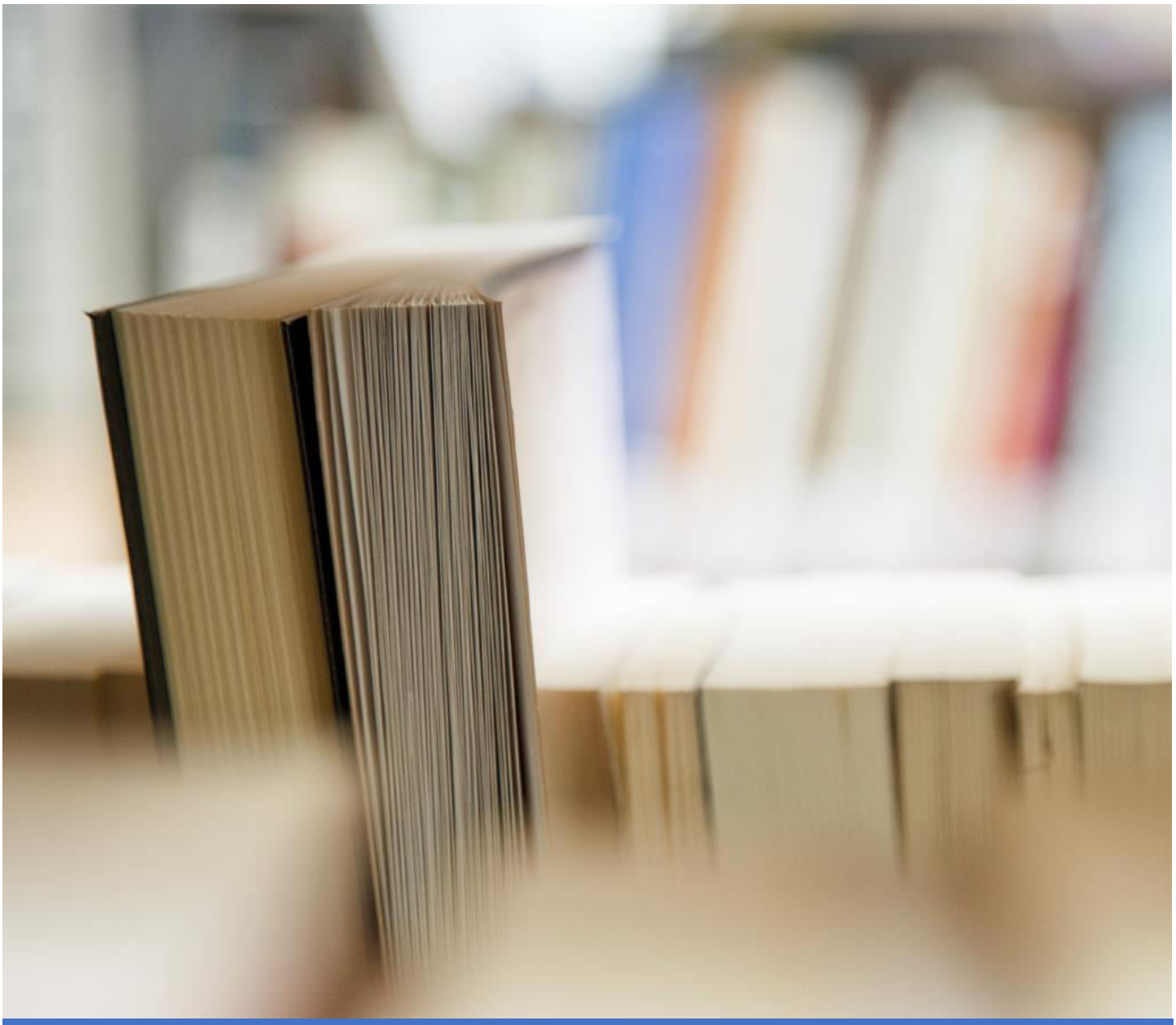


# Counters and Sequential Circuits

Experiment 7 and 8

BIMAL PARAJULI (20BDS0405)

7/17/21

CSE – 1003 (LAB)

Bimal Parajuli (20BDS0405)

# DIGITAL LOGIC DESIGN

## FLIP-FLOPS

EXPERIMENT #10

## Objectives:

1. To become familiar with flip-flops.
2. To implement and observe the operation of different flip-flops.

## Apparatus:

- IC type 7400 quad 2-input NAND gate
- IC type 7410 triple 3- input NAND gate
- IC type 7476 dual JK master-slave flip-flops.
- IC type 7474 dual D positive-edge-trigged flip-flops.
- Dual trace oscilloscope.

## Softwares Used

- LogicWorks 5
- Cadence Capture + PsPice

**Bimal Parajuli (20BDS0405)**

# Procedure:

1. ***In the pre-lab using LogicWorks construct the circuit shown in Fig. 1***

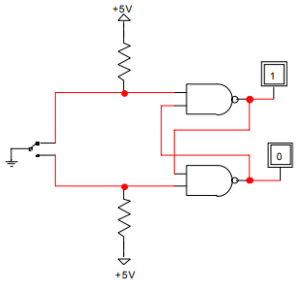
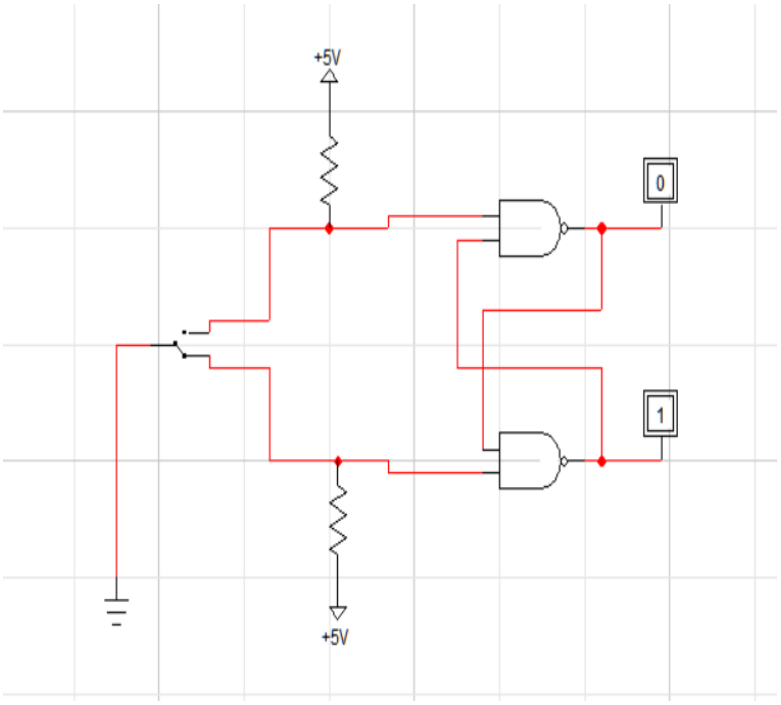
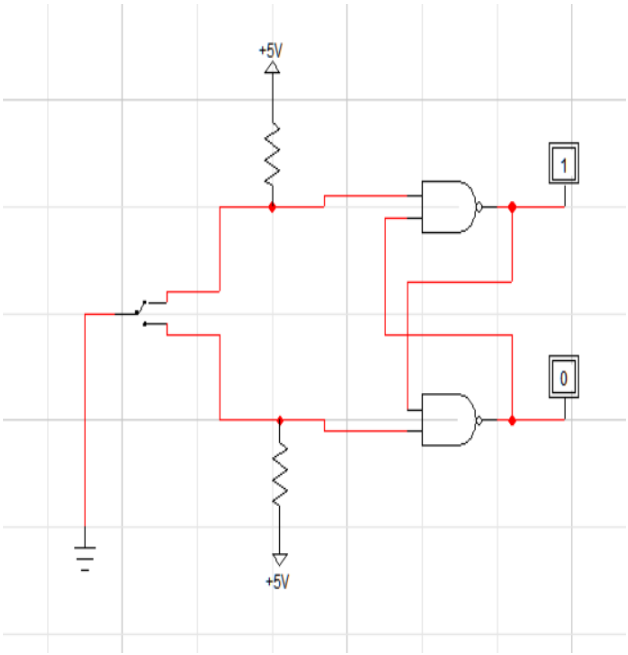



**Fig :1**

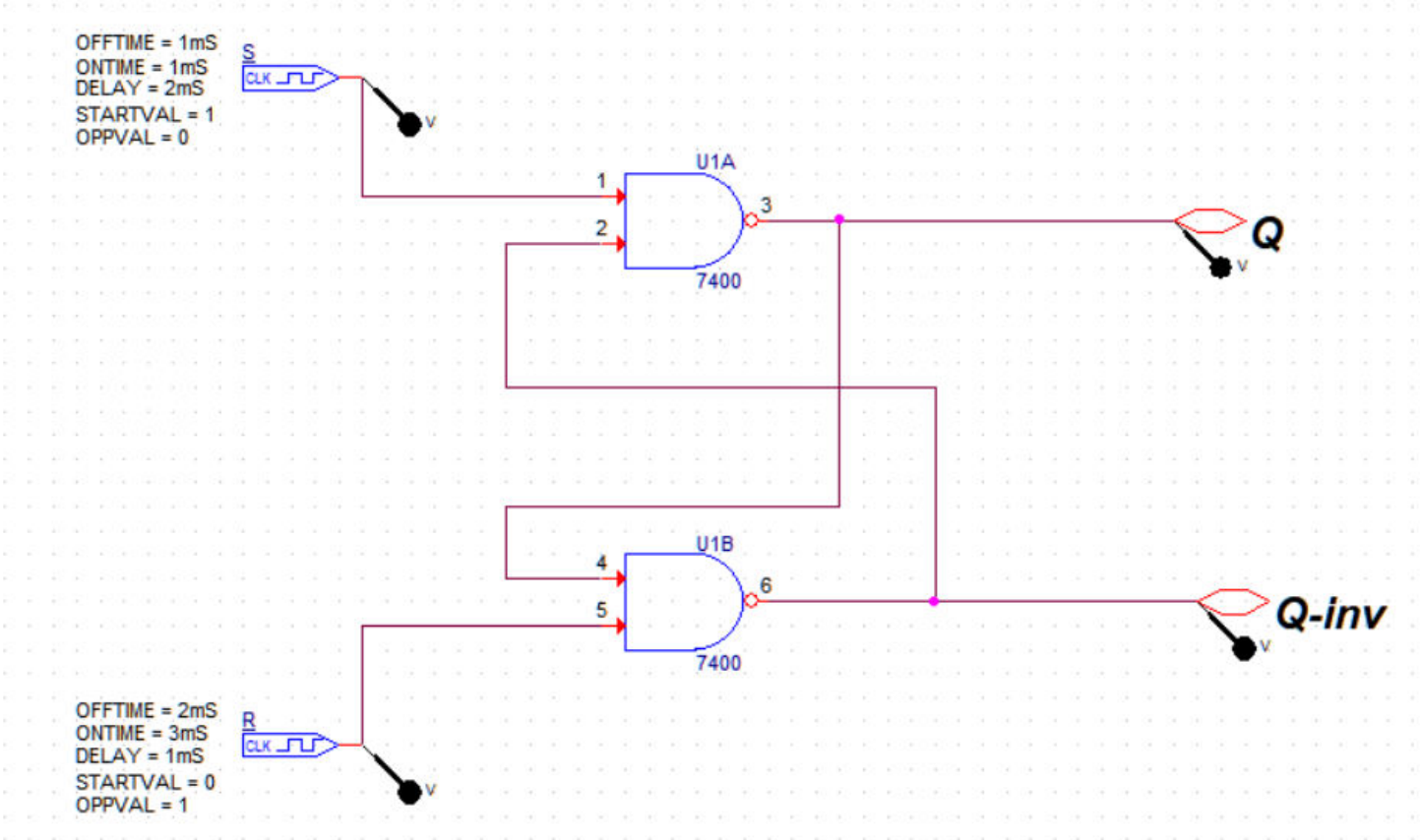


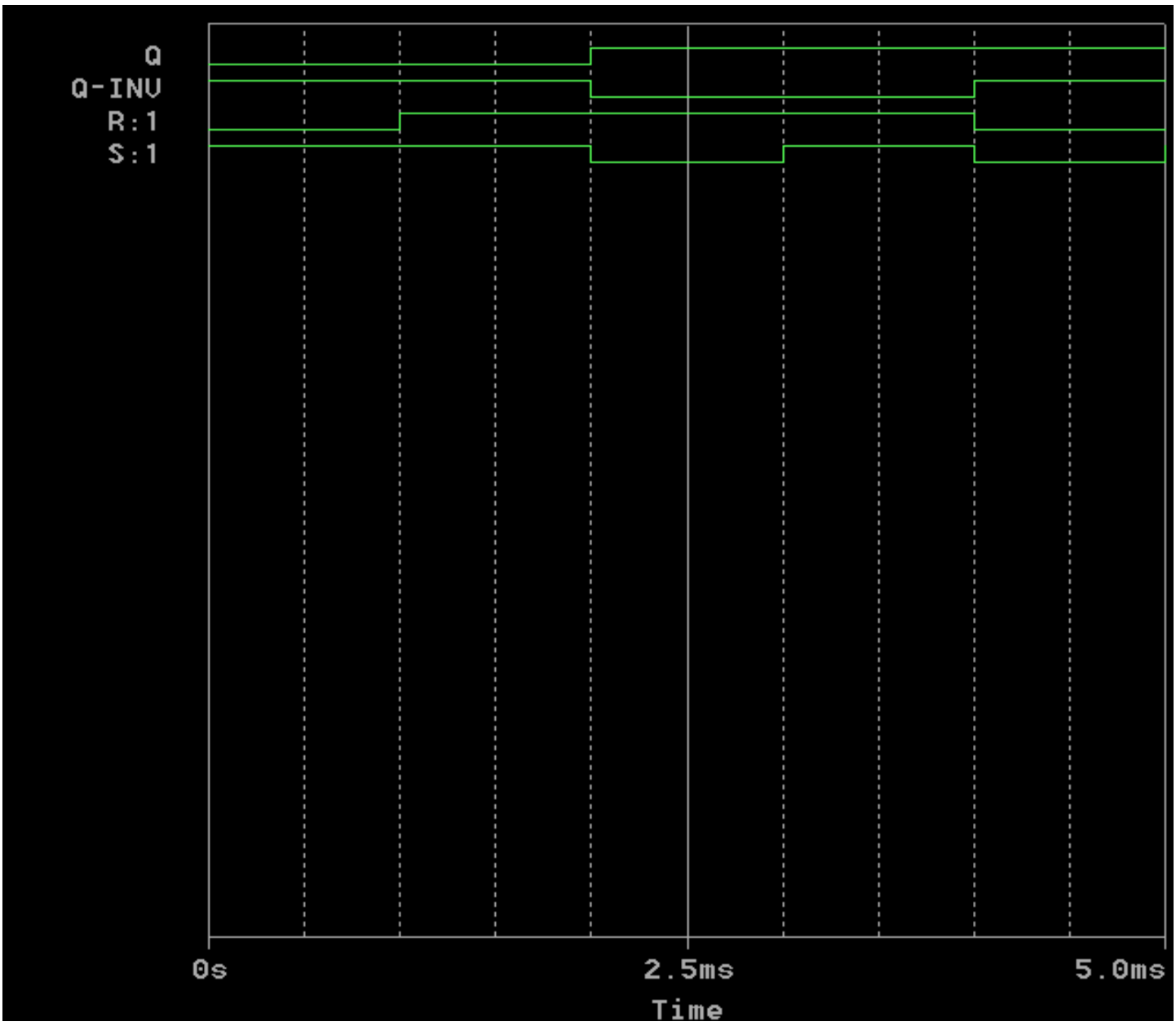




*Where we could use generic NAND gates or 74-00 and Binary Probes to simulate LEDs. Finally, we use SPDT for the bouncing switch. Using the simulated circuit fill in the truth table.*

| S | R | Q | Q′ | Cases |
|---|---|---|---|---|
| 1 | 0 | 0 | 1 | Reset |
| 1 | 1 | 0 | 1 | Memory |
| 0 | 1 | 1 | 0 | Set |
| 1 | 1 | 1 | 0 | Memory |
| 0 | 0 | 1 | 1 | Invalid |

*On doing laboratory simulation using OrCAD PsPice, we obtain the following circuit which is tabulated in the above table*




Q
Q-INV
R:1
S:1
0s
2.5ms
5.0ms
Time

***In the Lab, Build the RS latch shown in fig.2. Use SPDT switch S2 as a bouncing switch. Q and Q' Outputs are connected to LED 'S of the PB-503. Verify the truth table experimentally.***

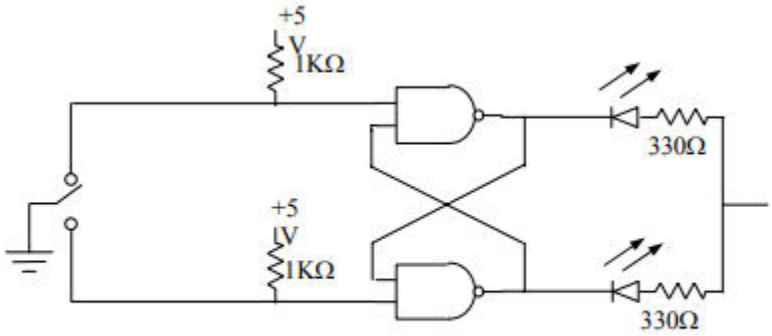
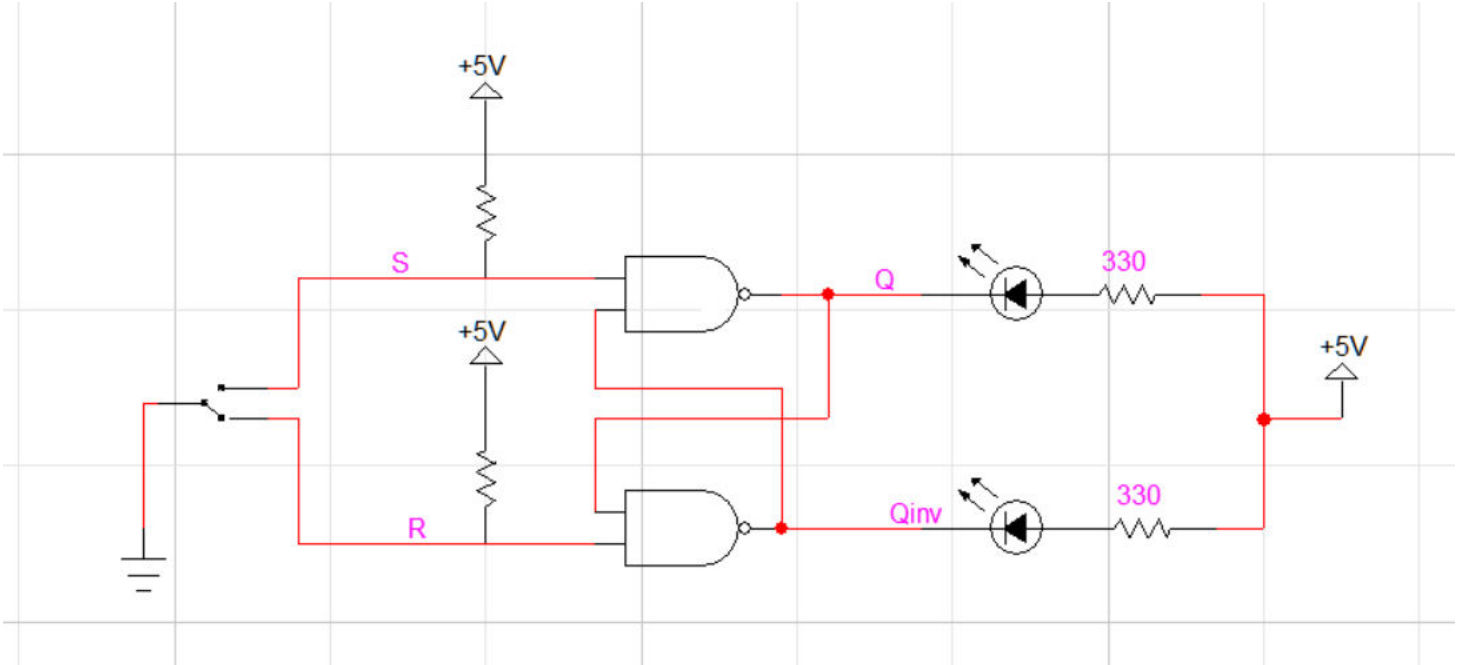
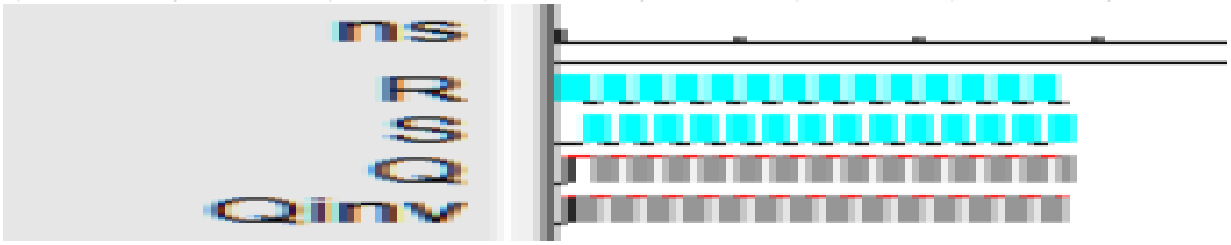



Fig. 2

*Following is the SR latch made in laboratory using SPDT switch input in LogicWorks 5*



2. ***Modify the basic R-S into a D latch by adding the steering gates and the inverter shown in Fig 3.***
   ***Connect the D input to the pulse generator of the Digi designer and set it at 1 Hz.***
   ***Connect the enable input to a high through 1k resistor. Observe the output; obtain the truth table experimentally then change the enable to a low.***
   ***Is the enable an active high or an active low? Leave the enable low and place a momentary short to ground first on one output and then on the other. What happens?***

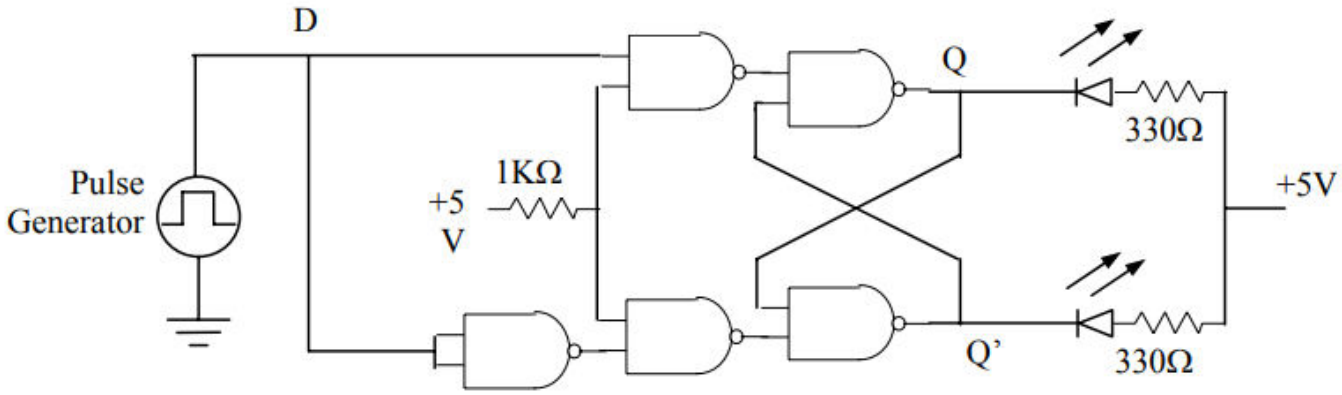
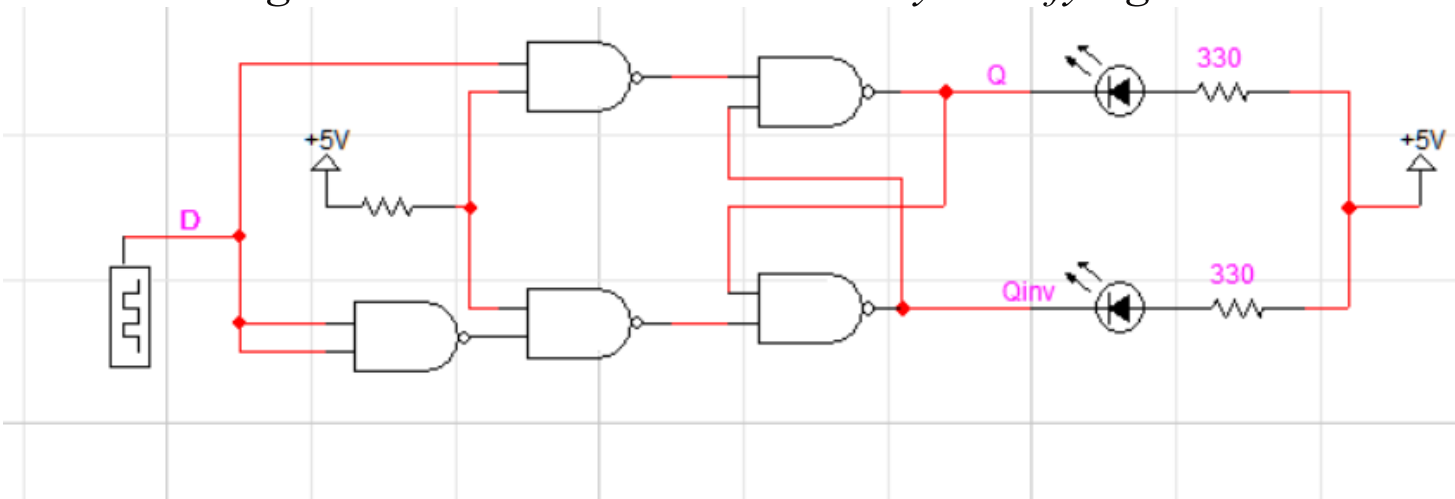
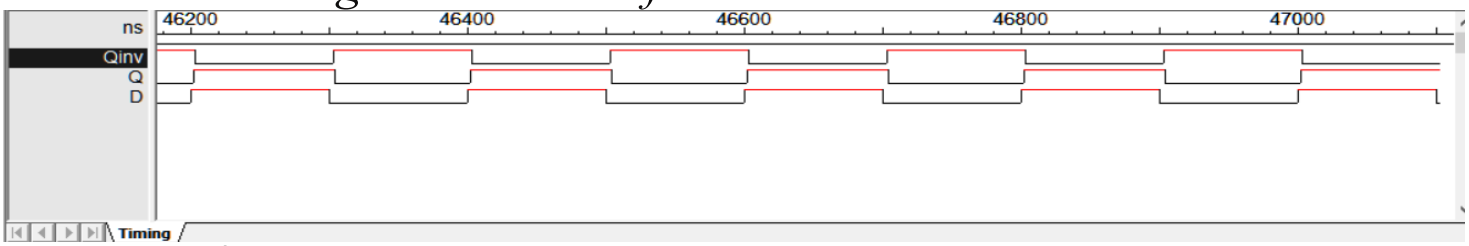



Fig: 3

*Following is the Basic D Latch created by modifying SR latch.*




*Following is the result of its simulation in time domain:*

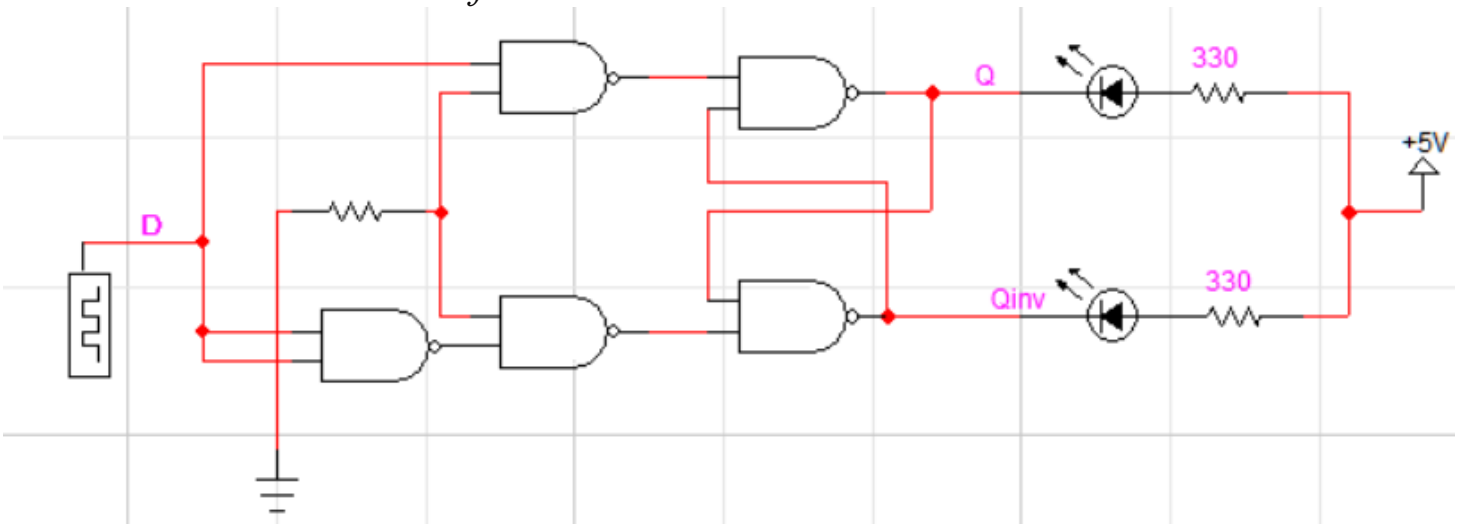


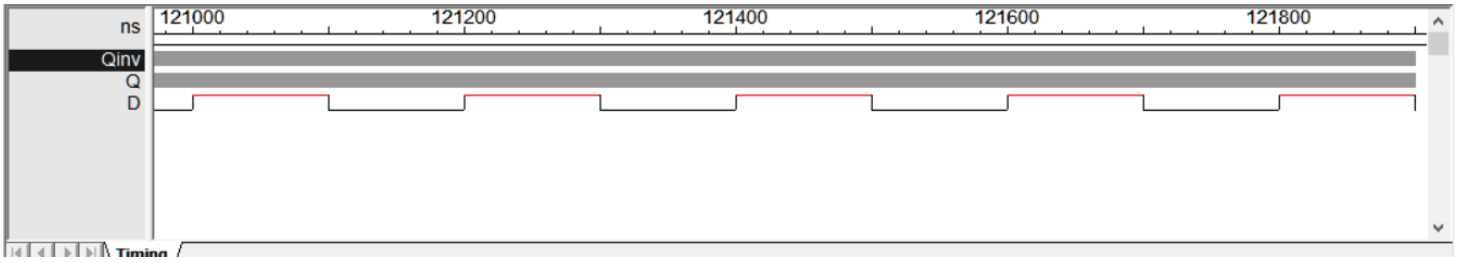
*Thus the following behavior is obtained:*

| *Enable* | *D* | *Q* | *Q'* | *Observation* |
|---|---|---|---|---|
| *0* | *X* | *Retained* | *Retained* | *No change* |
| *1* | *0* | *0* | *1* | *Reset* |
| *1* | *1* | *1* | *0* | *Set* |

*The enable of D latch is active high one. If we set enable as low, the output of the flip-flop stays the same and independent to the input value D.*

*Which is demonstrated as follows:*



2. ***The 7476 is a dual JK master-slave flip-flop with preset and clear inputs. The function table given in table 1 defines the operation of the flip-flop. The +ve transition of the CLOCK (CP) pulse changes the master flip-flop, and the (-ve) transition changes the slave flip-flop as well as the output of the circuit. In LogicWorks the chip 7476 is not available, however, the generic JK flip-flop behave in exactly the same way as the 7476. The "S" represents the Preset, the "R" represents the Clear, and C represents the clock pulse (CP). Verify the table by connecting Binary switches to R, S, J, K, and C. Notice that only the negative edge of the clock affects the outputs (Q, and Q').***

Table 1

| Input | | | | | Output | |
|---|---|---|---|---|---|---|
| Preset | Clear | Clock | J | K | Q | Q’ |
| 0 | 1 | X | X | X | 1 | 0 |
| 1 | 0 | X | X | X | 0 | 1 |
| 0 | 0 | X | X | X | 1 | 1 |
| 1 | 1 | ↓ | 0 | 0 | No change | |
| 1 | 1 | ↓ | 0 | 1 | 0 | 1 |
| 1 | 1 | ↓ | 1 | 0 | 1 | 0 |
| 1 | 1 | ↓ | 1 | 1 | Toggle | |

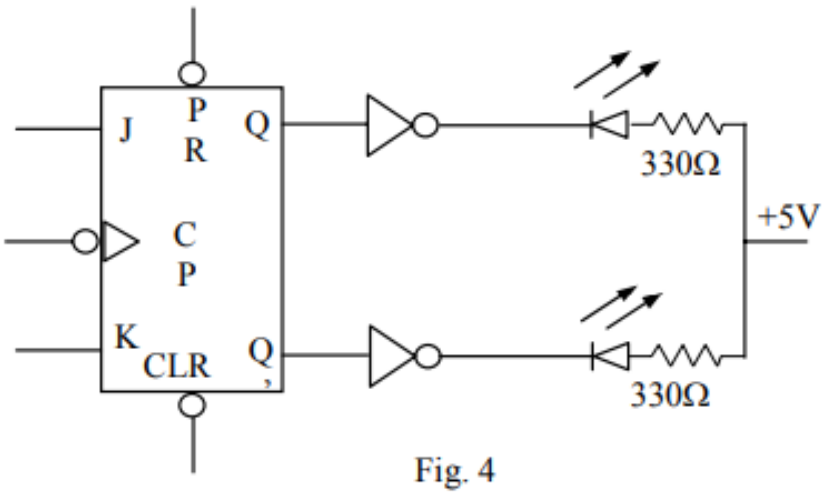


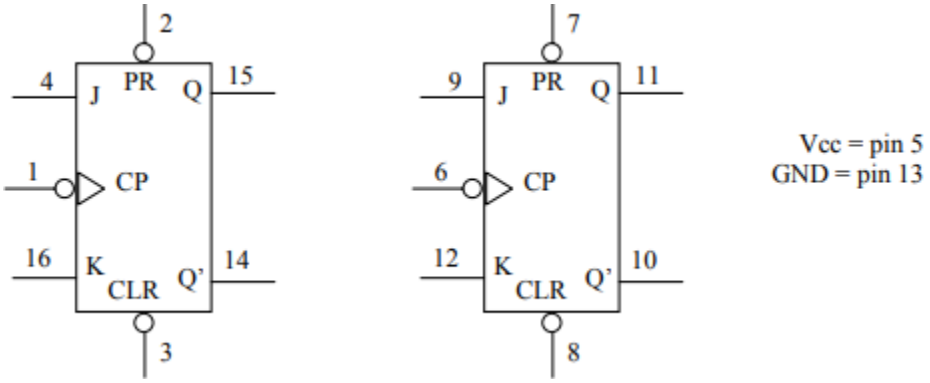

Fig. 4

***In the Lab, Construct the circuit of Fig 4. Look at the data sheet for the 7476 and determine the inactive logic required at the PRE and CLR inputs.***

***Connect the 7476 for the SET mode by connecting J = 1, K = 0. With CLOCK (CP) = O; test the effect of PRE, CLR by putting a 0 on each, one at a time.***

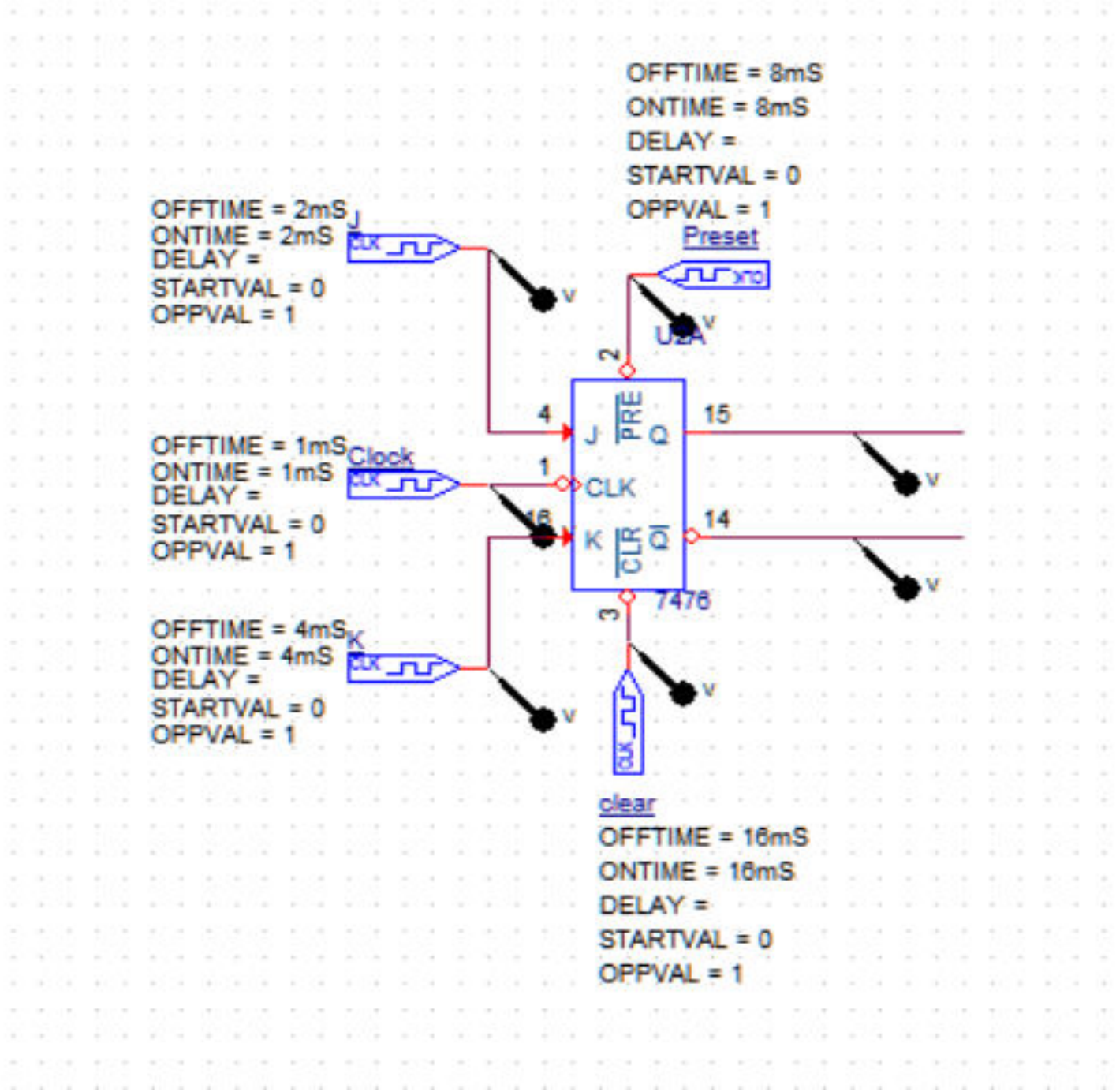
***Put CLR = 0, then pulse the clock (CP) by putting a HIGH then a LOW, on the clock. Does the CLR input override J input?***

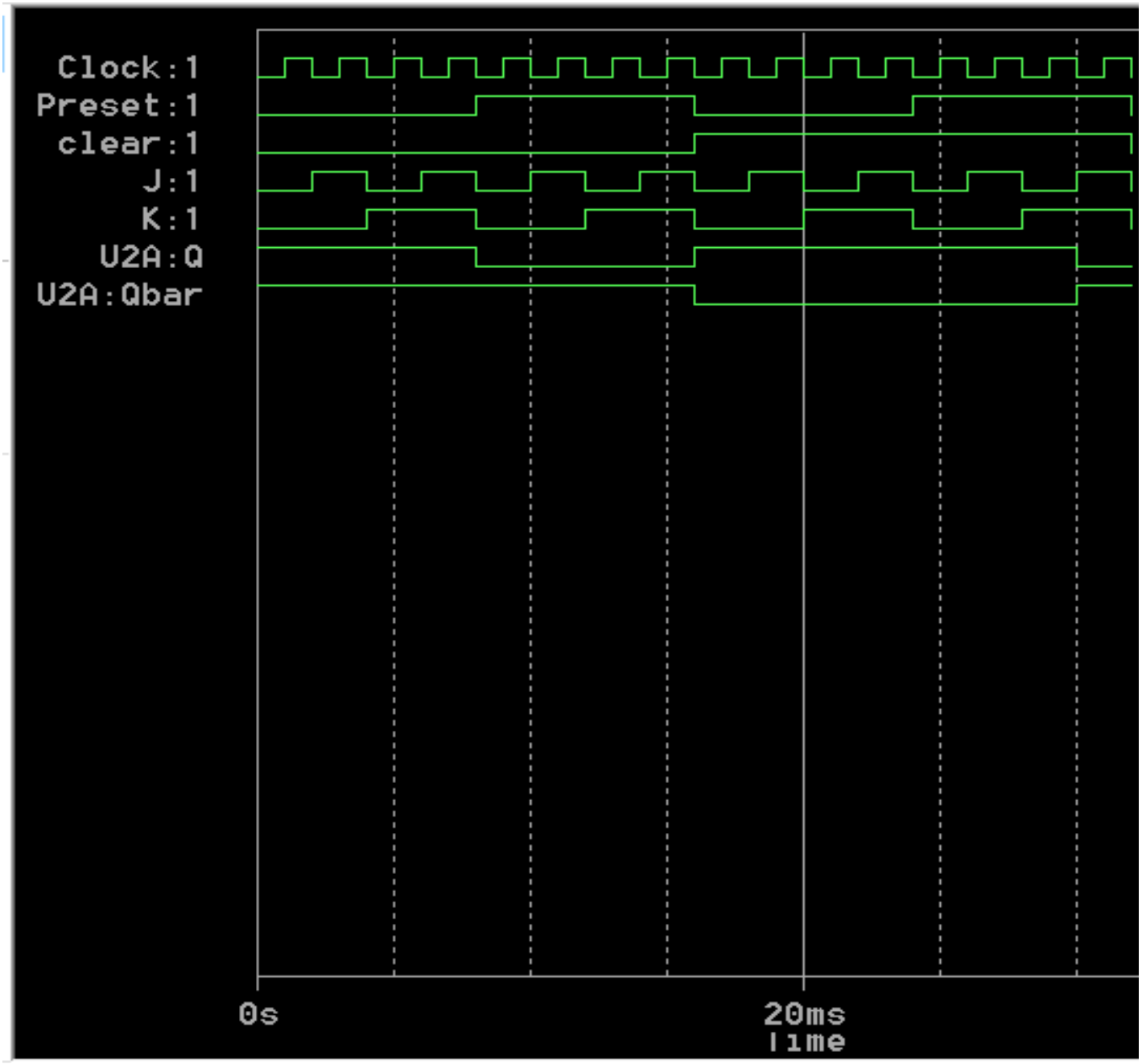
***Verify the operation of the JK flip flop by experimentally obtaining the characteristics Table.***



*Following is the arrangement to observe and study the behavior of JK flip flop with preset and clear inputs.*

*In the transient simulation alongside, we can observe how **Q** and **Q'** change with the change in clock, J, K, Preset and Clear.*




Clock:1
Preset:1
clear:1
J:1
K:1
U2A:Q
U2A:Qbar
0s
20ms
Time

## ***Observations:***

*When Preset =0 and Clear =0, always Q=1, Q'=1*

*When Preset =0 and Clear =1, always Q=1, Q'=0*

*When Preset =1 and Clear =0, always Q=0, Q'=1*

*When Preset =1 and Clear =1, Q depends on J and K at negative edge of the clock.*

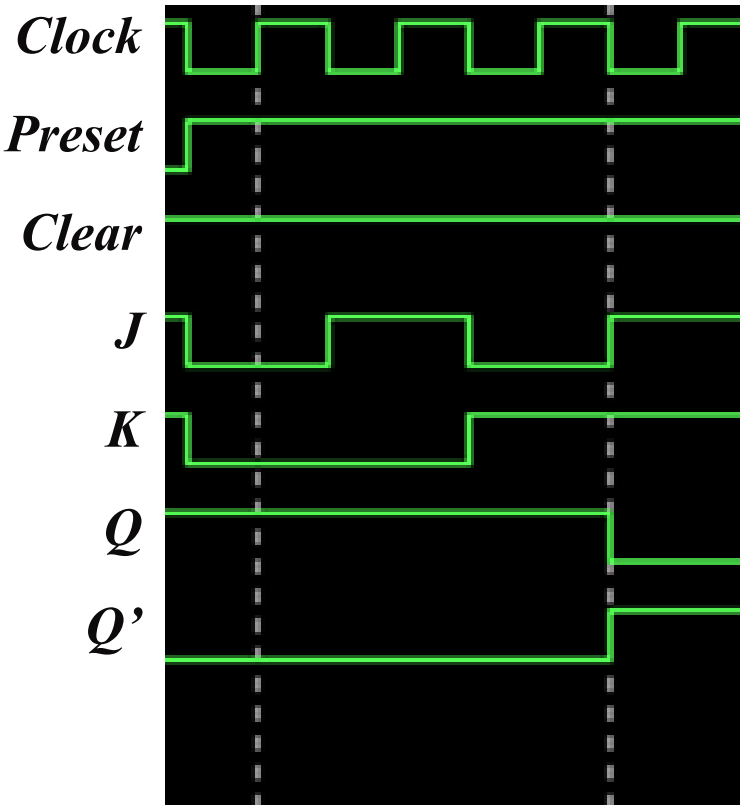
*When J=0, K=0: Q and Q' remains same*

*When J=0, K=1: Q=0 and Q'=1*

*When J=1, K=0: Q=1 and Q'=0*

*When J=1, K=1: Q and Q' toggles.*

*This is the small part of simulation when both Preset, Clear are 1.*



*Following characteristics table is obtained for JK flip flop:*

| 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 |

***Does the CLR input override the J input?***

- *Yes, CLR and preset can override J, K and Clock inputs. J, K and Clock are significant only when preset and clear inputs are set to 1, 1*

# DIGITAL LOGIC DESIGN EXPERIMENT #8

# CLOCKED SEQUENTIAL CIRCUITS AND COUNTERS

## OBJECTIVE:

- To design, build and test synchronous sequential circuits.
- To design, build, and test synchronous counters
- To design, build and test asynchronous counters

## APPARATUS:

- IC type 7476 dual JK master-slave flip-flops
- IC type 7400 quad 2-input NAND gates

## Softwares Used:

- Cadence Capture CIS Lite

# PROCEDURE:

## 1. *SYNCHRONOUS SEQUENTIAL CIRCUITS:*

***a) Design, construct and test a sequential circuit whose state is shown in Fig.1. Use JK flip-flops in the design.***

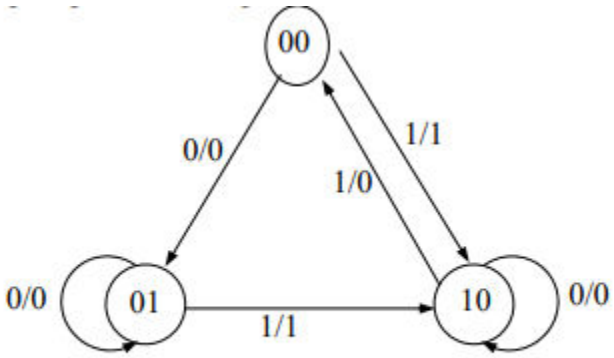



Fig. 1

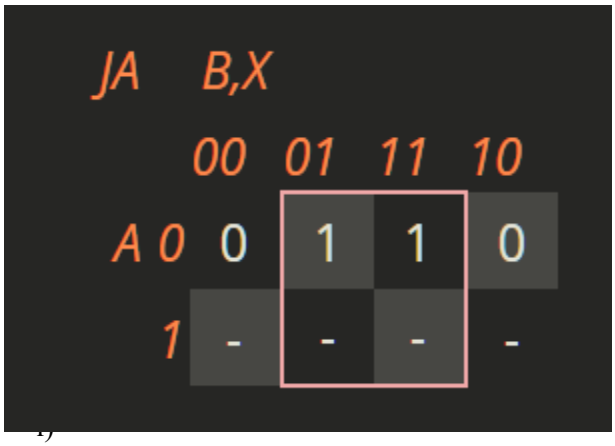
***The circuit has two flip-flops A, B, one input x and one output y. The circuit is to be designed by treating the unused states as don't care conditions. The final circuit must be analyzed to ensure that it is self-correcting. If not suggest a solution.***

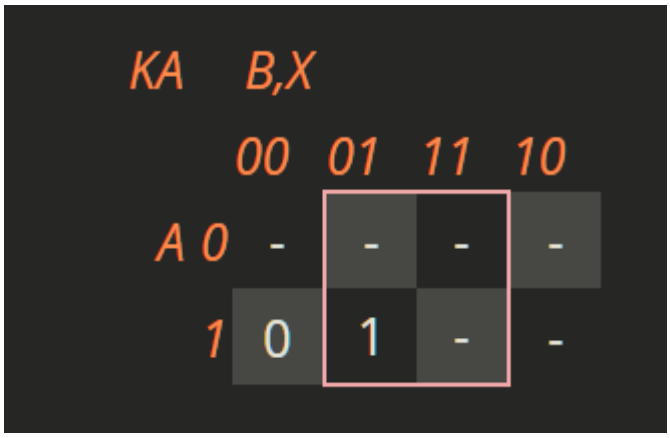
b) *Complete the excitation table shown in Table 1*

Table 1.

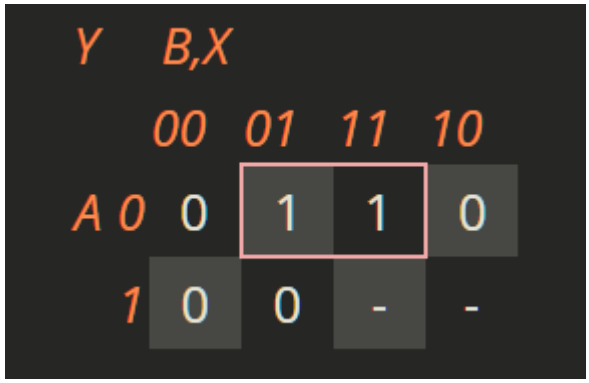
| Present state | | Input | Next state | | Output | Flip-flop input function | | | |
|---|---|---|---|---|---|---|---|---|---|
| A | B | X | A | B | Y | JA | KA | JB | KB |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | X | 1 | X |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | X | 0 | X |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | X | X | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | X | X | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | X | 0 | 0 | X |
| 1 | 0 | 1 | 0 | 0 | 0 | X | 1 | 0 | X |
| 1 | 1 | 0 | X | X | X | X | X | X | X |
| 1 | 1 | 1 | X | X | X | X | X | X | X |

c) *Using Karnaugh maps obtain minimal expressions for the flip-flop input functions JA, KA, JB, KB.*







From the above Kmap, Input functions are:

**JA = X**

**KA = X**

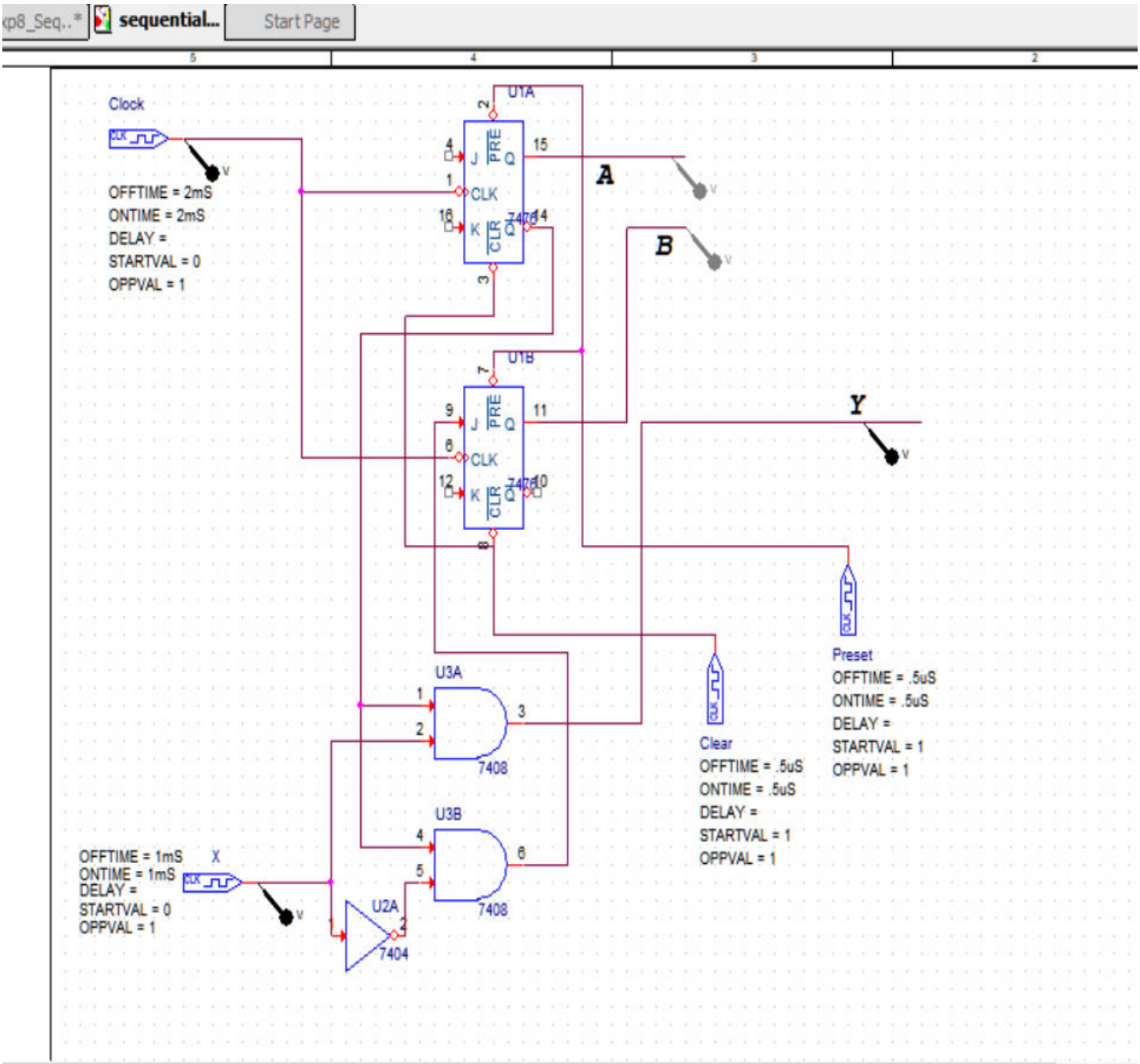
**JB = A'X'**

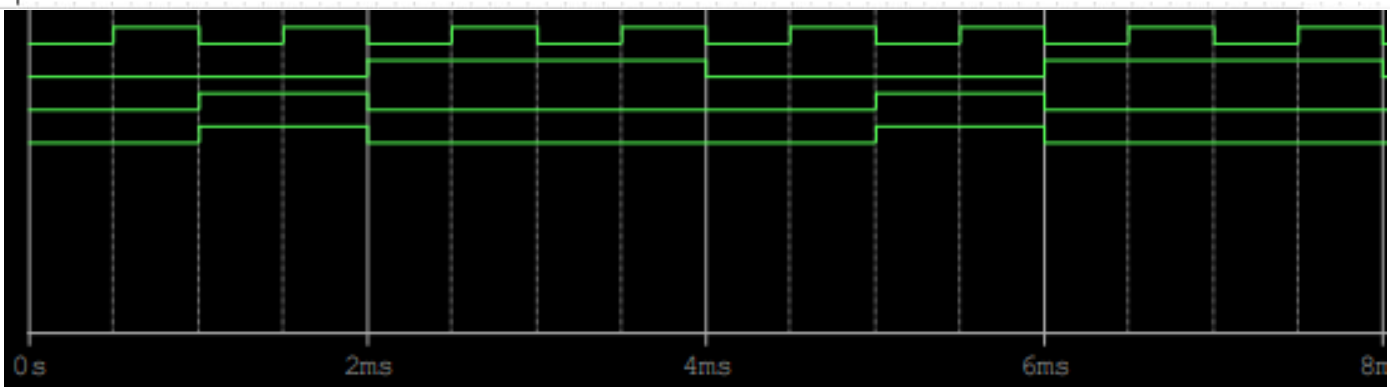
**KB = X**

Output function is:

**Y=A'X**

*d. Simulate the circuit using LogicWorks. LogicWorks does not have the JK master-slave flip-flop IC 7476. Use instead the generic JK flip-flop as you did in experiment 9. In the Lab, build the circuit and check the output to verify the state table values.*


sequential...
Start Page
Clock
OFFTIME = 2mS
ONTIME = 2mS
DELAY =
STARTVAL = 0
OPPVAL = 1
U1A
U1B
7476
A
B
Y
U3A
U3B
7408
U2A
7404
OFFTIME = 1mS
ONTIME = 1mS
DELAY =
STARTVAL = 0
OPPVAL = 1
X
Clear
OFFTIME = .5uS
ONTIME = .5uS
DELAY =
STARTVAL = 1
OPPVAL = 1
Preset
OFFTIME = .5uS
ONTIME = .5uS
DELAY =
STARTVAL = 1
OPPVAL = 1


0 s
2ms
4ms
6ms

## 2. *Synchronous Counters*

*Synchronous counters have all clock lines tied to a common clock causing all flip-flops to change at the same time. The count sequence of a counter can be analyzed by placing the counter into every possible number in the sequence and determining the next number in the sequence state diagram is developed as the analysis proceeds. (A state diagram is an illustration of the transitions that occur after each clock pulse).*

a) *In the pre-lab using LogicWorks and then in the lab using hardware chips, design a 2-bit gray code counter using JK flip-flops. The required sequence is the binary equivalent of (0-1-3-2-0). A state diagram for this counter is given in Fig. 2.*

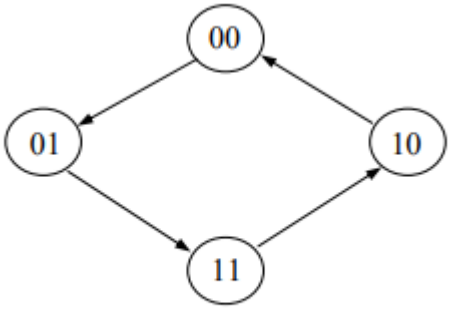
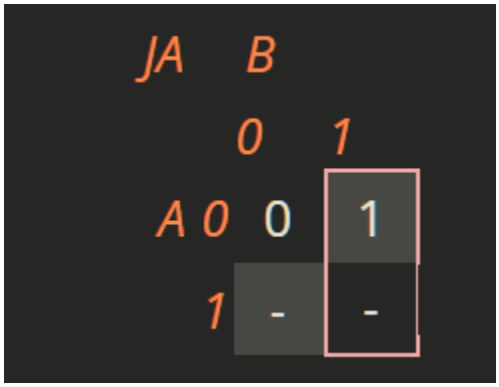


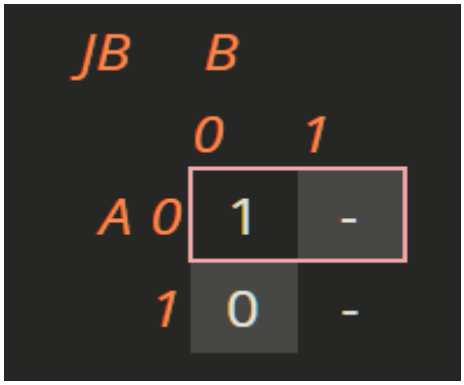

Fig. 2

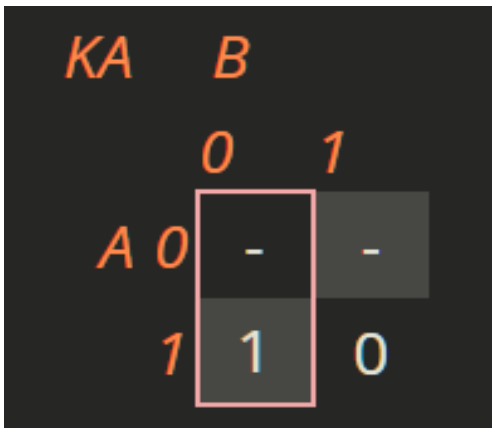
b) *Complete the excitation table (Table 2) for the counter and obtain logic expression for the JK flip-flop input functions.*

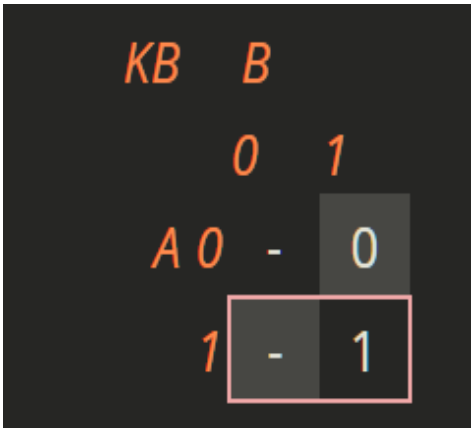
| Present_State | | Next state | | Flip Flop input functions | | | |
|---|---|---|---|---|---|---|---|
| A | B | A | B | JA | JB | KA | KB |
| 0 | 0 | 0 | 1 | 0 | X | X | X |
| 0 | 1 | 1 | 1 | X | X | X | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | X |
| 1 | 1 | 1 | 0 | X | 0 | 0 | 1 |

The respective KMaps of JA, JB, KA, KB are:



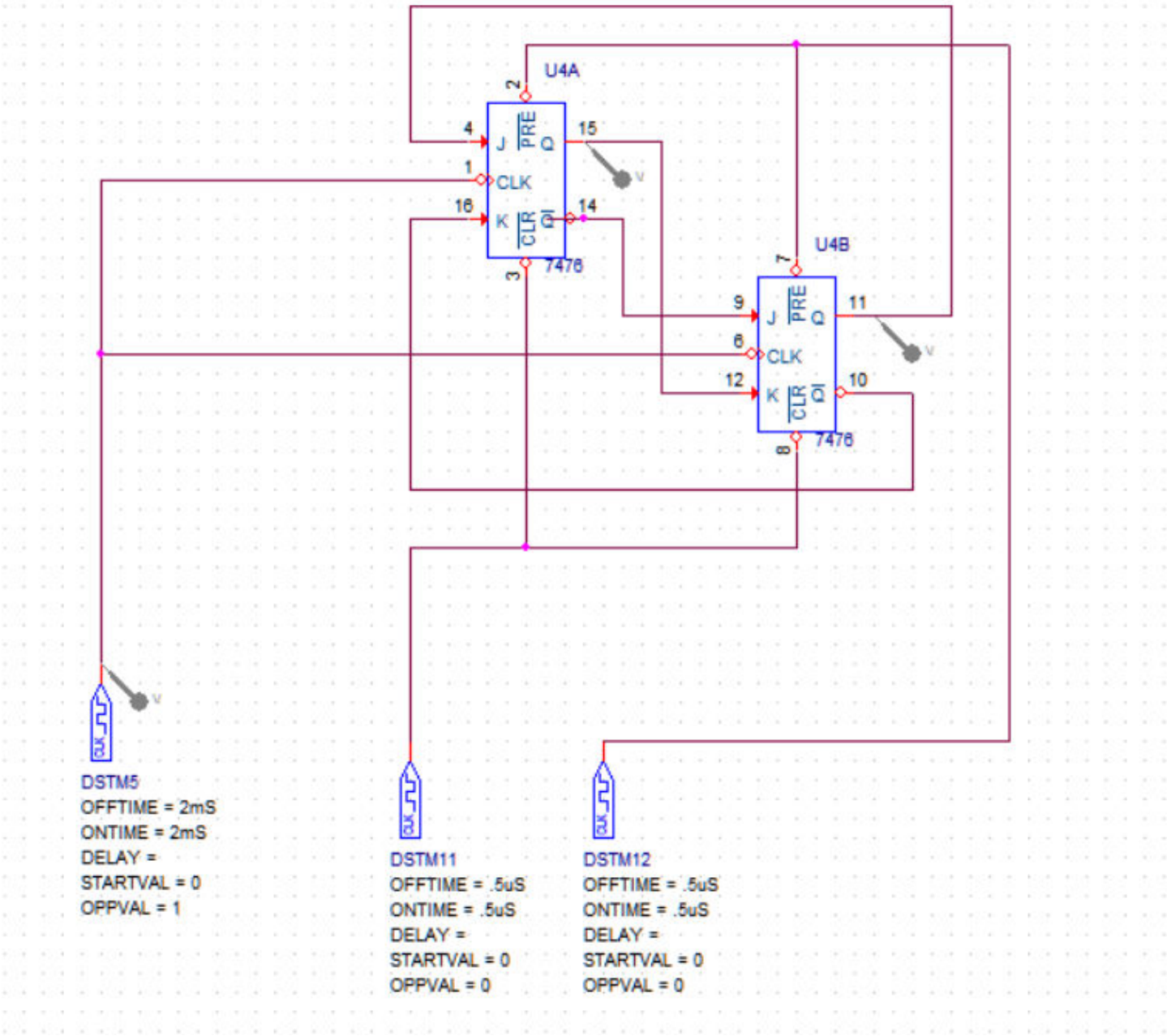




Flip-flop input functions are:

**JA=B** **KA=B'**

**JB=A'** **KB=A**

c) *In the lab, build the circuit and test it by pulsing it from the PB-503. Check that the output is the designed sequence.*



## 3. *Asynchronous Counters*

*Asynchronous counters are a series of flip-flops each clocked by the previous state, one after the other. Since all the stages of the counter are not clocked together, a ripple effect propagates as various flip-flops are clocked. For this reason they are called ripple counters. The modulus of a counter is the number of different output states the counter may take (i.e. Mod 4 means the counter has four output states).*

a) *In the pre-lab construct a 4-bit asynchronous counter shown in Fig.3. (It is also called binary ripple counter). Use four generic JK flip-flops. Connect four Binary Probes to Q outputs. Connect all R and S inputs to Logic 1 and connect a switch to the CP input.*

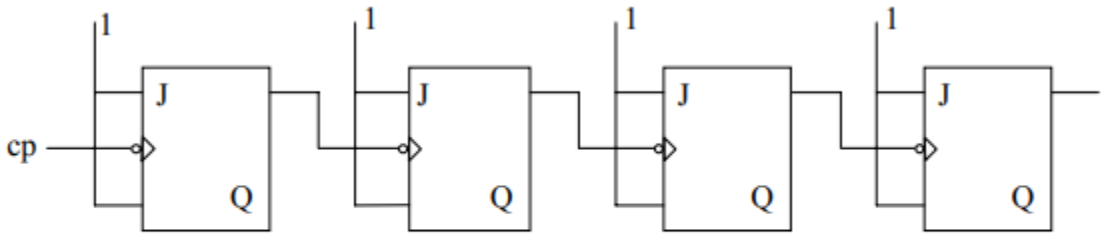
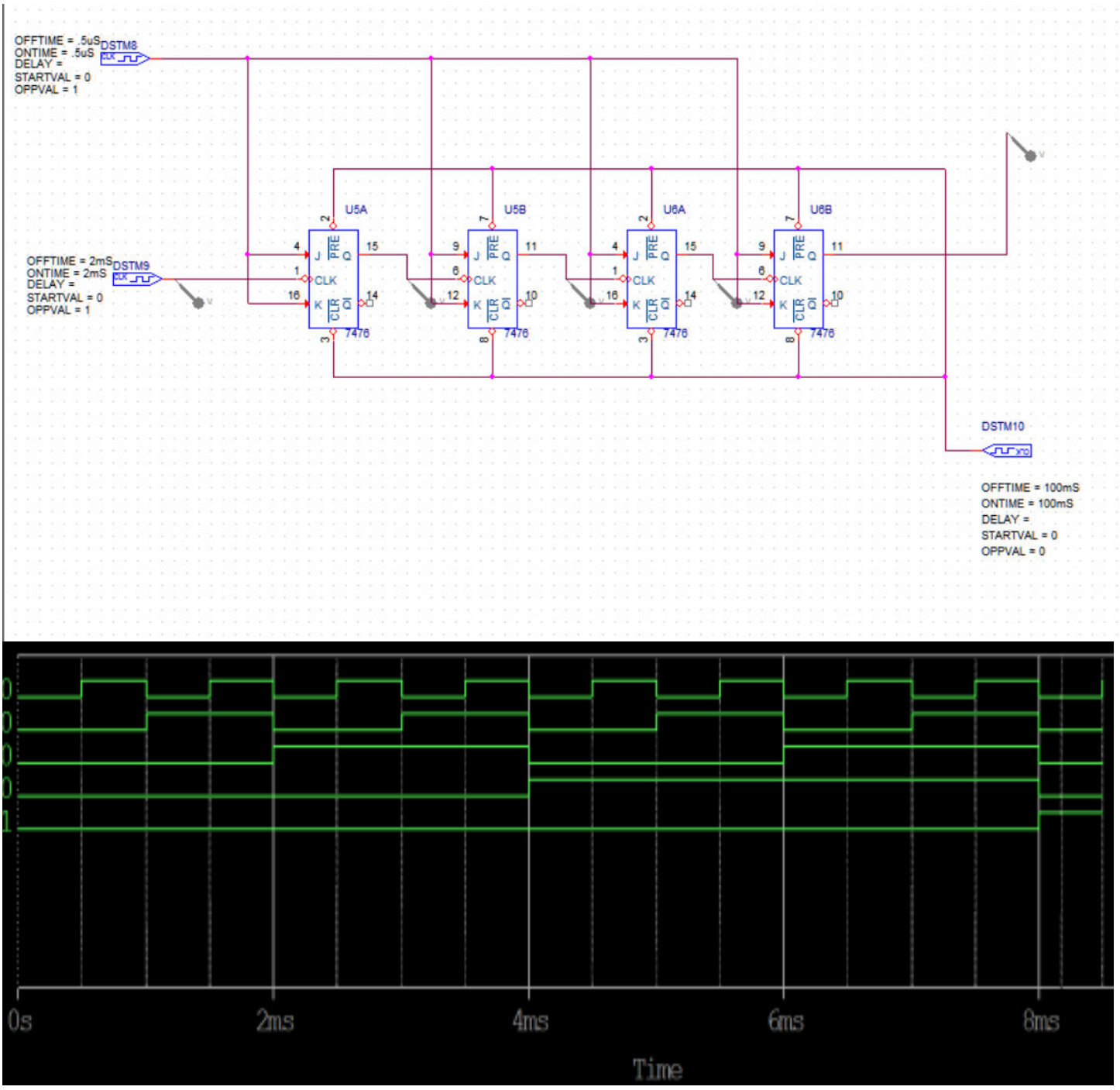



Fig. 3 4-bit ripple counter

b) In the Lab use two 7476 ICs to implement the design. Connect Q outputs of flip-flops to indicator lamps of the PB-503. Connect all clear (CLR) and preset (PRE) inputs to logic 1. Connect the CP input to the pulse output of the PB-503 and check the counter for proper operation.



c) Write down the count sequence in Table 3. Identify this count sequence (up or down). Comment on what happens after the application of 15 pulses to CP input.

Table 3. Count sequence for the 4-bit ripple counter.

| A | B | C | D |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 1 | 0 | `1 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |

The above given is up counter.

After the counter reaches upto 16, it resets to 0000 and the cycle again continues.