Digital Logic Design (EL-1005)

LABORATORY MANUAL

Spring-2025

LAB 10

Flip-Flops and Counters

______________________________	__________	___
STUDENT NAME	ROLL NO	SEC

____

INSTRUCTOR SIGNATURE& DATE

MARKS AWARDED: /03

____

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES (NUCES), KARACHI

OBJECTIVES:

- To study the operation of synchronous & Asynchronous Counter
- To study use of JK Flip Flop in design of counter
- Developed counter timing diagram
- To learn how to implement 2*2 bit multiplier circuit

APPARATUS: Logic trainer, Logic probe

COMPONENTS:ICs 74LS08, 74LS32, 74LS04, 74LS86, 74LS02, 74ls76

THEORY:

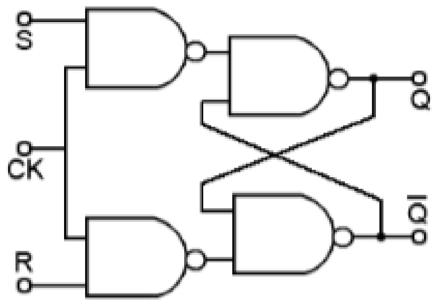
Flip-flop is a circuit that maintains a state until directed by input to change the state. A basic flip-flop can be constructed using four-NAND or four-NOR gates. Flip flop is popularly known as the basic digital memory circuit. It has its two states as logic 1(High) and logic 0(low) states. A flip flop is a sequential circuit which consist of single binary state of information or data. The digital circuit is a flip flop which has two outputs and are of opposite states. It is also known as a Bistable Multivibrator.

Types of flip-flops:

1. SR Flip Flop
2. JK Flip Flop
3. D Flip Flop
4. T Flip Flop

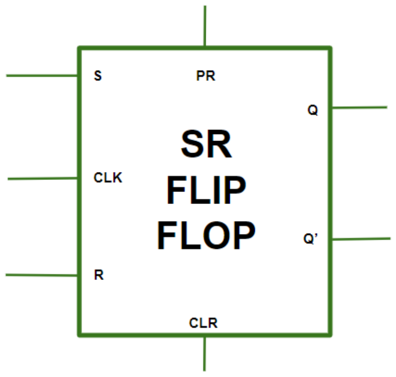
S-R Flip Flop :

In the flip flop, with the help of preset and clear when the power is switched ON, the states of the circuit keeps on changing, that is it is uncertain. It may come to set(Q=1) or reset(Q'=0) state. In many applications, it is desired to initially set or reset the flip flop that is the initial state of the flip flop that needs to be assigned. This thing is accomplished by the preset(PR) and the clear(CLR).

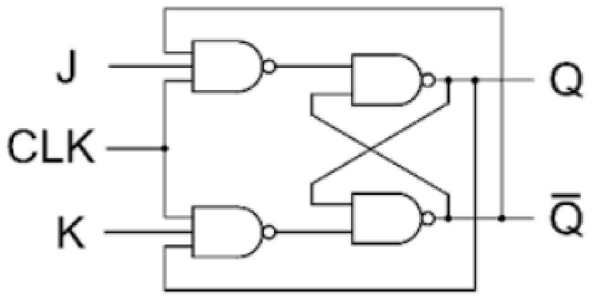


TRUTH TABLE

S	R	Q_N	Q_{N+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-

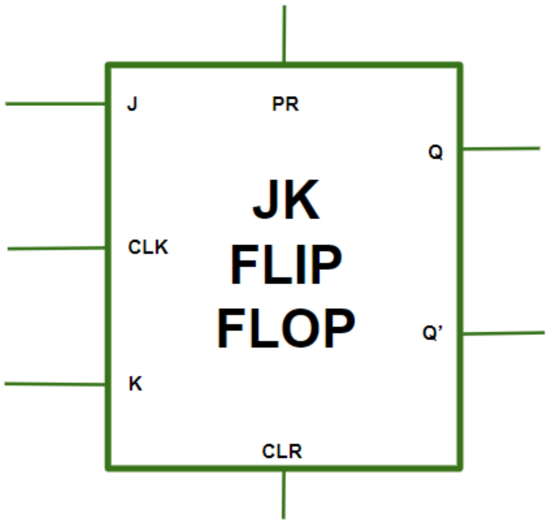


J-K Flip Flop:
In JK flip flops, the diagram over here represents the basic structure of the flip flop which consists of Clock (CLK), Clear (CLR), Preset (PR).

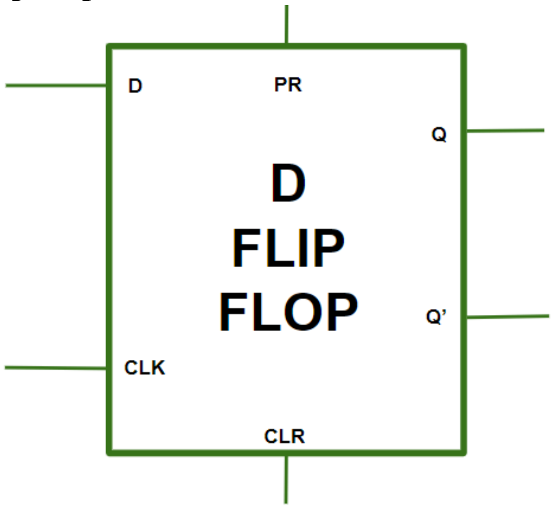


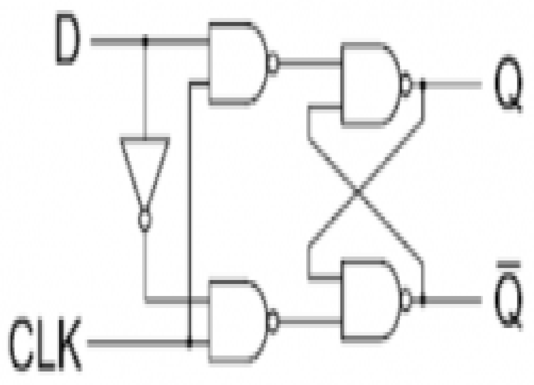
TRUTH TABLE

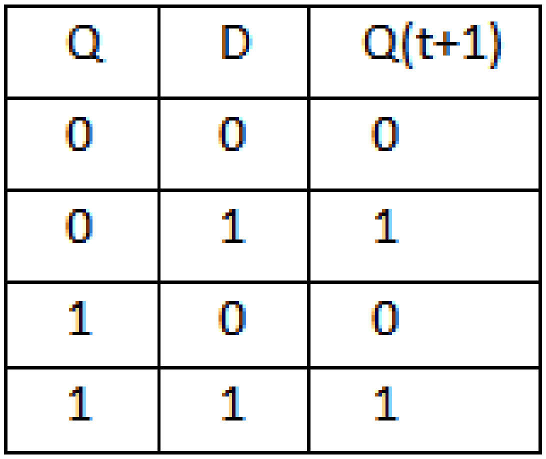
J	K	Q_N	Q_{N+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



D Flip Flop:

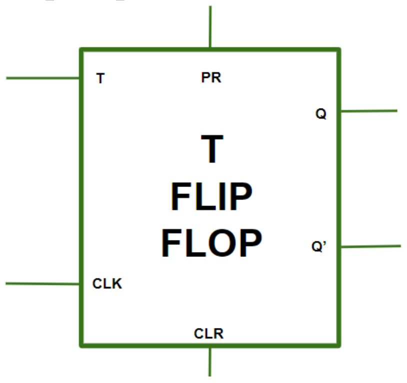


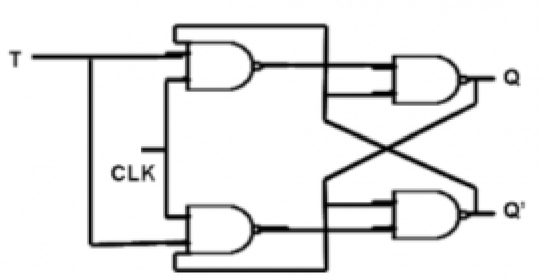


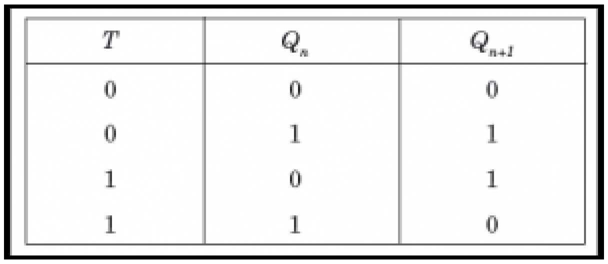


Q	D	Q(t+1)
0	0	0
0	1	1
1	0	0
1	1	1

T Flip Flop:







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

Difference between D and T Flip-Flops

- **D Flip-Flop**: The D flip-flop (Data flip-flop) stores and outputs the value of its data input (D). It captures the input value on the rising (or falling) edge of the clock signal and holds it until the next clock edge.
- **T Flip-Flop**: The T flip-flop (Toggle flip-flop) changes its output state (toggles) whenever its trigger input (T) transitions from one state to another. If T is high (1), the output toggles; if T is low (0), the output remains unchanged.

Difference between S-R and JK Flip Flops:
SR Flip-Flop: The SR flip-flop (Set-Reset flip-flop), also known as the RS flip-flop, has two inputs: S (set) and R (reset). It has two stable states, and its output changes based on the inputs. When S is high and R is low, the flip-flop sets (Q=1). Conversely, when R is high and S is low, the flip-flop resets (Q=0).

JK Flip-Flop: The JK flip-flop is a versatile sequential logic device. It has two inputs: J (set) and K (reset). The output of a JK flip-flop toggles when both J and K inputs are high (1) during a clock pulse. However, it can also function as a D flip-flop when J and K are tied together. Why use such a circuit well counters, frequency dividers, and shift all require a toggle output which this type of circuit can provide.

Counters:
A register that goes through a prescribed sequence of states upon the application of input pulses is called a counter. The input pulses may be clock pulses or may originate from some other source, and they may occur at fixed intervals of time or random intervals. The sequence of states may follow the binary number sequence or any other sequence of states: A counter that follows the binary number sequence is called a binary counter.

Counters are available in two categories: ripple counters and synchronous counters. In a ripple counter, the flip-flop output transition serves as a source for triggering other flip flops. In other words, the clock inputs of some or all of the flip flops are triggered not by the common clock pulses, but rather by the transition that occurs in other flip flop outputs. In a synchronous counter, the clock inputs of all of the flip flops receive the common clock pulse, and the change of state is determined from the present state of the counter.

J-K Flip FLOP As Counter

Starting with four J-K flip-flops connected in such a way to always be in the “toggle” mode, we need to determine how to connect the clock inputs in such a way so that each succeeding bit toggles when the bit before it transitions from 1 to 0. The Q outputs of each flip-flop will serve as the respective binary bits of the final, four-bit count:

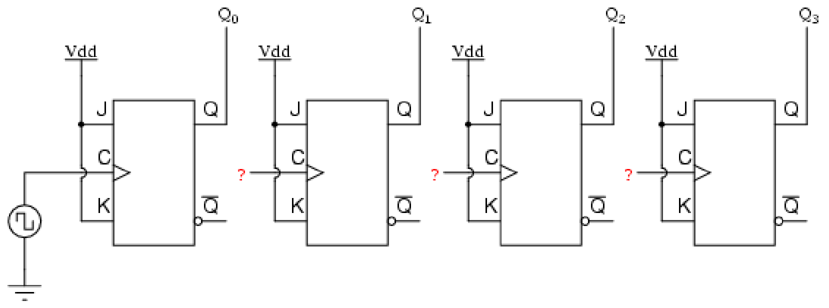


Fig:1 J-K Flip Flop as counter

If we used flip-flops with negative-edge triggering (bubble symbols on the clock inputs), we could simply connect the clock input of each flip-flop to the Q output of the flip-flop before it, so that when the bit before it changes from a 1 to a 0, the "falling edge" of that signal would "clock" the next flip-flop to toggle the next bit:

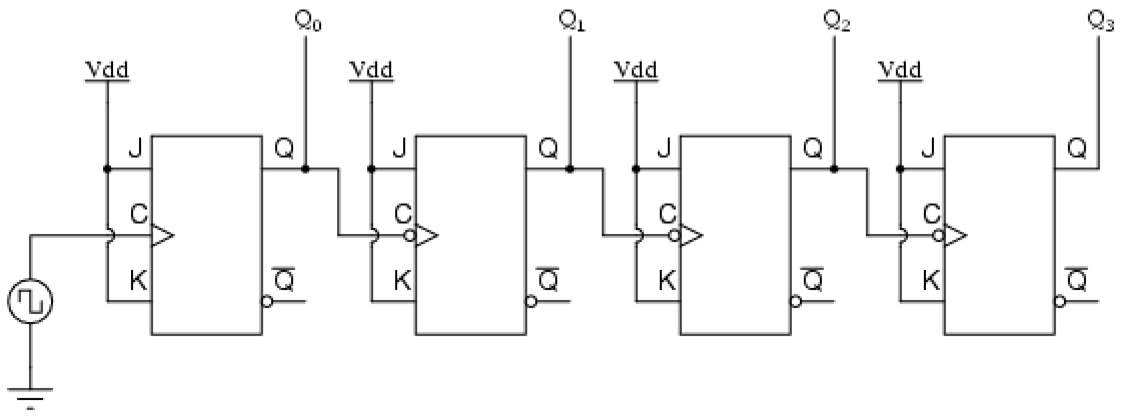


Fig:2 J-K Flip Flop as Asynchronous counter

This circuit would yield the following output waveforms, when "clocked" by a repetitive source of pulses from an oscillator:

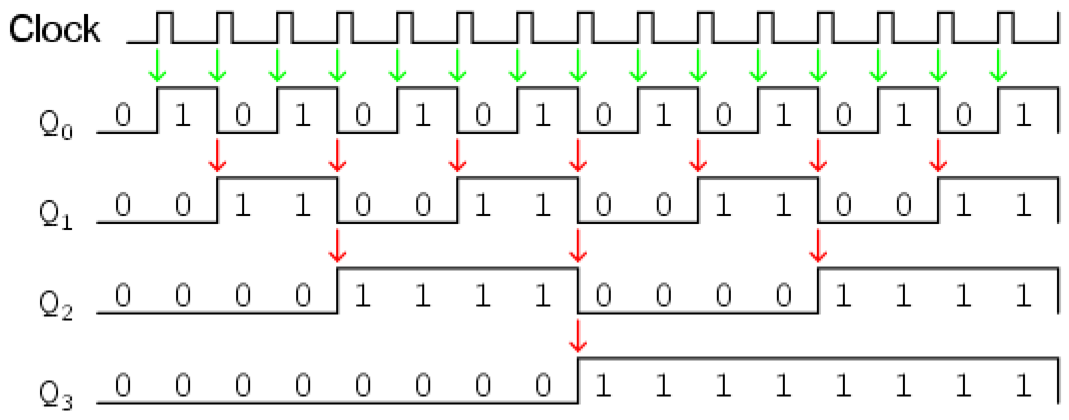


Fig3: Timing Diagram

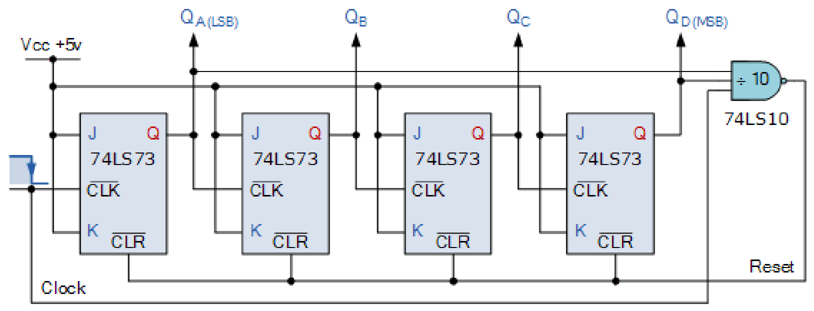
Asynchronous Counters

Asynchronous counters are those whose output is free from the clock signal. Because the flip flops in asynchronous counters are supplied with different clock signals, there may be delay in producing output.

The required number of logic gates to design asynchronous counters is very less. So they are simple in design. Another name for Asynchronous counters is “Ripple counters”.

The number of flip flops used in a ripple counter is depends up on the number of states of counter (ex: Mod 4, Mod 2 etc). The number of output states of counter is called “Modulus” or “MOD” of the counter. The maximum number of states that a counter can have is 2n where n represents the number of flip flops used in counter.

Asynchronous Decade Counter



This type of asynchronous counter counts upwards on each trailing edge of the input clock signal starting from 0000 until it reaches an output 1001 (decimal 9). Both outputs QA and QD are now equal to logic “1”. On the application of the next clock pulse, the output from the 74LS10 NAND gate changes state from logic “1” to a logic “0” level.

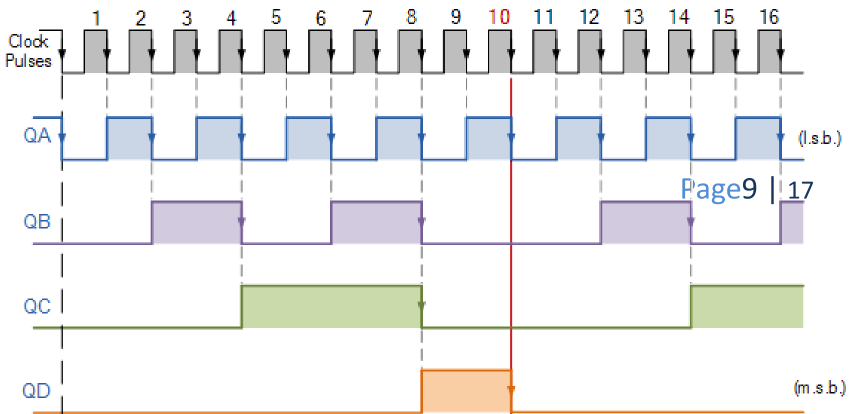
As the output of the NAND gate is connected to the CLEAR (CLR) inputs of all the 74LS73 J-K Flip-flops, this signal causes all of the Q outputs to be reset back to binary 0000 on the count of 10. As outputs QA and QD are now both equal to logic “0” as the flip-flop’s have just been reset, the output of the NAND gate returns back to a logic level “1” and the counter restarts again from 0000. We now have a decade or Modulo-10 up-counter.

Decade Counter Truth Table

Clock	**Output bit Pattern**				**Decimal**
Count	QD	QC	QB	QA	**Value**
1	0	0	0	0	0
2	0	0	0	1	1
3	0	0	1	0	2
4	0	0	1	1	3
5	0	1	0	0	4
6	0	1	0	1	5
7	0	1	1	0	6
8	0	1	1	1	7
9	1	0	0	0	8
10	1	0	0	1	9
11	Counter Resets its Outputs back to Zero				

Decade Counter Timing Diagram

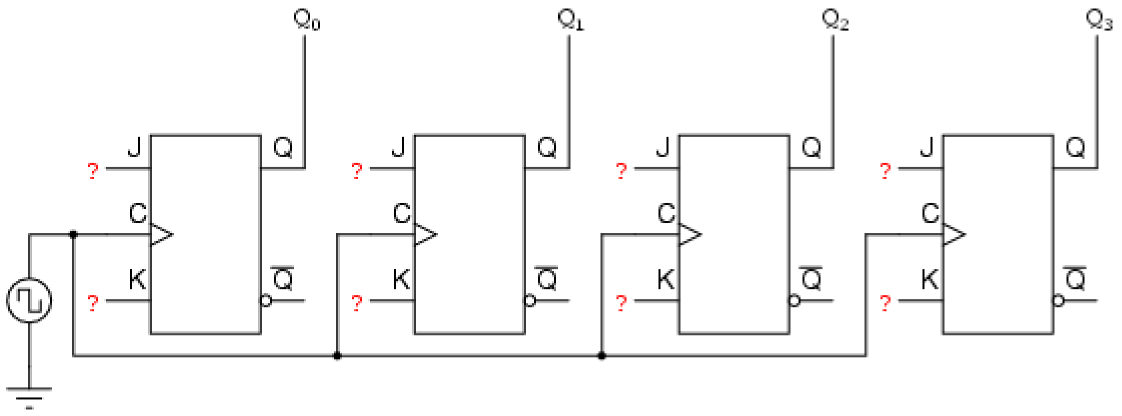
By using the same idea of truncating counter output sequences, the above



circuit could easily be adapted to other counting cycles be simply changing the connections to the inputs of the NAND gate or by using other logic gate combinations.

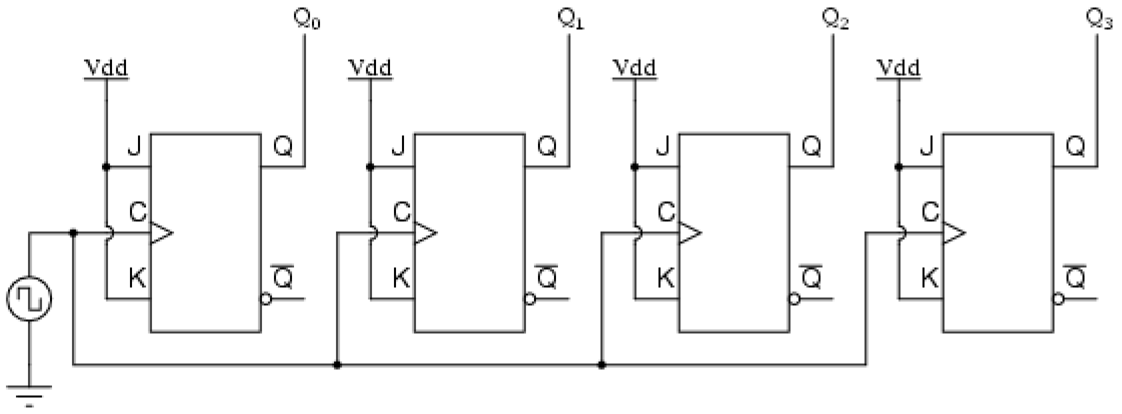
Synchronous counter

A *synchronous counter*, in contrast to an *asynchronous counter*, is one whose output bits change state simultaneously, with no ripple. The only way we can build such a counter circuit from J-K flip-flops is to connect all the clock inputs together, so that each and every flip-flop receives the exact same clock pulse at the exact same time:



Now, the question is, what do we do with the J and K inputs? We know that we still have to maintain the same divide-by-two frequency pattern in order to count in a binary sequence, and that this pattern is best achieved utilizing the "toggle" mode of the flip-flop, so the fact that the J and K inputs must both be (at times) "high" is clear.
However, if we simply connect all the J and K inputs to the positive rail of the power supply as we did in the asynchronous circuit, this would clearly not work because all the flip-flops would toggle at the same time: with each and every clock pulse!

This circuit will not function as a counter!

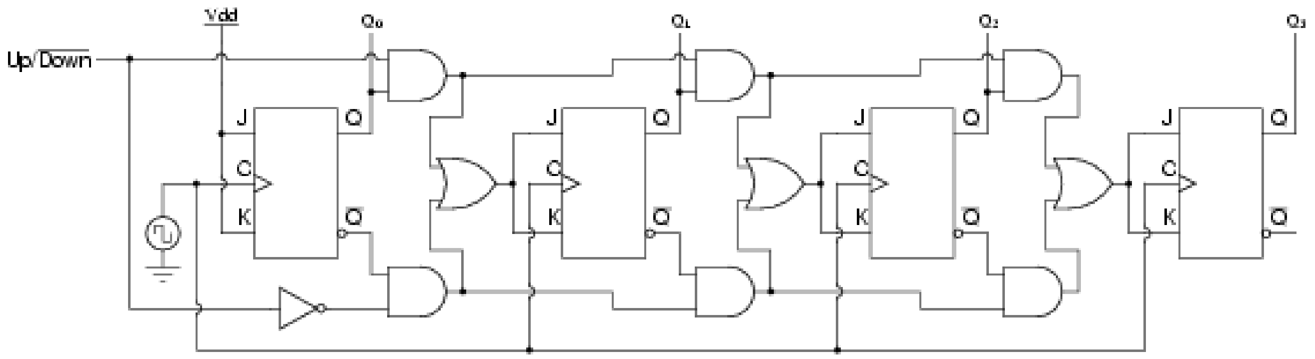


Let’s examine the four-bit binary counting sequence again, and see if there are any other patterns that predict the toggling of a bit. Asynchronous counter circuit design is based on the fact that each bit toggle happens at the same time that the preceding bit toggles from a “high” to a “low” (from 1 to 0).

Counter Circuit with Selectable “up” and “down” Count Modes

Taking this idea one step further, we can build a counter circuit with selectable between “up” and “down” count modes by having dual lines of AND gates detecting the appropriate bit conditions for an “up” and a “down” counting sequence, respectively, then use OR gates to combine the AND gate outputs to the J and K inputs of each succeeding flip-flop:

A four-bit synchronous "up/down" counter

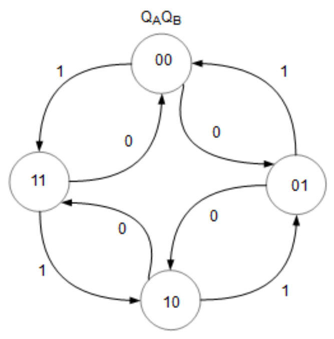


This circuit isn’t as complex as it might first appear. The Up/Down control input line simply enables either the upper string or lower string of AND gates to pass the Q/Q’ outputs to the succeeding stages of flip-flops.

<u>Counter Design</u>

The steps to design a Synchronous Counter using JK flip flops are:

1. Describe a general sequential circuit in terms of its basic parts and its input and outputs.
   Design a 2 bit up/down counter with an input D which determines the up/down function. Thus when D=0, the count sequence is 00,01,10,11,00 ... when D=1, the count sequence is 00,11,10,01,00 ...
2. Draw the state diagram for the given sequence.



3. Develop a next-state table for the specific counter sequence. Using the state diagram as a reference, fill up the present state and next state (yellow) columns. For this interactive table, you can modify the next state. After you have changed the values, press the **Calculate** button. Press the **Reset** button to reset the page to the initial data.

Present State			**Next State**		**JK flip flop inputs**									
D	Q_A	Q_B	Q_A	Q_B	J_A	K_A	J_B	K_B						
0	0	0	0	1	0	X	1	X						
0	0	1	1	0	1	X	X	1						
0	1	0	1	1	X	0	1	X						
0	1	1	0	0	X	1	X	1						
1	0	0	1	1	1	X	1	X						
1	0	1	0	0	0	X	X	1						
1	1	0	0	1	X	1	1	X						
1	1	1	1	0	X	0	X	1						

Next, the FF transition table (blue) is completed using data from the respective present state, next state and the JK flip flop transition table below. Click on any blue cell to understand how its value is obtained.
JK Flip Flop Truth Table

J	K	Q
0	0	Qo
0	1	0
1	0	1
1	1	Toggle

JK Flip Flop Transition Table

Q_N	Q_{N+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

The JK flip flop truth table is not required in the design procedure but has been included to explain how the JK flip flop transition table is obtained. Click on any green cell of the JK flip flop transition table to learn how its value has been derived.

Use K-map to derive the logic equations.

$J_A = D\ Q_B + D\ Q_B$

Q_AQ_B \ D	0	1
00	0	1
01	1	0
11	X	X
10	X	X

$K_A = D\ Q_B + D\ Q_B$

Q_AQ_B \ D	0	1
00	X	X
01	X	X
11	1	0
10	0	1

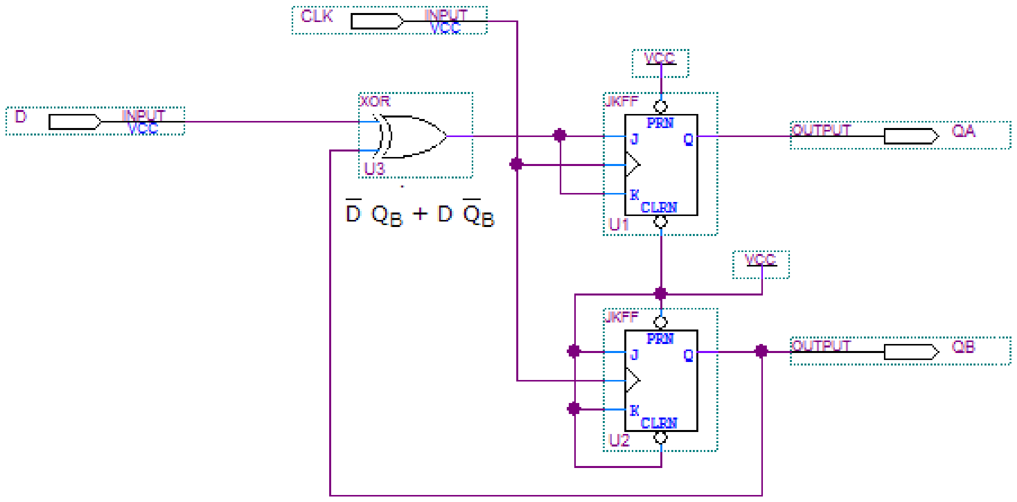
$J_B = 1$

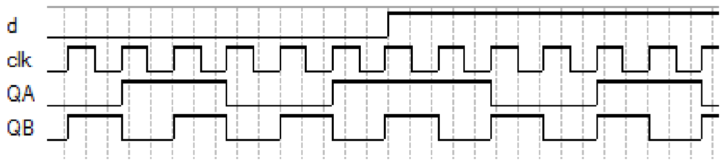
Q_AQ_B \ D	0	1
00	1	1
01	X	X
11	X	X
10	1	1

$K_B = 1$

Q_AQ_B \ D	0	1
00	X	X
01	1	1
11	1	1
10	X	X

Use the Boolean expressions to implement the counter.

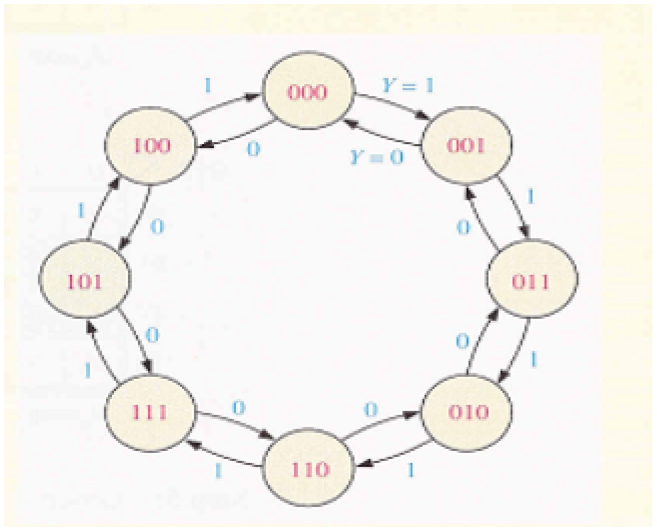




LAB TASK

Q1 : Implement all the above flip flops atleast once.

Q2) Design a 3 bit Up/DOWN Counter implement it in logism using JK-Flip Flop:



Truth Table: (JK- flip flop and states truth tables)

Diagram

Q3) Design a binary counter to count 0-63 number implement in logics work store the value 55 in the circuit

Q4: You are now familiar with the concepts of memory storage and counters. Implement a traffic signal with programmable counters (Set a value as how long the counter must count i.e. a 3 bit would only allow a timer of 16 seconds). Set a condition where if there are no cars present the lights should turn yellow and only turn on if any of the cars are present in atleast two intersections. There should also be a pedestrian counter.

Note: You can use the built in counter as logism would have an oscillation problem.