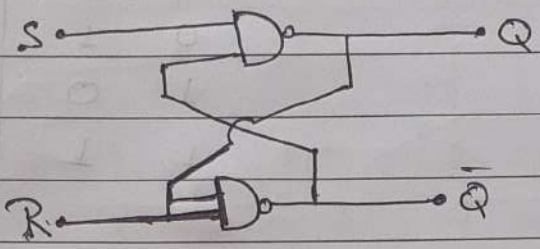
ELECTRONIC CIRCUIT AND SWITCHING

__/__/____

FLIP FLOPS:

* SR NAND Latch.



S	R	Q	$\bar{Q}$
		Invalid	
0	0	1	1
0	1	1	0
1	0	0	1
1	1	Memory.	

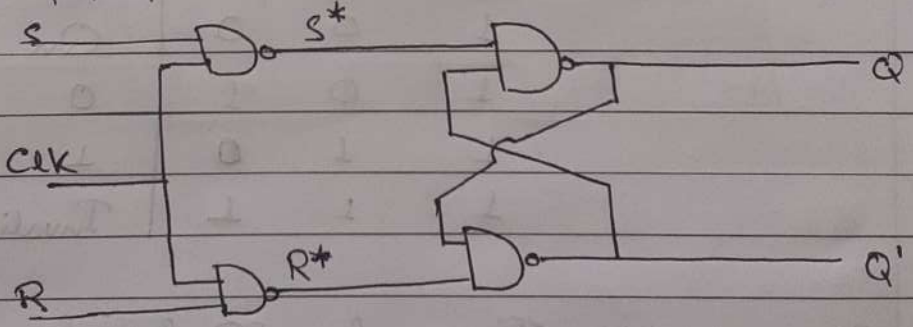
T.T. for NAND Gate

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

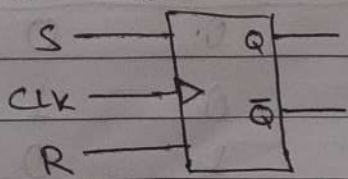
here, S is eqv. to A

i.e, when A is 0, output is always 1
when A is 1 & b is 0, output is 1
when A is 1 & b is 1, output is 0.
And, when B is 0, output is always 1.

SR flipflop:



Also, represented as,



$S^* = \bar{S} + \overline{Clk}$

$R^* = \bar{R} + \overline{Clk}$

Clk	S	R	Q	$\bar{Q}$
0	x	x	Memory	
1	0	0	Memory	
1	0	1	0	1
1	1	0	1	0
1	1	1	Invalid	

when Clk=0, $S^*=1$, $R^*=1$ } Always memory.

when Clk=1; S=0; R=0; $S^*=1$, $R^*=1$ } This is memory state.

Clk=1; S=0, R=1; $S^*=1$, $R^*=0$ } → Q=0; $\bar{Q}=1$ } Reset state.

Clk=1; S=1, R=0; $S^*=0$, $R^*=1$ } → Q=1; $\bar{Q}=0$ } set state

Clk=1; S=1, R=1; $S^*=0$, $R^*=0$ } → Invalid.

__/__/____

** T.T. for SR NAND LATCH:

S	R	Q	$\bar{Q}$	
0	0	Invalid		
0	1	1	0	→ Set
1	0	0	1	→ Reset
1	1	Memory.		

T.T. for SR flipflop:

Clk	S	R	Q	$\bar{Q}$	
0	x	x	Memory		
1	0	0	Memory		
1	0	1	0	1	→ Reset
1	1	0	1	0	→ Set
1	1	1	Invalid		

* Also written as

Clk	S	R	Q_{n+1} (Next State)
0	x	x	Q_n (Present State)
1	0	0	Q_n (or Memory)
1	0	1	0
1	1	0	1
1	1	1	Invalid

* Characteristic Table for SR flip flop.

Q_n	S	R	Q_{n+1}	
0	0	0	0	→ (why zero?)
0	0	1	0	
0	1	0	1	
0	1	1	Invalid	
1	0	0	1	→ (why one?)
1	0	1	0	
1	1	0	1	
1	1	1	Invalid	

These two are memory states Q_n. So whatever the value of Q_n, will be value of Q_{n+1}

__/__/____

* Excitation Table for SR flip flop.

Q_n	Q_{n+1}	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

↳ How?

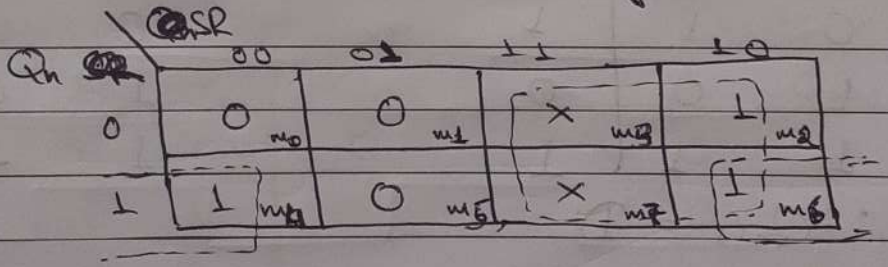
Looking at characteristics table, when $Q_n = 0$ & $Q_{n+1} = 0$, S is always 0; R is either 1 or 0.

Similarly for other cases, when $Q_n = 0$ & $Q_{n+1} = 1$; S is 1 and R is 0

when $Q_n = 1$ & $Q_{n+1} = 0$; S is 0 and R is 1

when $Q_n = 1$ & $Q_{n+1} = 1$; S is either 0 or 1 & R is always 0

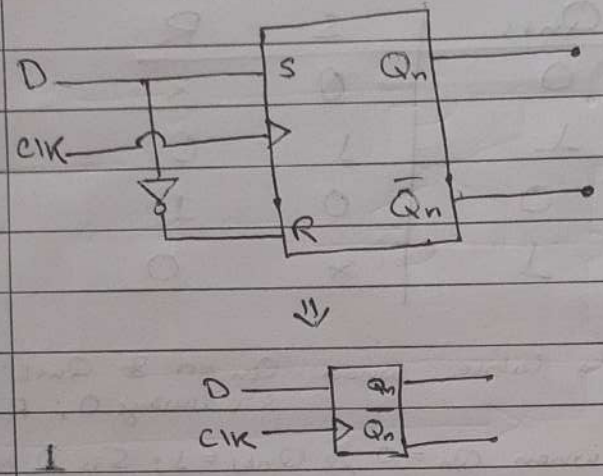
In characteristics table, making its K-Map for Q_{n+1}



$\therefore Q_{n+1} = S + Q_n\bar{R}$

__/__/____

* D-flipflop:



T₀ T₀

CLK	D	Q_{n+1}
0	x	Q_n
1	0	0
1	1	1

Characteristics Table

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

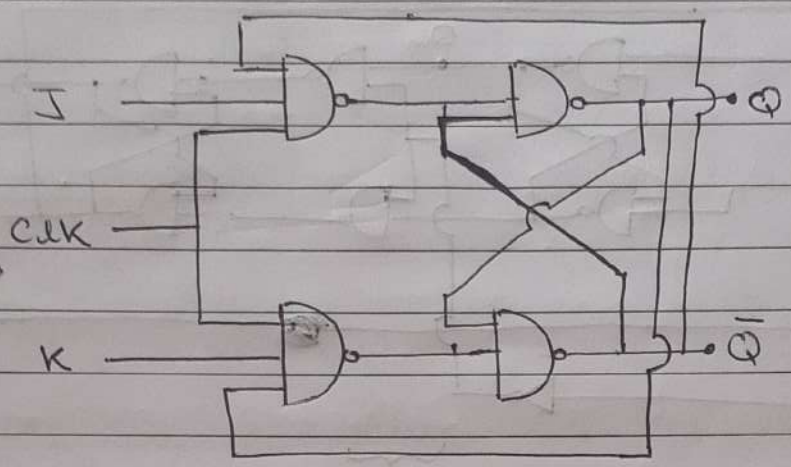
$\therefore Q_{n+1} = D$

Excitation Table.

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

__/__/____

* JK flip-flop



To T◦

Clk	J	K	Q_{n+1}
0	x	x	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	$\bar{Q}_n$

$\bar{Q}_n$ is known as Toggle

Characteristics Table.

Q_n	J	K	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Here, $Q_{n+1} = \bar{Q}_n \cdot J + Q_n \cdot \bar{K}$

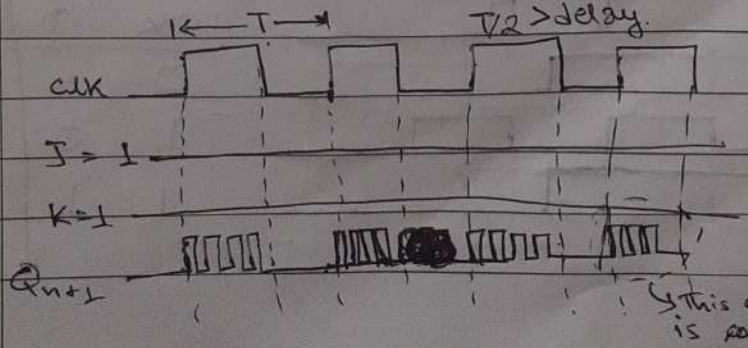
Excitation Table.

Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Here, $J = Q_{n+1}$

$K = \overline{Q_{n+1}}$

Racing.



Conditions to overcome Racing

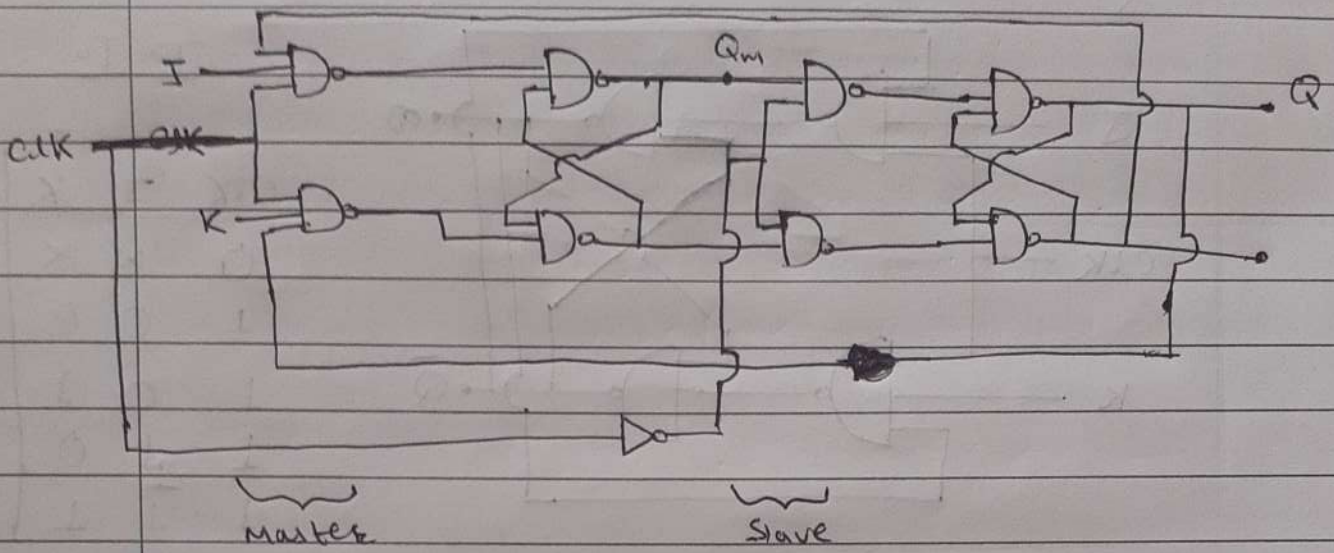
(i) T/2 < propagation delay of f.f.

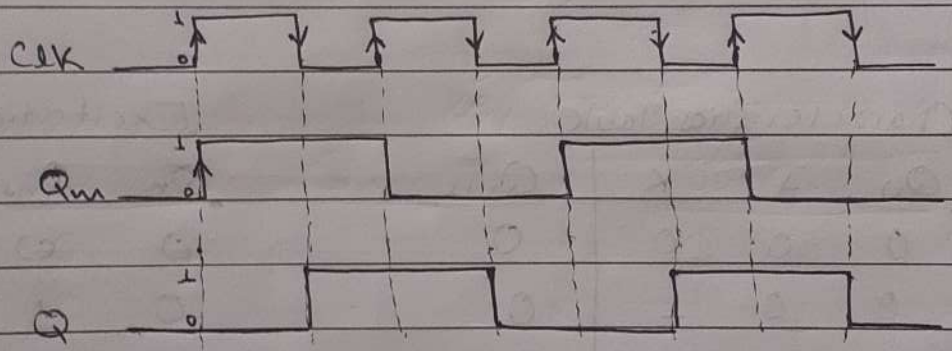
(ii) edge Triggering

(iii) Master-Slave

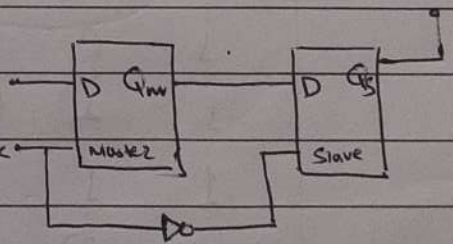
__/__/____

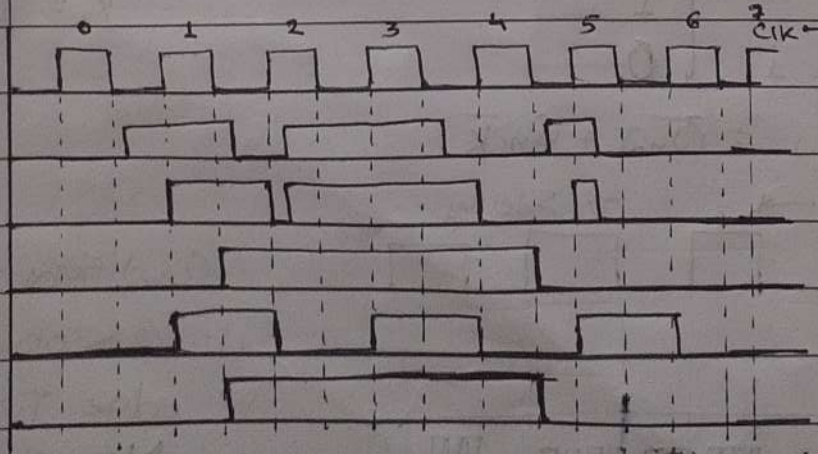
* Master-Slave Operation of JK flip flop





* Master-Slave D flip flop:

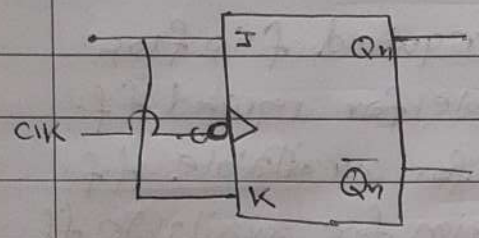




Clk	D	Q_{n+1}
0	x	Q_n
1	0	0
1	1	1

__/__/____

* T - flipflop :



T.T.

CLK	T	Q_{n+1}	
0	x	Q_n	} Memory state.
1	0	Q_n	
0	1	$\overline{Q_n}$	(Toggling)

Characteristics table.

Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Excitation ~~State~~ Table

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

$Q_{n+1} = Q_n \oplus T$

XOR

* Flip-flop Conversion.

STEPS:
1. Identify available and required flipflop.
2. Make characteristic table for required f.f.
3. Make excitation table for available ff.
4. Write boolean expression for available ff.
5. Draw the circuit (make available ff first and make req. changes)

Example. JK flipflop to D flipflop conversion.

Here,

Available f.f. is JK.

Required ff. is D.

Characteristic table for D ff.

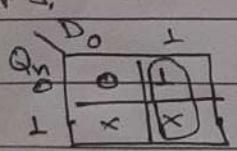
Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

Excitation table for JK ff.

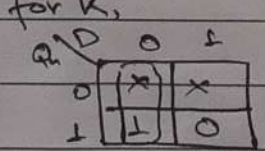
Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

using this

for J,

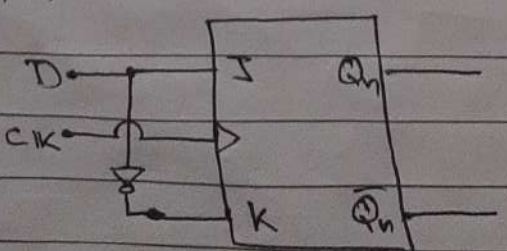


for K,



$\therefore$ $J = D$ $K = \bar{D}$ are req. boolean exp. for JK ff.

Circuit for D flipflop:



* T flipflop to D flipflop Conversion.

Here, Available ff. is T

Required ff. is D.

Characteristic table for D ff

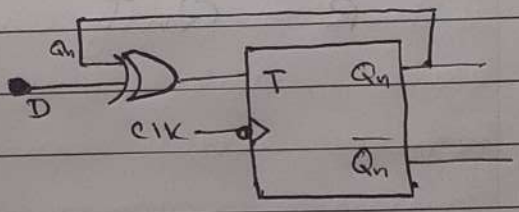
Q_n	D	Q_{n+1}	T
0	0	0	0
0	1	1	1
1	0	0	1
1	1	1	0

So, $T = D \oplus Q_n$

Excitation table for T ff.

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Circuit:



* SR → JK Conversion.

Available ff. is SR

Required ff is JK

Characteristic table for JK ff.

Q_n	J	K	Q_{n+1}	S	R
0	0	0	0	0	x
0	0	1	0	0	x
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	x	0
1	0	1	0	0	1
1	1	0	1	x	0
1	1	1	0	0	1

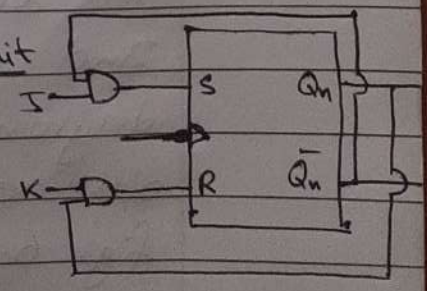
$S = \overline{Q_n} J$

$R = Q_n K$

Excitation table for SR ff.

Q_n	Q_{n+1}	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

Circuit



__/__/____

* SR → T flip flop Conversion.

Available ff is SR

Req. ff is T

Characteristic table for req. ff. T

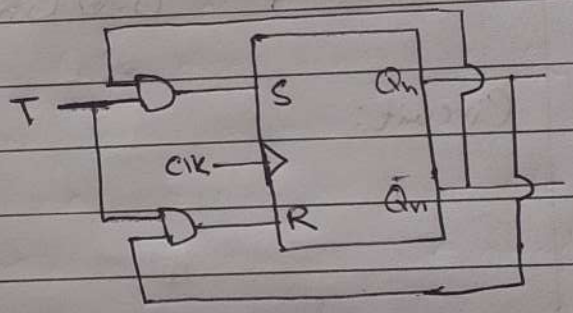
Q_n	T	Q_{n+1}	S	R
0	0	0	0	x
0	1	1	1	0
1	0	1	x	0
1	1	0	0	1

Excitation table for SR

Q_n	Q_{n+1}	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

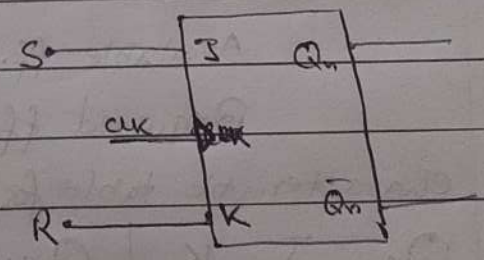
$S = \bar{Q}_n T$

$R = Q_n T$



* JK to SR : J = S

K = R



RESULT:

The truth table for Asynchronous Parallel Counter and its timing diagram were obtained as shown in the table 6.1 and fig. 6.7, respectively. And the truth table and the timing diagram for Synchronous Parallel Counter were obtained as shown in the table 6.2 and fig. 6.8, respectively

__/__/____

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

CACHAR, ASSAM

LABORATORY EXERCISE BOOK

B.TECH. IIIRD SEM.

NAME: SUBHOJIT GHIMIRE

SCH. ID.: 1912160

BRANCH: CSE - 'B'

SUBJECT: CIRCUIT AND SWITCHING LAB

CODE : EC-222

18/11/2020
Experiment-01

AIM: TO ANALYSE TRUTH TABLE OF BINARY TO GRAY AND GRAY TO BINARY CONVERTER USING COMBINATION OF NAND GATES AND TO UNDERSTAND THE WORKING OF BINARY TO GRAY AND GRAY TO BINARY CONVERTER WITH THE HELP OF LED DISPLAY

THEORY:

Binary Numbers is a default way to store numbers. On the other hand, Gray Code has property that two successive numbers differ in only one bit; because of this property Gray Code does the cycling through various states with minimal effort and is used in K-maps, error correction, communication etc. In computer science, many a times we need to convert binary code to gray code and vice versa. This conversion can be done by applying following rules.

1) Binary to Gray Conversion:

i. The MSB of Gray Code is always equal to the MSB of the given binary code.

ii. Other bits of the output in the gray code can be obtained by XORing binary code bit at that index and previous index.

There are four inputs and four outputs. The input variables are defined as B3, B2, B1, B0 and the output variables are defined as G3, G2, G1, G0. From the truth table, combinational table is designed. The logical expressions are defined as:

$B3 = G3$

$B2 \oplus B3 = G2$

$B1 \oplus B2 = G1$

$B0 \oplus B1 = G0$

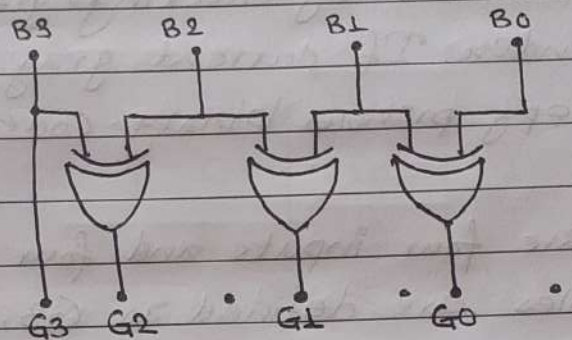


fig.10 Binary to Gray Code Converter Circuit

B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

fig.11 Binary to Gray Code Truth Table

__/__/____

2) Gray to Binary Conversion

i. The MSB of the binary code is always equal to the MSB of the given binary number.

ii. Other bits of the output binary code can be obtained by checking gray code bit at that index. If current gray code bit is 0, then copy previous binary code bit.

There are four inputs and four outputs. The input variables are defined as G3, G2, G1, G0 and the output variables are defined as B3, B2, B1, B0. From the truth table, combinational circuit is designed. The logical expressions are defined as:

$$G0 \oplus G1 \oplus G2 \oplus G3 = B0$$

$$G1 \oplus G2 \oplus G3 = B1$$

$$G2 \oplus G3 = B2$$

$$G3 = B3$$

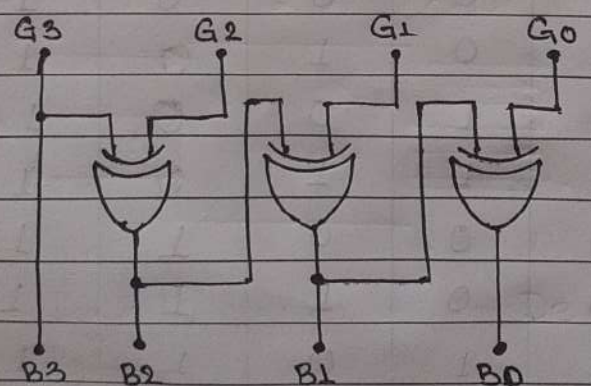


fig. 16: Gray to Binary Code Converter Circuit

__/__/____

G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
1	0	0	0	1	1	1	1
1	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	0	1	0	1	1
1	1	1	1	1	0	1	0

fig.1(iv): Gray to Binary Code Truth Table

RESULT:

The truth table for binary code to gray code conversion was obtained as shown in fig. (ii).

The truth table for gray code to binary code conversion was obtained as shown in fig. (iv).

CONCLUSION:

After entering all the values correctly, we got the Gray codes from the binary values.

18/11/2020
Experiment-02

AIM: To Analyse the Truth Table of 4x2 Decoder (De-multiplexer) using NOT (7404) and AND (7408) logic gate ICs and 2x4 Encoder using OR (7403) logic gate IC and to understand the working of 4x2 Decoder and 2x4 Encoder circuit with the help of LEDs display.

THEORY:

Binary Code of N digits can be used to store 2N distinct elements of coded information. This is what encoders and decoders are used for. Encoders convert 2N lines of input into a code of N bits and Decoders decode the N bits into 2N lines.

1) 2x4 Decoder (De-multiplexer):

A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines.

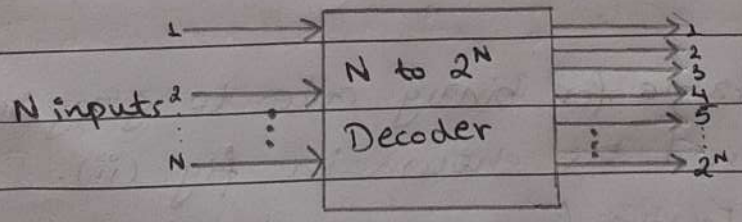


fig.2(1): Diagram of a Decoder.

- The 2 to 4 line decoder consists of an array of four AND gates. The 2 binary inputs labelled A and B are decoded into one of the 4 outputs, hence the description of 2 to 4 binary decoder. Each

output represents one of the minterms of the 2 input variables i.e, each output = 1 minterm.

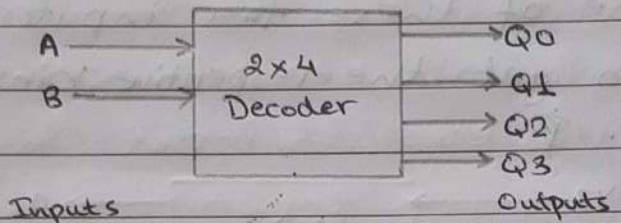


fig 2(ii): A 2×4 binary decoder.

A	B	Q0	Q1	Q2	Q3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

fig. 2(iii): Truth Table of 2×4 Decoder.

2) 4:2 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.

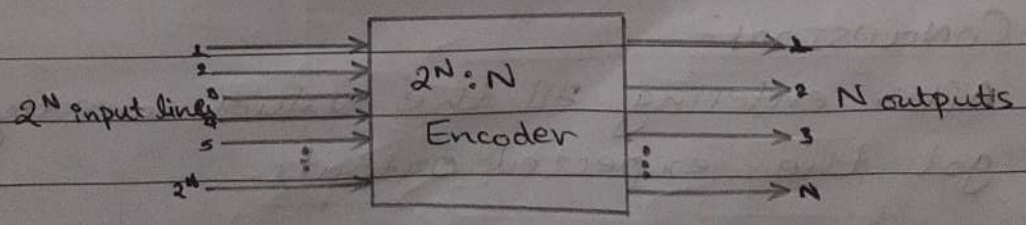


fig. 2(iv): Diagram of an Encoder.

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

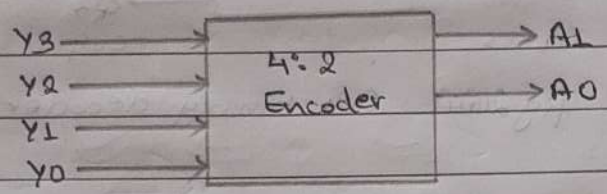


fig. 2(v): A 4:2 binary encoder

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

fig. 2(vi): Truth Table of 4:2 Encoder.

RESULT:

The truth table for 2x4 Decoder was obtained as shown in fig. 2(iii)

The truth table for 4:2 Encoder was obtained as shown in the fig. 2(vi)

CONCLUSION:

After entering all the values correctly, we got the expected outputs.

25/11/2020
Experiment-03

AIM: TO VERIFY THE TRUTH TABLE AND TIMING DIAGRAM OF RS, JK, T AND D FLIP-FLOPS BY USING NAND AND OR GATES ICs AND ANALYSE THE CIRCUIT OF RS, JK, T AND D FLIP-FLOPS WITH THE HELP OF LEDs DISPLAY.

THEORY:

A flip flop is an electronic circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications and many more.

1. RS Flipflop: The basic NAND gate RS flipflop circuit is used to store the data and thus provides feedback from both of its outputs again back to its inputs. The RS flipflop has three inputs - SET, RESET and its current output Q relating to its current state.

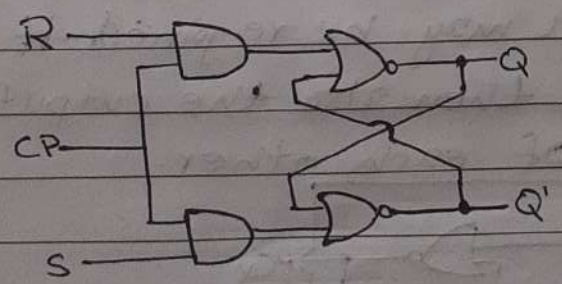


fig-3.1.: RS flipflop

CLK	S	R	Q	STATE
×	0	0	No Change	Previous
↑	0	1	0	Reset
↑	1	0	1	Set
↑	1	1	-	Forbidden

fig. 3.2: Characteristics table of RS flip flop.

__/__/____

2. D flipflop: A D flipflop has a single data input. This type of flipflop is obtained from the SR flipflop by connecting the R input through an inverter, and the S input is connected directly to data input. This modified clocked SR flipflop is known as D flipflop.

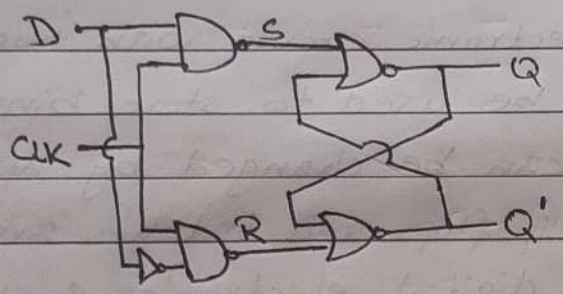


fig.: 3.3: D flipflop

D	reset	CLK	Q	Q'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1

fig.: 3.4: Characteristics table of D flipflop

3. JK flipflop: In a RS flipflop, the input R=S=1 leads to an indeterminate output. The RS flipflop circuit may be re-joined if both the inputs are 1 then also the outputs are complement of each other.

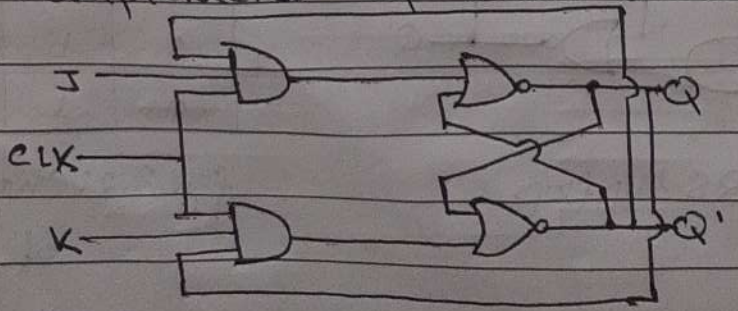
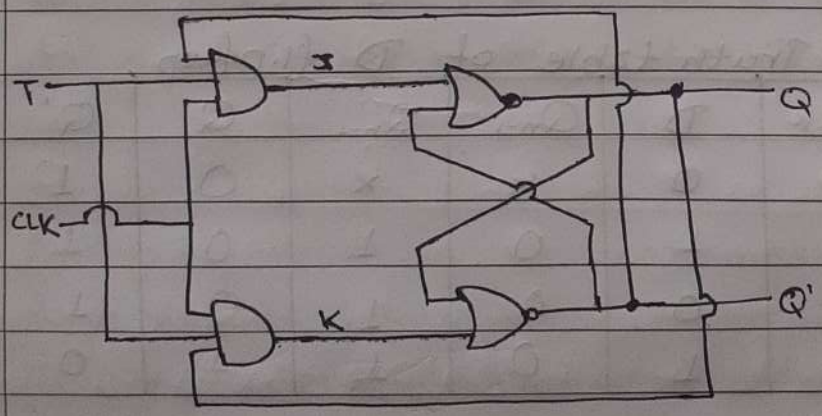


fig.: 3.5: JK flipflop

CLK	J	K	Q_n	Q_{n+1}	Inference
0	x	x	—	—	Latched
1	0	0	0	0	No change
1	0	0	1	1	No change
1	0	1	0	0	Reset
1	0	1	1	0	Reset
1	1	0	0	1	Set
1	1	0	1	1	Set
1	1	1	0	1	Toggle
1	1	1	1	0	Toggle

fig.: 3.6 : Characteristics table of JK flipflop

4. T Flipflop : T flipflop is also known as toggle flipflop. The T flipflop is modification of the JK flipflop. Both the JK flipflop are held at logic 1 and the clock signal continuous to change.



CLK	T	Q	Q'
1	0	Q	Q'
1	1	Q'	Q
0	x	Q	Q'

fig. 3.8: Characteristics Table of T flipflop.

fig.: 3.7: T flipflop

RESULT:

The truth table and clock diagram for SR, JK, D and T flipflops were obtained as shown in Table 3.1 - 3.4 and fig. 3.13 - 3.16, respectively.

__/__/____

OBSERVATIONS:

Table 3.1: Truth table of SR flipflop

S.No.	Clock	S	R	Q_{n-1}	Q'_{n-1}	Q	Q'	Remark
1	0	0	0	x	x	0	1	No Change
2	1	0	1	0	1	0	1	Reset
3	1	1	0	0	1	1	0	Set
4	1	1	1	1	0	0	1	Invalid

Table 3.2: Truth table of JK flipflop

S.No.	Clock	J	K	Q_{n-1}	Q'_{n-1}	Q	Q'	Remark
1	0	0	0	x	x	0	1	No change
2	1	0	0	0	1	0	1	No change
3	1	0	1	0	1	0	1	Reset
4	1	1	0	0	1	1	0	Set
5	1	1	1	1	0	0	1	Toggle

Table 3.3: Truth table of D flipflop

S.No.	Clock	D	Q_{n-1}	Q'_{n-1}	Q	Q'	Remark
1	0	0	x	x	0	1	No Change
2	0	1	0	1	0	1	No Change
3	1	0	0	1	0	1	Reset
4	1	1	0	1	1	0	Set

Table 3.4.: Truth table of T flipflop

S.No.	Clock	T	Q_{n-1}	Q'_{n-1}	Q	Q'	Remark
1.	0	0	x	x	0	1	No Change
2.	1	0	0	1	0	1	No Change
3.	1	1	0	1	1	0	Toggle.

25/11/2020
Experiment-04

AIM: To Analyse The Circuit And Truth Table of 4-Bit SIPO Shift Register By Using IC7474 (D Flipflop).

THEORY:

In SIPO (Serial In Parallel Out) shift registers, the data is stored into the register serially while it is retrieved from it in parallel-fashion. Figure 4.1. shows an n-bit synchronous SIPO shift register sensitive to positive edge of the clock pulse. Here the data word which is to be stored (Data IN) is fed serially at the input of the first flipflop (D1 of FF1). It is also seen that the inputs of all other flipflops (except the first flipflop FF1) are driven by the outputs of the preceding ones, like the input of FF2 is driven by the output of FF1. In this kind of shift register, the data stored within the register is obtained as a parallel output data word (Data out) at the individual output pins of the flip-flops (Q_1 to Q_n).

Analysing on the same grounds, one can note that the n-bit input data word is obtained as an n-bit output data word from the shift register at the rising edge of the n^{th} clock pulse. This working of the shift-register can be summarised as in Table 4.1 and the corresponding waveforms as given in figure 4.2.

In the right shift SIPO shift register, data bits shift from left to right for each clock pulse. However if the data bits are made to shift from right to left in the same design, one gets a left shift SIPO shift-register as shown in figure 4.3. Nevertheless, the basic working principle remains the same except the fact that now B_n down to B_1 is stored in Q_n down to Q_1 i.e, $Q_1 = B_1$, $Q_2 = B_2$, ... $Q_n = B_n$ at the n^{th} clock pulse.

OBSERVATION TABLE:

Table 4.2.: Truth Table of 4-bit of SIPO Shift Register

S.No.	Clock	Input Data	Q_3	Q_2	Q_1	Q_0
1	0	0	0	0	0	0
2	1	1	1	0	0	0
3	2	0	0	1	0	0
4	3	0	0	0	1	0
5	4	0	0	0	0	1
6	5	0	0	0	0	0

RESULT:

The truth table for 4-Bit Serial-In Parallel-Out Shift Register was obtained as shown in the Table 4.2.

01/12/2020
Experiment-05

AIM: ANALYSIS AND SYNTHESIS OF MULTI-BIT SEQUENTIAL CIRCUITS USING SHIFT REGISTERS.

THEORY:

A REGISTER capable of shifting information either to right or left is called a shift register. In a shift register, the flip-flops are connected in such a way that the binary bits are entered into the shift register, shifted from one location to another and finally shifted out. There are different types of shift registers, namely - Serial-In-Serial-Out (SISO), Serial-In-Parallel-Out (SIPO), Parallel-In-Parallel-Out (PIPO) and Parallel-In-Serial-Out (PISO).

4-Bit Shift Register:

The SN54/74LS95B is a 4-bit Shift Register with serial and parallel synchronous operating modes. These operating modes are controlled by a mode control input (S). The serial shift right and parallel load are activated by separate clock inputs which are selected by a mode control input (S). The data is transfered from the serial or parallel D inputs to the Q outputs synchronous with the HIGH to LOW transition of appropriate clock input.

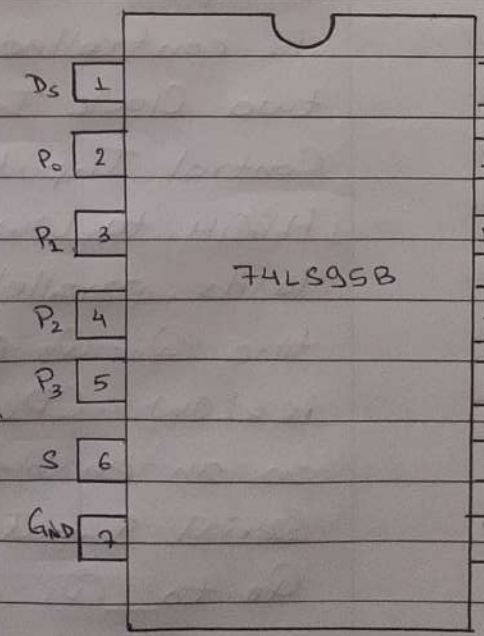


fig.: 5.1: 74LS95B, 4-bit Shift Register

PIN NAMES: (as seen in fig 5.1)

D_S : Serial Data Input

P_0-P_3 : Parallel Data Inputs

S : Mode Control Input

GND : Ground

CP_1 : Serial Clock (Active LOW Going Edge) Input

CP_2 : Parallel Clock (Active LOW Going Edge) Input

Q_0 - Q_3 : Parallel Outputs

V_{cc} : Voltage Common Collector.

FUNCTIONAL DESCRIPTION:

The SN54/74LS95B is a 4-bit Shift Register with serial and parallel synchronous operating modes. It has a Serial (D_S) and four parallel (P_0-P_3) Data inputs, and four Parallel (Q_0-Q_3) Data Outputs. The serial or parallel mode of operation is controlled by a Mode Control Input (S) and two Clock Inputs, CP_1 and CP_2. When the Mode Control Input (S) is HIGH, CP_2 is enabled. A HIGH to LOW transition on enabled CP_2 directly loads parallel data from the P_0-P_3 inputs to the Q_0-Q_3 Outputs. When the mode Control Input is LOW, CP_1 is enabled. A HIGH to LOW transition on an enabled CP_1 transfers the data from Serial Input (D_S) to Q_0 and Shifts the data in Q_0 to Q_1; Q_1 to Q_2; Q_2 to Q_3 respectively (right-shift). For normal operation, S should only change states when both Clock inputs are LOW. However, changing S from LOW to HIGH while

__/__/____

CP_2 is HIGH, or changing S from HIGH to LOW while CP_1 is HIGH and CP_2 is LOW will not cause any changes on the register outputs.

OPERATING MODES	INPUTS					OUTPUTS			
	S	CP_1	CP_2	D_s	P_n	Q_0	Q_1	Q_2	Q_3
Shift	L	⊐	x	l	x	L	q_0	q_1	q_2
	L	⊐	x	h	x	H	q_0	q_1	q_2
Parallel Load	H	x	⊐	x	P_n	P_0	P_1	P_2	P_3

figure 5.2: Truth Table

Here, (on the figure 5.2),

L: is low voltage level

H: is high voltage level

X: is don't care

l: is low voltage one set-up prior to the high to low clock transition

h: is high voltage one set-up prior to the high to low clock transition.

P_n: is the state of referenced input

RESULT:

The timing diagram was obtained as shown in figure 5.5 after inserting all the inputs. The multi bit sequential circuit was thus successfully analysed and synthesised using shift registers.

02/12/2020
Experiment-06

AIM: To verify the Truth Table and Timing Diagram of 4-bit Synchronous Parallel counter and 4-bit Asynchronous Parallel Counter by using JK flipflop ICs and analyse the circuit of 4-bit Synchronous Parallel Counter and 4-bit Asynchronous Parallel Counter with the help of LEDs Display.

THEORY:

A counter is a device which stores (and sometimes displays) the numer of time a particular event or process has occurred, often in relation to 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. Counters are broadly categorised into two - (i) Asynchronous counter.
(ii) Synchronous Counter.

1. Asynchronous Counter:

In asynchronous counter, we don't use universal clock, only first flipflop is driven by main clock and the clock inputs of rest of the following counters is driven by output of previous flipflops.

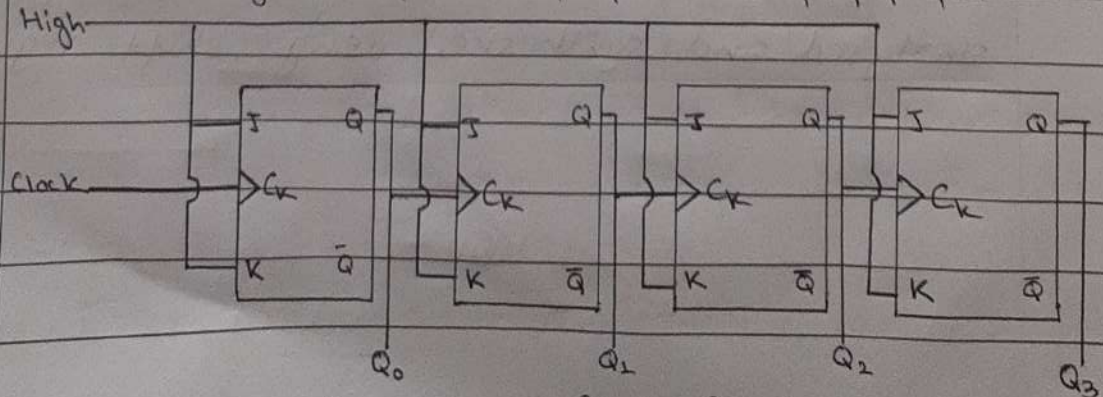


fig. 6.1: Asynchronous Counter Circuit

It is evident from the timing diagram that Q_0 is changing as soon as the rising edge of clock pulse is encountered; Q_1 is changing when rising edge of Q_0 is encountered (because Q_0 is like clock pulse for second flipflop) and so on. In this way, ripples are generated through Q_0, Q_1, Q_2 and Q_3, and hence this counter is also called RIPPLE counter.

2. Synchronous Counter:

Unlike the asynchronous counter, the synchronous counter has one global clock which drives each flipflop so output changes in parallel. The one advantage of synchronous over asynchronous counter is that it can operate on higher frequency than asynchronous counter as it does not have cumulative delay because of the same clock given to each flipflop.

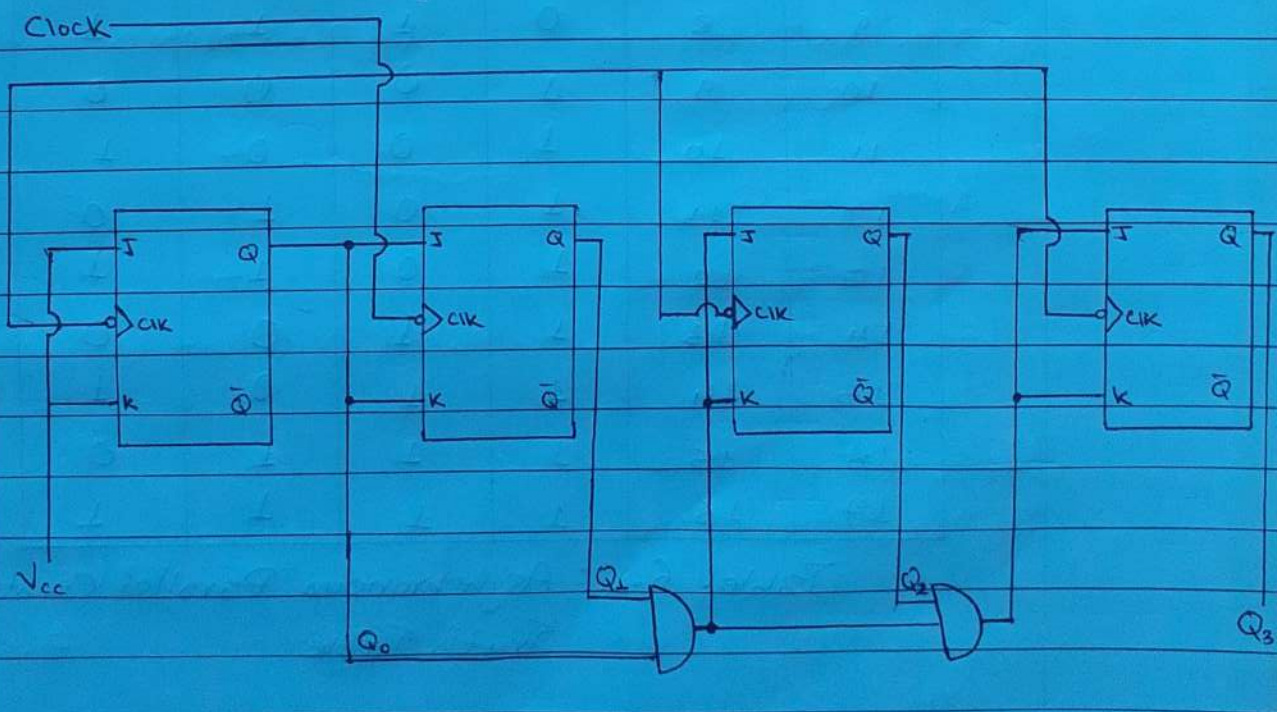


fig. 6.3: Synchronous Counter Circuit

From the timing diagram above, we see that Q_0 bit gives response to each falling edge of clock while Q_1 is dependent on Q_0; Q_2 is dependent on Q_1 and Q_0, and finally Q_3 is dependent of Q_2, Q_1 and Q_0.

OBSERVATIONS:

Serial No	Clock	Q_3	Q_2	Q_1	Q_0
1	0	x	x	x	x
2	1	0	0	0	0
3	2	0	0	0	1
4	3	0	0	1	0
5	4	0	0	1	1
6	5	0	1	0	0
7	6	0	1	0	1
8	7	0	1	1	0
9	8	0	1	1	1
10	9	1	0	0	0
11	10	1	0	0	1
12	11	1	0	1	0
13	12	1	0	1	1
14	13	1	1	0	0
15	14	1	1	0	1
16	15	1	1	1	0
17	16	1	1	1	1

Table: 6.1: Asynchronous Parallel Counter Truth Table

Table 6.2: Synchronous Parallel Counter Truth Table

09/12/2020
Experiment-07

AIM: ANALYSIS AND SYNTHESIS OF SEQUENTIAL CIRCUITS USING BASIC FLIP-FLOPS.

THEORY:

The logic circuits whose outputs at any instant of time depend not only on the present input ~~by~~ but also on the past outputs are called sequential circuits. The simplest kind of sequential circuit which is capable of storing one bit of information is called latch. The operation of basic latch can be modified, by providing an additional control input that determines, when the state of the circuit is to be changed. The latch with additional control input is called the flip-flop. The additional control input is either the clock or enable input. There are four basic types of flip-flops:

1) SR flipflop
2) JK flipflop
3) D flipflop
4) T flipflop

1) SR Flipflops: The SR flipflop basically consists of two NOR gates and also two NAND gates. It can also be made using only NAND gates.

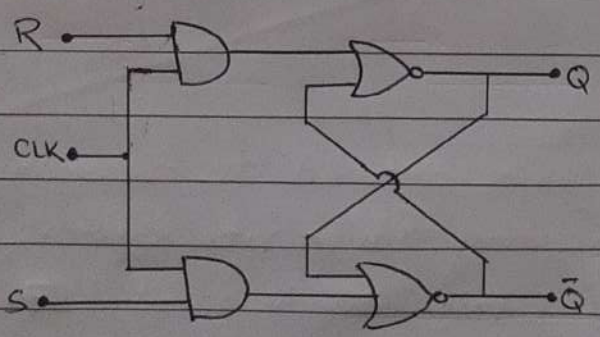


fig. 7.1: Clocked NOR based SR flipflop

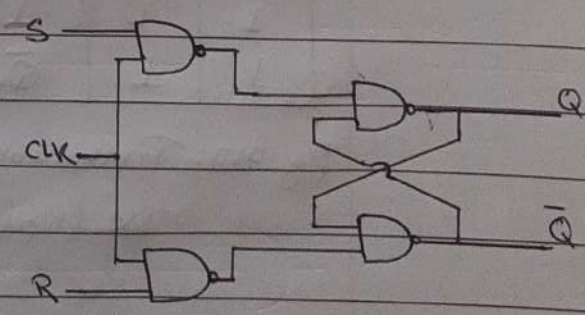


fig. 7.2: Clocked NAND based SR Flipflop

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	Not Used

fig. 7.3: Truth Table of SR flipflop

2) JK Flipflop: The JK flipflop is basically a gated SR flipflop with the addition of a clock input circuitry.

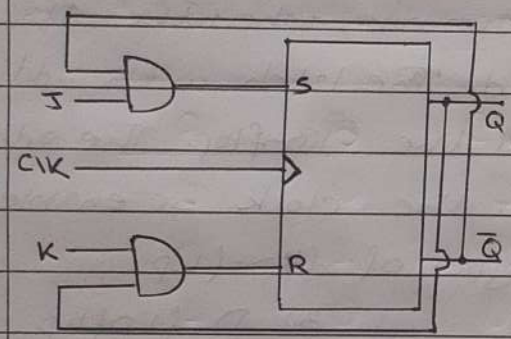


fig. 7.5: JK flipflop using SR flipflop

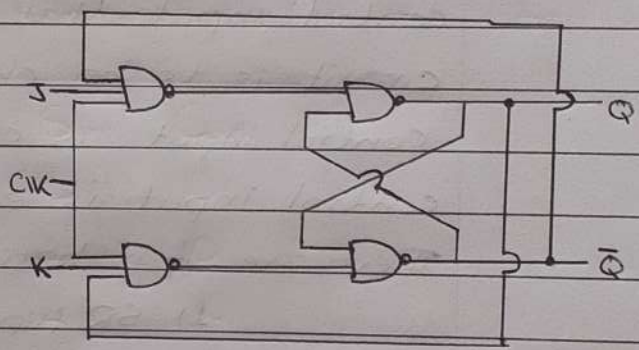
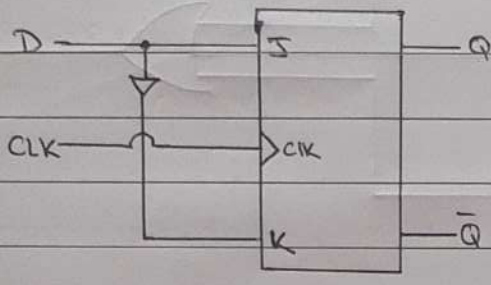


fig. 7.6: NAND based JK flipflop

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	$\bar{Q}_t$

fig. 7.7: Truth table of JK flipflop

3) D flipflops: D flipflop captures the D-input value at the specified edge (i.e., rising or falling) of the clock.



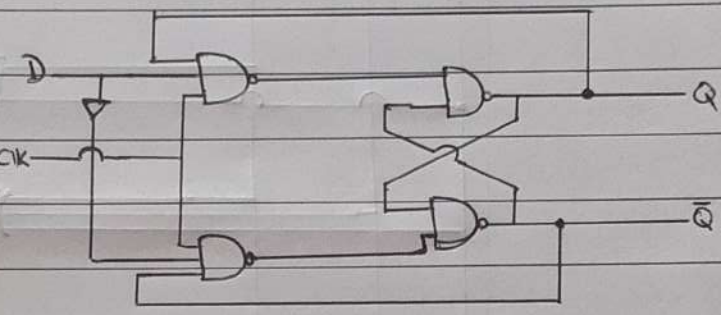


fig: 7.9: D flipflop using JK flipflop fig. 7.10: NAND-Based D flipflop

D	Q_{t+1}
0	1
1	1

fig: 7.11: Truthtable of D flipflop

Synthesis using flipflop:

We can verify the operation of a serial (sequential) adder (1 bit full adder) carry output of a one bit full adder can be fed back to the input of a D flipflop. The output of this flipflop can be fed back to the carry input of that adder.

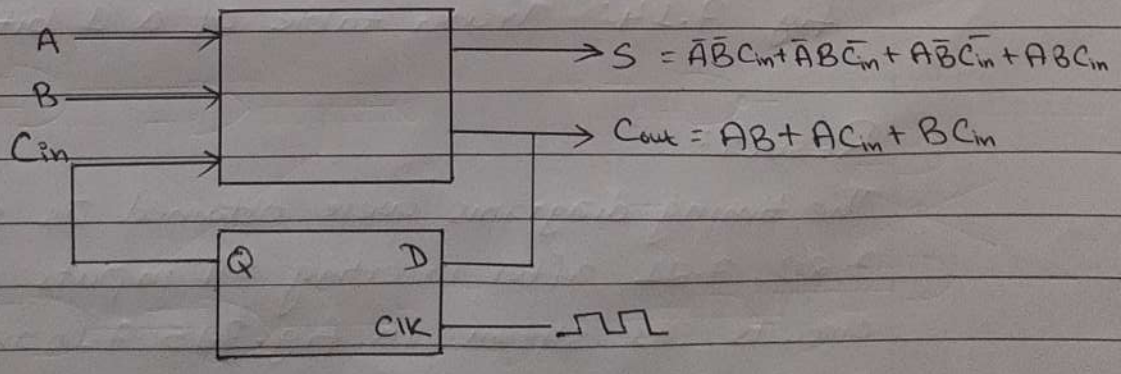


fig 7.12: Verification of the functionality of a combinational circuit using sequential element (flipflop)

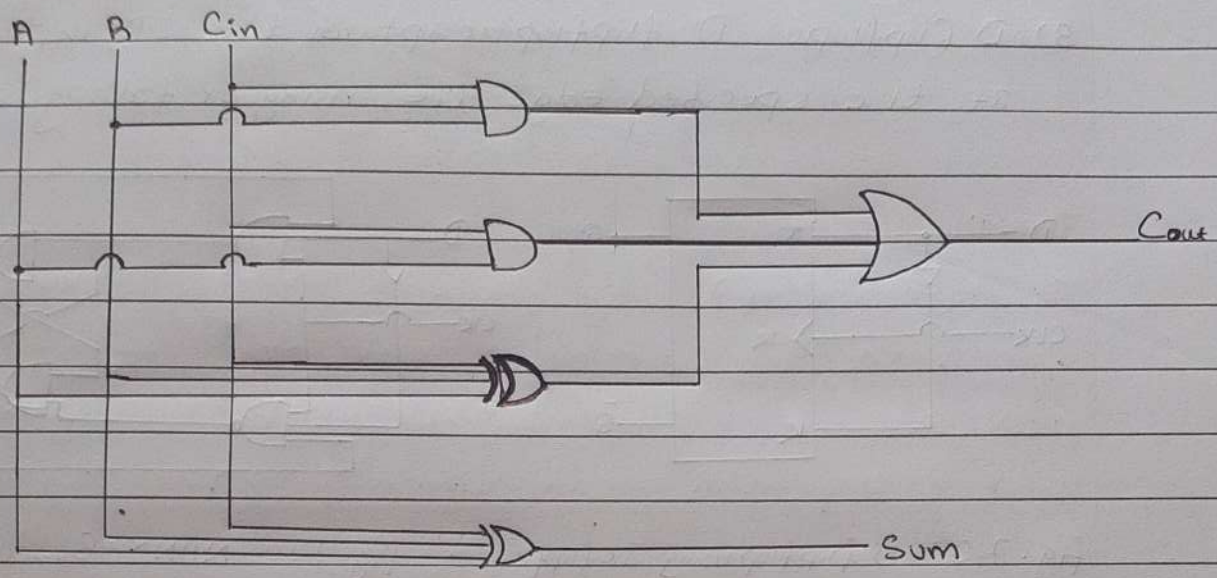


fig. 7.13: Gate diagram of Combinational Circuit (1 bit full adder)

A	B	Cin	Sum	Cout
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

fig. 7.14: Truth table of a 1 bit full adder.

RESULT:

The timing diagrams were obtained as shown in fig. 7.19, 7.20, 7.21 after inserting all inputs correctly in the circuits as shown in fig. 7.15, 7.16 and 7.17. In fig. 7.18, Serial 1 bit full adder was made using D flip flop. Hence, the sequential circuits were analysed and synthesised using the basic gates.