Başar Demir

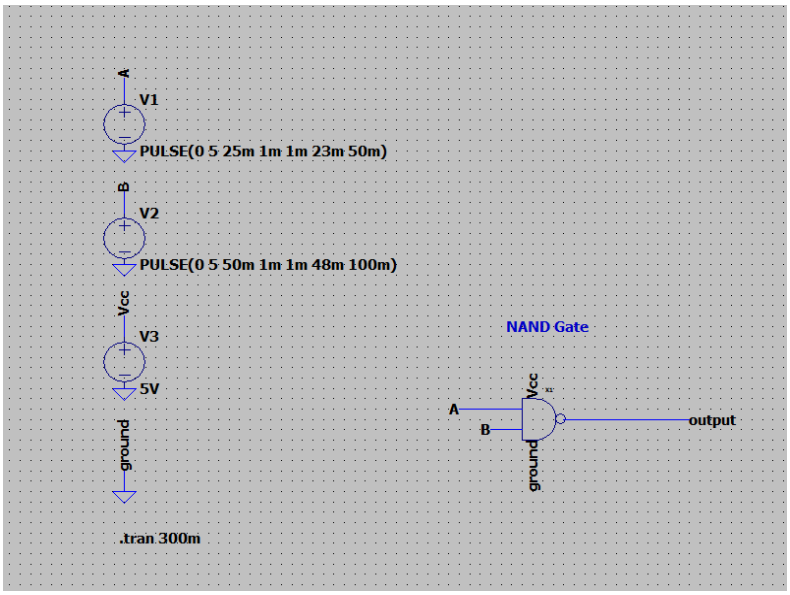
150180080

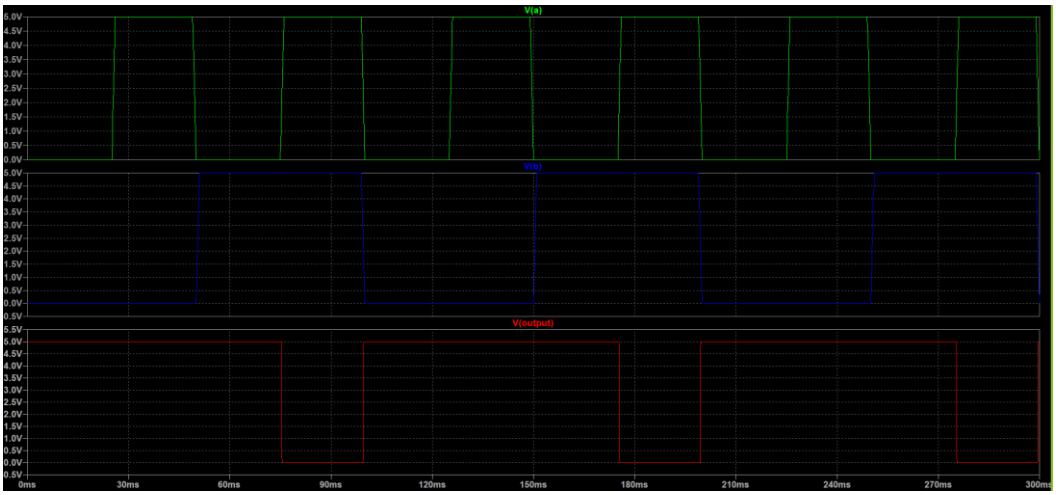
EHB-311 Experiment-6

1) NAND Gate

I directly used given NAND gate and output was came as expected.

A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0





2) NOT-AND-OR

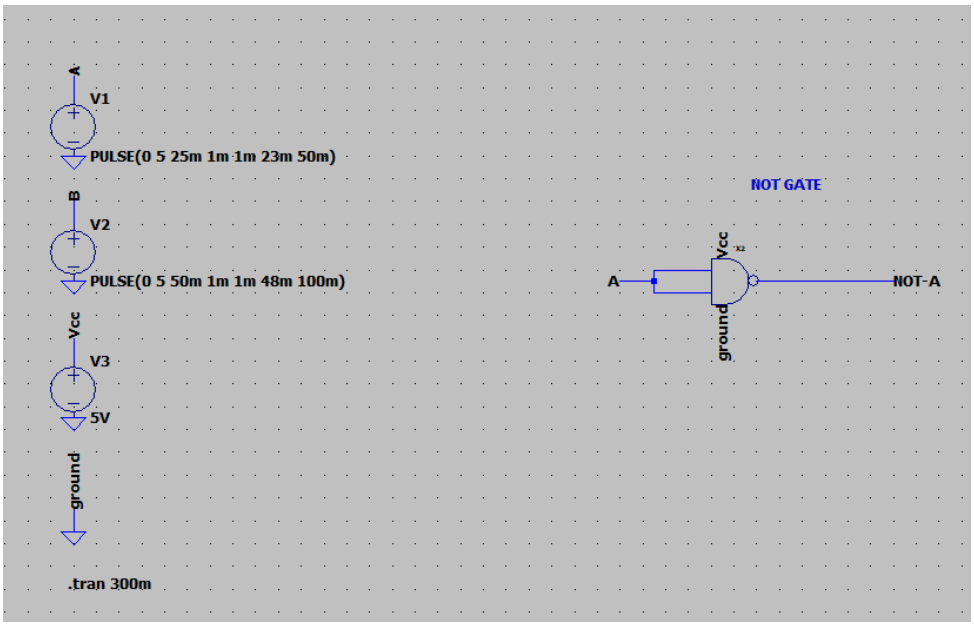
I separately implement all gates and I will discuss them in different sub-sections.

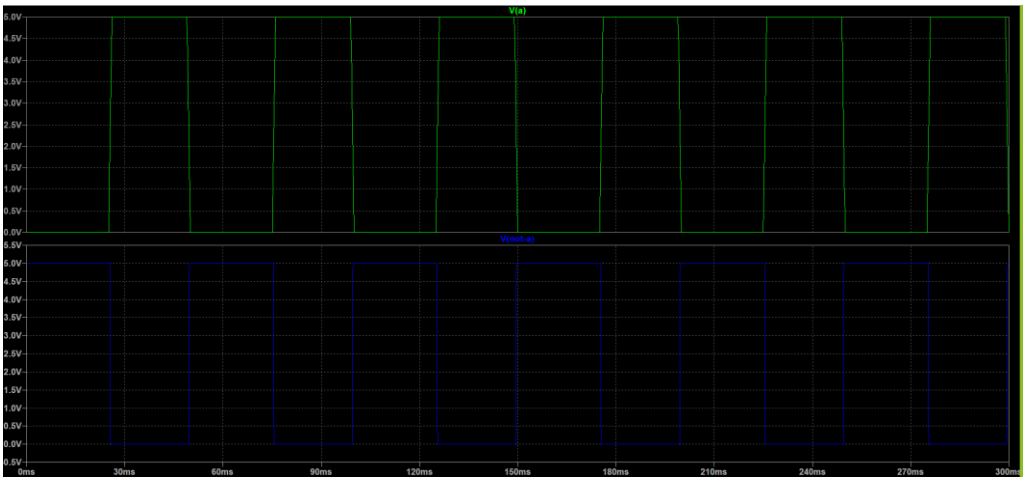
a. NOT Gate

If we give same value to both inputs of the NAND gate, it gives inverted value as input.

A NAND A = (A*A)' = (A)'

It gives inverted value of the input as seen in plot.



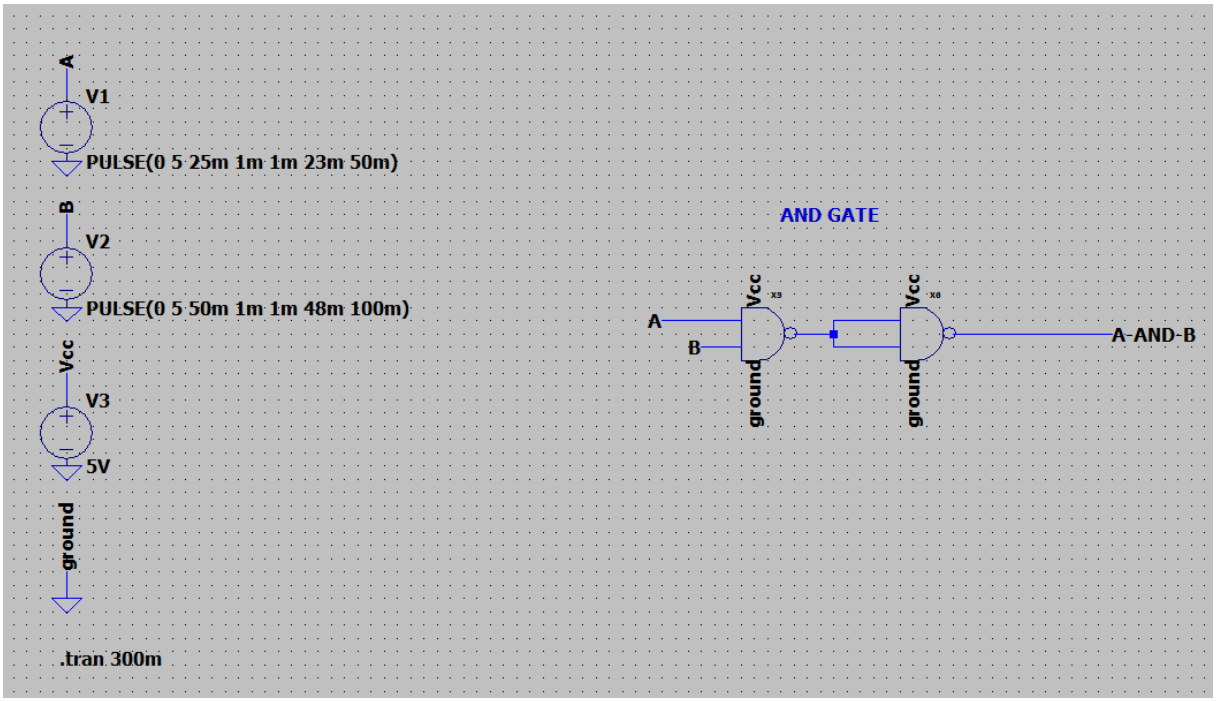


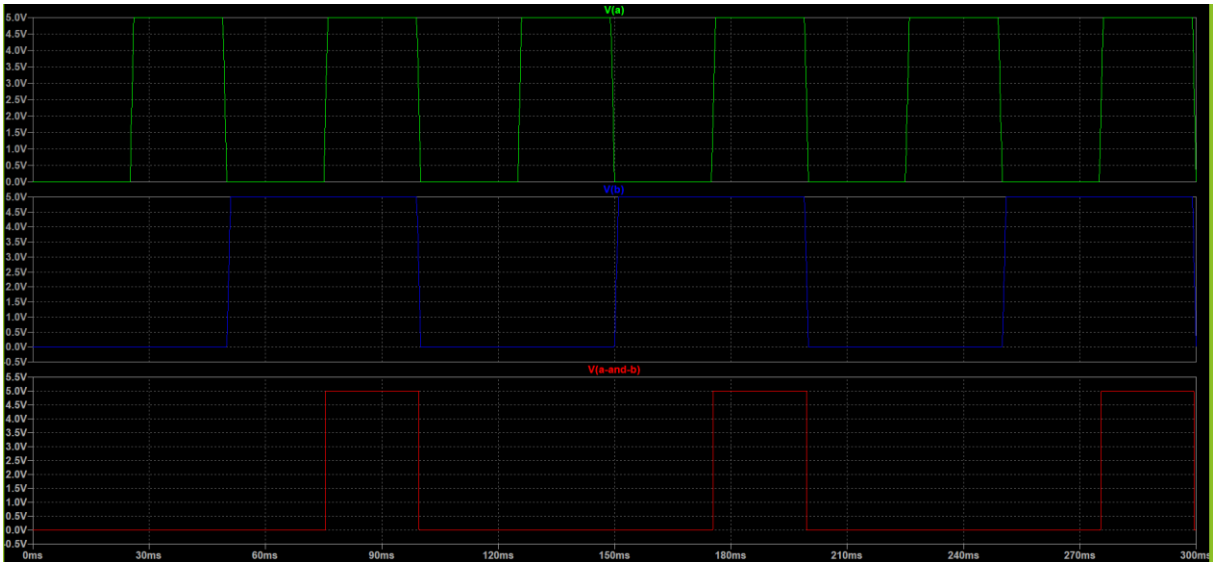
b. AND Gate

To implement NAND Gate from AND Gate, we can basically get not value of the NAND Gate.

A NAND B = NOT(A AND B)

I used NOT gate that was implemented for previous sub-section. It gives AND value corresponding to input values as seen in plot.



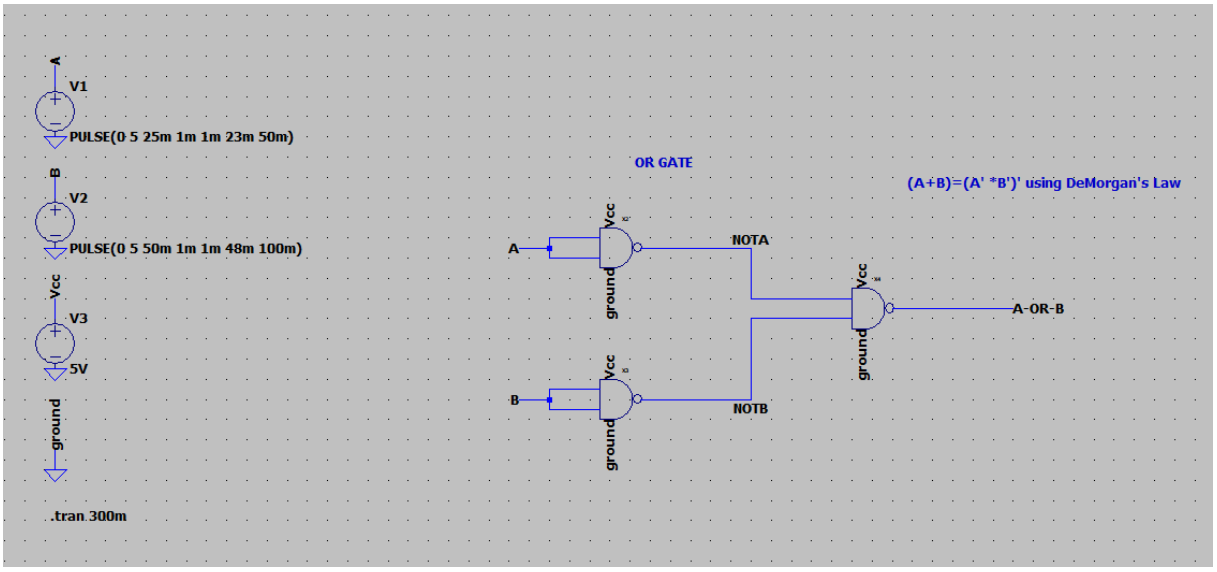


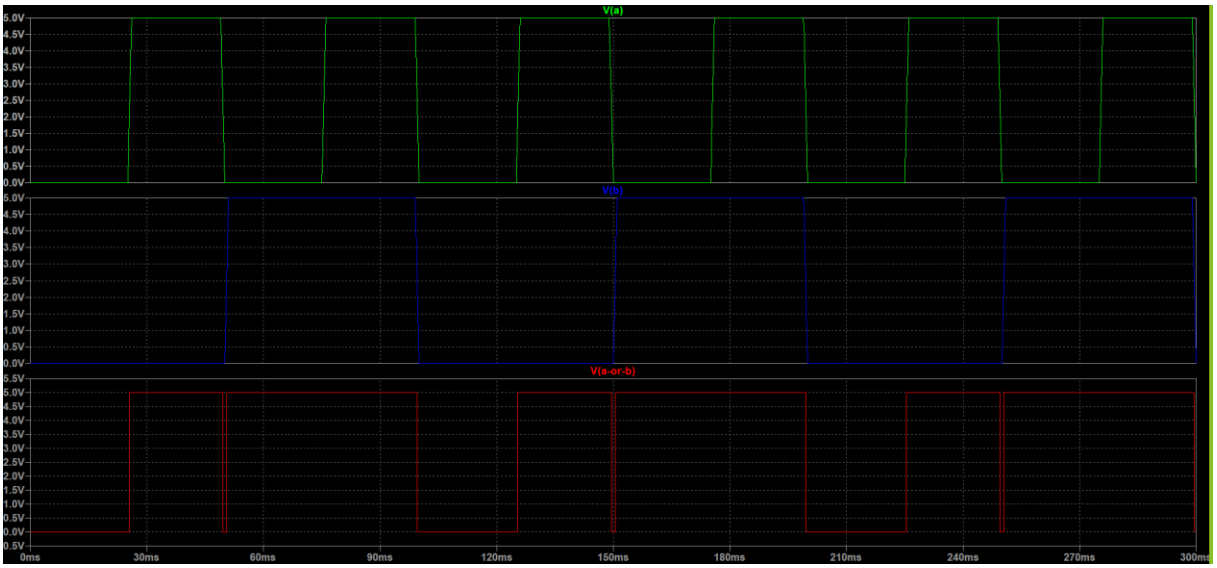
c. OR Gate

To implement OR Gate from AND Gate, I used De Morgan's Law. Firstly, I performed two inverse operations and then, I applied law for inside of parenthesis.

A OR B = ((A+B)')'= **(A'*B')' = A' NAND B'**

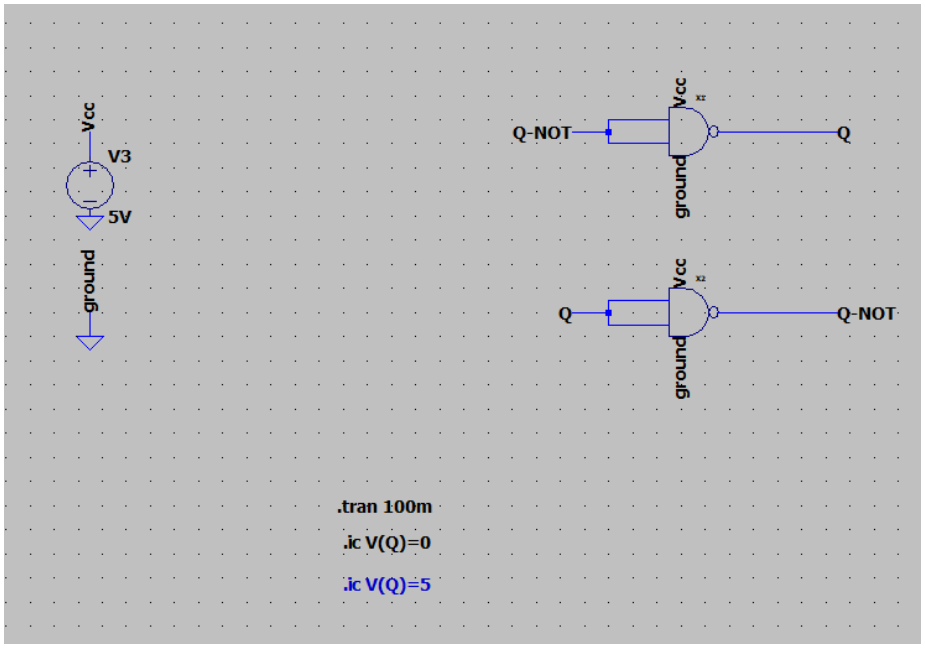
By this way, I have implemented OR gate using NAND gates.





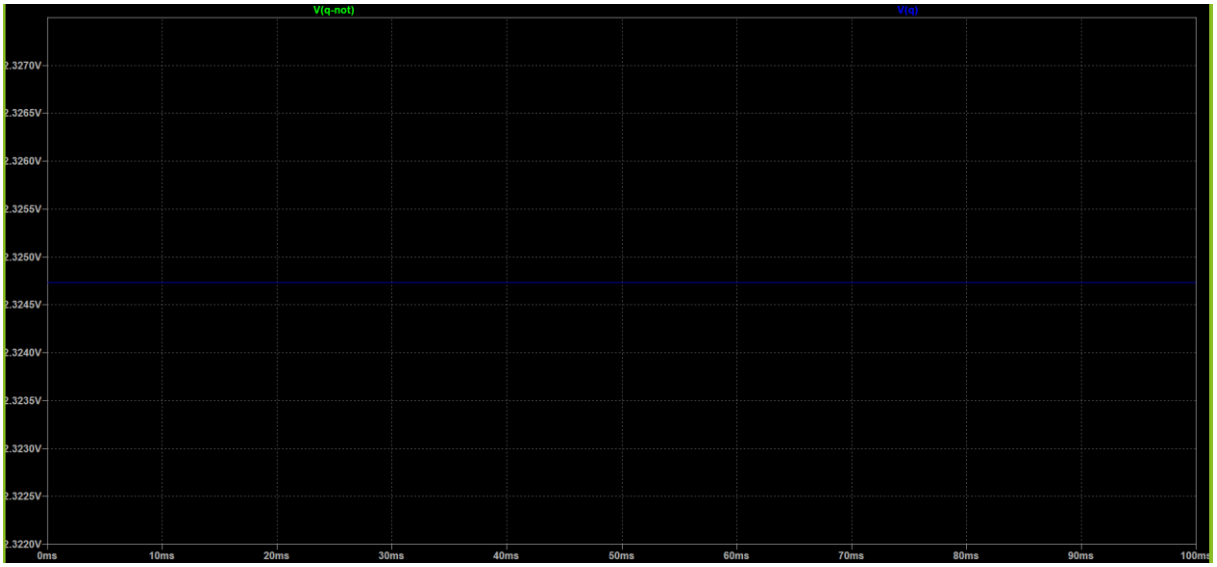
3) Cross Coupled Inverter

It is a primitive memory unit that consist of two inverters. By this way, we can preserve our value between cross coupled inverters.



a) Q(initial)=unknown

If we do not give initial value to the circuit, it gives output around **2.32V** and this situation makes circuit **useless**. Its states cannot be determined.



b) Q(initial)=0V

For this case, we can preserve our 0V value using inverters. It always gives 0 in Q and 1 in Q' output. Because inverters always face with same values and their output do not change.

c) Q(initial)=5V

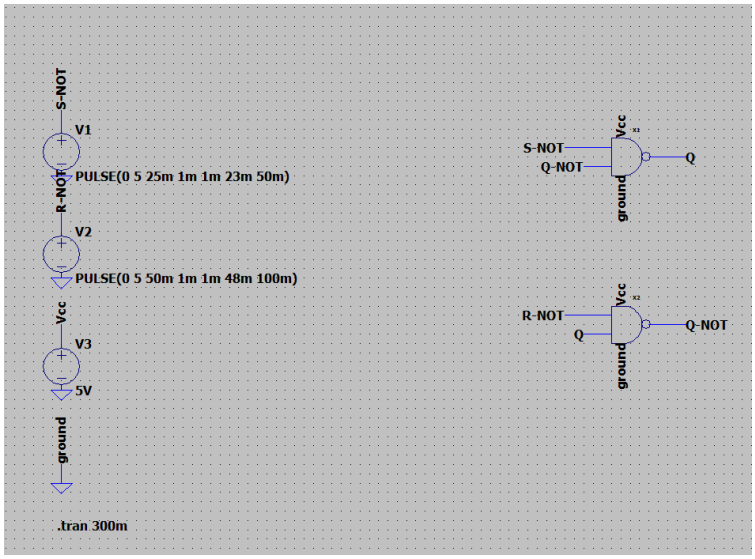
For this case, we can preserve our 5V value using inverters. It has same logic with 0V case. We can always observe 5V in Q output.

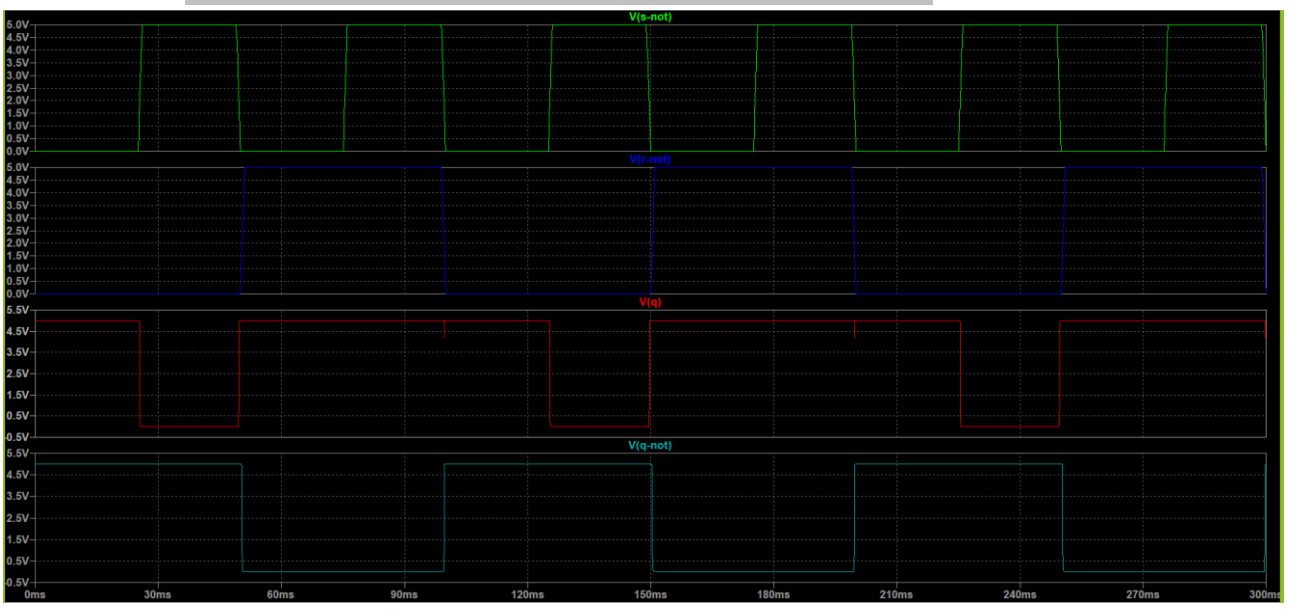
4) SR Latch

SR Latch is a memory unit that has reset and set properties. The corresponding input output combinations is given below. For this circuit we used NOT values of S and R. Therefore, I used this values in my table.

S’	R’	Q(t)	Q(t+1)
0	0	Q(t)	Forbidden (Q and Q’ is 1)
0	1	Q(t)	1
1	0	Q(t)	0
1	1	Q(t)	Q(t)

We can observe same behaviour in this plot. It works as expected.





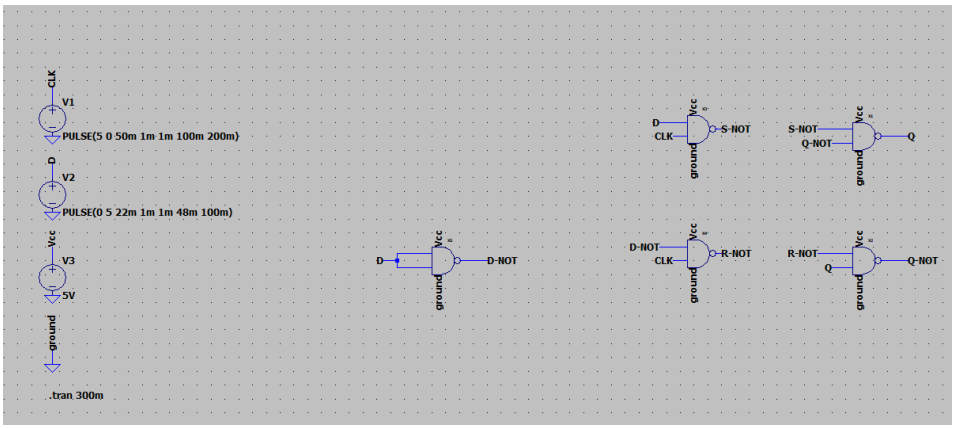
5) D Latch

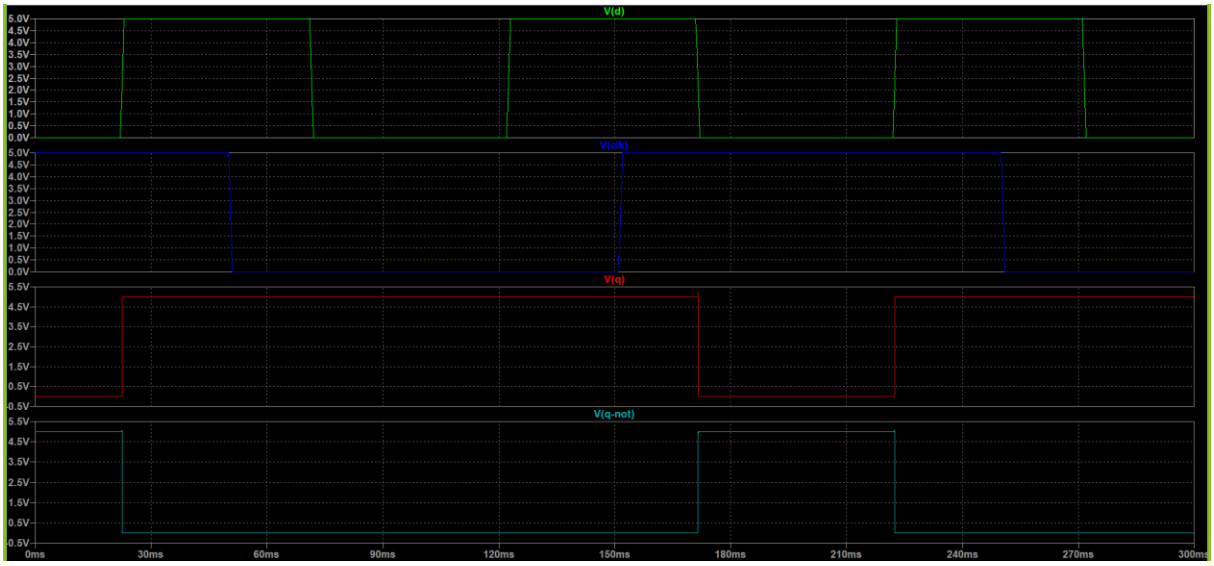
D latch is a memory unit that is constructed on SR-Latch. In SR Latch we have forbidden state that is useless. In D Latch, we directly get a 1 input called D and we give proper S and R values to SR-Latch. For example, if 1 is given as a D value than S will be 1 and R will be 0. By this way, we get rid of the undesirable situation (S=1, R=1).

With clock input, we can provide a preserve condition that gives same output at Q(t+1) and Q(t). It is similar to S=0, R=0 condition. If clock is 1, output is directly equal to D input. Otherwise, it preserves data.

D latch working mechanism is given in table below. It preserves value when clock is 0, otherwise it directly gives D value as an output.

D	CLK	Q(t)	Q(t+1)
insignificant	**0**	**Q(t)**	**Q(t)**
0	**1**	**Q(t)**	**0**
1	**1**	**Q(t)**	**1**





6) D Flip-Flop

D Flip-Flop is a memory unit that is dependent to rising edge of the clock. It consists of two latches called master and slave. Inverted value of the clock is given to master and slave is directly dependent to clock value.

When CLK=0, master loads D input to itself as I stated in previous part and slave preservers its value. When CLK=1, slave reads value of the master's output and it takes it as a D input. Master preserves its value.

This mechanism provides a circuit that changes output value when rising edge occurs. We can observe this situation in the given plot, when CLK changes from 0 to 1, it gives input value as an output of the circuit. Otherwise, it preserves its value.

D	CLK	Q(t)	Q(t+1)
insignificant	**0**	**Q(t)**	**Q(t)**
insignificant	**1**	**Q(t)**	**Q(t)**
insignificant	**1->0**	**Q(t)**	**Q(t)**
D	**0->1**	**Q(t)**	**D**