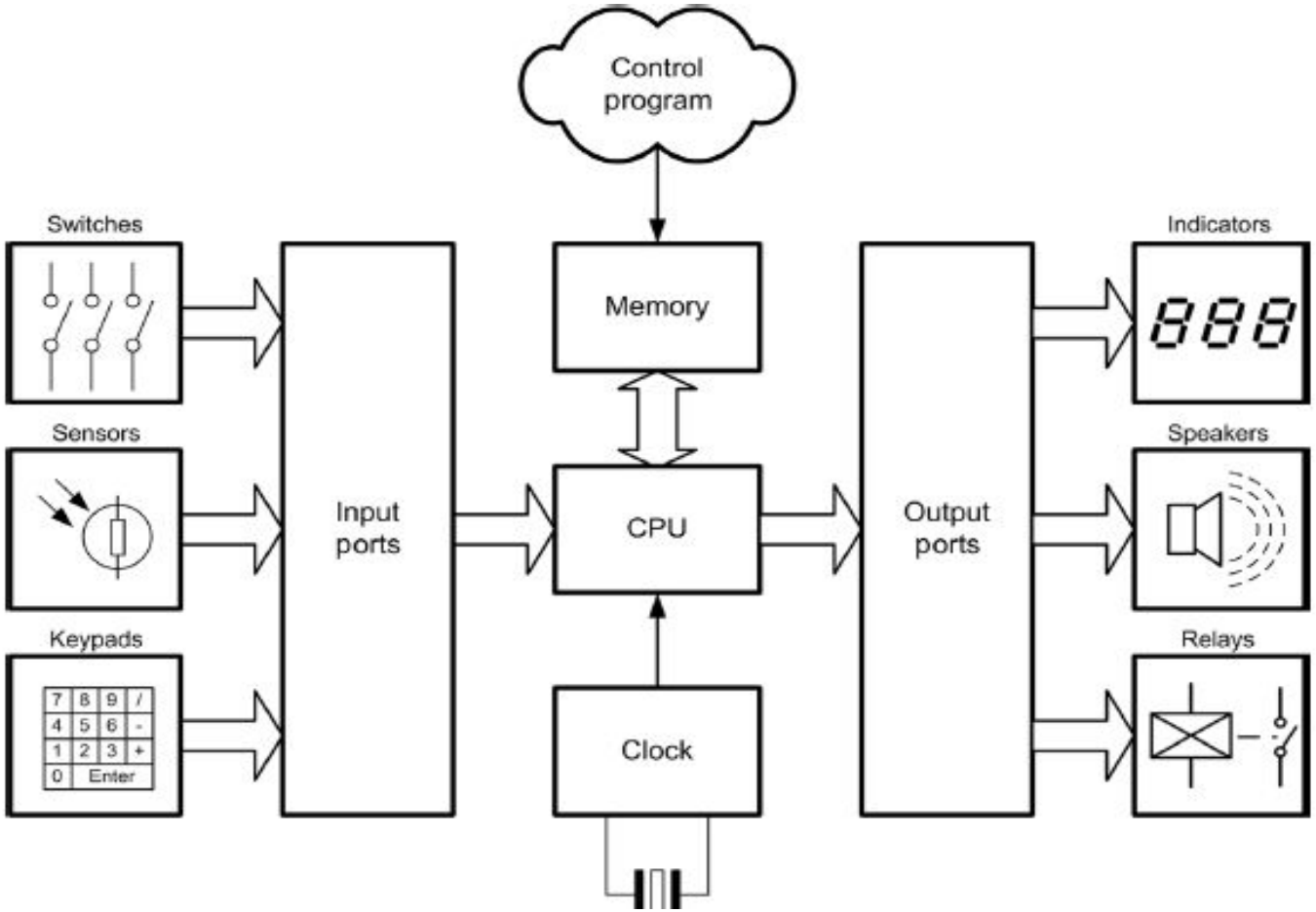
- CMOS RAM
  - Data stored in RAM is lost when power switched off
  - In CMOS RAM, data is kept alive using a small battery
  - It retains important data- time and date

Data Storage

- The data stored in the memory device is represented in kilobytes(KB)
- 1 Kilobyte = 1024 bytes . It is nearest power of 2.(2^{10} =1024)
- Semiconductor ROM capacity = address range * number of bits stored at each address
- Ex: 2k x 8 bits(2KiloBytes) , 4k x 8 bits (4 Kilobytes)

Microcontroller system

Microcontroller system



Typical Microcontroller system

- The sensed quantities(temperature, position) are converted to electrical signals using sensors
- Sensor outputs are either digital or analogue in nature
- These signals passed to microcontroller. Microcontroller also accepts user options
- Software instructions control the operation of the microprocessor and output signals sent to controlled devices
- Controlled device converts electrical energy from one form into another form
- In real world, systems are self regulating called closed loop system

  Ex. Heating control system

Typical Microcontroller system

- Micro controllers have a CPU which performs arithmetic, logical and timing operations
- Some input sources are switches, sensors, keypads
- Some output devices are LED indicators, LED 7 segment displays, motors and actuators, relays, transistor drivers and other solid-state switching devices

Input devices

- Input device is keyboard, mouse scanner and modem
- Switches or contacts which make and break
- Sensors which provide logic level outputs
- To connect an input device to a microcontroller, the device must provide a logic compatible signal.
- In microcontroller inputs can only accept digital input signals

  0v= logic 0 and 5v = logic 1 signal
- Some devices sense analogue quantities
- Some microcontrollers provide ADC and provide digital data.
- Resolution of the ADC depends on the number of bits used (8, 10 or 12 bits)

Output devices

- They are used to communicate information to the outside world
- Common output devices are flat screen display, printers and modems
- Microcontroller uses output devices such as LEDs, piezoelectric sounders, relays and motors
- DAC are required at the output of the microcontroller
- The resolution of the DAC also depends on the number of bits

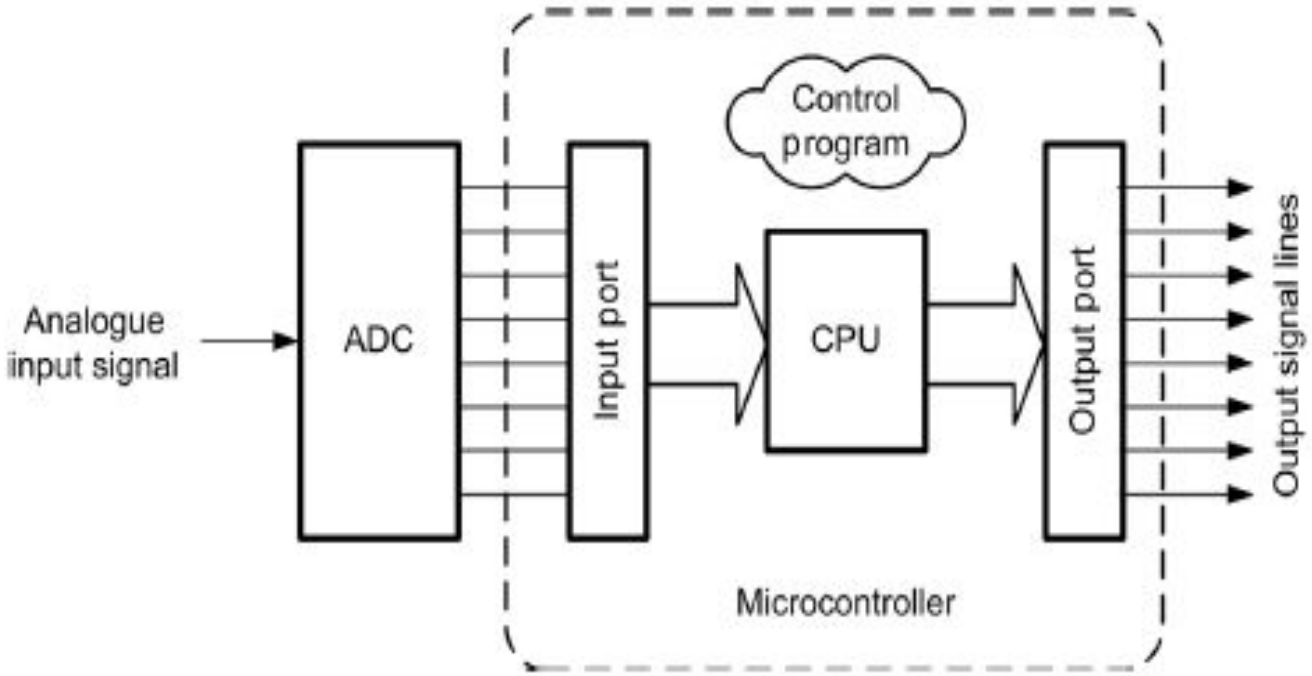
Interface Circuit

- When the output and input signals are not logically compatible interface circuits are needed
- Also, when a load requires more current than is available from standard logic device or output port
- They provide the additional current drive

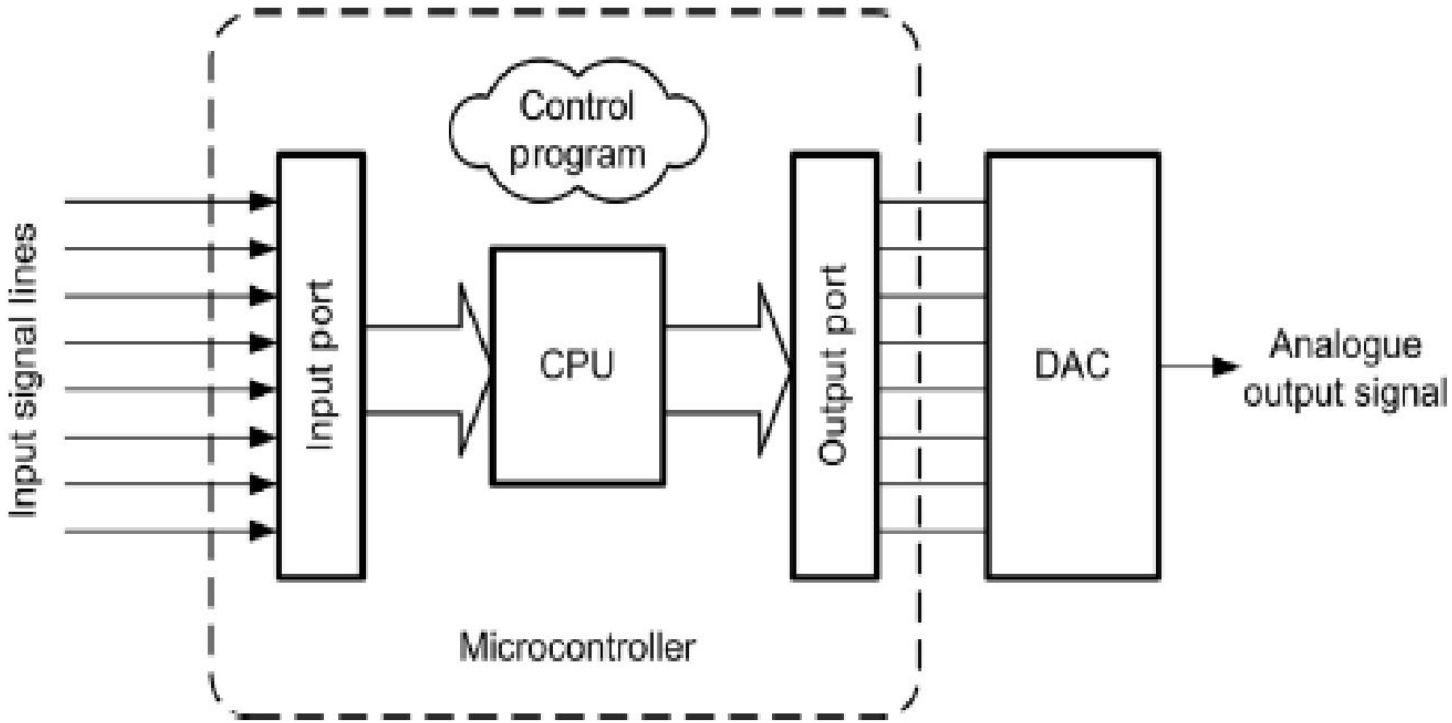
Ex: Interface circuits allow a microcontroller to interface to an AC mains

connected load. To control a central heating system

Ex: An analog input signal connected to a microcontroller via ADC



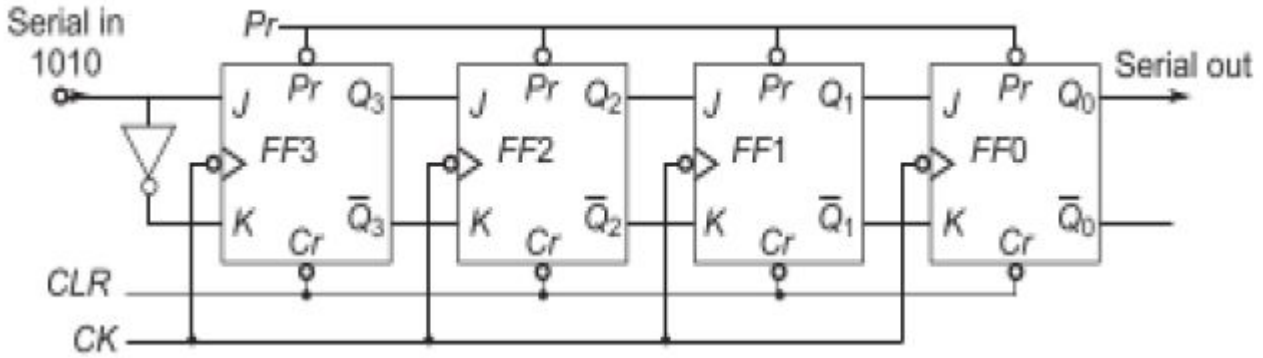
Ex: An analog output signal produced by connecting DAC to microcontroller



Shift Register

- Registers are the devices used to store multiple bits of information
- Flip-flops are used to store 1-bit of information, for n-bit of information to be stored in the registers, registers use n-number of flip-flops.
- Using the shift registers along with storing the information, stored information can be read parallel or series.
- Shift registers are used to write/read the information either in series(1-bit at a time) or in parallel (all bits of data at a time).

Block diagram of shift register (SISO):



- JK flip-flop with active low Preset and Clear is used to construct the 4-bit shift register.
- Negative edge of the clock is used to trigger the operation of the flip-flops
- 4-bit data of 1010 is sent to the shift register serially (1-bit per clock) through serial in. Data 1010 is fed in with LSB first
- Each bit is shifted to the through the 4 flip-flops and output through serial out after the occurrence of 4 negative edges of the clock.

Operation of Shift Register:

Clock pulse	Serial in	Q_3	Q_2	Q_1	Q_0 (serial out)	
0	0	0	0	0	0	
1	1	1	0	0	0	
2	0	0	1	0	0	
3	1	1	0	1	0	Data entered
4	0	0	1	0	1	
5	0	0	0	1	0	
6	0	0	0	0	1	
7	0	0	0	0	0	1 Register cleared

Register types

- Serial In Serial Out(SISO)
- Serial In Parallel Out (SIPO)
- Parallel In Serial Out(PISO)
- Parallel In Parallel Out(PIPO)

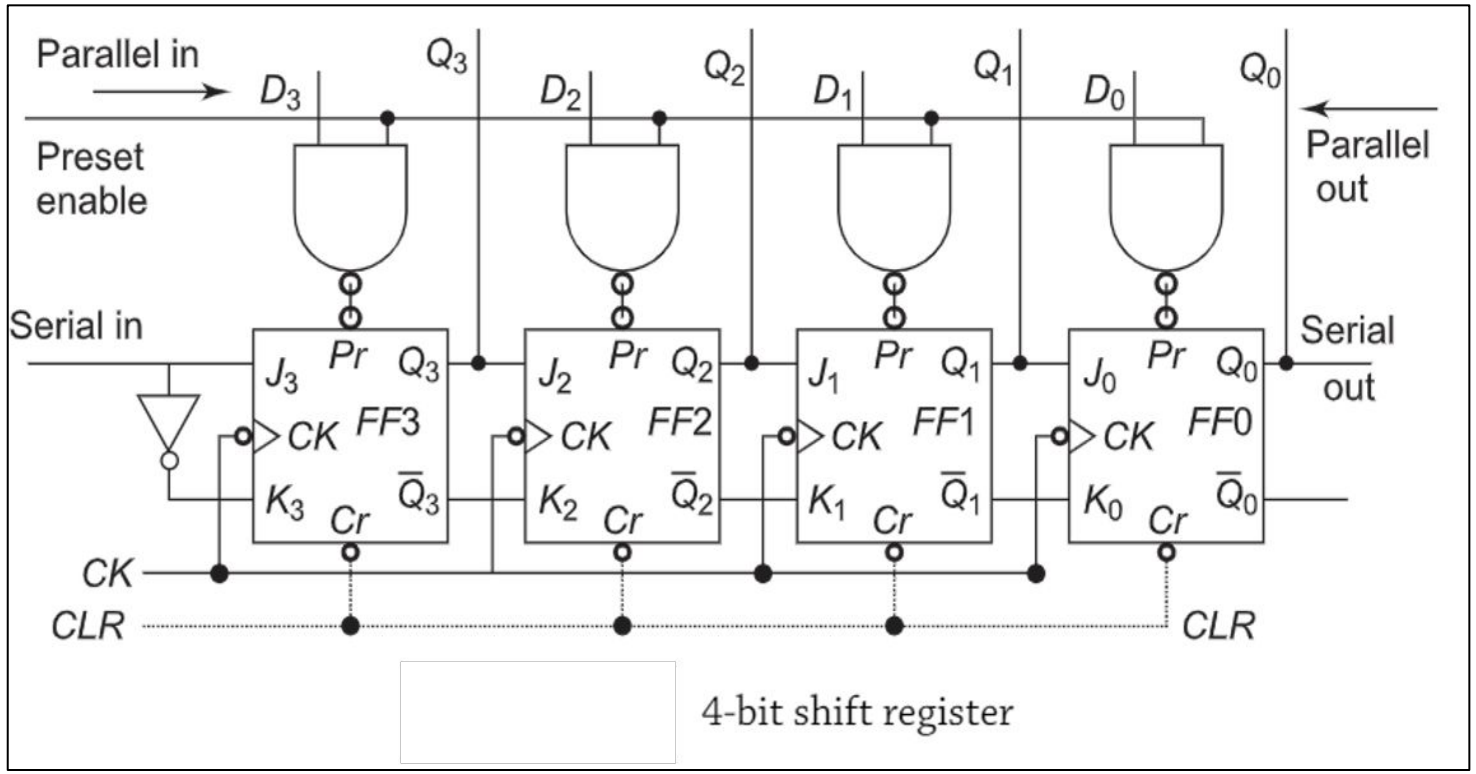
Register types

- Parallel out – output connections taken from Q3 to Q0
- Parallel in – After clearing the FFs, "Parallel in" is set '1'. Data fed simultaneously on D3-D0

Any D_i input causes

$D_i = 0, P_r = 1$ and $Q_i = 0$

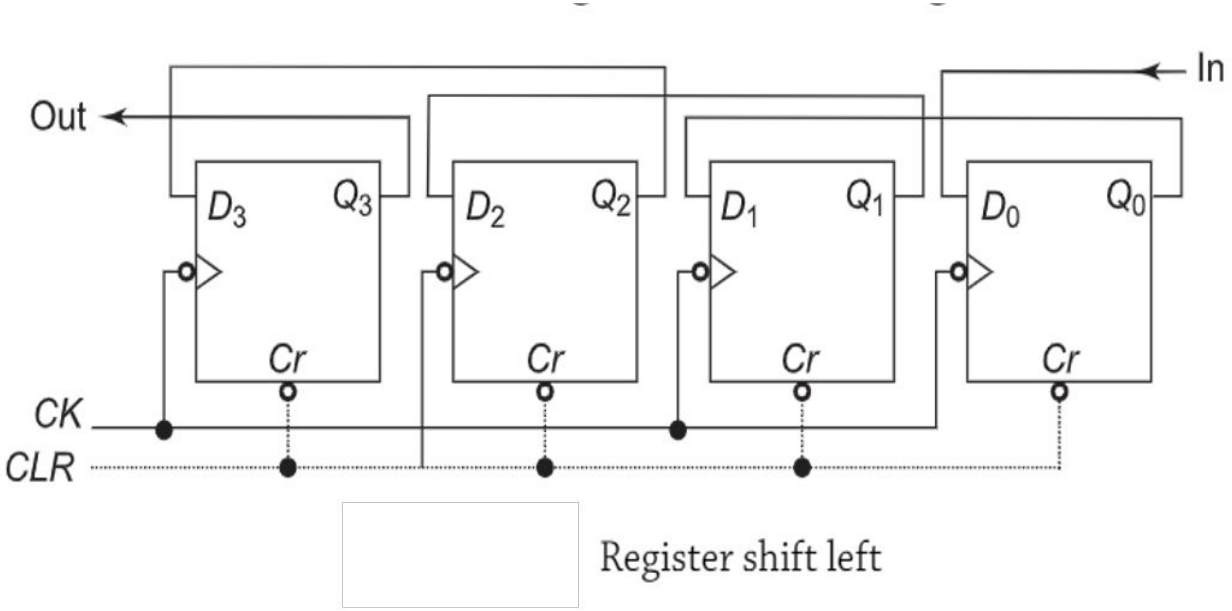
$D_i = 1, P_r = 0$ and $Q_i = 1$



4-bit shift register

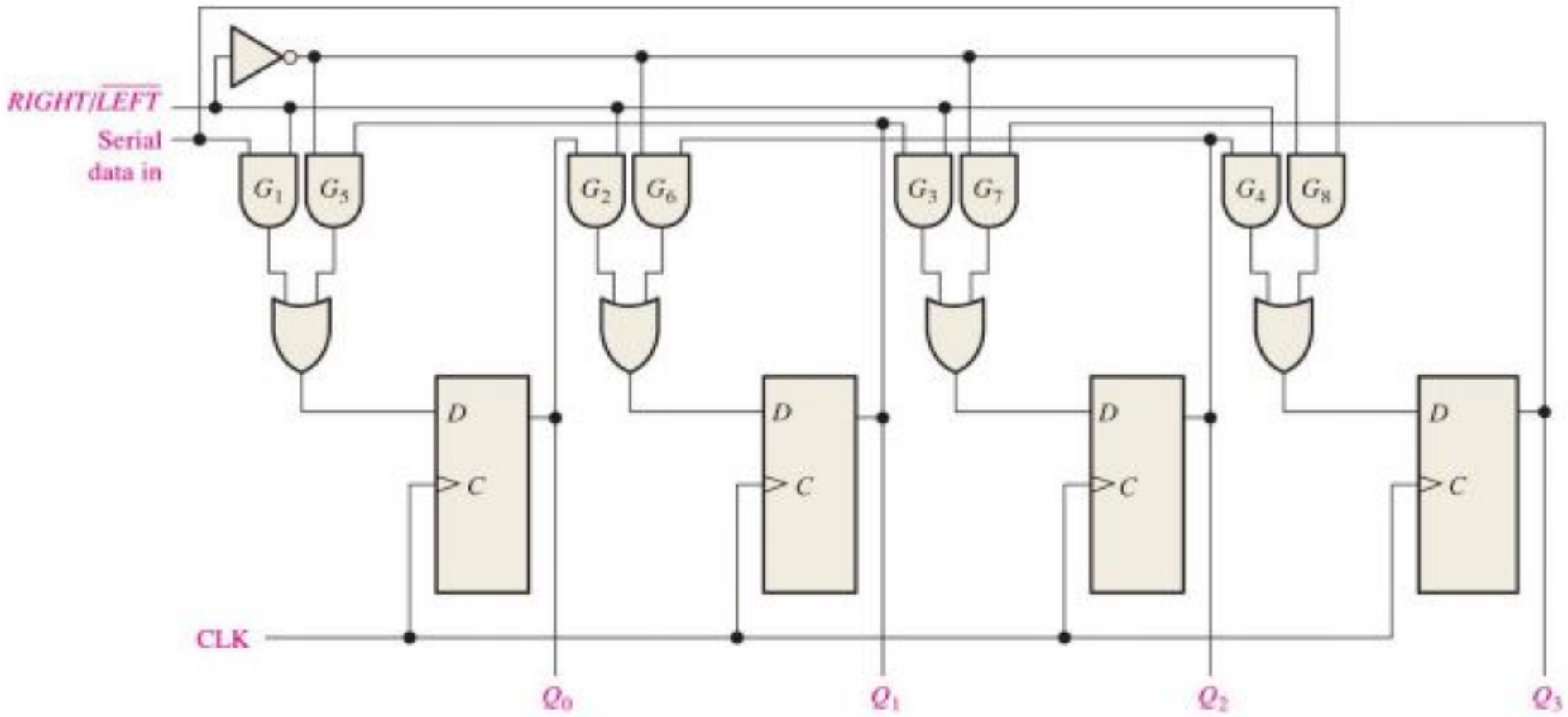
Shifting of Data to Left

- Previously, the data was shifted from left by one bit at clock pulse by feeding output at FF0 to input FF1, and FF1 to FF2, FF2 to FF3.
- In the below fig, On the occurrence of a pulse, the data word will shift one bit to left.



Register shift left

Bidirectional Shift Registers



Counters

- A counter is a circuit that counts the number of occurrence of an input.
- Each count, a binary number, is called a state of the counter.
- A counter counting in terms of **n-bits** has **2^n** different states.
- The number of different states of a counter is known as the modulus of the counter. Thus an n-bit counter is a **modulo-2^n** counter.
- Counter circuits are primarily constituted of **flip-flops**, along with **combinational elements** are used for the generation of counter signals.

Counters

Counter can be divided into two major categories:

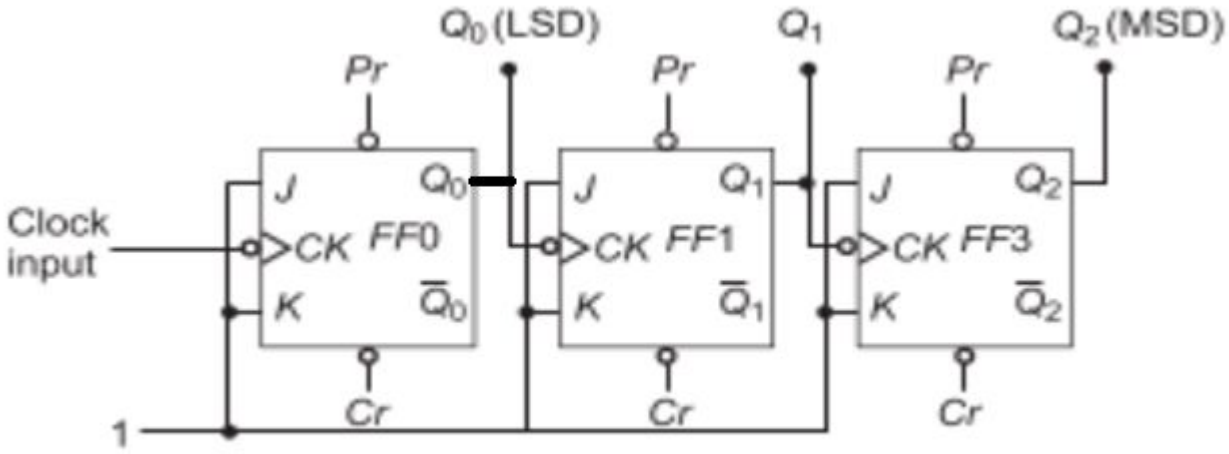
1. **Asynchronous counter (Ripple counter)**: all the flip-flops are not clocked simultaneously.
2. **Synchronous counter:** all the flip-flops are clocked simultaneously.

Asynchronous Counters

The below figure shows a 3-bit (modulo 8) asynchronous counter.

Operation: All the flip flops are initially cleared. The clock pulses are then applied.

- With each negative edge of the clock pulse (1 to 0), Q0 toggles.
- For each negative edge (1 to 0) of Q0, Q1 toggles.
- Similarly for 1 to 0 transition of Q1 , Q2 toggles.



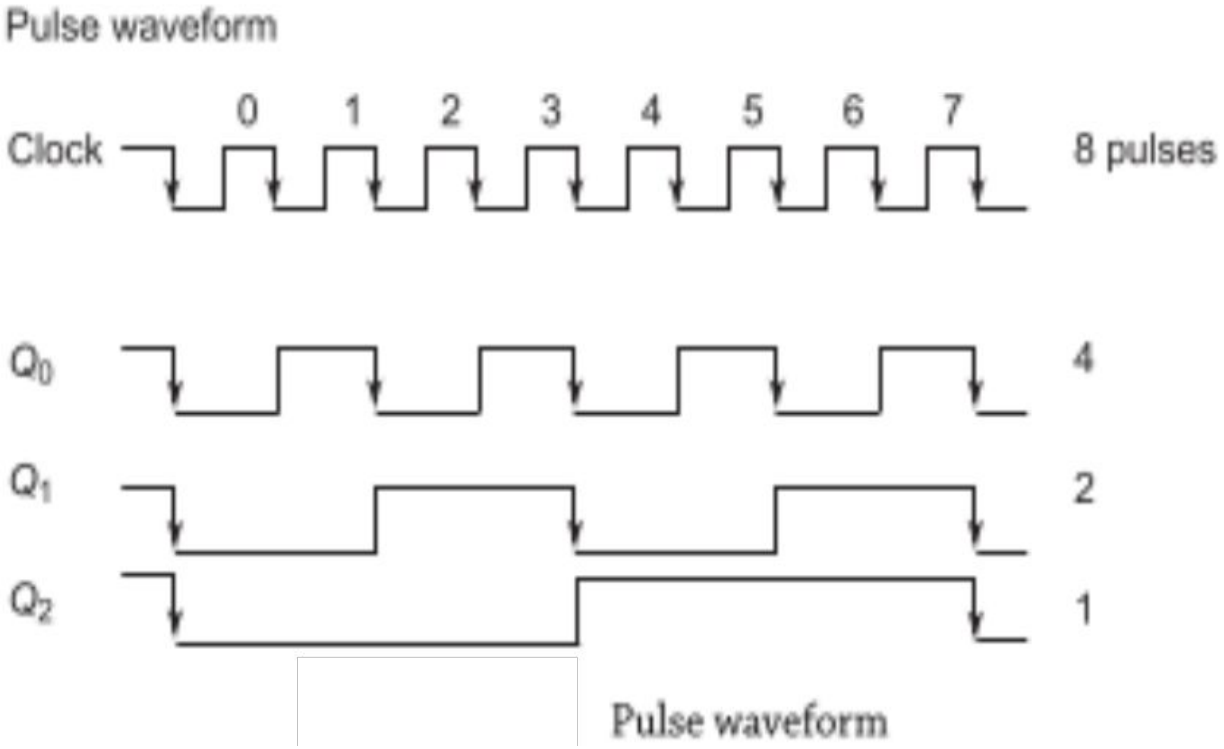
3-bit asynchronous counter (UP)

Operation of Asynchronous Counter

Input (Clock)	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

Observation

- The 3-bit counter counts the 8-clock pulses, negative edge occurs at Q_0 4 times, Q_1 2 times and Q_3 1 time.
- The frequency of the clock pulse reduces by a factor of 2 at each stage.



Counters

- **Down counter**: $\overline{Q}s$ are connected to the next flip-flop. As $\overline{Q}$ toggles from 1 to 0, the corresponding Q toggles 1 to 0, which is count down.

- **Up-down counter**: For up/down count, CK is connected to Q and $\overline{Q}$ through AND-OR logic.