Faculty of Engineering and Technology

Department of Electrical and Computer Engineering

ENCS 211

Lab Report for Experiment #5

Sequential Logical circuits

Student Name: Yousef Anini

Student ID: 1140119

Instructor: Dr. Wasel Ghanem

Section: 6

Date: 16/11/2016

Table of contents:

I)Abstract :

The aim of the experiment is to understand the differences between combinational and sequential logic circuits, and to study the operating principles and applications of various flip flops, registers and counters and how to construct them.

II)Theory :

a) Sequential Circuits:

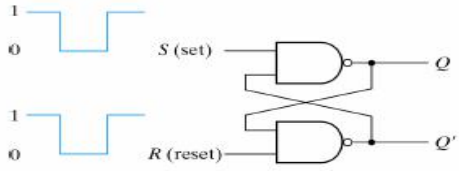
Circuits that contain memory elements to save the output and to use it in the next input. In other words, the output state of a "sequential logic circuit" is a function of the following three states, the "present input", the "past input" and/or the "past output".

The word "Sequential" means that things happen in a "sequence", one after another. The actual clock signal determines when things will happen next. It can be constructed from standard circuits such as: Flip flops, Latches and Counters and which themselves can be made by simply connecting together universal NAND gate and/or NOR gate in a particular combinational way to produce the required sequential circuit.

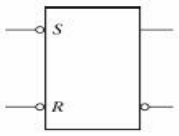
b) Latches:

Latches form one class of flip-flops. This class is characterized by the fact that the timing of the output changes is not controlled.

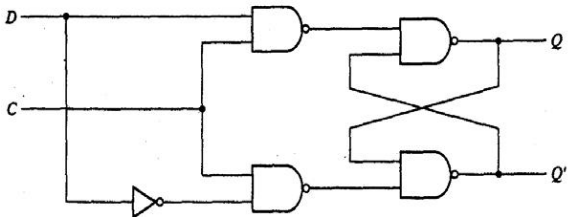
- **The SR (set-Reset) Latch:** It is a circuit with two cross-coupled NOR or NAND gates. The condition that is undefined is when both inputs are equal to 0 at the same time. There is a type that comes control input C .If C=0 the output does not change regarding less the inputs values. If C= 1the circuit will work normally.



S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	

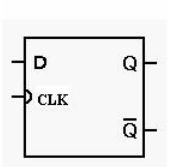


- **The D Latch :**To eliminate the undefined condition of the indeterminate state in the RS latch, the D latch was developed



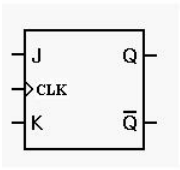
C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

- **Flip-Flops:** used for storage binary information as the latches, but the different is: The output change in the flip-flop occurs only at the clock edge while in the latch it occurs at the clock level. A flip-flop can be implemented using two separated latches. There are several types of flip-flops, the common ones are D, T, and JK flip flops.



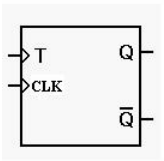
D Flip-Flop

D	*Q*(*t* + 1)	
0	0	Reset
1	1	Set



JK Flip-Flop

J	*K*	*Q*(*t* + 1)	
0	0	*Q*(*t*)	No change
0	1	0	Reset
1	0	1	Set
1	1	*Q*′(*t*)	Complement

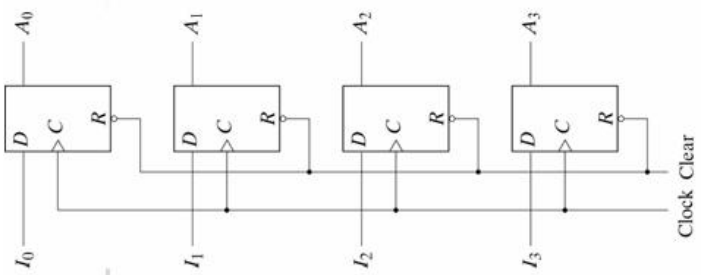


T Flip-Flop

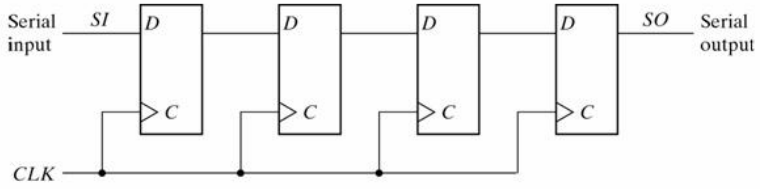
T	*Q*(*t* + 1)	
0	*Q*(*t*)	No change
1	*Q*′(*t*)	Complement

c) Registers:

Used to hold binary entities. It is a collection of flip flops; N- bit register consists of N flip flops. All the flip-flops are driven by a common clock, and all are reset simultaneously.

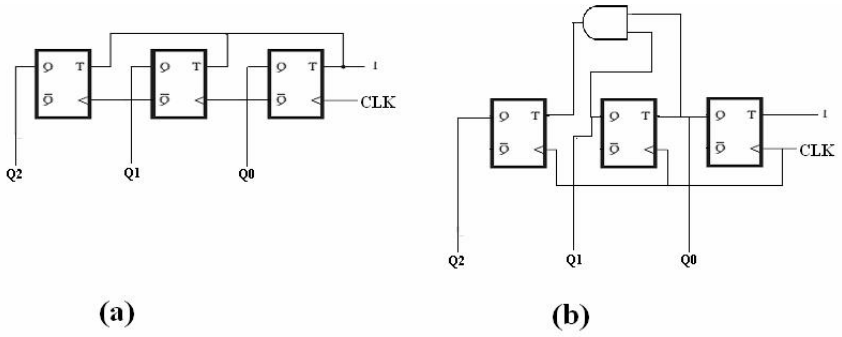


Shift register is a group of flip-flops connected in a chain so that the output from one flip flop becomes the input of the next flip-flop.



d) Counters:

A special-purpose register; that goes through a prescribed Sequence of states. The counters are classified into two categories: Ripple and Synchronous counters. In ripple counters, there is no common clock; the flip-flop output transition serves as a source for triggering other flip-flops. In synchronous counters, all flip flops receive a common clock.



III) Procedure and Discussion:

1.Latches and Flip flops :

a) Constructing RS latch with Basic Logic Gates

We used KL-26006 module to implement the circuit in Fig.1.Then, we connected Switches to inputs(S, R), and connected outputs (Q, Q') to logic indicator.

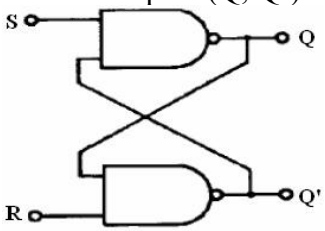


Fig.1: RS latch

The Following results were taken:

S	R	Q	Q'	State
0	**0**	**1**	**1**	**Indeterminate**
0	**1**	**1**	**0**	**Set**
1	**0**	**0**	**1**	**Reset**
1	**1**	**0**	**1**	**No change**

It is an active low SR latch. There is no clock for synchronization. Each state is declared in the table.

---

b) Constructing RS latch with control input

We used KL-26006 module to implement the circuit in Fig.2.Then, we connected CK2 (control input) to logic 1 (+5V). The Following results were taken:

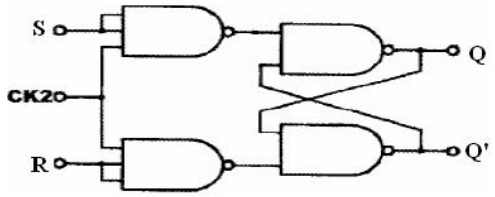


Fig.2: RS Latch with control input

S	R	Q	Q'
0	0	1	0
0	1	0	1
1	0	1	0
1	1	1	1

It is an active high SR latch with control. Each state is declared in the table.

---

c) Constructing D latch with RS latch

We used KL-26006 module to implement the circuit in Fig.3.Then, we connected CK2 to Pulser switch. The Following results were taken:

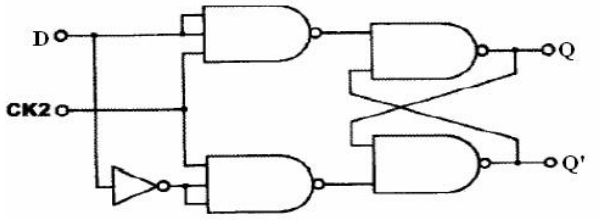


Fig.3: D Latch

CK2	D	Q	Q'	State
0	0	0	1	No Change
0	1	0	1	
⎍	0	0	1	Q= D Q'= D'
⎍	1	1	0	

Q =D when a pulse is provided form CK2, otherwise Q maintain its previous value.

d) Constructing JK latch with RS latch

We used KL-26006 module to implement the circuit in Fig.4.Then, we connected CK2 to Pulser switch. The Following results were taken:

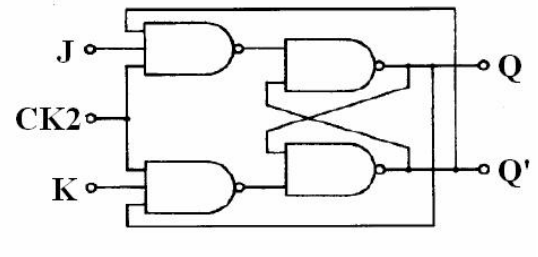


Fig.4: JK Latch with RS Latch

J	K	Q	Q'
0	0	1	0
0	1	0	1
1	0	1	0
1	1	0	1

This latch is constructed using RS latch and 2 NAND gates connected to a clock. The circuit works on the positive edge of the clock.

---

e) Constructing JK Flip-flop with master- slave RS latches

We used KL-26006 module to implement the circuit in Fig.5.Then, we connected CK1 to Pulser switch. The Following results were taken:

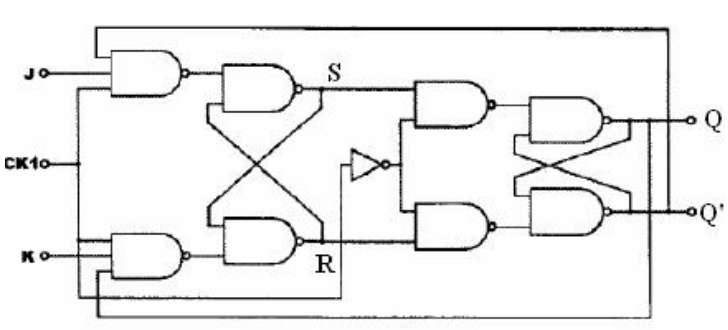


Fig.5: JK Flip-Flop

Clk	K	J	S	R	Q	Q'	->	S	R	Q	Q'
⎍	0	0	1	0	1	0	->	1	0	1	0
⎍	0	1	1	0	1	0	->	1	0	1	0
⎍	1	0	1	0	1	0	->	0	1	0	1
⎍	1	1	0	1	0	1	->	1	0	0	1
⎍	1	1	1	0	0	1	->	1	0	1	0

This JK Flip-Flop is constructed using 2 RS latches connected as master-slave The input signals (J, K) are connected to the gated “master” SR flip flop which “locks” the input condition while the clock (Clk) input is “HIGH” at logic level “1”. As the clock input of the “slave” flip flop is the inverse (complement) of the “master” clock input, the “slave” SR flip flop does not toggle. The outputs from the “master” flip flop are only “seen” by the gated “slave” flip flop when the clock input goes “LOW” to logic level “0”.

2. Registers:

a) Constructing Shift Register with D Flip-Flops

We used KL-26006 module (block a) to construct the circuit shown in Figure.6.

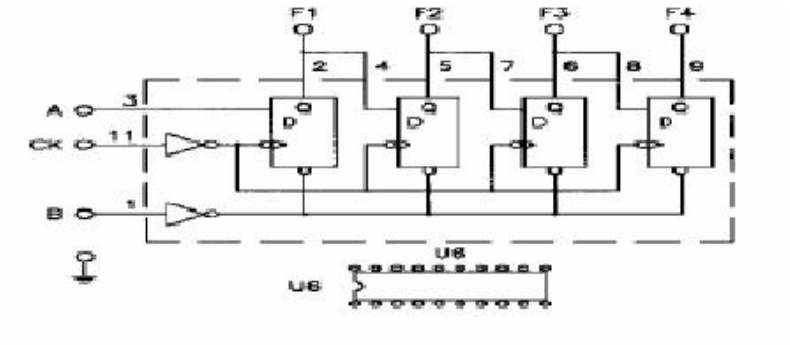


Fig.6: Shift Right Register

1)Connect A (serial input) to sw1. Connect B (clear) to sw2. Connect the outputs F1, F2, F3 and F4 to L1L2L3L4. Connect CK to Pulser Switch. Set SW0 to 0 to clear the register, and then keep it on , then the results are :

clock1: 1000
clock2: 1100
clock3: 1110
clock4: 1111

2)To load the value 1011 on the register, the following steps were applied:

1. SW1 = 1, apply clock pulse & output is: 1000
2. SW1 = 1, apply clock pulse & output is: 1100
3. SW1 = 0, apply clock pulse & output is: 0110
4. SW1 = 1, apply clock pulse & output is: 1011

---

b) 4-Bit Shift Register with serial and parallel load

We used KL-26006 module (Block b) .It is IC 7495 which is 4-Bit Shift Register with serial and parallel synchronous operating modes. It has a Serial input and four Parallel (A–D) Data inputs and four Parallel Data outputs (QA–QD). Figure.7 (a) shows the top view of IC7495 package and Figure.7 (b) shows block b in KL-26006 module.

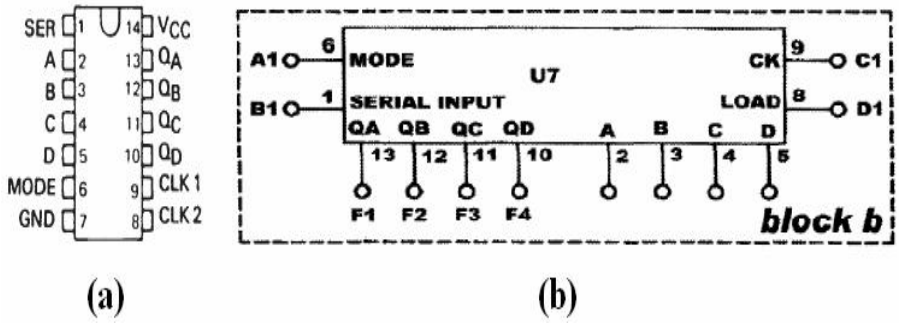


Fig.7: IC7495 shift register with serial and parallel load.

1. Shift- right serial- input:

To enable the shift-right operation which is controlled by CK C1 (pin 9), the MODE control input A1 (pin 6) must be 0. In this mode CK depends only on the SERIAL INPUT B1 (pin 1) to shift the inputs A-D to the right.

B1 was connected to SW0, A1 to GND, inputs A-D to SW4-SW1, and outputs QA-QD to L3-L0 and CK to pulser switch. The value 1101 was loaded to the register (ABCD = 1101) and then four clock pulses were applied and resulted after clock1: QAQBQCQD = 0110 , & after clock4: QAQBQCQD = 0000

2. Parallel- load Register:

To enable the parallel load operation which is controlled by LOAD D1 (pin 8), the MODE control input A1 (pin 6) must be 1. In this mode LOAD depends only on the parallel inputs A-D to load them into the parallel outputs QA-QD. A1 was connected to +5VCC, inputs A-D to SW4-SW1, outputs QA-QD to L3-L0 and LOAD to pulser switch.

We set this value on the inputs ABCD = 1101, and then we applied a clock pulse & the output was QAQBQCQD = 1101.

Task 1

- We make a shift left register using (IC7495)by taking the output in reverse order of the shift left mode; since the mode is shift right with output instead of $F_1F_2F_3F_4$ (to logic indicators $L_3L_2L_1L_0$) take them $F_4F_3F_2F_1$ to logic indicators $L_3L_2L_1L_0$.
- We make shift left operation using parallel mode by connecting $F_4F_3F_2F_1$ to logic indicators $L_3L_2L_1L_0$, and connecting the inputs ABCD to $F_2F_3F_4$ & SI(serial input) and load any value using serial input pin

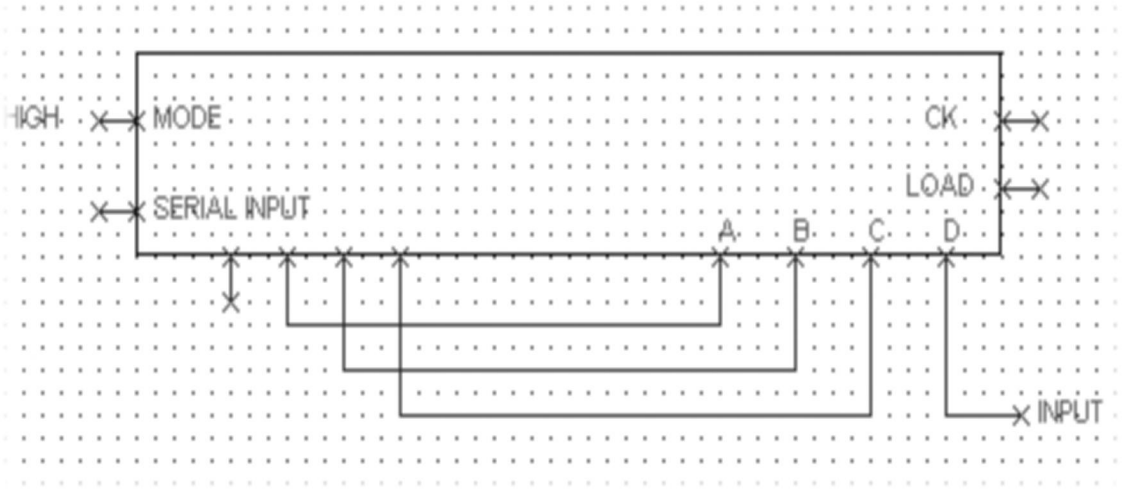


Fig. (Serial Shift-Left)

3. Counters:

a) 2-bit Synchronous Counter.

We used KL-26007 module (block c), which contains three separated JK flip flops to implement the 2-bit synchronous counter shown in fig.8. We connected CLK input to Pulser switch, connected counter outputs (Q1, Q0) to indication lamps, applied clock pulses to CLK input and observed and recorded the outputs in the Table(a). We applied counter outputs (Q1, Q0) to **D1 display** inputs. We observed and recorded the outputs in Table (b).

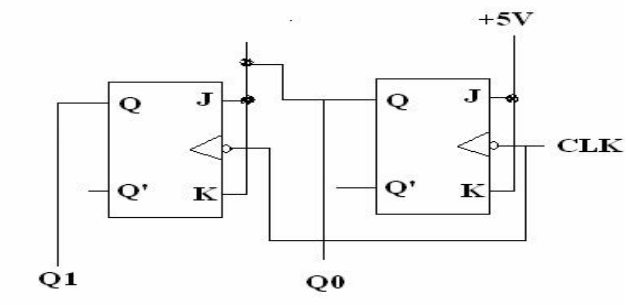


Figure.8: 2-bit Synchronous Counter

Clk	Q1	Q0	D1
⎍	0	0	0
⎍	0	1	1
⎍	1	0	2
⎍	1	1	3
⎍	0	0	0
⎍	0	1	1
⎍	1	0	2
⎍	1	1	3
⎍	0	0	0
⎍	0	1	1

b) 3-bit (divide-by-eight) Ripple Counter.

We used KL-26007 module (block c), which contains three separated JK flip flops to implement the 3-bit (divide by eight) Ripple counter shown in Figure.9. We connected CLK input to Pulser switch. Connected counter outputs (Q2,Q1, Q0) to indication lamps. We applied clock pulses to CLK input. And observed and recorded the outputs in Table. (A). We applied counter outputs (Q2, Q1, Q0) to **D1 display** inputs. And observed and recorded the outputs in Table. (B).

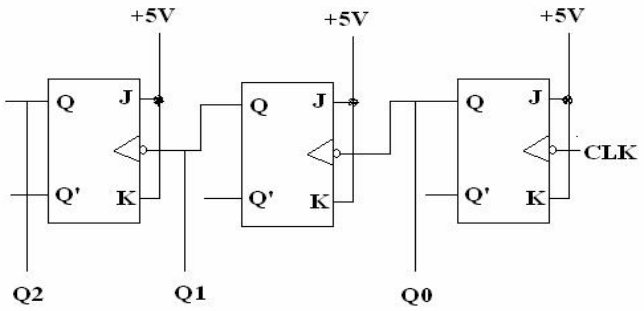


Fig.9: 3-bit Ripple Counter

CLK	Q2	Q1	Q0	D1
⎍	0	0	0	0
⎍	0	0	1	1
⎍	0	1	0	2
⎍	0	1	1	3
⎍	1	0	0	4
⎍	1	0	1	5
⎍	1	1	0	6
⎍	1	1	1	7
⎍	0	0	0	0
⎍	0	0	1	1

This counter counts from 0 to 7 then reset to zero and start counting again, (Asynchronous); since not the same clock is connected to all flip flops. The next value is set to Q0 of the counter only when a clock is applied and other digits vary depending on the varying of previous digit.

Task 2

In order to make a three bit synchronous counter, we make the design below:

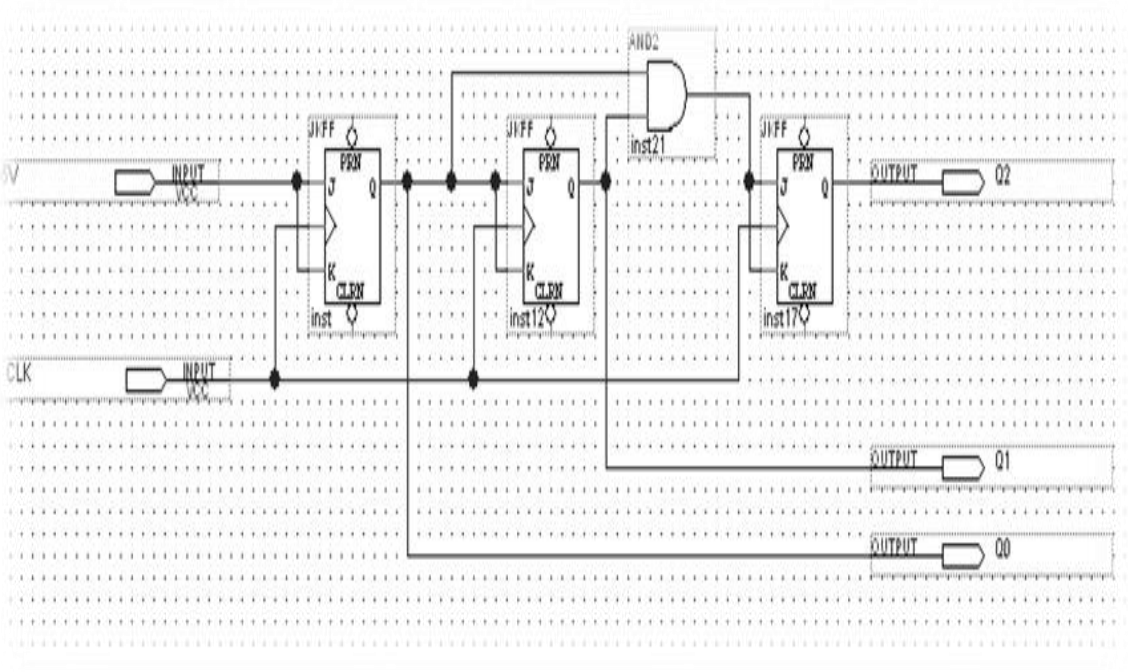


Fig. (3-bit Synchronous counter)

---

c) BCD Counter.

First, we located BCD counter (IC 7490) on KL-26007 module (block b). Then, we connected the circuit shown in Figure.10. Then, we connected RO (1) and RO (2) (pins 2& 3) to 0v, note that RO (1) and RO (2) are two inputs to internal NAND gate which used to reset the counter. We connected clock INPUT 1 (pin 14) to Pulser switch. Connected the outputs (A, B, C, and D) to indication lamps. We applied clock pulses to INPUT 1, and observed the count sequence (0000-1001). We applied counter outputs (A, B, C, and D)to **D1 display** inputs. Applied clock pulses to INPUT 1. Observed the count sequence on the display. Connected the INPUT 1 to the output of the clock generator. Observe the count sequence on the display.

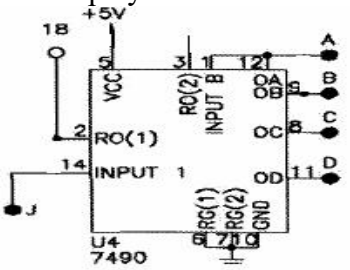


Fig.10(IC 7490) BCD Counter.

The counter works perfectly and counts from 0000 to 1001(1-9) when a clock is provided.

d) Divide-by-8 counter using BCD chip counter

Connect RO (2) (pin3) to +5V, and connect RO (1) (pin2) to QD (pin11) output. This will make counter reset after 111 (or 7) ➔before the value of D becomes 1 at the first time (DCBA=1000//8) the pin reset of the counter is 0 (active low), so the counter is reset DCBA=0000, then starts the count again. Then, we connect clock INPUT 1 (pin 14) to Pulser switch. Connect the outputs A, B, C, and D to indication lamps. Apply clock pulses to INPUT 1, and observe the count sequence (0000-0111). Apply counter outputs A, B, C, and D to **D1 display** inputs. Apply clock pulses to INPUT 1.

Task 3

To make the BCD counter counts from 0000 to 0101, we reset the counter just before it becomes 6. We did that by connecting C & B pins to RO (1) & RO (2).

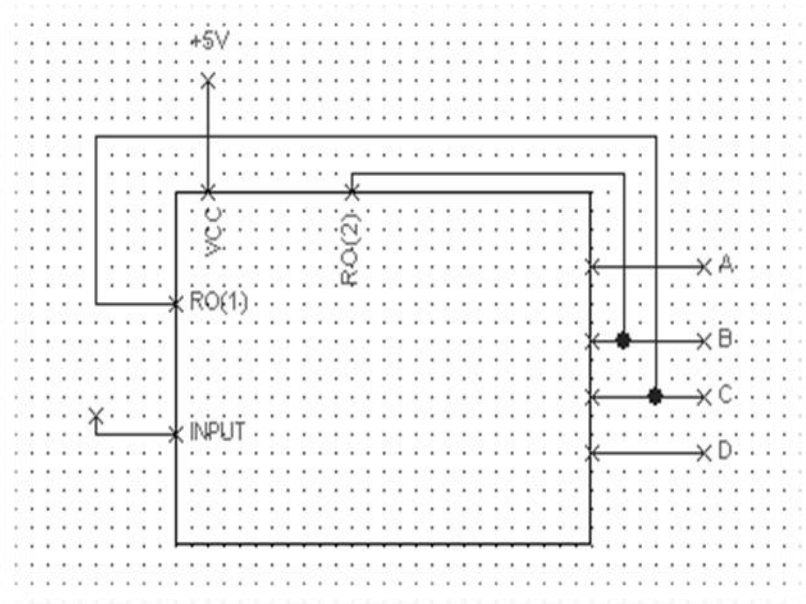


Fig.: (0-5 Counter)

To make the BCD counter counts from 0000 to 0100, we reset the counter just before it becomes 5. We did that by connecting C & A pins to RO (1) & RO (2).

IV. Conclusion and Discussion Questions

In this experiment we learned about a new kind of circuits (The Sequential Circuits). How it works, and many applications on such as counters and registers. Also we learned about the latches and flip flops and how to build a flip flop using latches. And we learned about how registers work and how we build it, finally we studied the differences between ripple and synchronous counters.

The latches are useful for storing binary information; since they are faster than flip flops (flip flops are master-slave latches), and there size also less than flip flop's size and they consume less power. But they make many problems; they make the system asynchronous so increase the probability of glitches in the circuits that use them; since they are level triggering chips that's why they are rarely used in the sequential circuits design.

The main disadvantages of SR latches is when S=R=1. In this state the latch make a set and reset to itself at the same time, so Q=Q'=1, and this is indeterminate state. So it is the responsibility of the designers to make sure that S & R are not equal to 1 at the same time.

The difference between "synchronous" and "ripple":

Ripple counter: the clock is not connected to all flip-flops. In other words, there is no synchronization; since the output of flip-flop transition serves when the output of triggering flip-flop changes.

Synchronous counter: there are common clock that connected to all flip-flops. All flip-flop's output are adjusted only at the clock.

We faced some troubles with the Pulser switch.Also we faced some troubles with some IC's which turned out to be faulty after testing it and we had to replace it.

V.References:

- ✓ M.Morris Mano, Micheal D. Clietti ,Digital Design with an introduction to the verilog hdl ,5th edition ,Pearson,2013.
- ✓ ENCS 211 Lab Manual.
- ✓ http://www.electronics-tutorials.ws/sequential/seq_2.html
- ✓ Some of the designs were made using Quartus program V.13.1.