EE 477 Laboratory #3

Student Name: Vaibhav Murali

Student ID: 9952684924

Student Email ID: vmurali@usc.edu

Date: 12/03/2018

Area of Neuron 1: 4825.65 um^2

Delay of Neuron 1: 4 ns

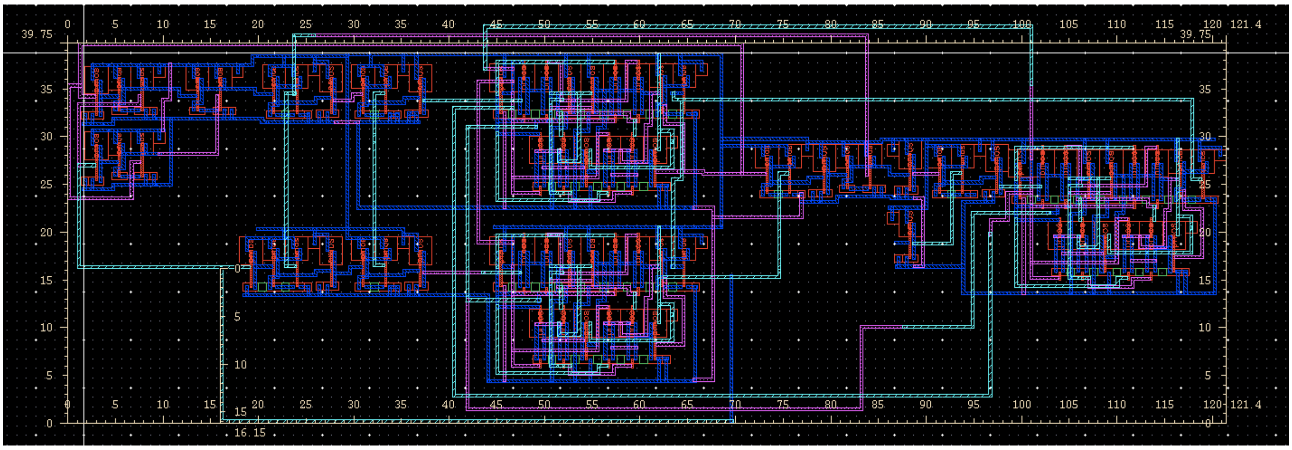
Area - Delay product of Neuron 1: 19302.6 um^2 ns

Area of Neuron 2: 5046.2475 um^2

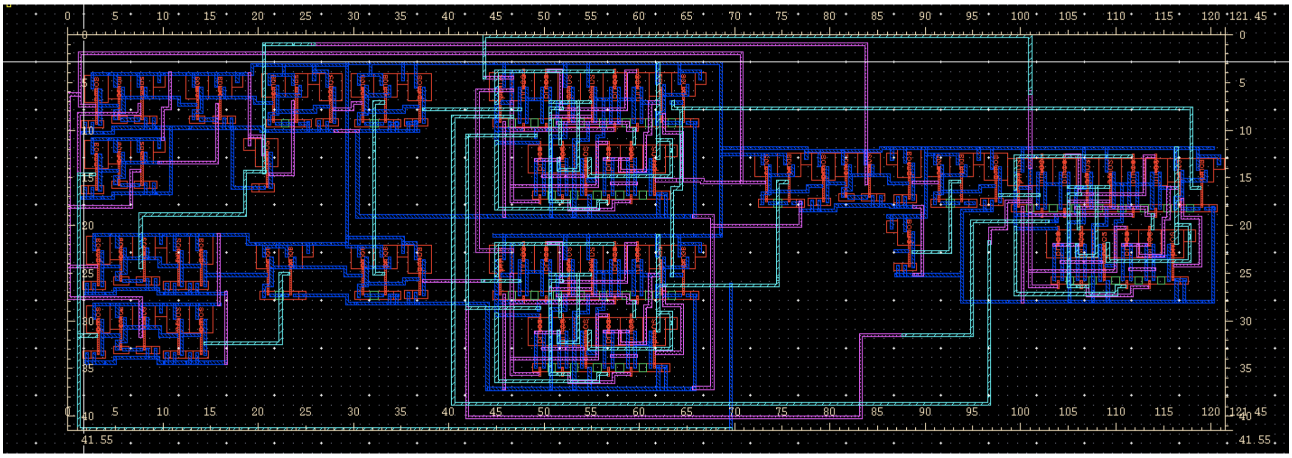
Delay of Neuron 2: 4 ns

Area - Delay product of Neuron 2: 20184.99 um^2 ns

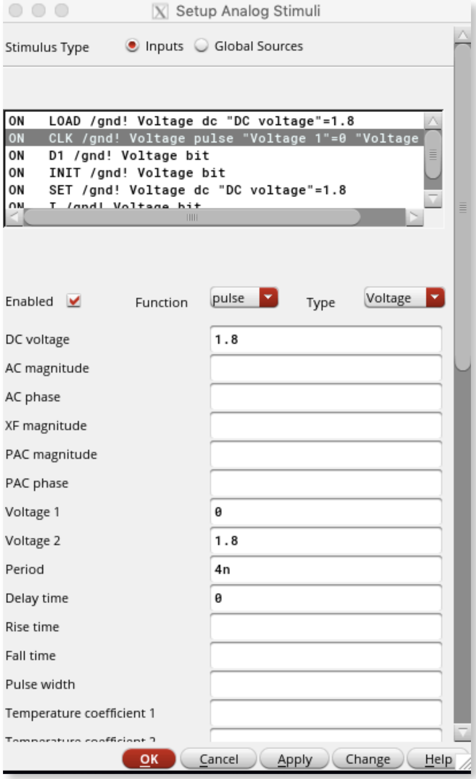
Neuron 1:



Neuron 2:



Clock with similar specifications were used for both the neurons as follows:



Setup Analog Stimuli

Stimulus Type ● Inputs ○ Global Sources

```
ON   LOAD /gnd! Voltage dc "DC voltage"=1.8
ON   CLK /gnd! Voltage pulse "Voltage 1"=0 "Voltage
ON   D1 /gnd! Voltage bit
ON   INIT /gnd! Voltage bit
ON   SET /gnd! Voltage dc "DC voltage"=1.8
ON   I /gnd! Voltage bit
```

Enabled ☑ Function pulse Type Voltage

DC voltage	1.8
AC magnitude	
AC phase	
XF magnitude	
PAC magnitude	
PAC phase	
Voltage 1	0
Voltage 2	1.8
Period	4n
Delay time	0
Rise time	
Fall time	
Pulse width	
Temperature coefficient 1	
Temperature coefficient 2	

Table of Contents

Previous Labs:

Lab 1:

In this lab, we built simple schematics and layout for Inverter ,NAND gate and transmission gate which we used in the subsequent labs.

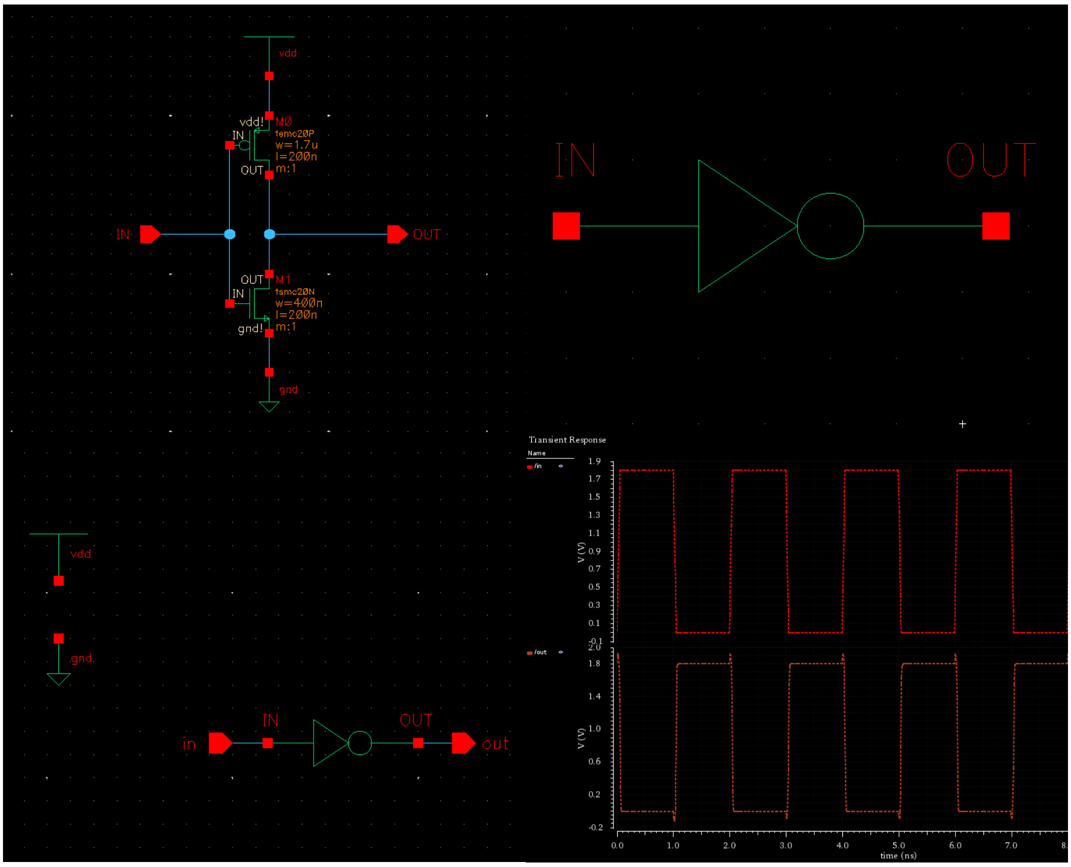
Inverter

Design:

NMOS transistor in the inverter is unit sized, and the PMOS transistors sized so that the C point where both transistors are in saturation occurs when Vin is about 0.8v Since the NMOS transistor is unit sized the width of the diffusion is 4λ. Here λ equals 100nm therefore the NMOS is sized to 400nm. The C point is about 0.8V, here K_p equals K_n.

51.9 W_p/L_p = 219.4 W_n/L_n
=> Wp= 4.3 Wn (since L_p = L_n equal to 200nm)
When W_n equals 400nm, W_p almost equals 1720nm for equal K_p and K_n.

Schematic and Symbol:



With the above design, schematic for the inverter was made and then converted into a symbol which was tested with a simple logic. This symbol was used in the subsequent labs.

NAND Gate

A 2 input NAND gate was constructed using PMOS and NMOS transistors with the following design.

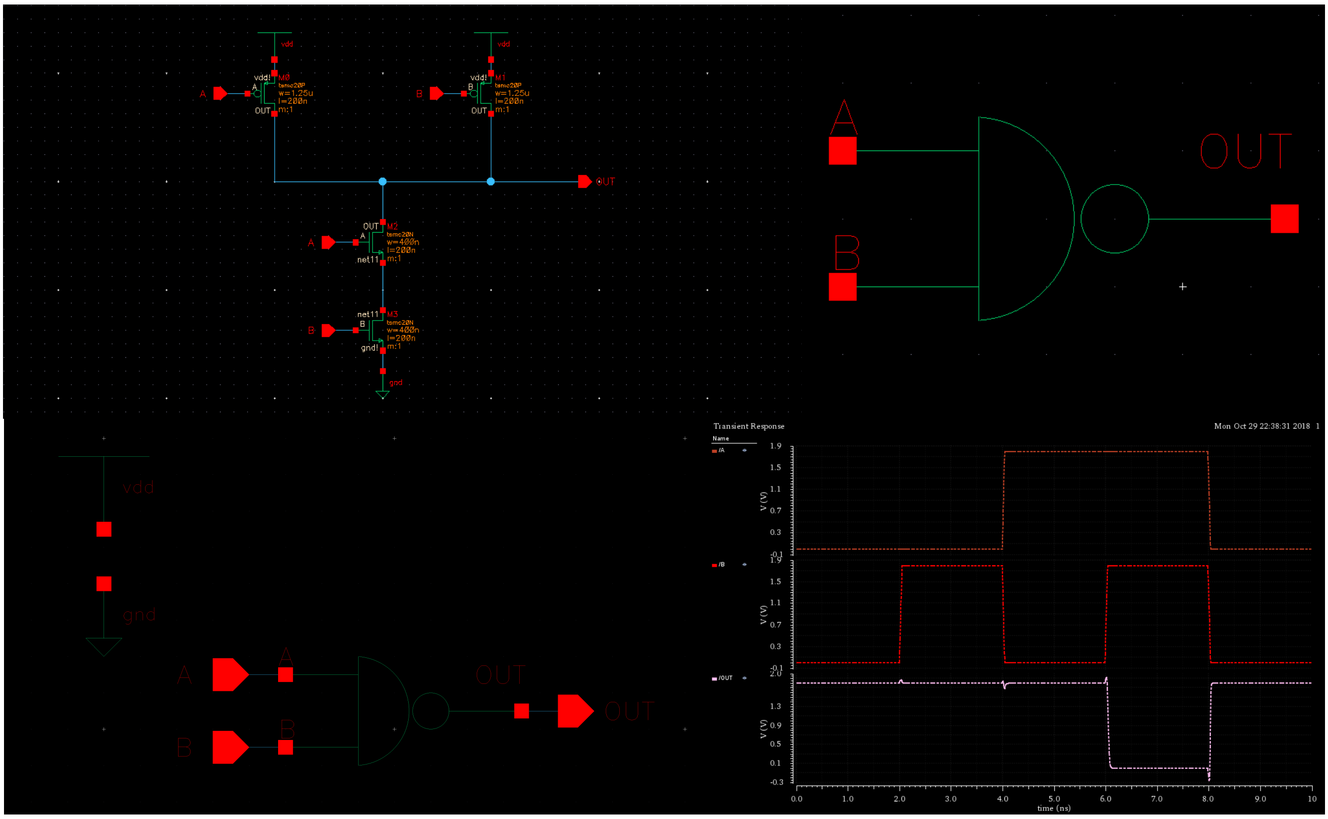
Design:

The 2 input NAND gate uses unit size NMOS transistors and the PMOS transistors are made wide enough to have rise and fall times within 3x each other in the worst case.

Fall Time < 3 x Rise Time
$2 \times R_n < 3 \times R_p$
$2/K_n < 3/K_p$
$2 \times K_p < 3 \times K_n$
$(2 \times 51.9) \times Wp < (3 \times 219.4) \times Wn$
$W_p < 6.34\ W_n$

Since NMOS is unit sized, Wn= 400nm. This implies Wp must be under 2.5um, therefore each PMOS is designed to have a width of 1.25um to satisfy the above condition.

Schematic and Symbol:



With the above design, schematic for the NAND gate was made and then converted into a symbol which was tested with a simple logic. This symbol was used in the subsequent labs.

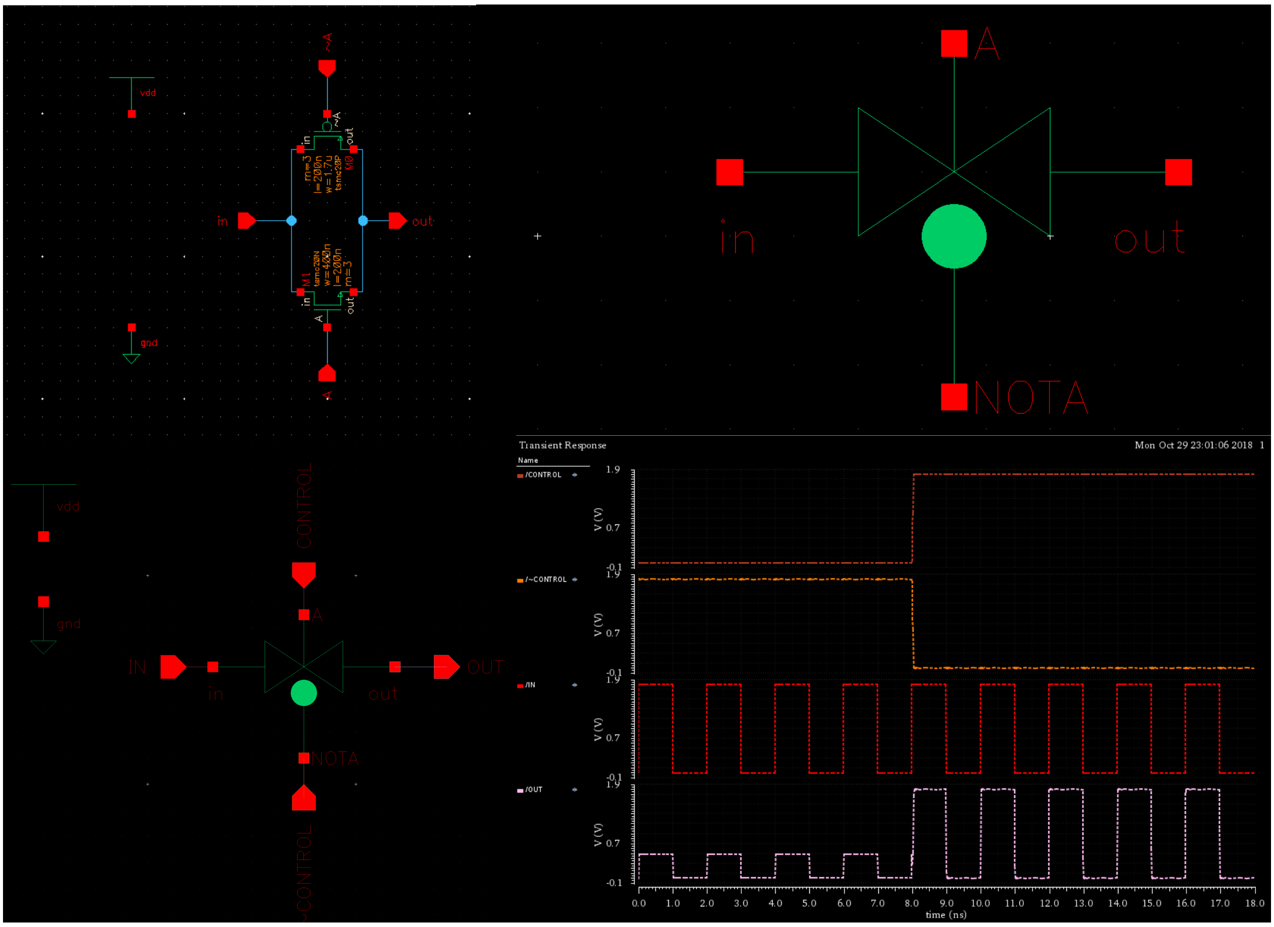
Transmission Gate:

The sizing was similar to Inverter 1

Design:

PMOS SIZE: 1.7um
NMOS SIZE: 400nm

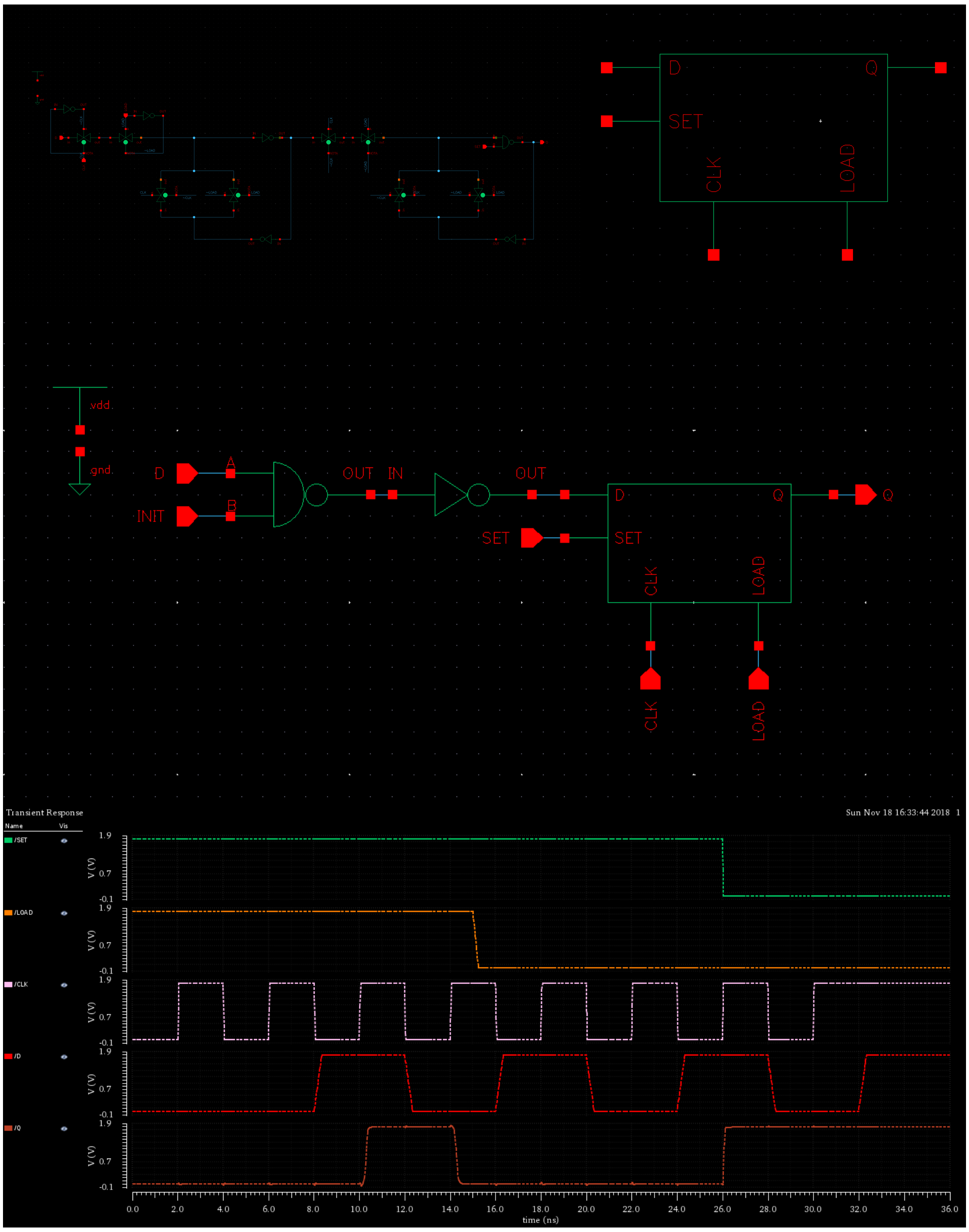
Schematic and Symbol:



With the above design, schematic for the Transmission gate was made and then converted into a symbol which was tested with a simple logic. This symbol was used in the subsequent labs.

<u>Lab 2:</u>

In this lab, we built simple schematic and layout for a positive edge triggered D flipflop. There is LOAD control signal which when asserted allows D to move to the output on the positive edge of the clock. A multiplexer has been used at the beginning of the flip flop to provide a recirculating path for Q when LOAD is unasserted. An active low Set signal has been provided. It has the highest priority and it is synchronous. When Set is asserted, the output would go high on the positive edge of the clock. Two latches have been used in master slave configuration to construct the flip flop. The first latch is made up of 2 transmission gates and 2 inverters. One set of transmission gate and inverter together form the feedforward path and the other set forms the feedback path to recirculate the value. The second latch is made up of 2 transmission gates and an inverter and a NAND gate. The NAND gate is controlled by the output of the first set of latch and by SET.



With the above design, schematic for the positive edge triggered D flipflop was made and then converted into a symbol which was tested with a simple logic. This symbol was used in the subsequent lab.

Lab 3

In this lab we have designed two special purpose digital circuits that mimic neurons (brain cells).

There two neurons are presented with 5 inputs:

- Data input to the neuron, **D** that could be different every clock cycle
- Single bit inhibitory input **I**
- **Load** control signal, that allows the output firing flip flop to be loaded with a new value
- **Set** control signal that resets the flip flop by loading Vdd
- **Clock** with duty cycle of your choice

The circuit that was designed to achieve this was a Mealy Machine, where the output (the firing flip flop) is derived from the present state and the inputs during the current clock cycle. The next state of the flip flop is also a function of the present state and the inputs.

Mealy Machine

A **sequence detector** is a sequential state machine which takes an input string of bits and generates an output 1 whenever the target sequence has been detected.In a Mealy machine, output depends on the present state and the external input (**D**). Hence in the diagram, the output is written outside the states, along with inputs. Sequence detector is of two types:

- Overlapping
- Non-Overlapping

In an overlapping sequence detector the last bit of one sequence becomes the first bit of next sequence. However, in non-overlapping sequence detector the last bit of one sequence does not become the first bit of next sequence.

For non overlapping 101 case

Input : 0110101011001

Output: 0000100010000

For overlapping 101 case

Input: 0110101011001

Output: 0000101010000

In this lab, we've designed schematic and layout for non-overlapping 1111 and 1001 Mealy sequence detector.

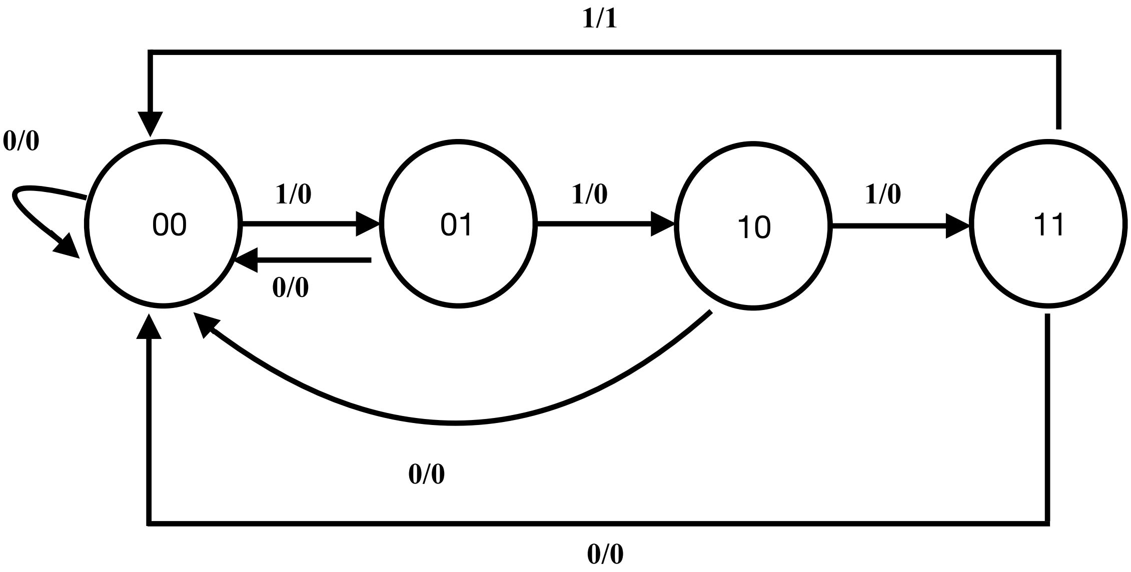
Neuron 1

The neurons fires when their input D1 have the sequence 1111 and I = 0. Each neuron loads 1 into the output flip flop for one clock cycle when the input sequence is correct and then resets it by loading 0 into the output flip flop on the next clock cycle. When I = 0, it prevents the neuron from firing even when the input sequence has 1111.

Design:

First we design a Mealy machine to detect the sequence 1111 irrespective of I. This requires 2 flip-flops. Next the output of this is given to another flipflop which fires when the sequence is 1111 and I = 0. This final output is represented by AP1.

State diagram for 1111 Sequence



State table for 1111 Sequence

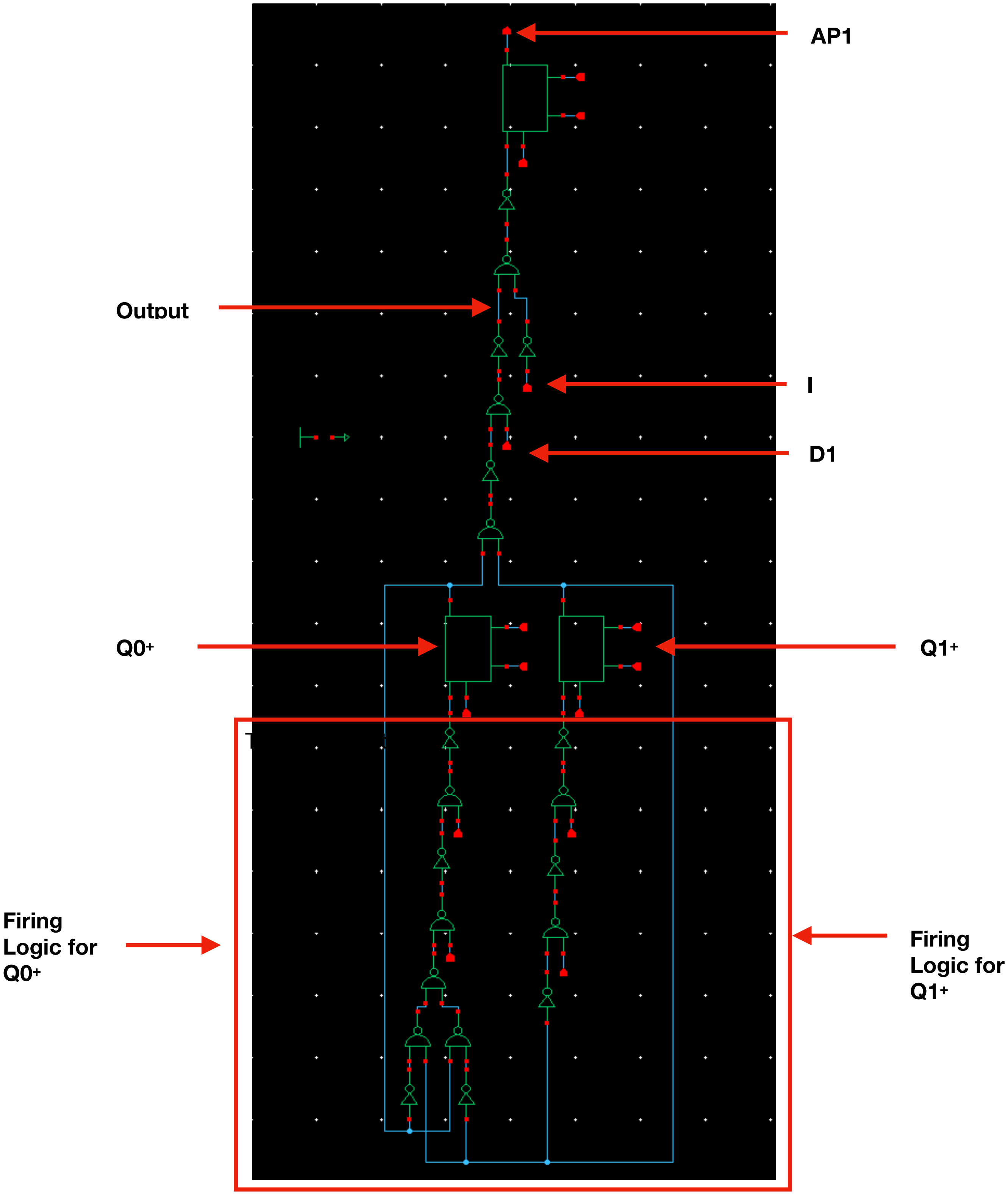
D1	Q0 Present State	Q1 Present State	$Q0^+$ Next State	$Q1^+$ Next State	Output
0	0	0	0	0	0
1	0	0	0	1	0
0	0	1	0	0	0
1	0	1	1	0	0
0	1	0	0	0	0
1	1	0	1	1	0
0	1	1	0	0	0
1	1	1	0	0	1

Logic Expression for Qo^{+} $Q1^{+}$ and Output to construct the mealy machine:

$Q0^{+}$ = D1.[~Q0.Q1 + Q0.~Q1] = **D1 AND [Q0 XOR Q1]**

$Q1^{+}$ = D1.~Q0.~Q1 + D.Q0.~Q1 = D1.~Q1.[Q0 + ~Q0] = **D1 AND ~Q1**

Output= **D1 AND Q0 AND Q1**

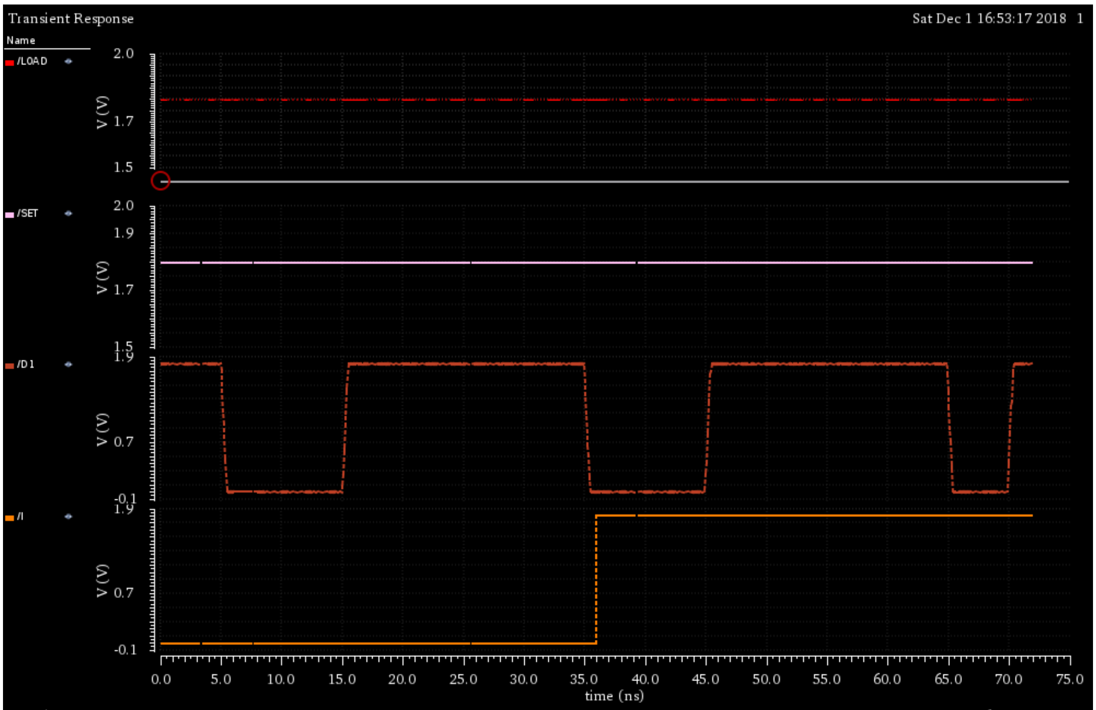


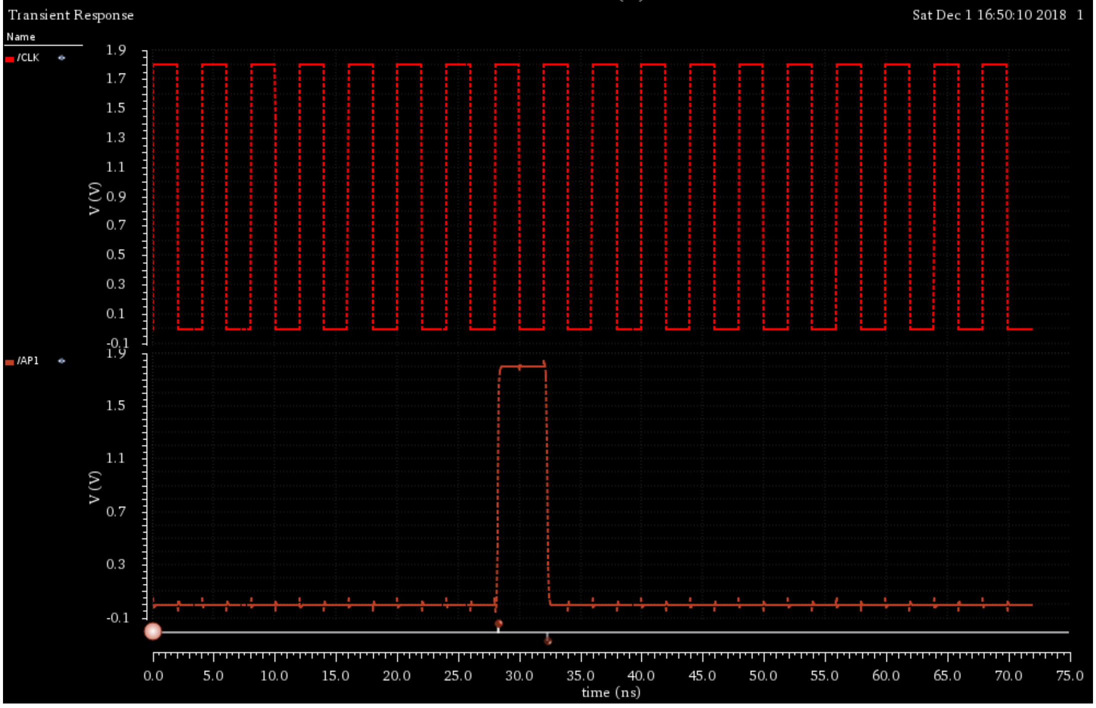
Inhibitory Logic:

The output coming from Q0⁺ & Q1⁺ flip-flops are fed to an AND gate which has its other input as ~I. When I is one, it makes the output of AND gate to be zero since one of its input is zero, by this way when I is activated it prevents the final flipflop from firing irrespective of the output coming from the Q0⁺ & Q1⁺ flip-flops.

Schematic Simulation:

Here you can see that the neuron fires when the input sequence is 1111 and I = 0. Also we can see when I = 1, the neuron doesn't fire despite the sequence to be 1111.





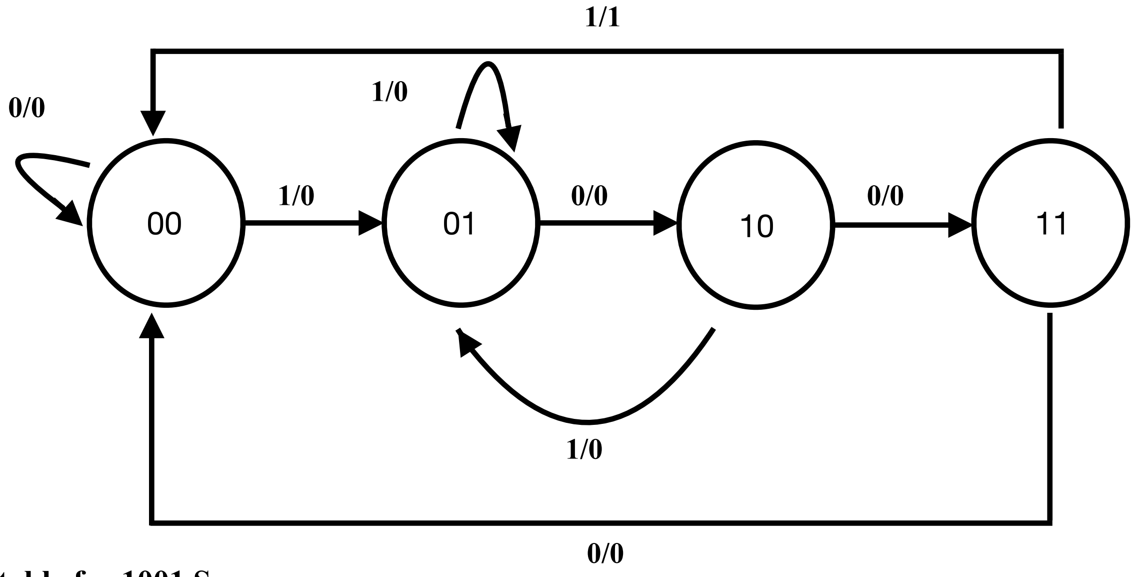
Neuron 2

The neurons fires when their input D2 have the sequence 1001 and I = 0. Each neuron loads 1 into the output flip flop for one clock cycle when the input sequence is correct and then resets it by loading 0 into the output flip flop on the next clock cycle. When I = 0, it prevents the neuron from firing even when the input sequence has 1001.

Design:

First we design a Mealy machine to detect the sequence 1001 irrespective of I. This requires 2 flip-flops. Next the output of this is given to another flipflop which fires when the sequence is 1001 and I = 0. This final output is represented by AP2.

State diagram for 1001 Sequence



State table for 1001 Sequence

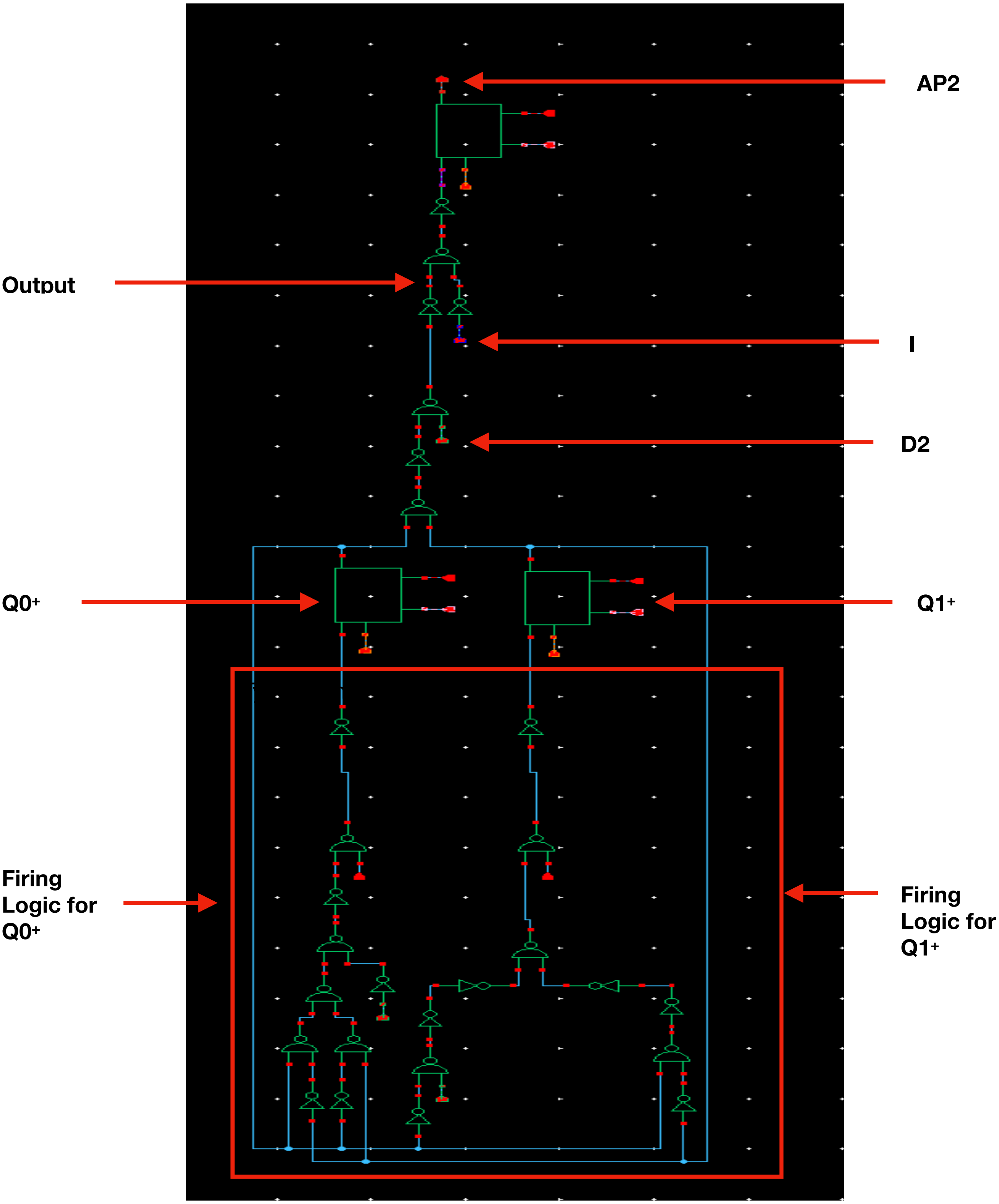
D2	Q0 Present State	Q1 Present State	Q0+ Next State	Q1+ Next State	Output
0	0	0	0	0	0
1	0	0	0	1	0
0	0	1	1	0	0
1	0	1	0	1	0
0	1	0	1	1	0
1	1	0	0	1	0
0	1	1	0	0	0
1	1	1	0	0	1

<u>Logic Expression for Qo$^+$ Q1$^+$ and Output to construct the mealy machine:</u>

Q0$^+$ = ~D2.[~Q0.Q1 + Q0.~Q1] = **~D2 AND [Q0 XOR Q1]**

Q1$^+$ = D2.~Q0.~Q1 + D2.~Q0.Q1 + ~D2.Q0.~Q1 + D2.Q0.~Q1 = **[D2 AND ~Q0] OR [Q0 OR ~Q1]**

Output= D2 AND Q0 AND Q1

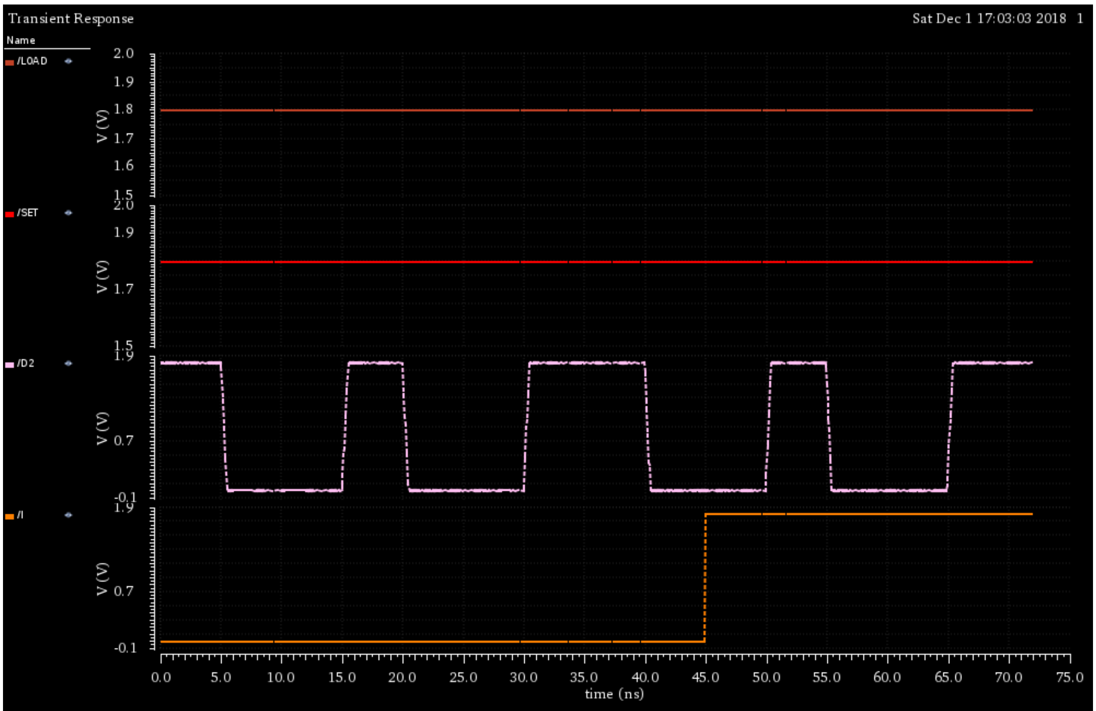


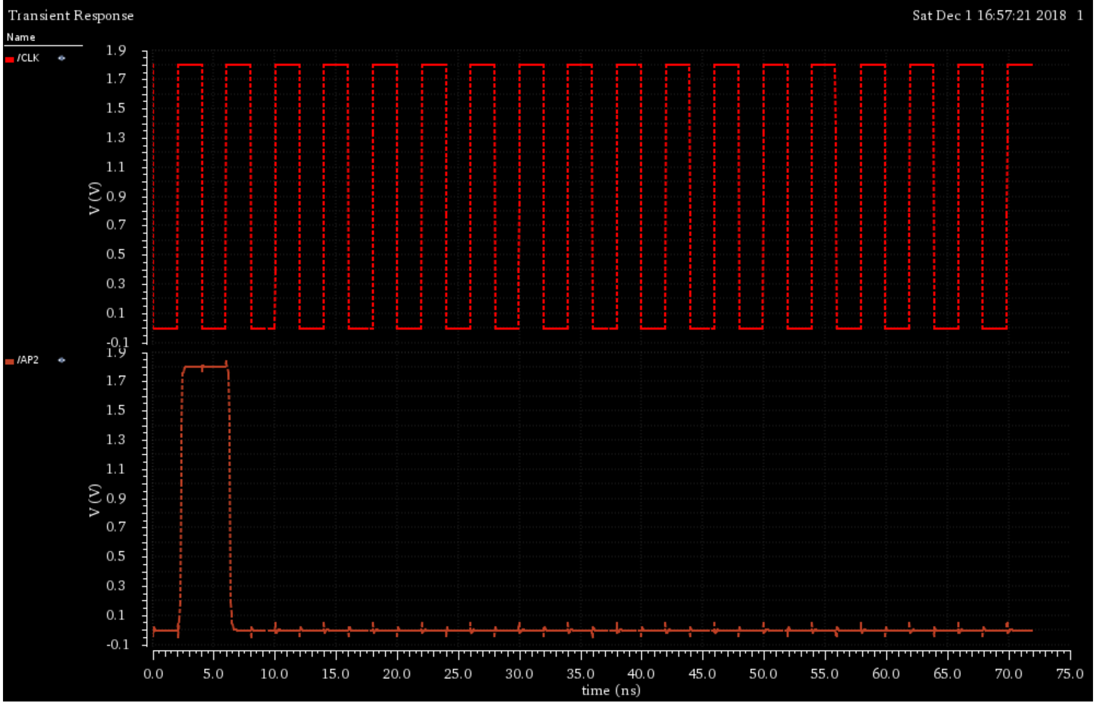
Inhibitory Logic:

The output coming from Q0⁺ & Q1⁺ flip-flops are fed to an AND gate which has its other input as ~I. When I is one, it makes the output of AND gate to be zero since one of its input is zero, by this way when I is activated it prevents the final flipflop from firing irrespective of the output coming from the Q0⁺ & Q1⁺ flip-flops.

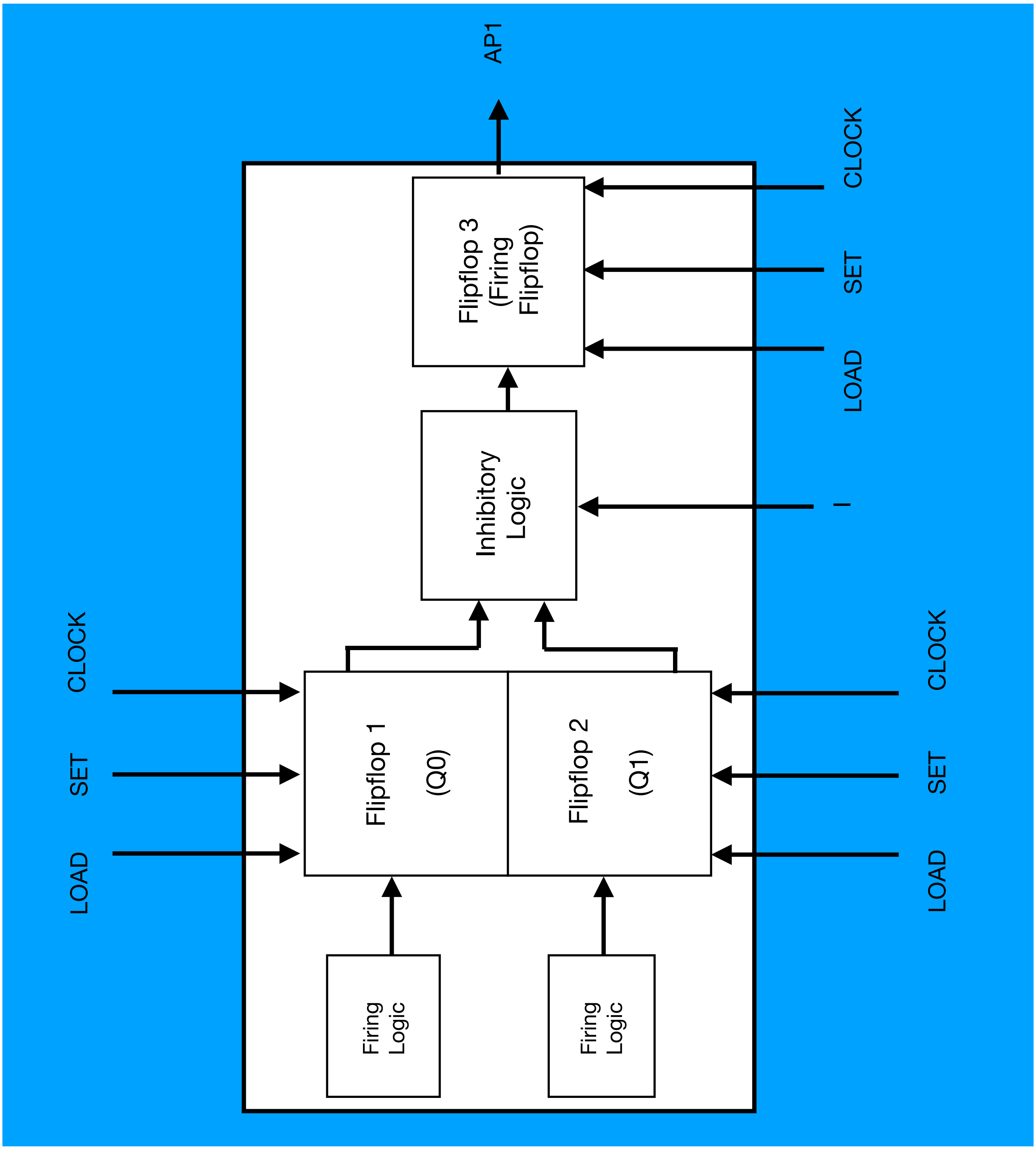
Schematic Simulation:

Here you can see that the neuron fires when the input sequence is 1001 and I = 0. Also we can see when I = 1, the neuron doesn't fire despite the sequence to be 1001.



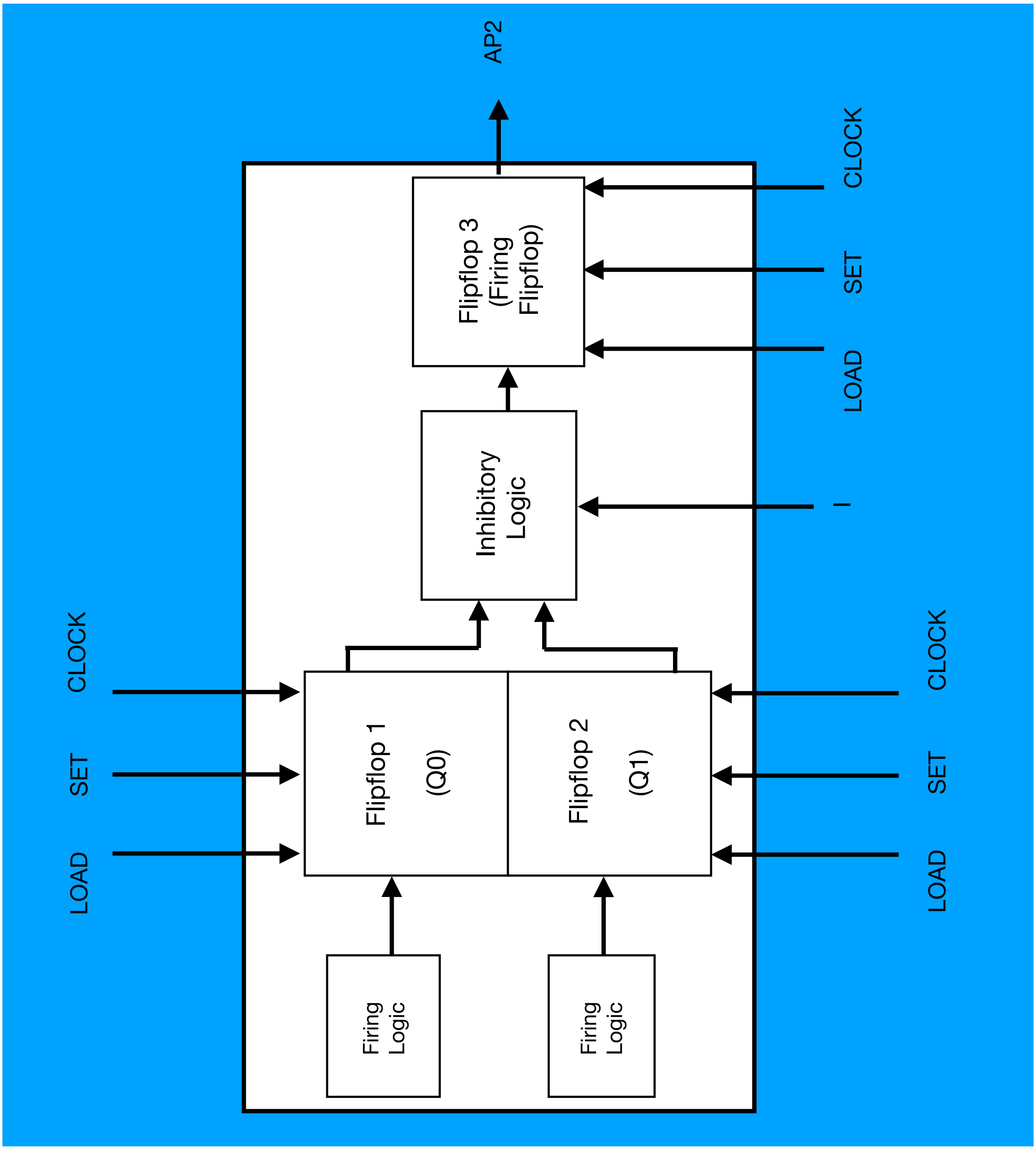


Neuron 1 (1111 Sequence detector) Floor Plan



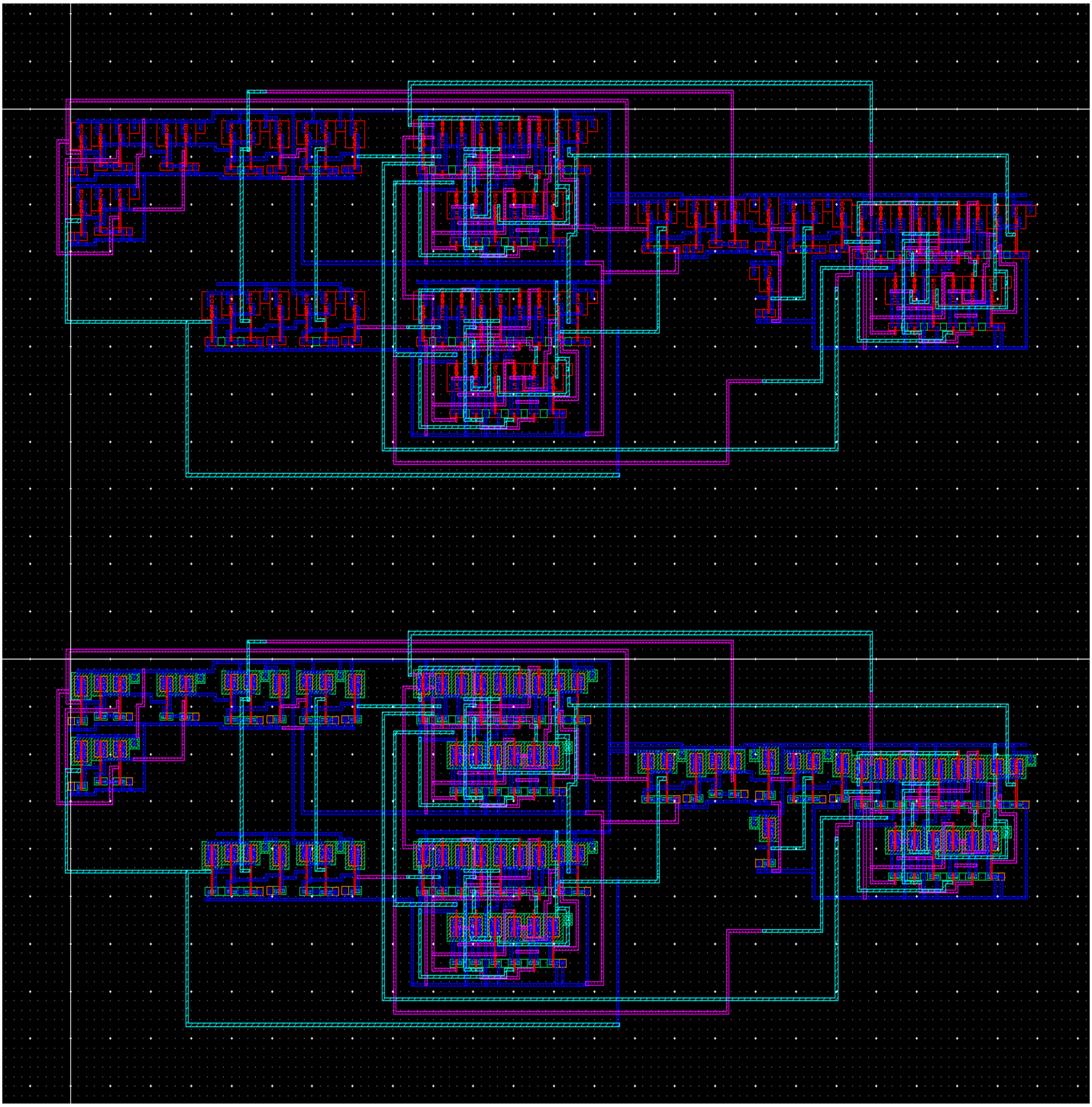
The figure represents floorplan for a Mealy machine which detects the sequence 1111. It involves flip-flops and logic circuits.The two flip-flops Q0 and Q1 are used to check for the sequence 1111 from the input (D1) and fires when the sequence matches.This output is fed to the inhibitory logic which involves inhibitory signal I. It prevents the neuron from firing when activated. The output of inhibitory logic is fed to the firing flipflop which fires when the sequence 1111 is detected and the inhibitory signal I is low which is confirmed through the inhibitory logic circuit. Firing logic is designed in such a way that it makes the output of flip-flops Q0 and Q1 to fire when the condition is met. It involves NAND gates and inverters. The output of firing logic is fed to an AND gate before feeding to the flipflop. This is done to initialise the states of the two flip-flops to zero.

Neuron 2 (1001 Sequence detector) Floor Plan



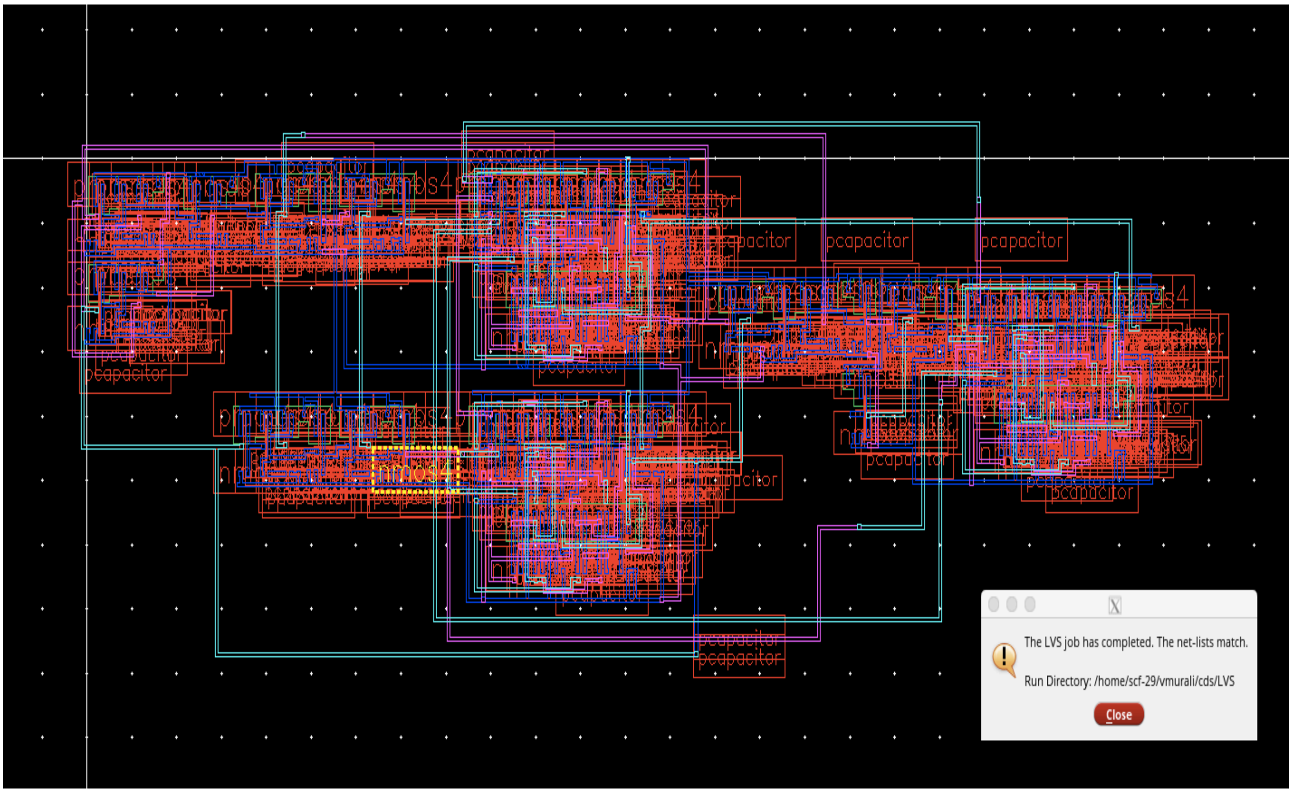
The figure represents floorplan for a Mealy machine which detects the sequence 1001. It involves flip-flops and logic circuits.The two flip-flops Q0 and Q1 are used to check for the sequence 1001 from the input (D2) and fires when the sequence matches.This output is fed to the inhibitory logic which involves inhibitory signal I. It prevents the neuron from firing when activated. The output of inhibitory logic is fed to the firing flipflop which fires when the sequence 1001 is detected and the inhibitory signal I is low which is confirmed through the inhibitory logic circuit. Firing logic is designed in such a way that it makes the output of flip-flops Q0 and Q1 to fire when the condition is met. It involves NAND gates and inverters. The output of firing logic is fed to an AND gate before feeding to the flipflop. This is done to initialise the states of the two flip-flops to zero.

Neuron 1 (1111 Sequence detector) Layout



- Wn equals 400nm, Wp almost equals 1720nm for inverter, transmission gate and Wn equals 400nm, Wp almost equals 1250nm for NAND gate
- M1_POLY is used to give input to the poly (gate) of a device (PMOS/NMOS)
- PTAP and NTAP is used to prevent the current from flowing unnecessarily between P Substrate and N impurity and vice versa.

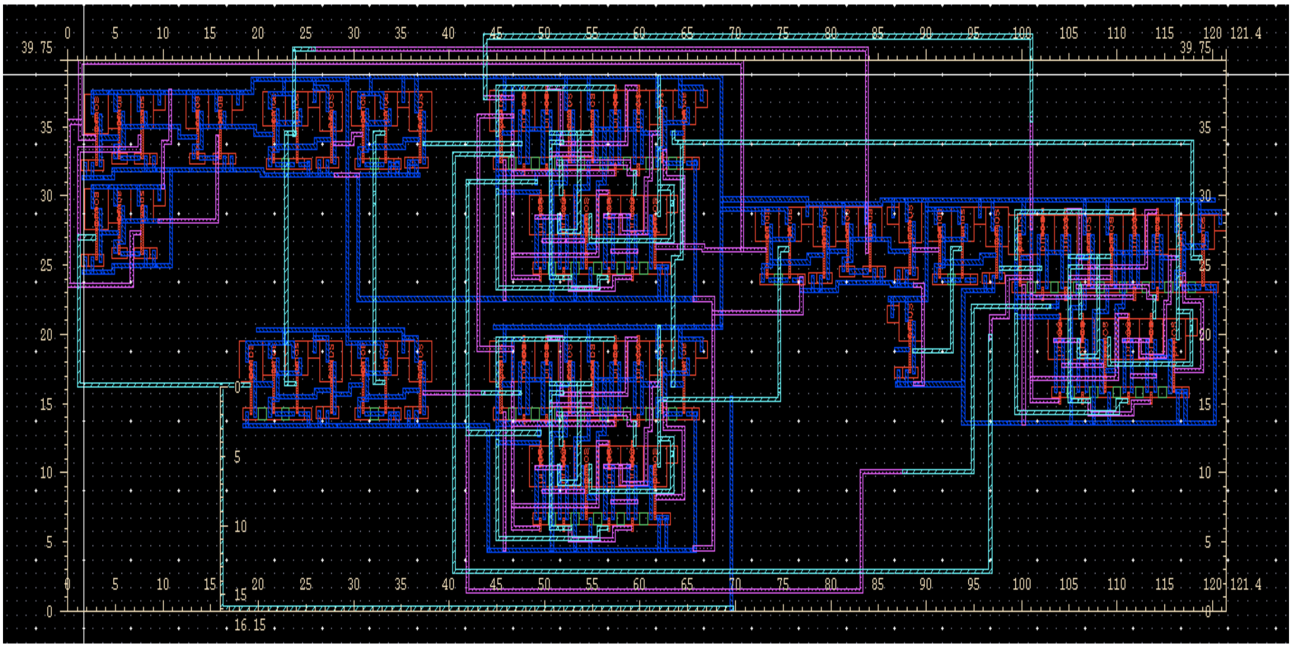
Neuron 1 (1111 Sequence detector) LVS



The LVS job has completed. The net-lists match.

Run Directory: /home/scf-29/vmurali/cds/LVS

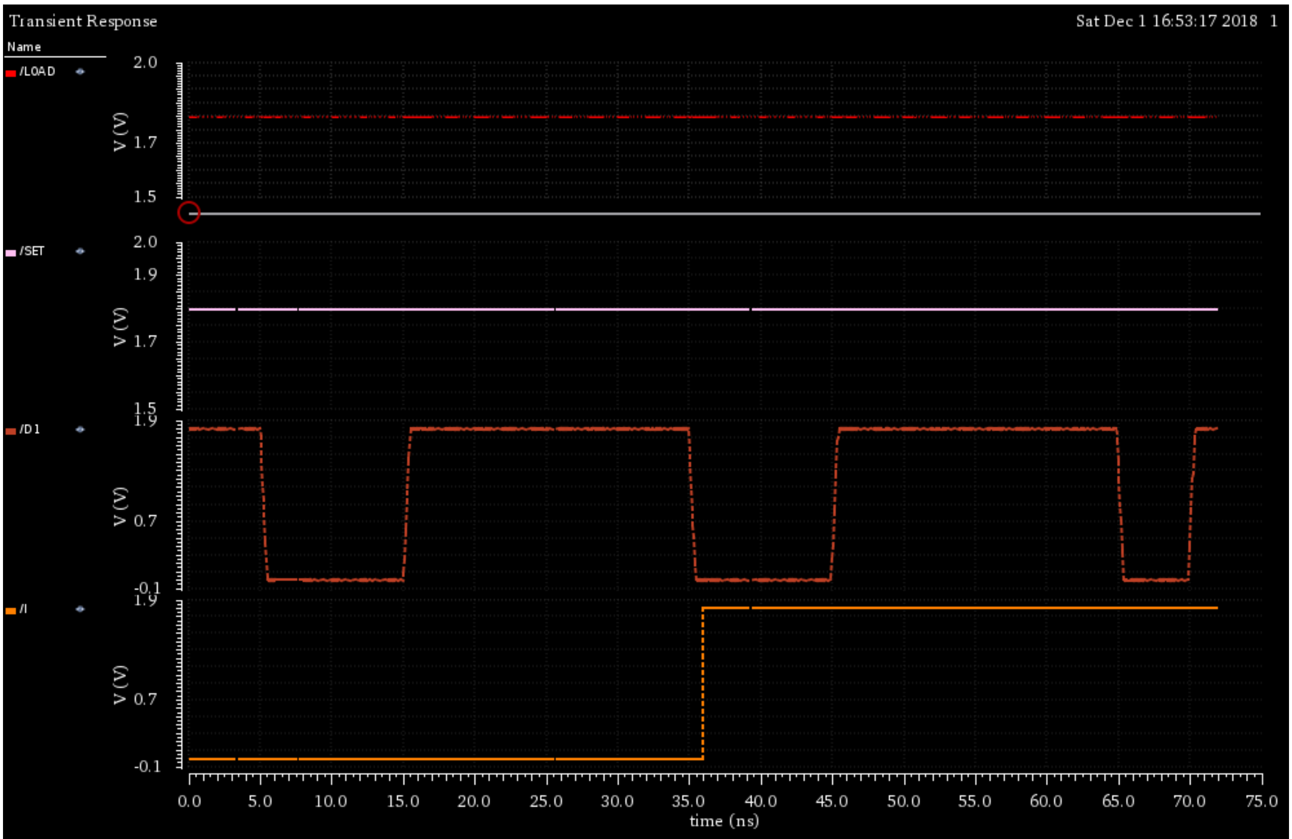
Neuron 1 (1111 Sequence detector) Area

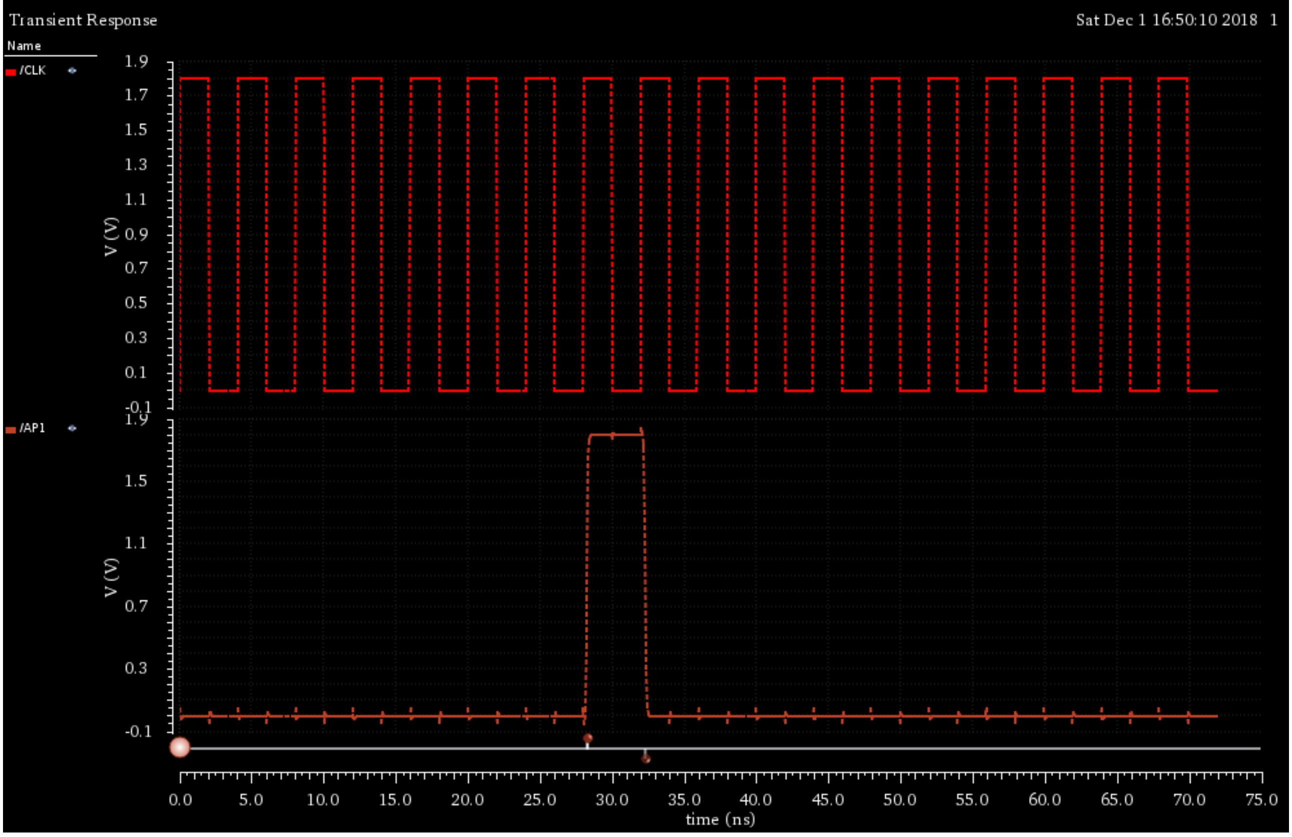


Cell Height (um)	Cell Width (um)	Cell Area (um2)
39.75	121.4	4825.65

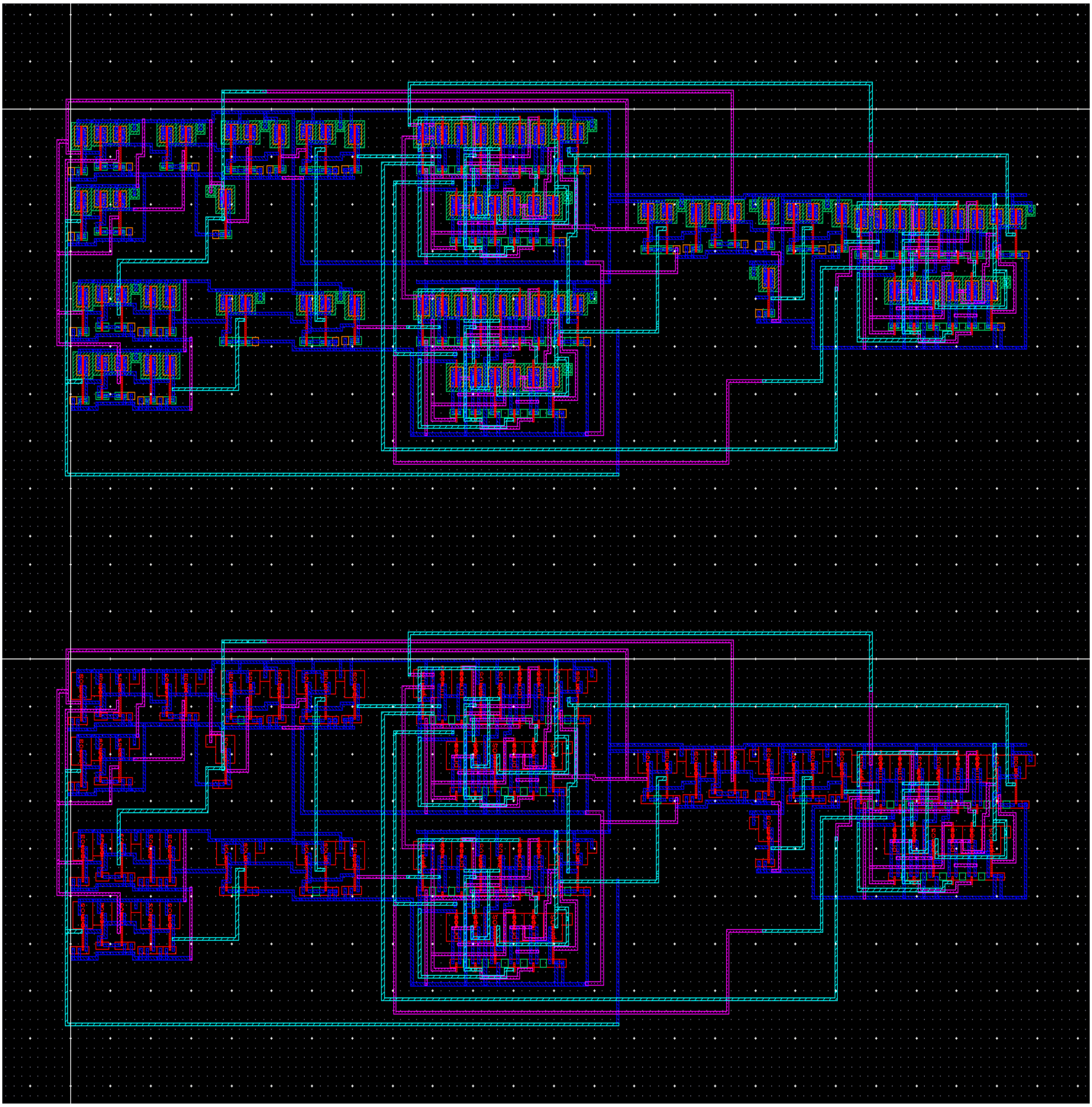
Neuron 1 (1111 Sequence detector) Layout Simulation

Here you can see that the neuron fires when the input sequence is 1111 and I = 0. Also we can see when I = 1, the neuron doesn't fire despite the sequence to be 1111.





Neuron 2 (1001 Sequence detector) Layout

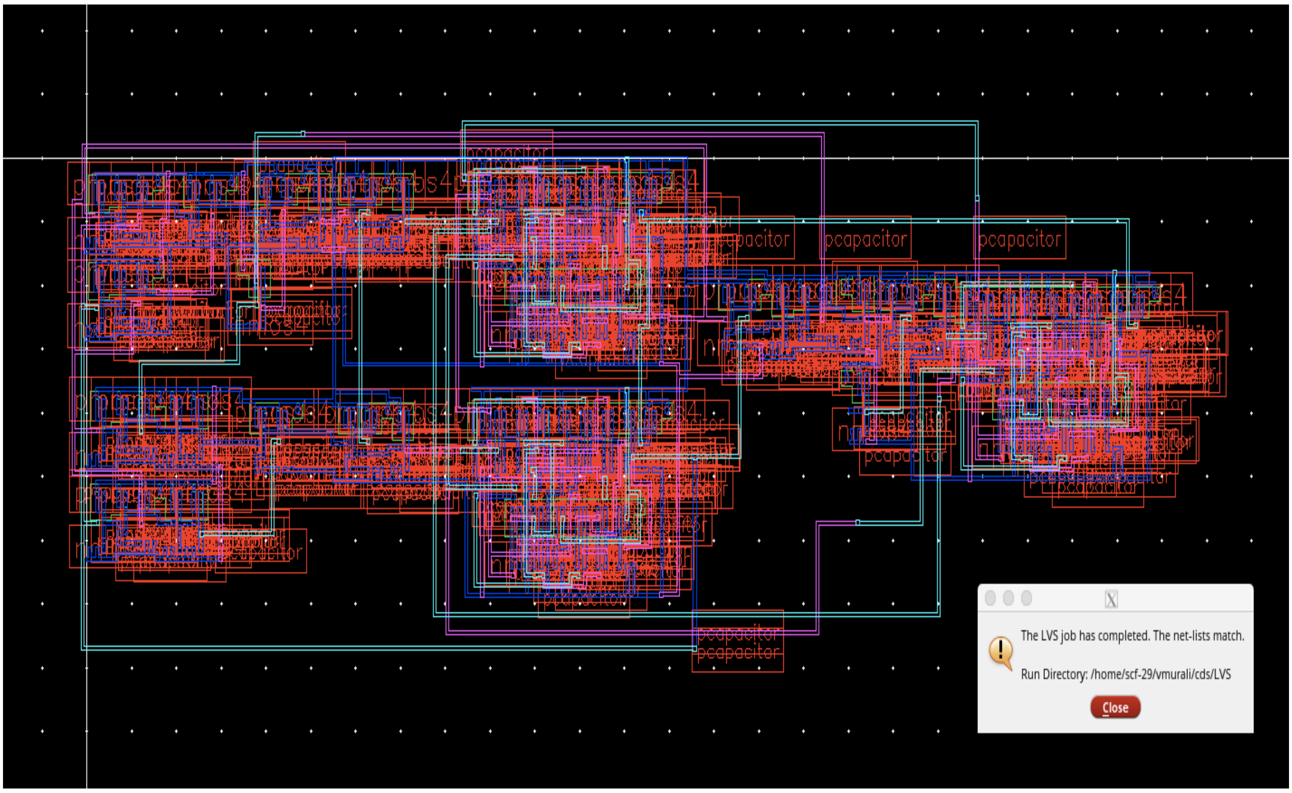


- Wn equals 400nm, Wp almost equals 1720nm for inverter, transmission gate and Wn equals 400nm, Wp almost equals 1250nm for NAND gate

- M1_POLY is used to give input to the poly (gate) of a device (PMOS/NMOS)

- PTAP and NTAP is used to prevent the current from flowing unnecessarily between P Substrate and N impurity and vice versa.

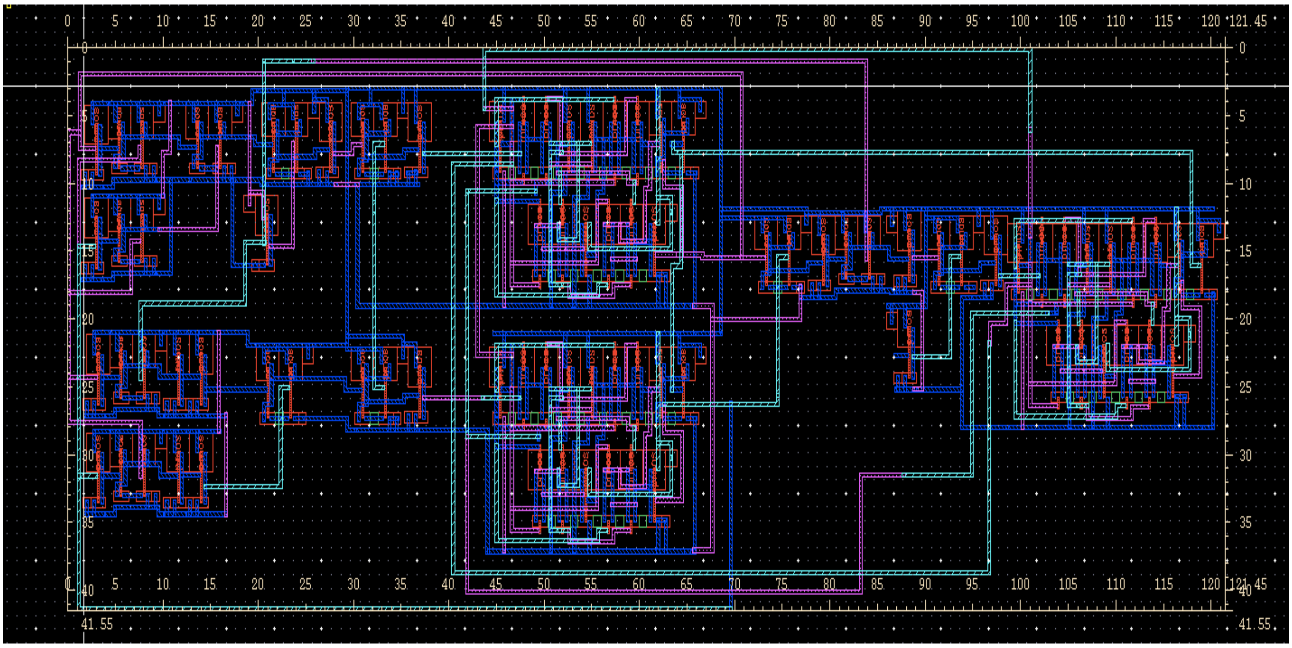
Neuron 2 (1001 Sequence detector) LVS



The LVS job has completed. The net-lists match.

Run Directory: /home/scf-29/vmurali/cds/LVS

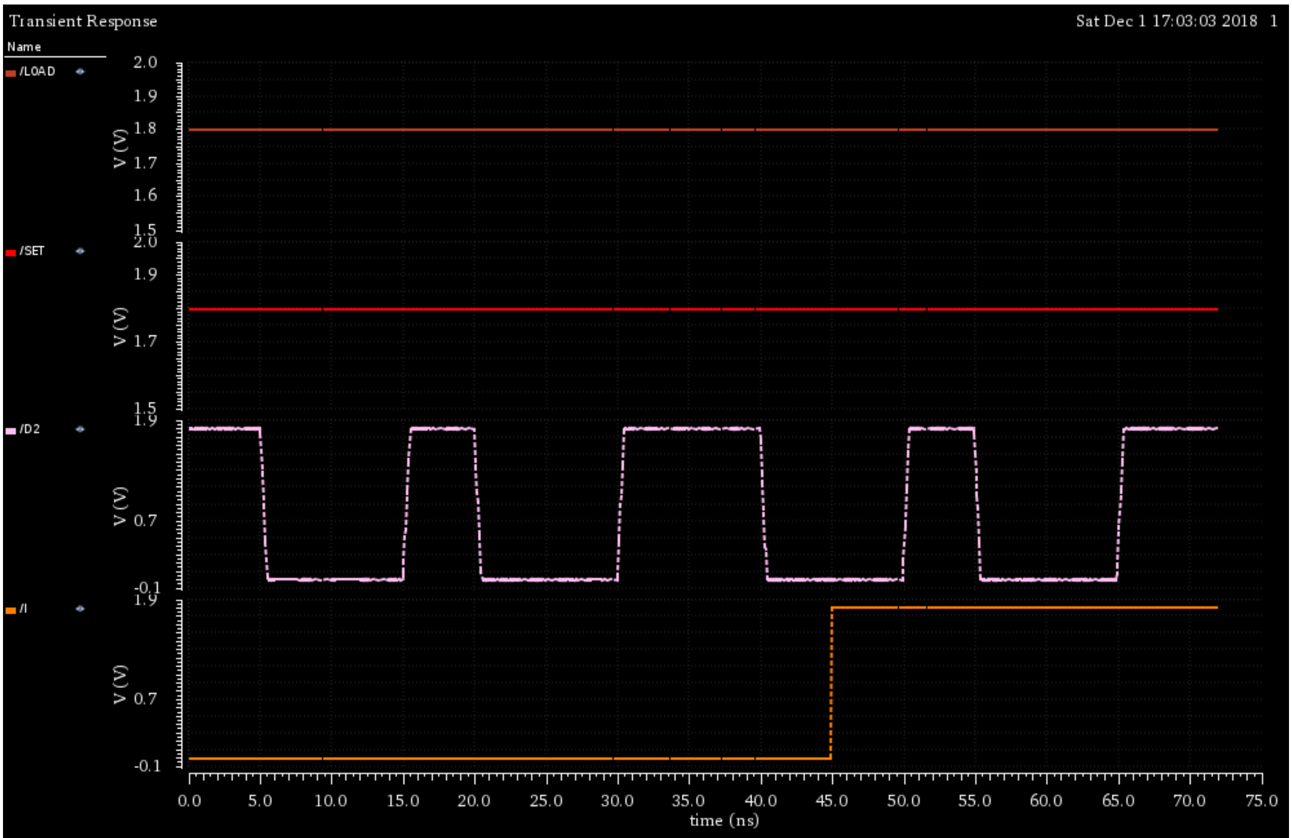
Neuron 2 (1001 Sequence detector) Area

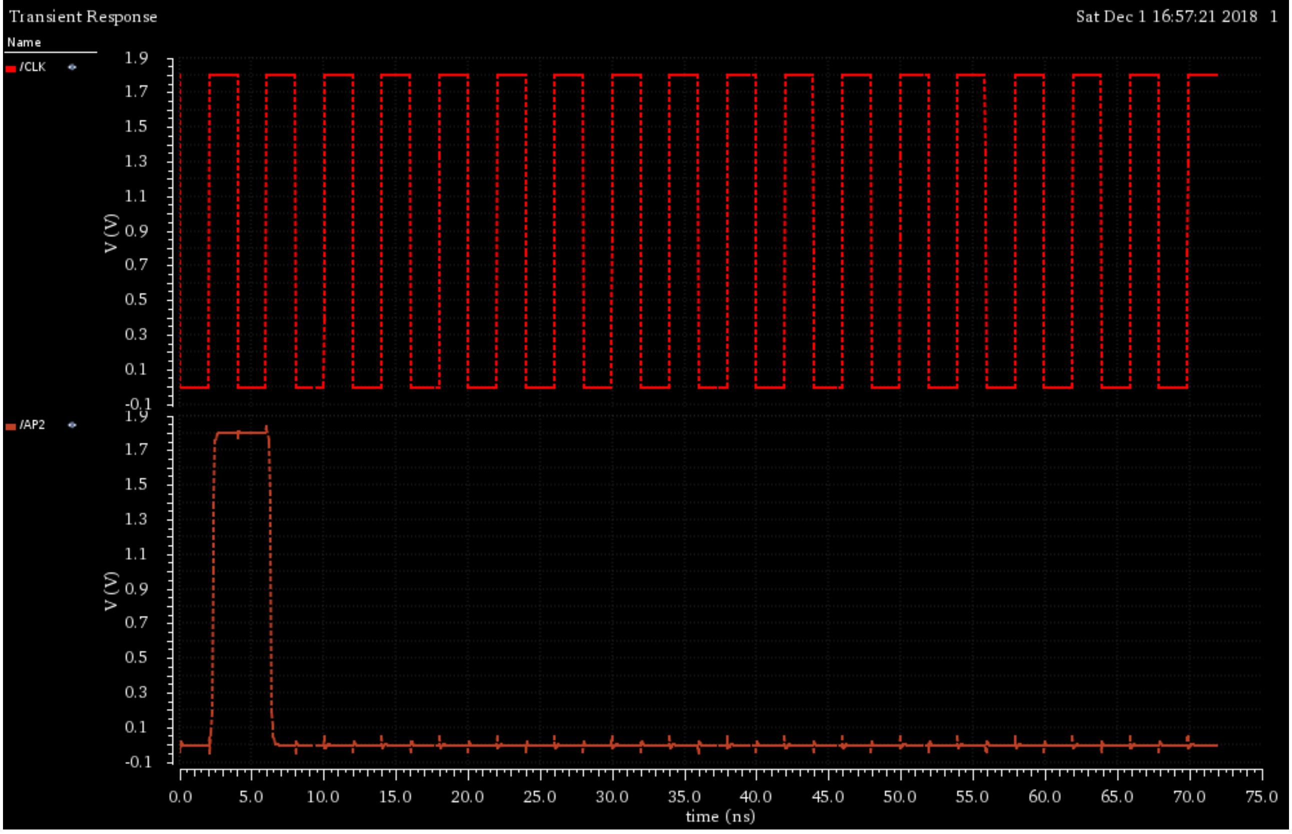


Cell Height (um)	Cell Width (um)	Cell Area (um^2)
41.55	121.45	5046.2475

Neuron 2 (1001 Sequence detector) Layout Simulation

Here you can see that the neuron fires when the input sequence is 1001 and I = 0. Also we can see when I = 1, the neuron doesn't fire despite the sequence to be 1001.





Description of simulation using spectre

Spectre is a SPICE-class circuit simulator. It provides the basic SPICE analyses and component models. I've used Spectre simulator to simulate my schematic and layout designs . The stimuli for my schematic and layout are same same for each layout. They are:

For schematic and layout
Load: DC (Voltage= 1.8V)
CLK: Pulse (DC Voltage = 1.8, Voltage 1= 0V, Voltage 2= 0V, Period= 4n)
Set: DC (Voltage= 1.8V)
Init: Bit (Period= 2n, Rise Time= 0.05n, Fall Time= 0.05n, Pattern: 001111111111111111111111111111111111)
I: Bit (Period= 4n, Rise Time= 0.05n, Fall Time= 0.05n, Pattern: 000000000111111111)

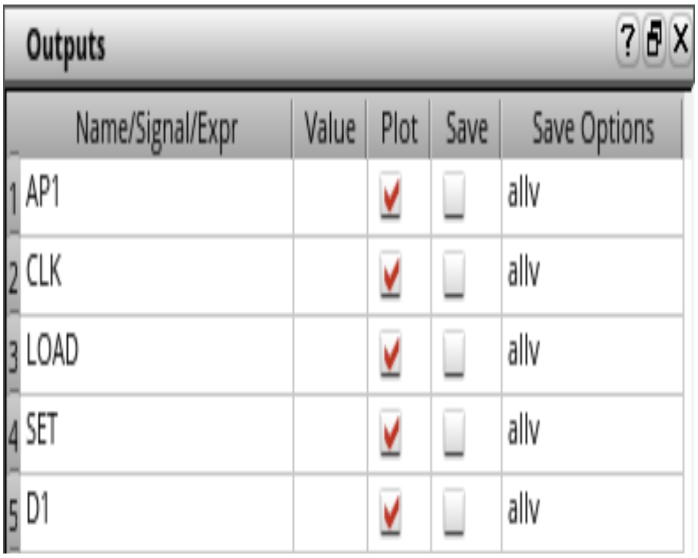
For neuron 1
D1: Bit (Period= 5n, Rise Time= 0.5n, Fall Time= 0.5n, Pattern: 100111100111101111100111100111101111)

For neuron 2
D2: Bit (Period= 5n, Rise Time= 0.5n, Fall Time= 0.5n, Pattern: 1001001100100110010011001001)

For Layout
vdd!= 1.8V
gnd!= 0V

Output of neuron 1: AP1
Output of neuron 2: AP2

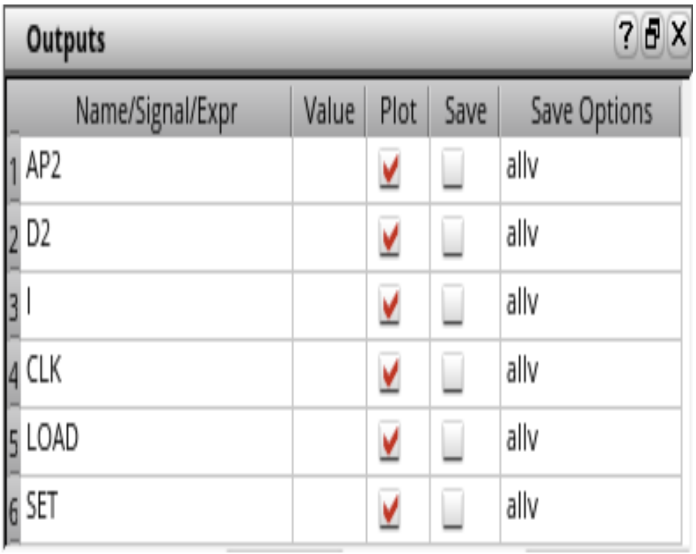
For both the neurons, transient analysis was performed for 72 ns.



Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	AP1		☑	□	allv
2	CLK		☑	□	allv
3	LOAD		☑	□	allv
4	SET		☑	□	allv
5	D1		☑	□	allv

For Neuron 1



Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	AP2		☑	□	allv
2	D2		☑	□	allv
3	I		☑	□	allv
4	CLK		☑	□	allv
5	LOAD		☑	□	allv
6	SET		☑	□	allv

For Neuron 2