Experiment 8
Introduction to Latches and Flip-Flops and registers

Introduction:

The logic circuits that have been used until now were combinational logic circuits since the output of the device depends on the input data. Sequential logic circuits are defined as circuits whose outputs depend both on the present values of the inputs and the previous state of the circuits. Latches and flip-flops are basic sequential circuit whose operation we will investigate during this experiment. The difference between these two sequential devices is that flip-flop's output changes only at specific times determined by a clocking signal, while latch's output changes independent of a clocking signal.
Sequential circuits form the basis of registers, memories, and state machines, which in turn are vital functional units in digital design.

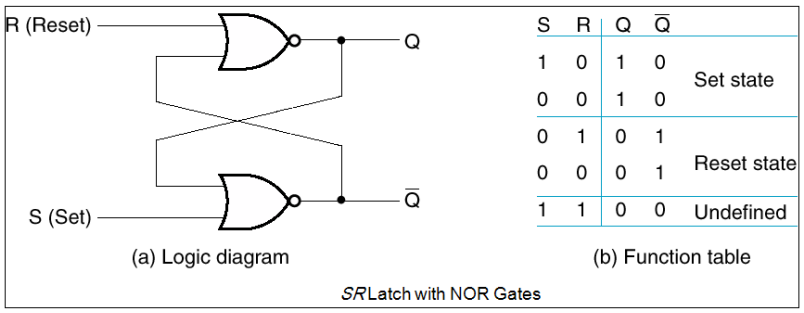
Objectives:

- Design, build, and test various sequential logic circuits.
- An in-depth study of the operation of S-R, J-K, master-slave, and edge-triggered latches and flip-flops.
- An introduction to commercially available flip-flops.

Procedure:

1- The S-R Latch

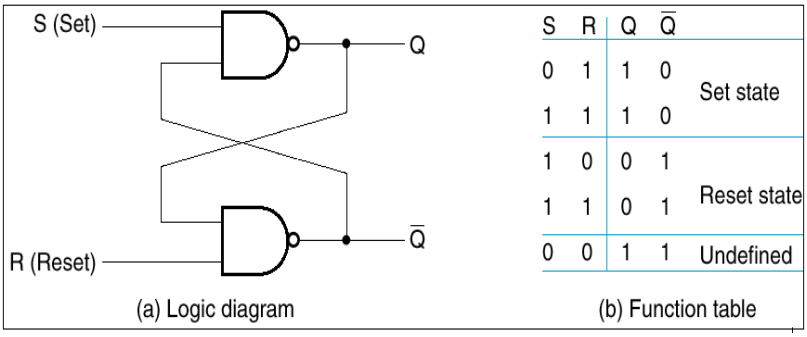
The most basic sequential unit is the S-R latch. From this basic circuit flip-flops are constructed, and from flip-flops, the registers, memories, and state machines can be made. The basic S-R latch has two inputs, S and R, and two outputs, Q and Q`.



S	R	Q	$\overline{Q}$	
1	0	1	0	Set state
0	0	1	0	
0	1	0	1	Reset state
0	0	0	1	
1	1	0	0	Undefined

Figure(1)

Similar SR latch can be made from NANDs as follow:



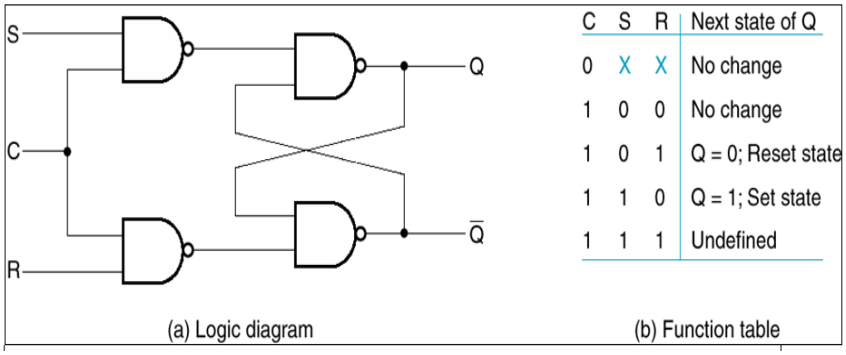
S	R	Q	$\overline{Q}$	
0	1	1	0	Set state
1	1	1	0	
1	0	0	1	Reset state
1	1	0	1	
0	0	1	1	Undefined

Figure(2)

2- The S-R Latch with Clock (S-R Flip-Flop)

To achieve synchronous operation, the latch should change state only on the proper clock signal. For example, assume that the latch should change state only when the clock signal goes high, else the latch holds its value independently from the value of S and R.

So we can adjust the circuit we have implemented above to have a third input (Clk).

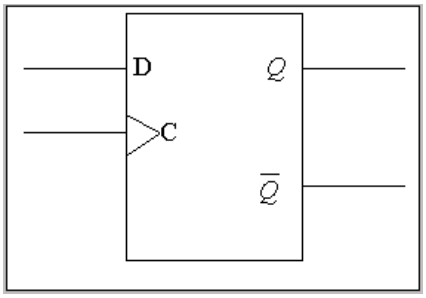


C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	Q = 0; Reset state
1	1	0	Q = 1; Set state
1	1	1	Undefined

(b) Function table

Figure(3)

3- D Flip-Flop

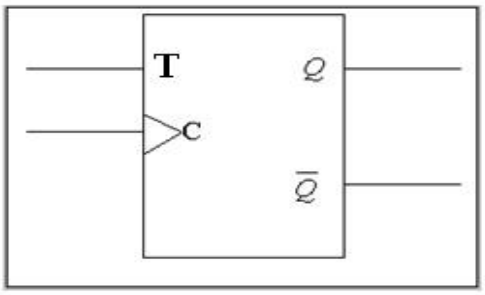


Figure(4)

D	C	Q	$\overline{Q}$	Operation
0	Rising edge	0	1	Reset (stores 0)
1	Rising edge	1	0	Set (stores (1)

- Notice that a D flip flop can be made from S-R flip flop by ensuring that the S and R outputs are the complement of each other at all times.

4- T Flip-Flop

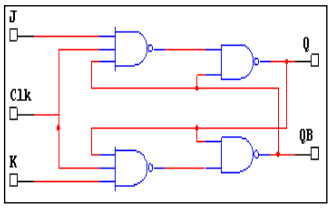


Figure(5)

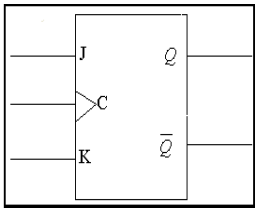
T	C	Q	Operation
0	Rising edge	0	Q **(No change)**
1	Rising edge	1	$\overline{Q}$ **(complement)**

5- J-K Flip-Flop

The J-K flip-flop is simply an S-R flip-flops that has been modified so that both inputs can be active at the same time. Where in the S-R flip-flop this condition was considered invalid, in the J-K flip-flop this condition toggles the output on successive clock cycles.



Figure(6)

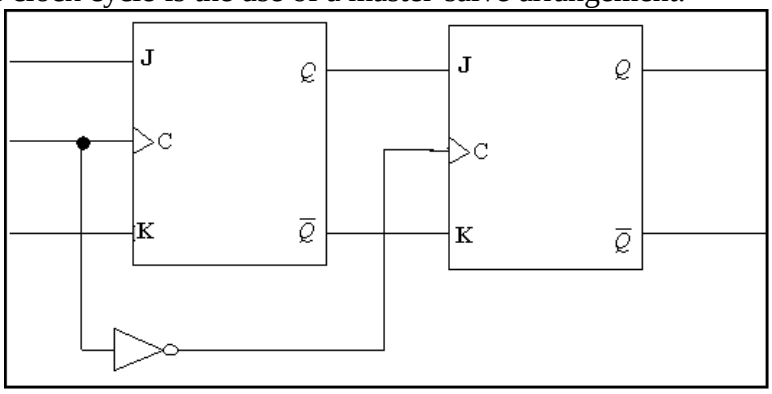


Figure(7)

J	K	C	Q	$\overline{Q}$	Operation
0	0	Rising edge	Q_0	$\overline{Q_0}$	Hold (no change)
0	1	Rising edge	0	1	Reset
1	0	Rising edge	1	0	Set
1	1	Rising edge	$\overline{Q_0}$	Q_0	Toggle

6- Master-Slave Flip-Flop

There is a slight problem with using a clock pulse. During the time the clock is high, the flip-flop performs identically to the regular asynchronous latch. Thus, if the inputs changed multiple times while the clock was high, the state of the latch could also change multiple times. One technique for eliminating multiple- state transition during a single clock cycle is the use of a master-salve arrangement.



Figure(8)

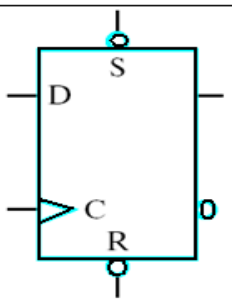
The left or master Latch in Figure above forms the inputs to the flip-flop, and the right or slave latch forms the outputs of the flip-flop. The master latch looks at the inputs while the clock is high. When the clock returns low, the slave latch is enabled, using the outputs of the master latch as its inputs. Thus the inputs are "read" while the clock is high and transferred to the outputs when the clock returns low.

7- Direct inputs:

- **Set/Reset independent of clock**
  - Direct set or preset
  - Direct reset or clear

S	R	C	D	Q	$\overline{Q}$
0	1	X	X	1	0
1	0	X	X	0	1
0	0	X	X	Undefined	
1	1	↑	0	0	1
1	1	↑	1	1	0

(b) Function table

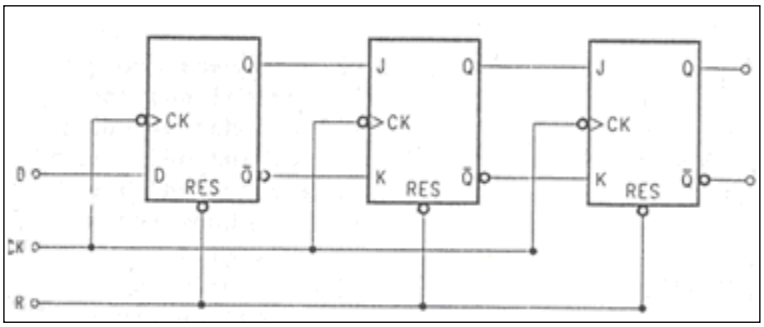


(c) Simplified Symbol

Figure(9)

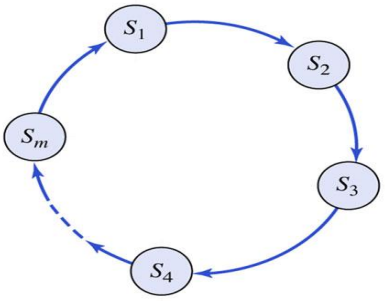
8- 3-Stage Shift Register

A group of cascaded FFs used to store related bits of information is known as a register. A register that is used to store information arriving from a source is called a shift register. Each FF output of a shift register is connected to the input of the next FF, and a common clock pulse is applied to all FFs. Hence, the shift register is a synchronous sequential circuit. The storage capacity of a register is the number of bits of digital data it can store. Each FF in a register represents one-bit storage capacity, therefore, the number of FFs in a register determine its total storage capacity.



Counter:

Counter: is essentially a register that goes through a predetermined sequence of states.
The gates in the counter are connected in such a way as to produce the prescribed sequence of binary states.
The counting sequence is often depicted by a graph called a **state diagram**.
A counter with m-states has the following state diagram:



Each node Si denotes the states of the counter and the arrows in the graph denote the order in which the states occur.
Counters are available in two categories: **ripple (Asynchronous) counters** and **synchronous counters**.

1) Ripple (Asynchronous) Counter:

In a ripple counter, the flip-flop output transition serves as a source for triggering other flip-flops; In other words, clock inputs of the flip-flops are triggered by output transitions of other
Flip-flops, rather than a common clock signal.
The output of each FF is connected to the clock input of the next flip-flop in sequence.

3-Stage Asynchronous Binary Counter

In the previous experiment, the edge−triggered JK FF was wired to operate as a toggle. Every time a clock pulse was detected at the input, the output changed state. After two clock pulses were detected, the output of the FF returned to its original state. As a result, there were two state changes of the output and the frequency of the input clock was divided by two. Therefore two events occurred, the number of clock pulses was counted and the frequency of the output was divided by 2. The circuit of Figure 3 contains the logic diagram for a three bit asynchronous binary counter with Q2 being the MSB. The frequency of the input clock is divided by two for the first FF and divided by two for the second FF and then divided by two again for the third FF. The frequency at Q2 has been divided by eight or 2n were n is the number of FFs in the circuit. There are also eight states in the truth table. This factor 2^n is also called the **Modulus or MOD** of the counter. Since this counter has 3 FFs, it is referred to as a MOD 8 counter. The MOD of any counter may be modified by connecting the proper combinational logic between the outputs of the appropriate FF and the Clear input. To convert the counter in Figure 3 to a MOD 7 counter, NAND the Q_0, Q1, Q2 inputs and connect the output of the NAND gate to the CLEAR input (active low input) of all the FFs. Figure 3 is an asynchronous device since the preceding FF must complete one cycle to provide the clock pulse for the next FF in the counter. The FFs do not change state at the same time and this creates a ripple effect in the way that the output of each FF changes state. This ripple effect is more noticeable in a MOD 16 or higher counter when the count resets from 15 or the maximum count back to 0. Another name for the asynchronous counter is the Ripple Counter.

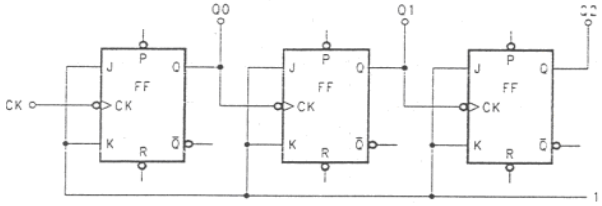


Figure (11)

- Advantages of Ripple Counters:
  - Simple hardware and design.
- Disadvantages of Ripple Counters:
  - They are asynchronous circuits, and can be unreliable and delay dependent, if more logic is added.
  - Large ripple counters are slow circuits due to the length of time required for the ripple to occur.

2) Synchronous Binary Counter

In the previous Asynchronous binary counter example, we saw that the output of one counter stage is connected directly to the input of the next counter stage and so on along the chain, and as a result the asynchronous counter suffers from what is known as "Propagation Delay". However, with **Synchronous Counters**, the external clock signal is connected to the clock input of EVERY individual flip-flop within the counter so that all of the flip-flops are clocked together simultaneously (in parallel) at the same time giving a fixed time relationship. This results in all the individual output bits changing state at exactly the same time with no ripple effect and therefore, no propagation delay.

3) BCD Counter (74LS160)

Since some digital functions are performed in BCD, the decade counter is often used.

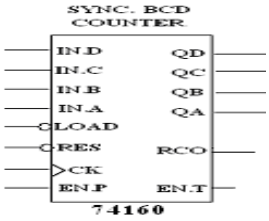


Figure 5

- CK: The counter clock.
- EN.P, EN.T: Are the two active high enable signals.
- QAQBQCQD: 4-bit counter output.
- RES: Active low reset input; when RES = 0 then the output QAQBQCQD = 0000.
- LOAD: active low load input; if:
  - LOAD = 0 then the counter start counting from the value on the inputs (IN.A, IN.B, IN.C, IN.D) to 9 each clock cycle.
  - LOAD = 1 then the counter start counting from the value on QAQBQCQD to 9 each clock cycle.
- IN.A, IN.B, IN.C, IN.D: The starting count value when LOAD = 0.

Ex1:

IN.A IN.B IN.C IN.D = 0101

CLOCK	Counter output	LOAD
1	0	1
2	1	1
3	2	1
4	3	1
5	4	1
6	5	1
7	6	1
8	7	1
9	8	1
10	9	1
11	0	1

EX2:

IN.A IN.B IN.C IN.D = 0101

CLOCK	Counter output	LOAD
1	5	0
2	5	0
3	6	1
4	7	1
5	8	1
6	9	1
7	0	1
8	5	0
9	5	0
10	5	0
11	6	1

EX3:

IN.A IN.B IN.C IN.D = 0101

CLOCK	Counter output	LOAD
1	5	0
2	6	1
3	7	1
4	8	1
5	9	1
6	5	0
7	6	1
8	7	1
9	8	1
10	9	1
11	5	0

- Note that if you want to count like 5,6,7,8,9,5,6,…as in EX3 above then the LOAD input should be 0 when counter output reaches 9 to take the loaded value (5) not (0).
- RCO: The output that become "1" when the value on QAQBQCQD = 1001 and "0" otherwise.
  This input is used to ensure that the load input become "0" when the counter output reaches "9" by connecting RCO to inverter (because LOAD IS ACTIVE LOW) and the output of the inverter to the LOAD input.