**Exp:1.** Design and implementation of an inverter

**AIM:** To design, simulate and draw the layout of CMOS inverter.

**TOOLS:**

1.Personal Computer

2.Microwind Software 3.5 vertion

**THEORY:**

A CMOS inverter contains a PMOS and a NMOS transistor connected at the drain and gate terminals, a supply voltage VDD at the PMOS source terminal, and a ground connected at the NMOS source terminal, were VIN is connected to the gate terminals and VOUT is connected to the drain terminals.

When VIN is low, the NMOS is "off", while the PMOS stays "on": instantly charging VOUT to logic high. When Vin is high, the NMOS is "on and the PMOS is "on: draining the voltage at VOUT to logic low.

**CIRCUIT DIAGRAM:**



|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| A | NOT A |
| 0 | 1 |
| 1 | 0 |

**Fig 1:Inverter Symbol Fig 2:Truthtable**

**PROCEDURE:**

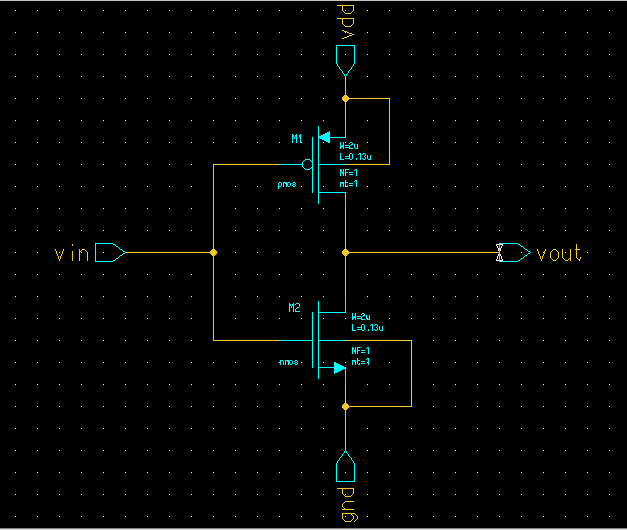
1. Connect the Circuit as shown in the circuit diagram using Pyxis schematic.
2. Create a simulation schematic for simulation.
3. Generate symbol for the schematic.

4.Add necessary ports to the symbol to analyze waveforms.

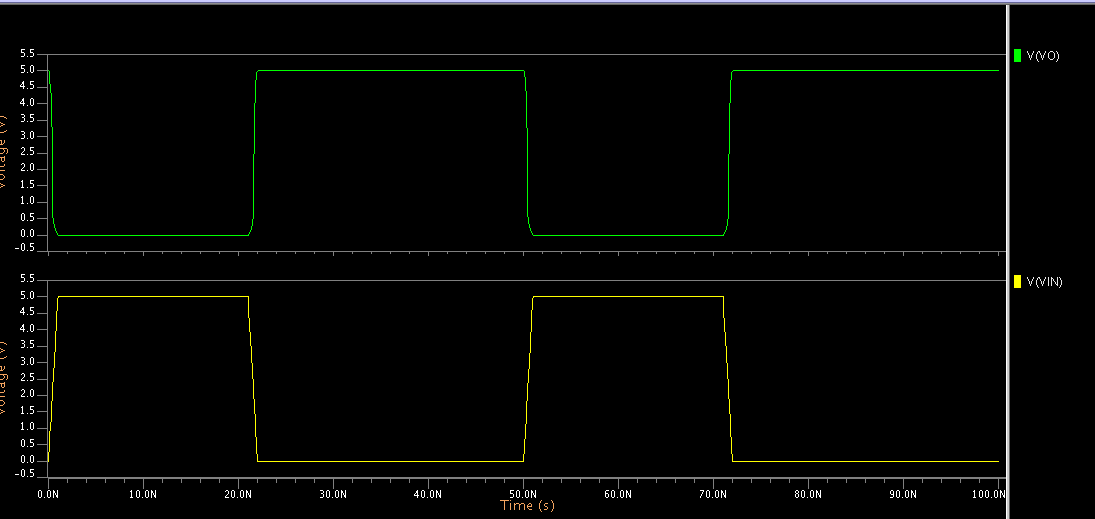
5.Run the Simulation and observe results in EZwave.

6.Draw the Layout for the circuit using PyxisLayout.

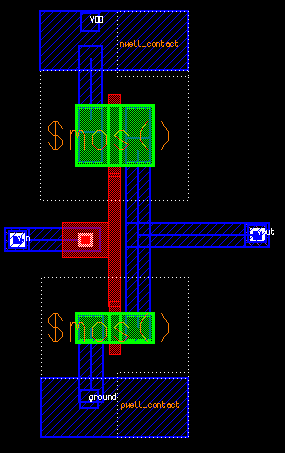
Observe the layout.

**SCHEMATIC DIAGRAM**

**SIMULATION WAVEFORMS:**



**LAYOUT:**



**RESULT:**

**Exp:2. Design and Implementation of Universal Gates**

**AIM:** To design, simulate and draw the layout of Universal Gates (NAND&NOR)

**TOOLS:**

* 1. PersonalComputer

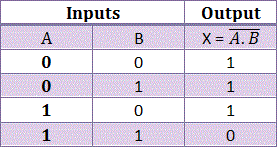
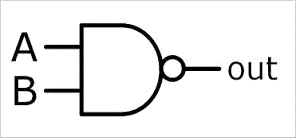
2. DSCH3 (Schematic Editor)

3. MICROWIND3.5 (Layout Editor)

**THEORY: CMOS NAND GATE**

A universal gate is a gate which can implement any Boolean function without need to use any other gate type. The NAND and NOR gates are universal gates

NAND GATE: When output of an AND gate is inverted through a NOT gate, the operation is called NAND operation. The logic gate which performs this NAND operation is called NAND gate. The basis logical construction of the NAND gate is similar to AND gate but one bubble is drawn at the output point of the AND gate. The output of this gate is just reverse of that of a similar AND gate. Operation: We know that the output of the AND gate is only high or 1, when all the inputs are high or 1. In all other cases, the output of AND gate is low or 0. In the case NAND, the case is a just opposite, here, the output is only low or 0 when and only when all inputs of the gate are 1 and in all other cases, the output of NAND gate is high or1.

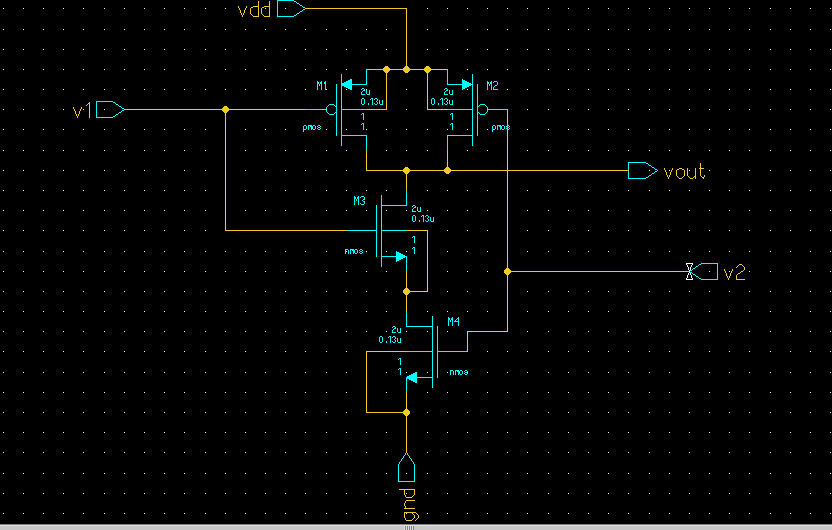
..**CIRCUIT DIAGRAM:**

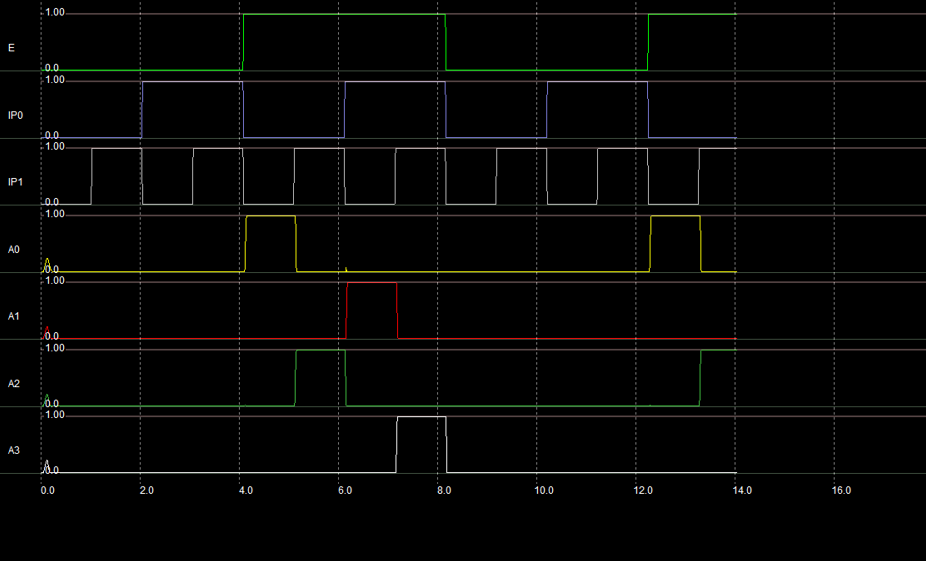
**Fig1:NANDsymbol Fig 2: Truthtable**

**PROCEDURE:**

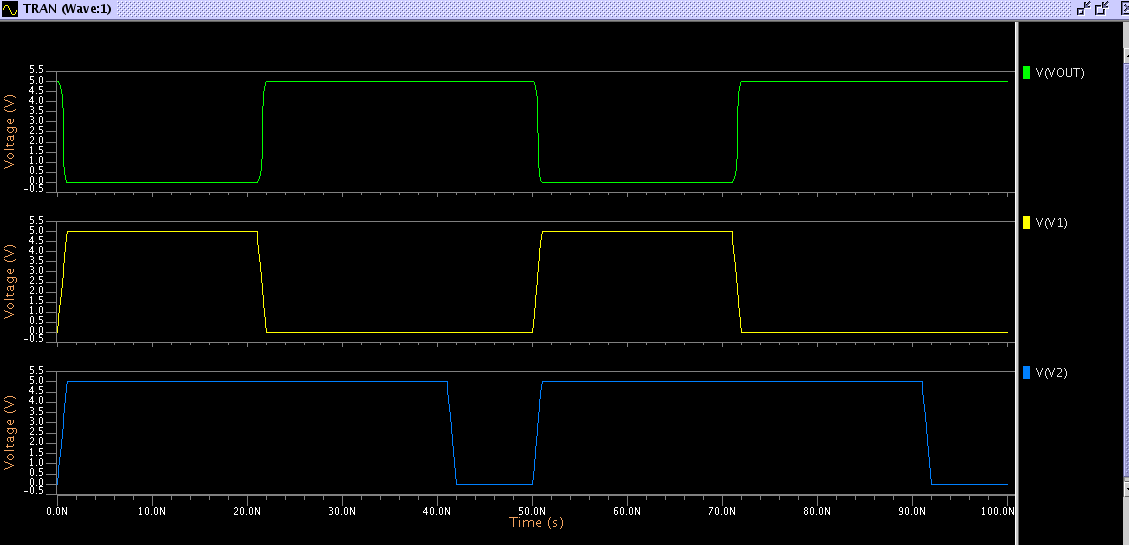
1. Connect the Circuit as shown in the circuit diagram using Pyxisschematic.
2. Create a simulation schematic for simulation.
3. Generate symbol for the schematic.
4. Add necessary ports to the symbol to analyze waveforms.
5. Run the Simulation and observe results in EZwave.
6. Draw the Layout for the circuit using Pyxis Layout.
7. Observe the layout.

**SCHEMATIC DIAGRAM:**

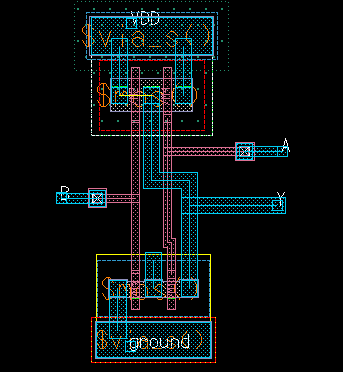




**SIMULATION WAVEFORMS:**



**LAYOUT:**

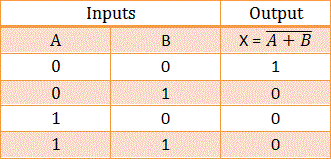


**THEORY: CMOS NOR GATE**

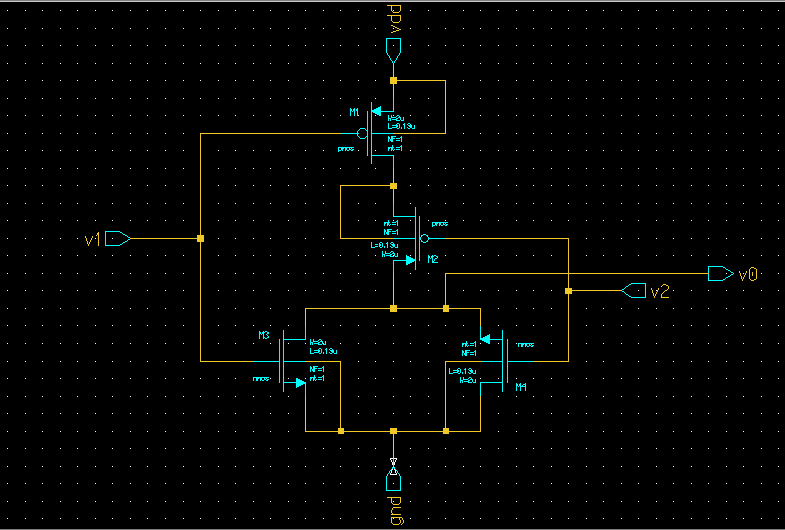
NOR gate means NOT OR gate. In a NOR gate an OR gate is inverted through

a NOT gate. Actually an inverted OR operation is NOR operation and the logic gate performs this operation is called NOR gate. The basic logic construction of the NOR gate is similar to OR gate but one bubble is drawn at the output point of the OR gate in the case of OR gate. NOR gate means not an OR gate which means output of this gate is just reverse of that of a similar OR gate.

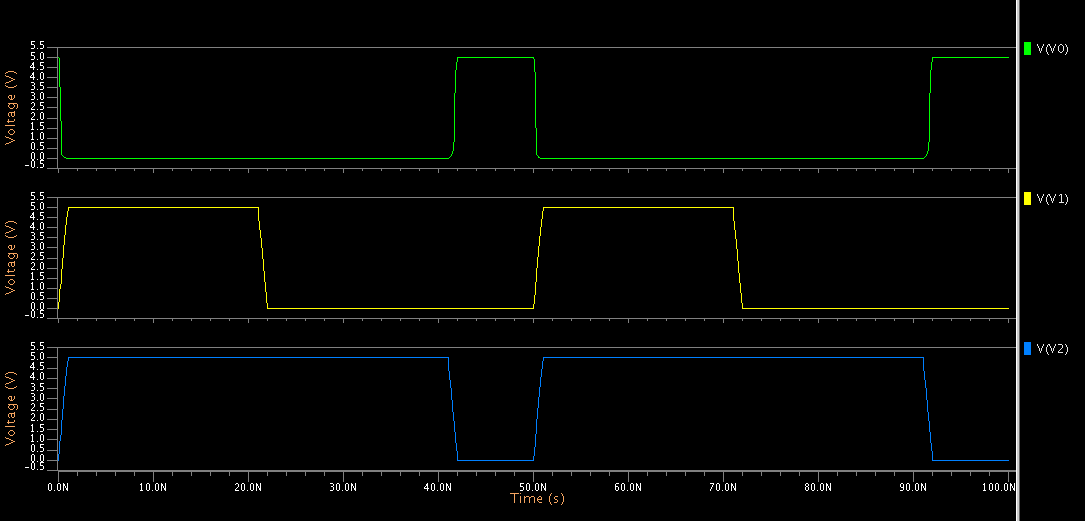
Operation: We know that output an OR gate is 0 only when all inputs of OR gate is 0. But in the case of NOR gate the output is 1 only when all inputs are 0.

 **Fig 3:NORsymbol Fig 4: Truthtable**

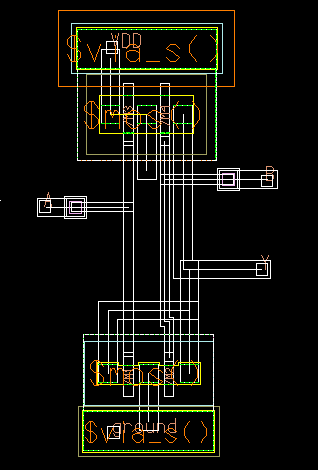
**SCHEMATIC DIAGRAM:**



**SIMULATED WAVEFORMS:**



**LAYOUT:**



**RESULT:**

**EXP:3. Design and Implementation of Full Adder**

**AIM:** To design and simulate the full adder.

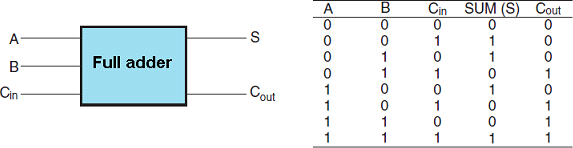
**TOOLS:**

1. PersonalComputer

2. DSCH3 (Schematic Editor)

3. MICROWIND3.5 (Layout Editor)

**THEORY:**

An adder is a digital logic circuit in electronics that implements addition of two one 1 bit numbers. The full adder circuit has three inputs: A,B and C, which add the three input numbers and generate a carry and sum. The first two inputs are A and B and the third input is an input carry as C-IN. We can also design a full adder using two half adders. The full adder is the basic building block of the ripple adders and many other type of adders. the adders are mainly used in digital systems to perform arithmetic operations.

**Fig1: Full adderBlockDiagram Fig 2: TruthTable**

**PROCEDURE:**

1. Connect the Circuit as shown in the circuit diagram using Pyxisschematic.
2. Create a simulation schematic for simulation.
3. Generate symbol for the schematic.

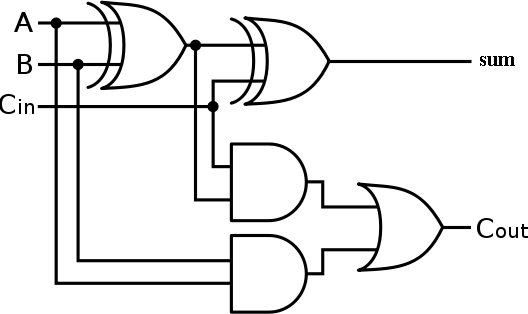
4.Add necessary ports to the symbol to analyze waveforms.

5.Run the Simulation and observe results in EZ wave.

6.Draw the Layout for the circuit using P yxis Layout.

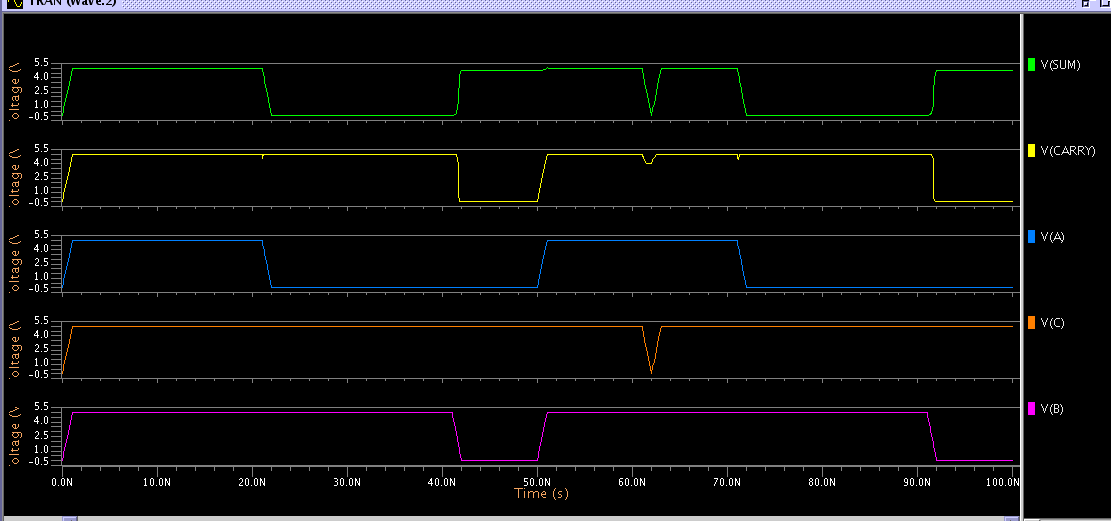
7.Observe the layout.

**SCHEMATIC DIAGRAM**

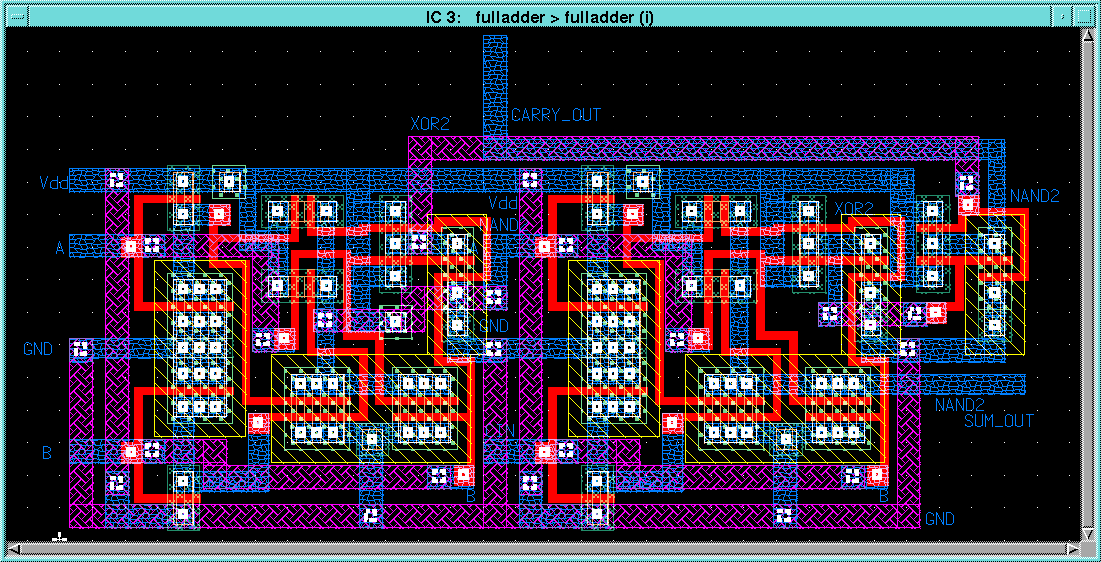


**Fig3: Logic diagram of Full adder using two Half Adders**

**SIMULATION WAVEFORMS:**



**LAYOUT:**



**RESULT:**

**Exp:4. Design and Implementation of Full Subtractor**

**AIM:** To design, simulate and draw the layout of CMOS FULL SUBTRACTOR

**TOOLS:**

1. Personal Computer

2. DSCH3 (Schematic Editor)

3. MICROWIND3.5 (Layout Editor)

.

**THEORY :**

A full subtractor is a multiple output combinational logical network which performs a subtraction between two binary bits considering that a ‘1’ might have been borrowed by a lower significant stage. Along with the minuend ‘a’ and the subtrahend ‘b’, the third input is the borrow bit ‘c’, from the previous stage of subtraction. The combinational logic network of the full subtractor thus has three inputs and two outputs. The two outputs produced are the difference bit output ‘d’ and a final borrow ‘bo’ respectively. The Boolean expression for full subtractor is given by,

Difference = a ^ b^c Borrow = a’b + a’c + bc..

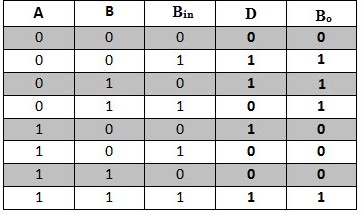
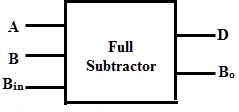
CIRCUITDIAGRAMS:

Fig1: Full subtract or block Diagram Fig 2: TruthTable

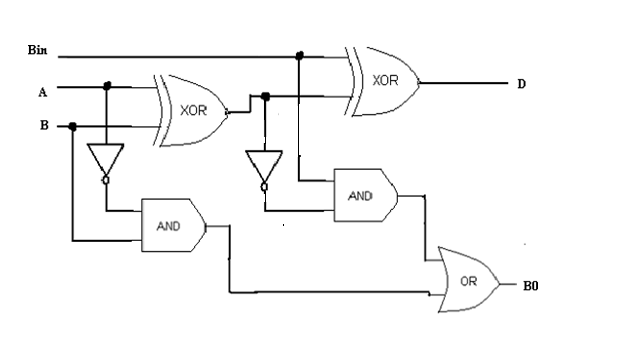


Fig2:Logic diagram of full subtractor using two half subtractors

**PROCEDURE:**

1.Connect the Circuit as shown in the circuit diagram using Pyxisschematic.

2.Create a simulation schematic for simulation.

3.Generate symbol for the schematic.

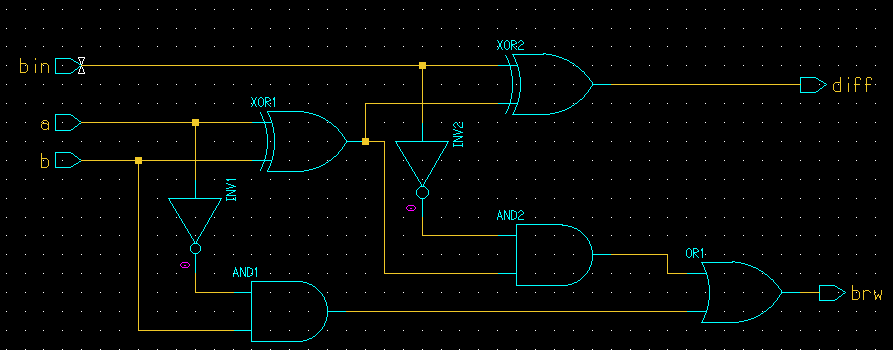
4.Add necessary ports to the symbol to analyze waveforms.

5.Run the Simulation and observe results in EZwave.

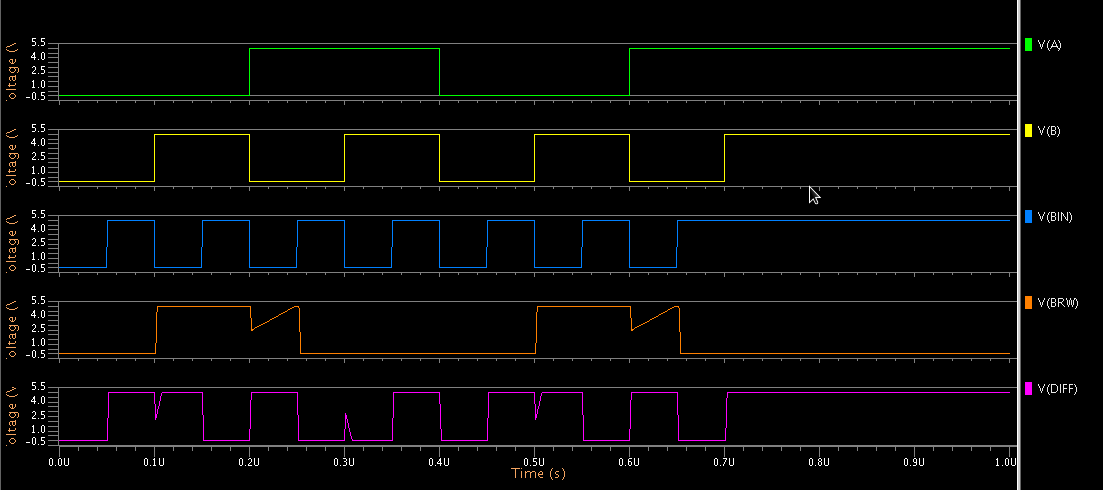
6.Draw the Layout for the circuit using Pyxis Layout.

7.Observe the layout.

**SCHEMATICDIAGRAM:**



**SIMULATED WAVEFORMS:**

**LAYOUT:**

**RESULT:**

**Exp:5. Design and Implementation of Encoder**

**AIM**: To design, simulate and draw the layout of Encoder

**TOOLS:**

1. PersonalComputer

2. DSCH3 (Schematic Editor)

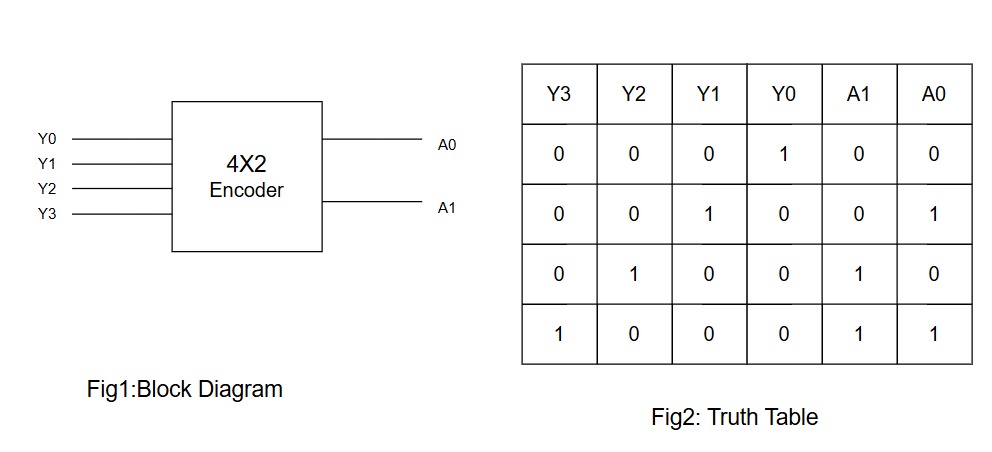
3. MICROWIND3.5 (Layout Editor)

.

**THEORY :**

Now you will design a 4 to 2 encoder, using the same steps you went through to create the 2 to 4 decoder. Include your derived equations and the schematic in your report.

Any input combination other than the four combinations listed in the truth table will create a “don’t care” output. Place an 'X' in the "don't care" locations, populate and derive equations using the following Karnaugh map. Recall that you can group “don’t care” outputs with ones, in order to reduce the amount of hardware required.



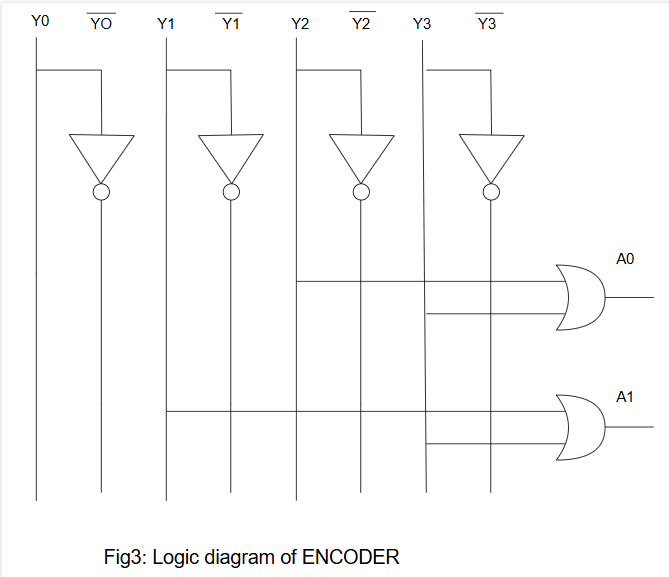
**PROCEDURE:**

1.Connect the Circuit as shown in the circuit diagram using Pyxisschematic.

2.Create a simulation schematic for simulation.

3.Generate symbol for the schematic.

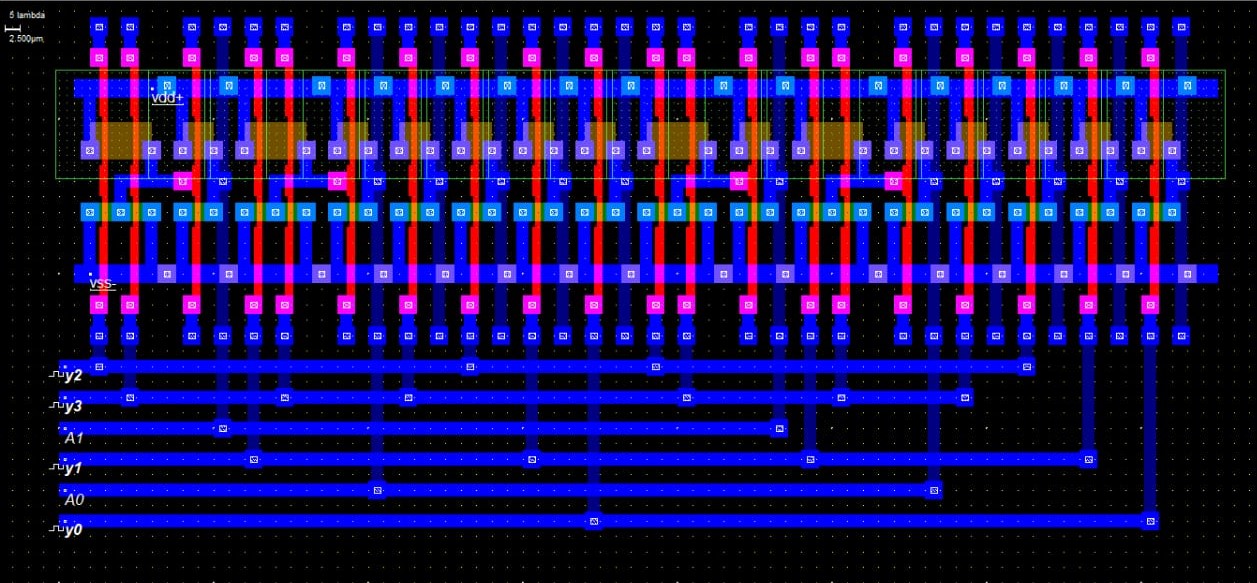
1. Add necessary ports to the symbol to analyze waveforms.
2. Run the Simulation and observe results in EZwave.
3. Draw the Layout for the circuit using Pyxis Layout.
4. Observe the layout.



**SIMULATED WAVEFORMS:**



**Lay out:**



**RESULT:**

**Exp:6. Design and Implementation of Decoder**

**AIM**: To design, simulate and draw the layout of Decoder

**TOOLS:**

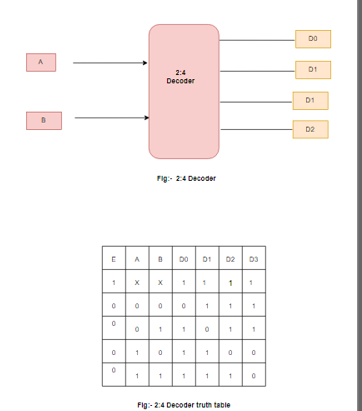
1. PersonalComputer

2. DSCH3 (Schematic Editor)

3. MICROWIND3.5 (Layout Editor)

**THEORY :**

A [decoder](https://www.geeksforgeeks.org/encoders-and-decoders-in-digital-logic/) is a combinational logic circuit that has ‘n’ input signal lines and 2n output lines. In the 2:4 decoder, we have 2 input lines and 4 output lines. In addition, we provide ‘*enable*‘ to the input to ensure the decoder is functioning whenever enable is 1 and it is turned off when enable is 0. The truth table, logic diagram, and logic symbol are given below:



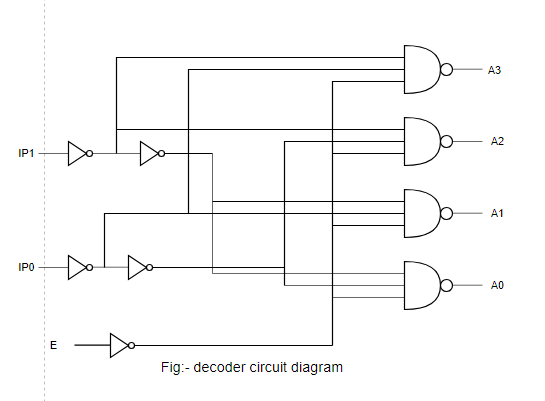
**PROCEDURE:**

1.Connect the Circuit as shown in the circuit diagram using Pyxisschematic.

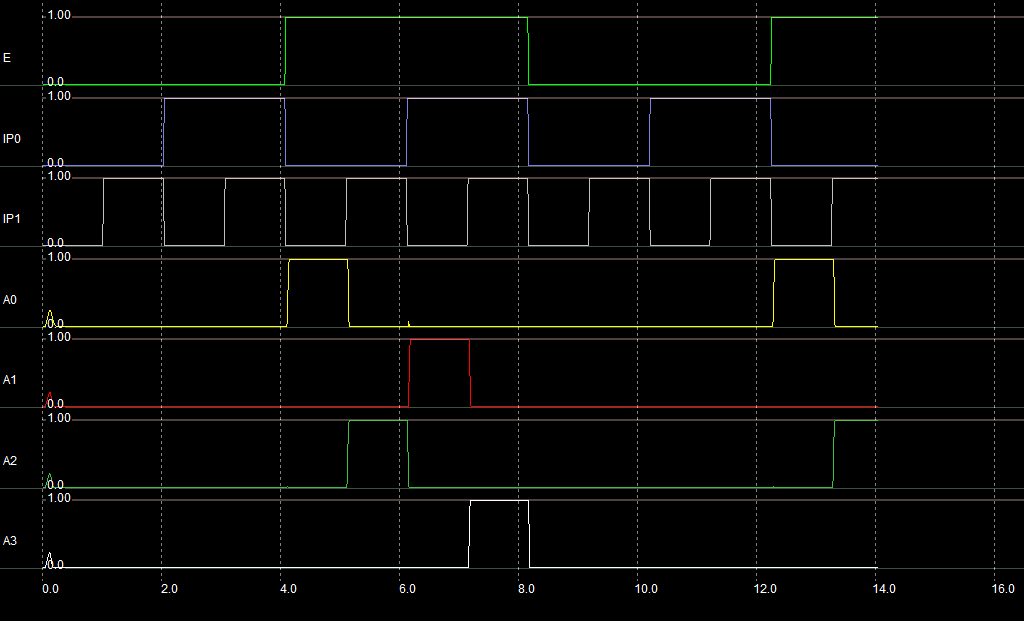
2.Create a simulation schematic for simulation.

3.Generate symbol for the schematic.

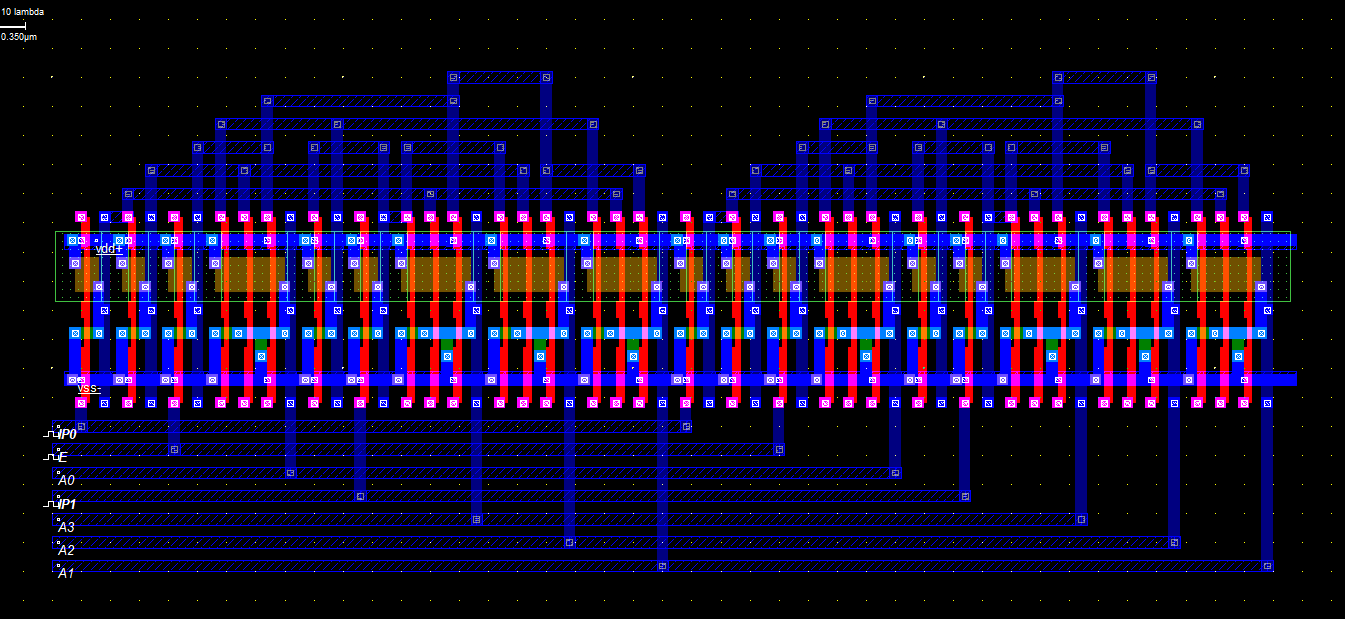
1. Add necessary ports to the symbol to analyze waveforms.
2. Run the Simulation and observe results in EZwave.
3. Draw the Layout for the circuit using Pyxis Layout.
4. Observe the layout.



**SIMULATED WAVEFORMS:**



**Lay out:-**

**RESULT:**

**Exp:7. Design and Implementationof SR**

**Latch**

**AIM:** To design, simulate and draw the layout CMOS **TOOLS:**

1.PersonalComputer

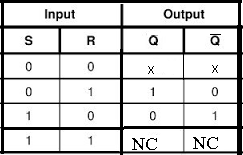
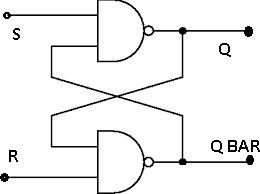
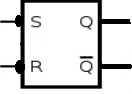
2.Software - Mentor Graphics-Pyxis, Eldo, CalibreToolsVersion-v10.2

3.Cent OS - 6Version

**THEORY:**

S R latch has two inputs S and R and two outputs Q and . The state of this latch is determined by condition of Q. If Q is 1 the latch is said to be SET and if Q is 0 the latch is said to be RESET. This S R Latch can be designed either by two cross-coupled [NAND gates](http://www.electrical4u.com/nand-gate/) or two-cross coupled [NOR gates](http://www.electrical4u.com/nor-gate/). When we design this latch by using [NOR gates,](http://www.electrical4u.com/nor-gate/) it will be an active high S-R latch. That means it is SET when S = 1. When we design this latch by using [NAND gates,](http://www.electrical4u.com/nand-gate/) it will be an active low S-R latch. That means it is SET when S = 0. S R Flip Flop is also called SET RESET Flip Flop.

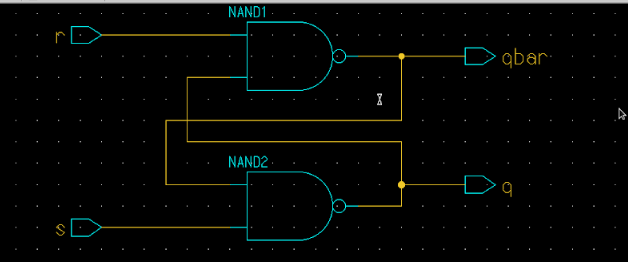
**CIRCUIT DIAGRAMS:**

**Fig1: symbol Fig 2: Logic diagram Fig 3: Truth table**

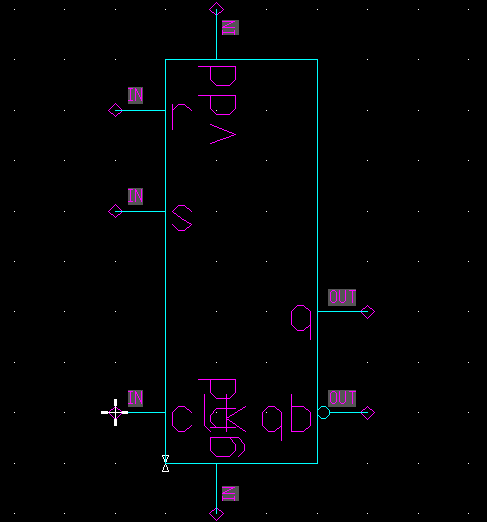
**PROCEDURE:**

1. Connect the Circuit as shown in the circuit diagram using Pyxis schematic.
2. Create a simulation schematic for simulation.
3. Generate symbol for the schematic.
4. Add necessary ports to the symbol to analyze waveforms.
5. Run the Simulation and observe results in EZwave.
6. Draw the Layout for the circuit using Pyxis Layout.
7. Observe the layout.

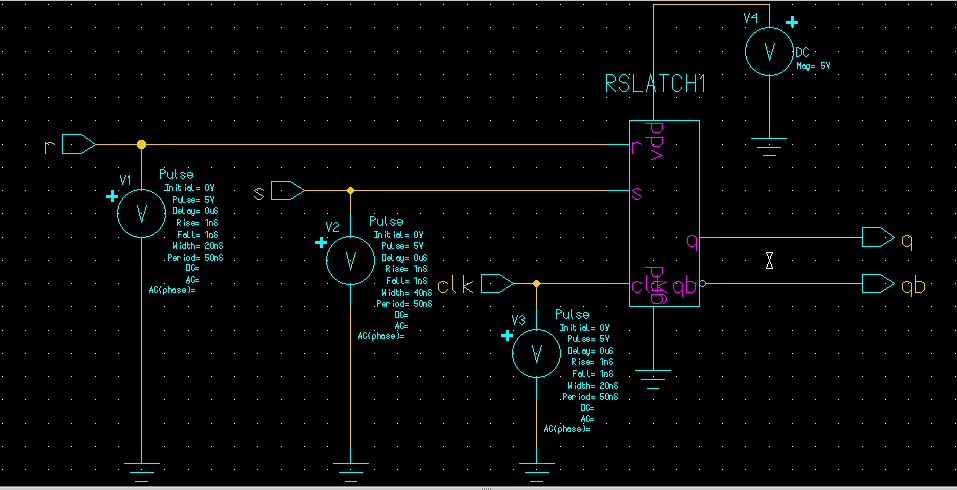
**SCHEMATIC DIAGRAM:**



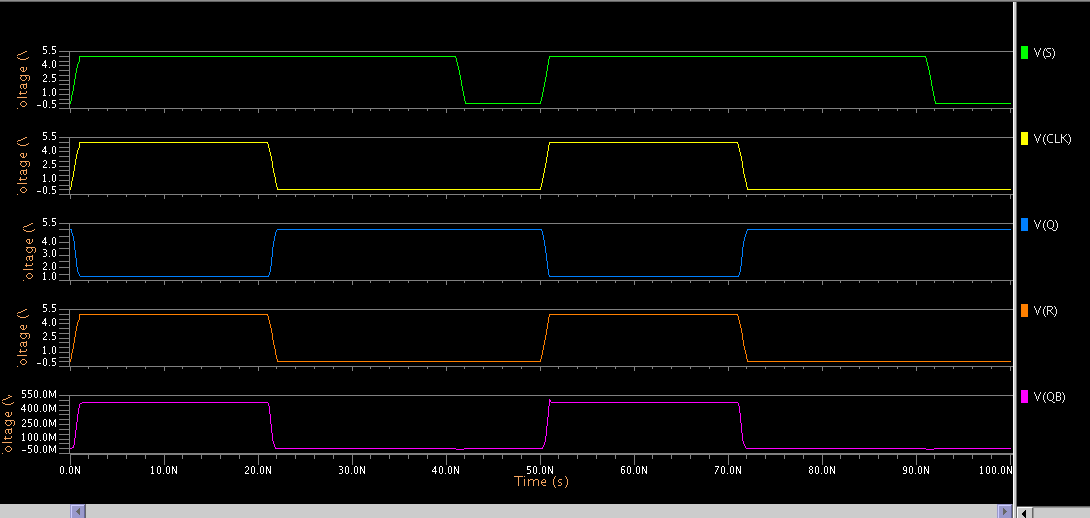
**SYMBOL:**



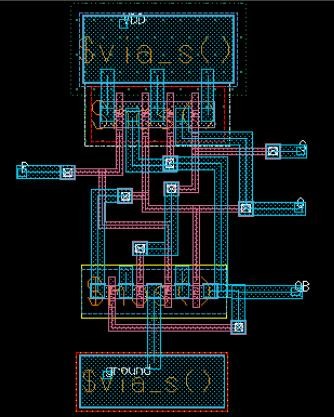
**TESTBENCH SCHEMATIC:**



**SIMULATION WAVEFORMS:**



**LAYOUT:**



**RESULT:**

**Exp:8. Design and Implementation of D Latch**

**AIM**: To design, Simulate and draw the layout of D latch.

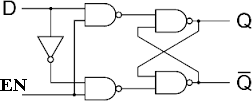
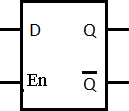
**TOOLS:**

* 1. PersonalComputer
  2. Software - Mentor Graphics-Pyxis, Eldo , CalibreTools
  3. Version-v10.2
  4. Cent OS - 6Version

**THEORY:**

D-LATCH is a recognizable SR latch with enable, with an inverter added to generate S and R inputs from the single D(data )input. It has only a single data input. That data input is connected to the S input of RS-latch ,while the inverse of D is connected to the R input. This prevents that the troublesome situation in SR latch.. To allow the flip flop to be in holding state, a D-flip flop has a second input called “enable”. D-latch is used in applications where the each bit is presented on single line and like to store on somewhere.

**CIRCUIT DIAGRAMS:**



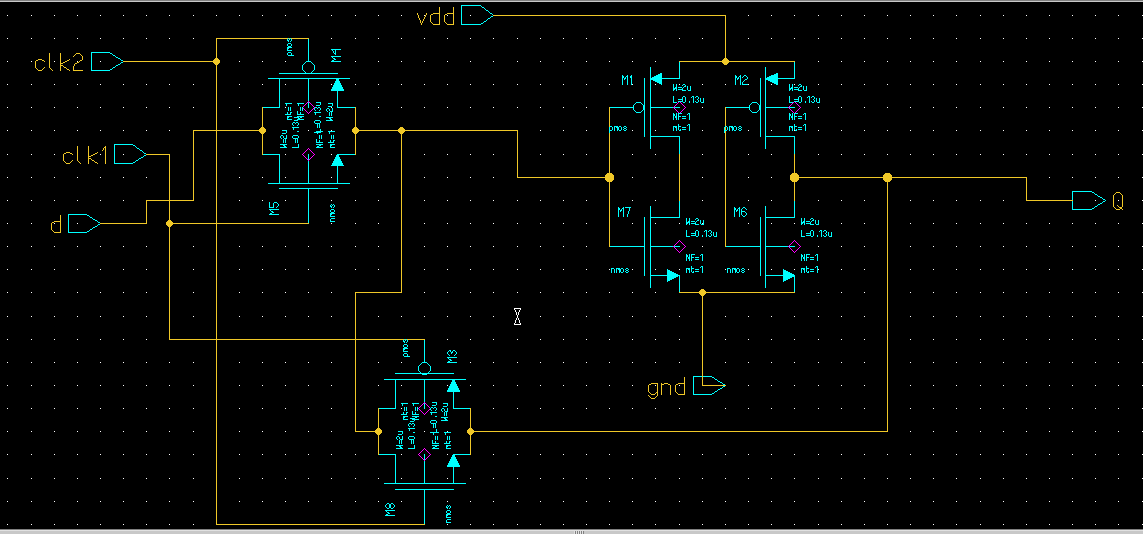
|  |  |  |  |
| --- | --- | --- | --- |
| **EN** | **D** | **Q** | **Q’** |
| **1** | **0** | **0** | **1** |
| **1** | **1** | **1** | **0** |
| **0** | **X** | **lastQ** | **Last Q’** |

**Fig1:D-latch Symbol Fig 2:Logic Diagram Fig 3: Truth Table**

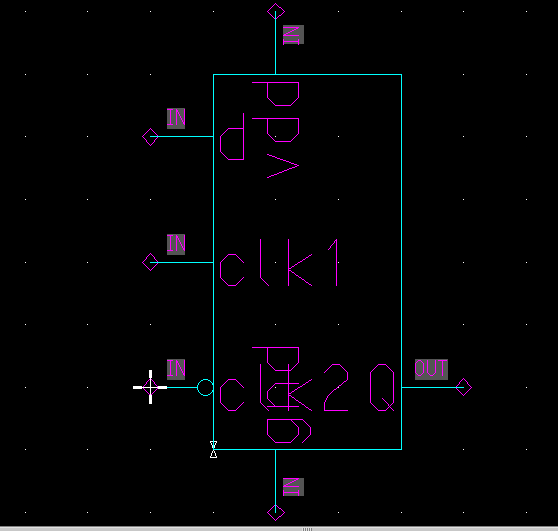
**PROCEDURE:**

1. Connect the Circuit as shown in the circuit diagram using Pyxis schematic.
2. Create a simulation schematic for imulation.
3. Generate symbol for theschematic.
4. Add necessary ports to the symbol to analyze waveforms.
5. Run the Simulation and observe results in EZwave.
6. Draw the Layout for the circuit using Pyxis Layout.
7. Observe the layout.

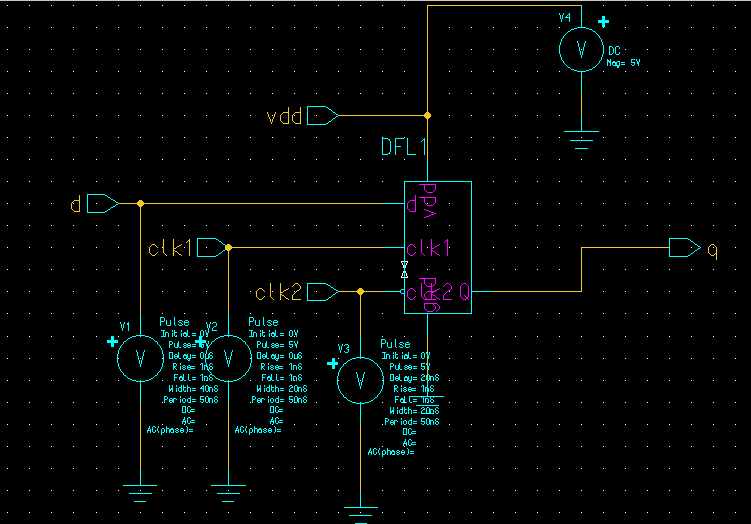
**SCHEMATIC DIAGRAM:**



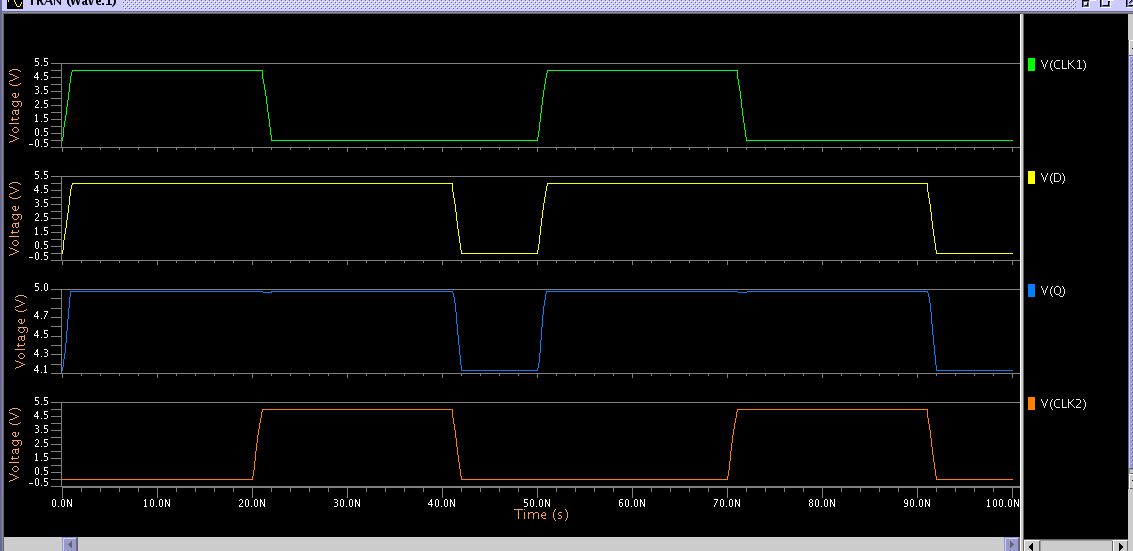
**SYMBOL:**



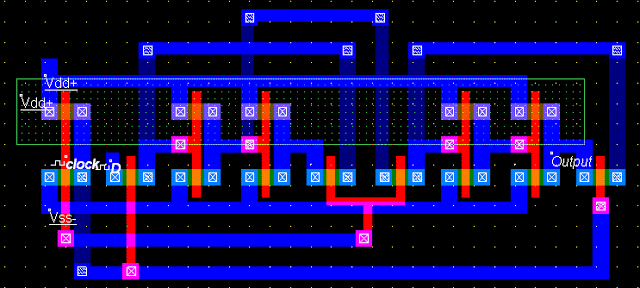
**TESTBENCH SCHEMATIC:**



**WAVE FORMS :**



**LAYOUT :**



**RESULT:**

**Exp:9.Design and Implementation of Asynchronous Counter**

**AIM:** To design, simulate and draw the layout of Asynchronous Counter

**TOOLS:**

* 1. PersonalComputer
  2. Software - Mentor Graphics-Pyxis, Eldo , CalibreTools
  3. Version-v10.2
  4. OS – CentOS6Linux.

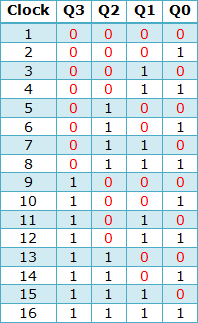
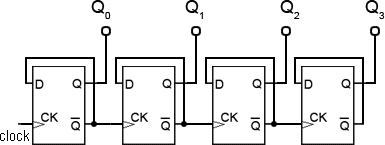
**THEORY:**

A ripple counter is an asynchronous counter where only the first flip-flop is clocked by an external clock. All subsequent flip-flops are clocked by the output of the preceding flip-flop. Asynchronous counters are also called ripple-counters because of the way the clock pulse ripples it way through the flip-flops. There are many ways to implement the ripple counter depending on the characteristics of the flip flops used and the requirements of the count sequence.

* Clock Trigger: Positive edged or Negativeedged,
* JK or D-flip-flops
* Count Direction: Up, Down, orUp/Down

Asynchronous counters are slower than synchronous counters because of the delay in the transmission of the pulses from flip-flop to flip-flop. It can be implemented using [D-typeflip-flops](http://electronics-course.com/d-flip-flop)or JK-type flip-flops.For each D register we must connect the notQ output to the D input, and the clock signal of each stage (except for thefirst) is simply carried out by the previous Q output. The first stage receives the clock signal. For the reset, we use the reset of our D registers and we connect them together

**CIRCUIT DIAGRAMS:**

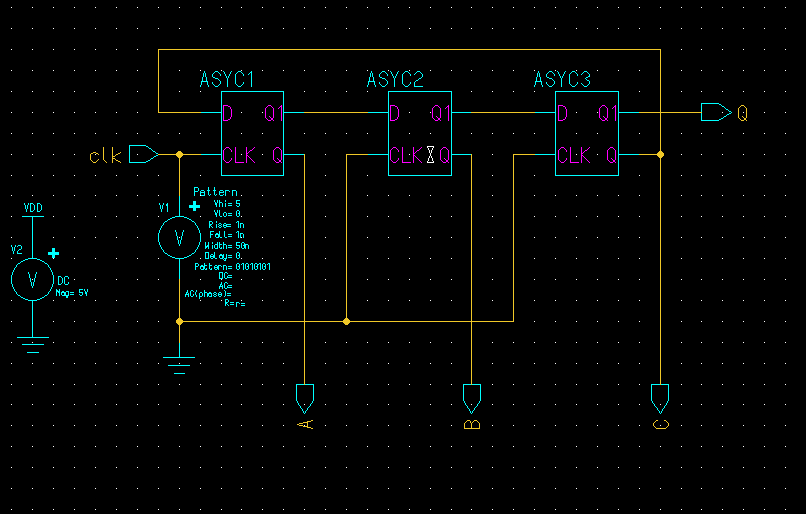


**Fig 1 : 4-bitAsynchronouscounter Fig 2 : Truthtable**

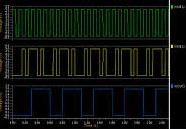
**PROCEDURE:**

1. Connect the Circuit as shown in the circuit diagram using Pyxis schematic.
2. Create a simulation schematic for simulation.
3. Generate symbol for theschematic.
4. Add necessary ports to the symbol to analyze waveforms.
5. Run the Simulation and observe results in EZwave.
6. Draw the Layout for the circuit using Pyxis Layout.
7. Observe the layout.

**ASYNCHROUNOUS COUNTER TEST BENCH**



**SIMULATION WAVEFORMS:**

LAYOUT:

RESULT: