WRIGHT STATE UNIVERSITY

EE 4540L/6540L/CEG4322L/CEG6322L

FALL 2023

TA: Kanchan Vissamsetty

Lab section: Lab1

UID: U01102112

Name: Vamshikrishna Bavirishetty

"I have neither given nor received aid on this assignment, nor have I observed any violation of the Honor code"

Signature: Vamshi **Date:09/15/2023**

Report due date: 09/18/2023

AIM/OBJECTIVE:

The main objective of this lab is to become familiar with cadence Virtuoso tool and to design a 4-Bit prime number detector. Where it can identify whether a 4-Bit binary number ranging from 0 to 15 is a prime number or not. Here's a clear outline of the aim and objectives.

PROCEDURE

The first step in this lab is to draw a truth table with four inputs A3, A2, A1, A0 and F as the output. Then mark prime numbers in between 0-15 as 1 and others as 0.

Truth Table

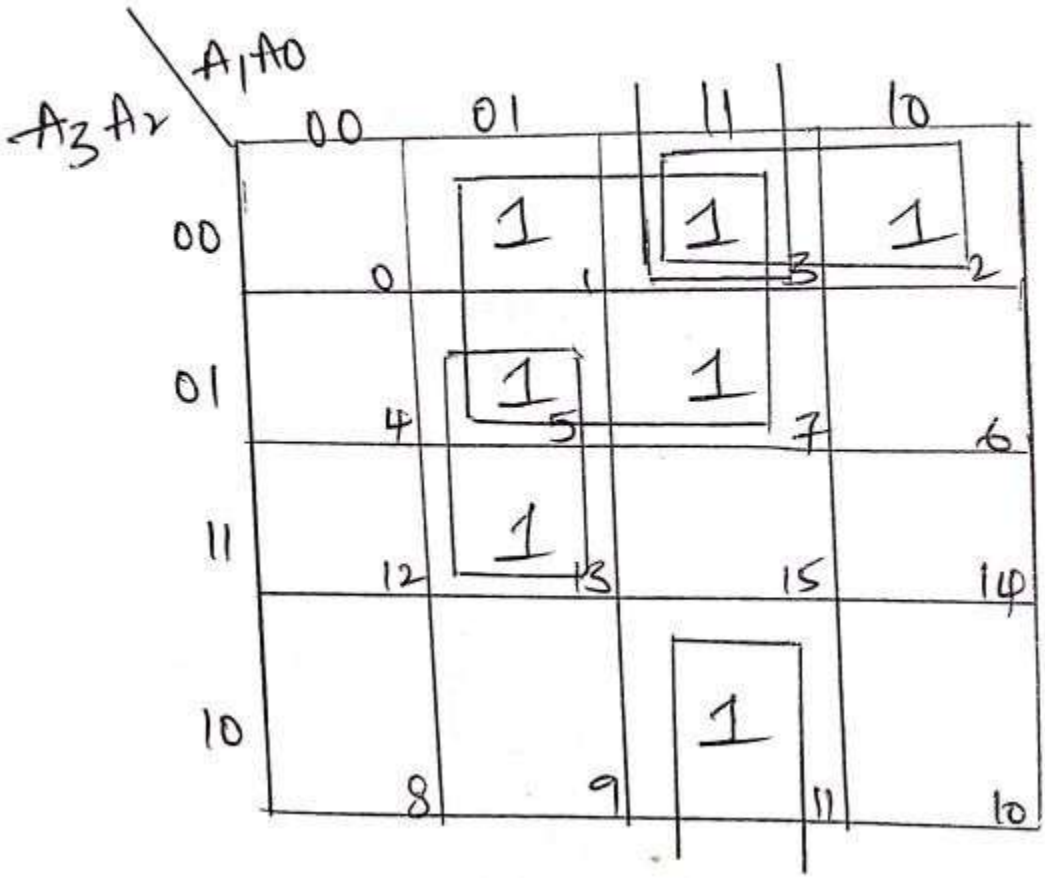
A3	A2	A1	A0	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

After drawing the truth table I extracted the F equation from it.

$$F = \bar{A}_3\bar{A}_2\bar{A}_1A_0 + \bar{A}_3\bar{A}_2A_1\bar{A}_0 + \bar{A}_3\bar{A}_2A_1A_0 + \bar{A}_3A_2\bar{A}_1A_0 + \bar{A}_3A_2A_1A_0 + A_3\bar{A}_2A_1A_0 + A_3A_2\bar{A}_1A_0$$

The above equation requires more logic gates, so I had used K-Map for reducing this equation.

K-Map simplification:

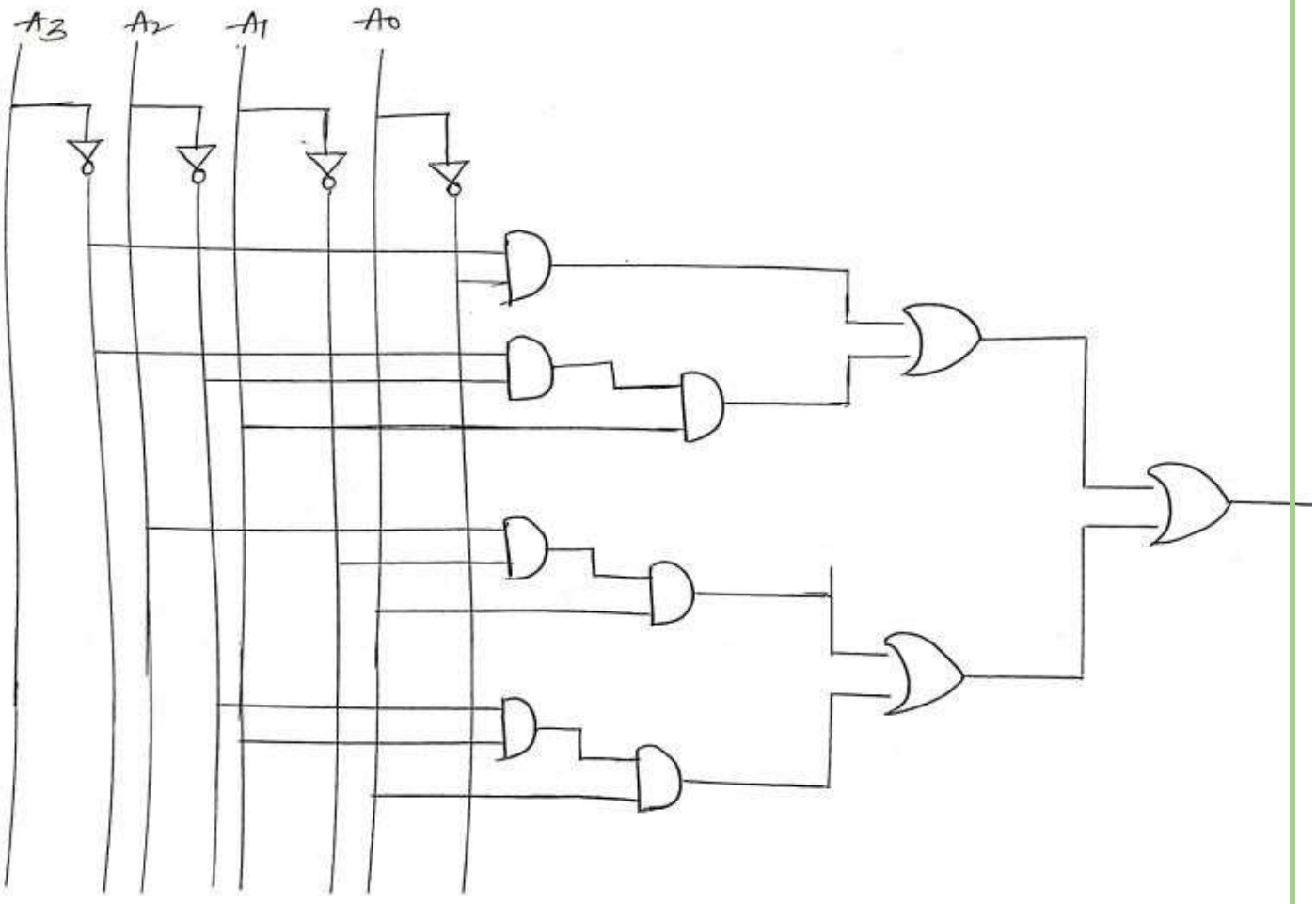


After solving the K-Map the equation will be reduced and below is the reduced equation.

Simplified equation:

$$\overline{A_3}A_0 + \overline{A_3}\,\overline{A_2}A_1 + A_2\overline{A_1}A_0 + \overline{A_2}A_1A_0$$

With this equation we can design a logic diagram where we will implement this logic diagram in the Cadence tool.

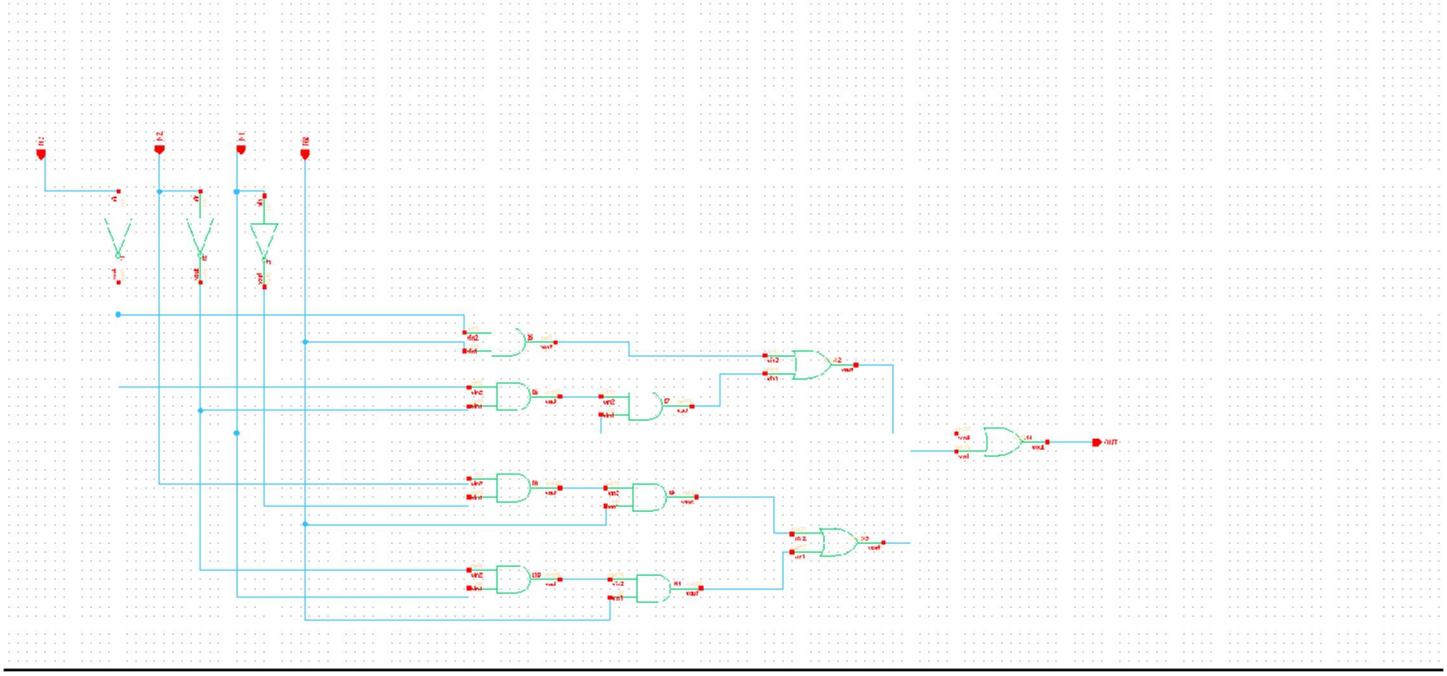


Here in the above logic diagram the maximum inputs for the logic gates are only 2. Because of this reason I have used the extra logic gates.

Procedure to implement this logic diagram in cadence tool:

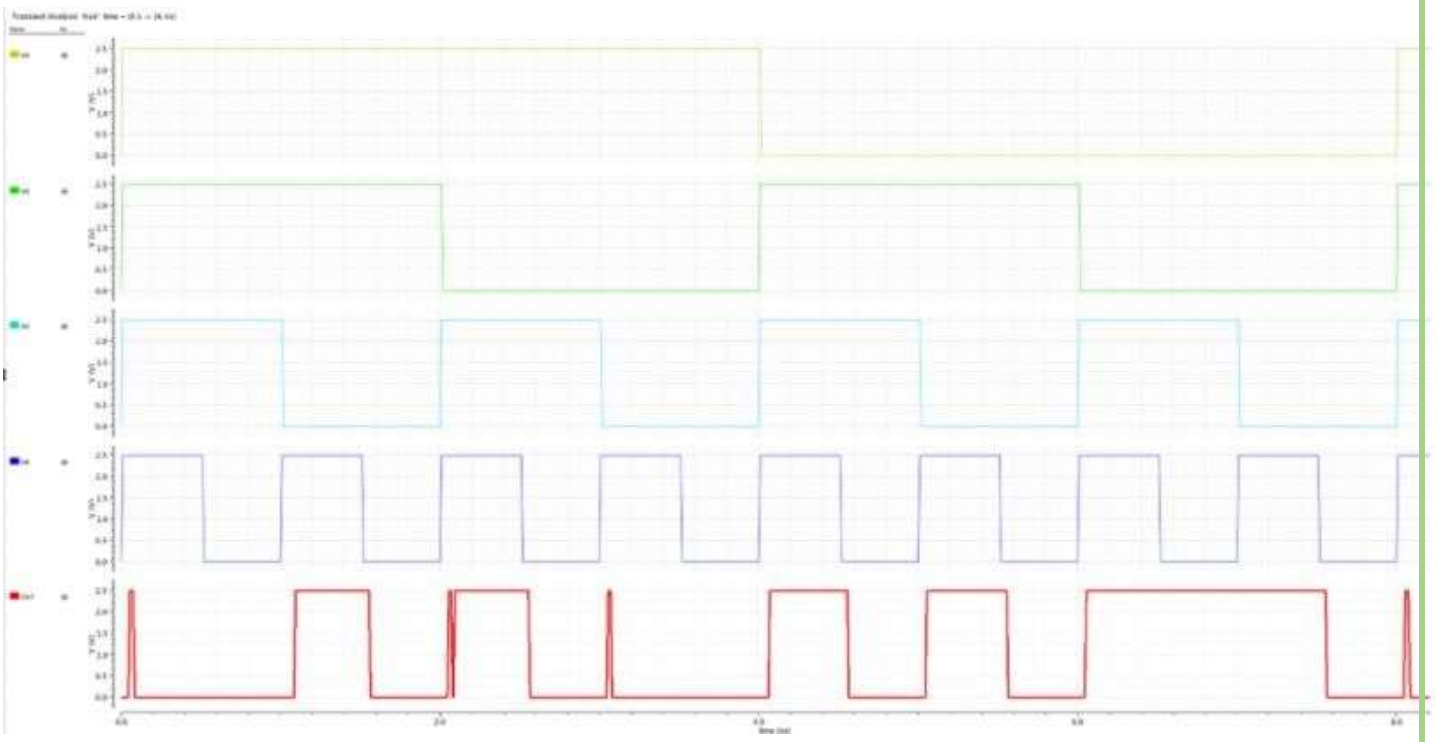
- First switch on to the linux machine.
- Then open the terminal and type (virtuoso &).
- Create a new library and name it.
- Next step is to create a new cell in the library we have created.
- And now start implementing the logic diagram in the new schematic.
- To place the logic gates, go to the create option and click on instance then in the component browser browse for the required logic gates and place them in the schematic.
- After placing all the logic gates connect then accordingly using a wire.
- And to create input and output pins click on create and then pin further Add pin window appears where pin names can be given by specifying the direction.
- The next step is to simulate the design. So, for simulation click on launch and then click on ADE Explorer and then select create a new view, now a new window opens.
- Now select setup, then click on stimuli. In this window select the pulse option.
- In the period filed enter values as 1n, 2n, 4n and 8n.
- In the pulse width field give the values as 0.5n, 1n, 2n and 4n.
- Enter these above values for input signals N0, N1, N2 and N3.
- Set the voltage0= 0, voltage 1= 2.5, rise time = 10ps and fall time = 10ps for all signals.
- For running the simulation select analysis then click on tran. And enter stop time as 16n, and then select enable option and click on ok.
- Then select session and then save the state.
- Next step is to click on simulation and then select Netlist and run in the ADE window.
- To see the waveform click on tools and select results browser.
- After that a new window opens then under tran we can select the input output and power signals.
- To view the graph clearly click on split current strip option.

Schematic Screenshot:



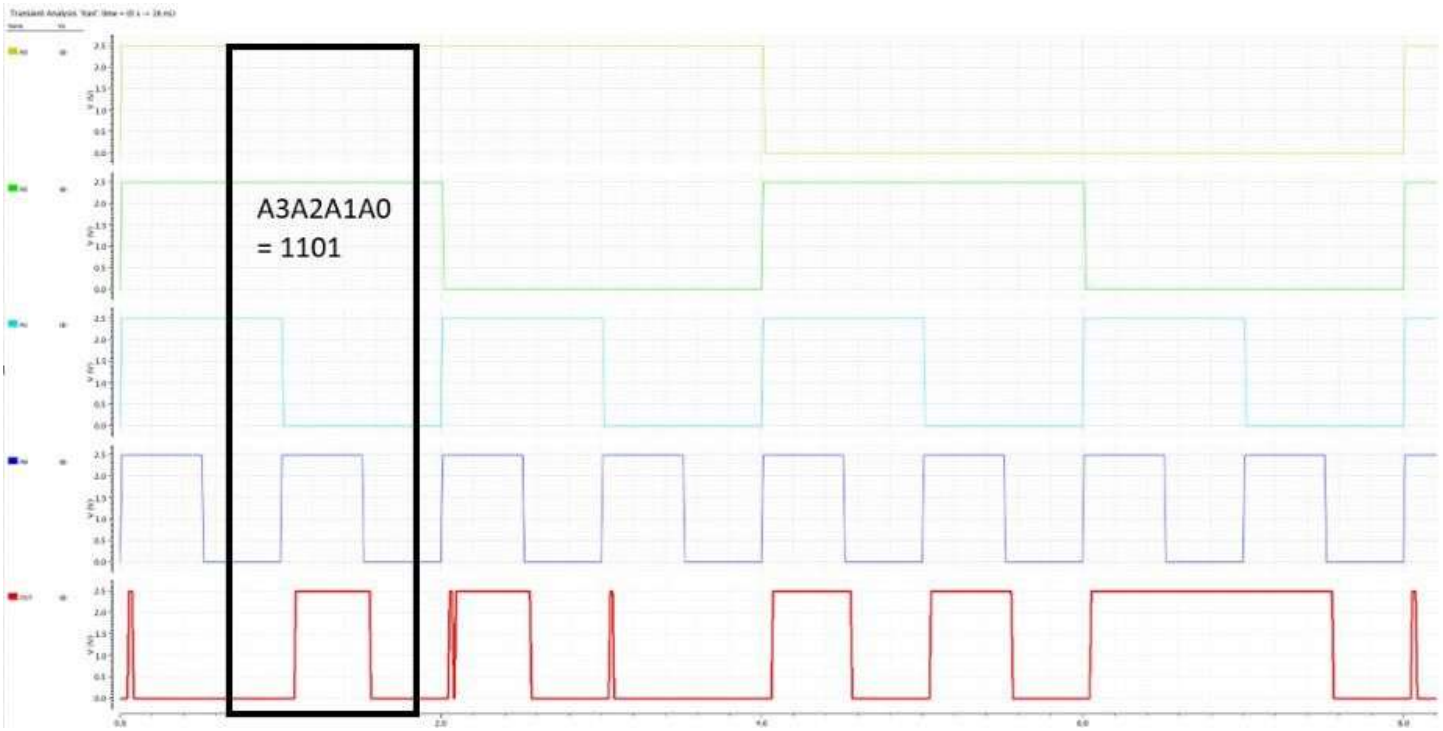
The above is the schematic of the logic diagram where I had implemented in the Cadence Virtuoso tool.

RESULT WAVEFORM:

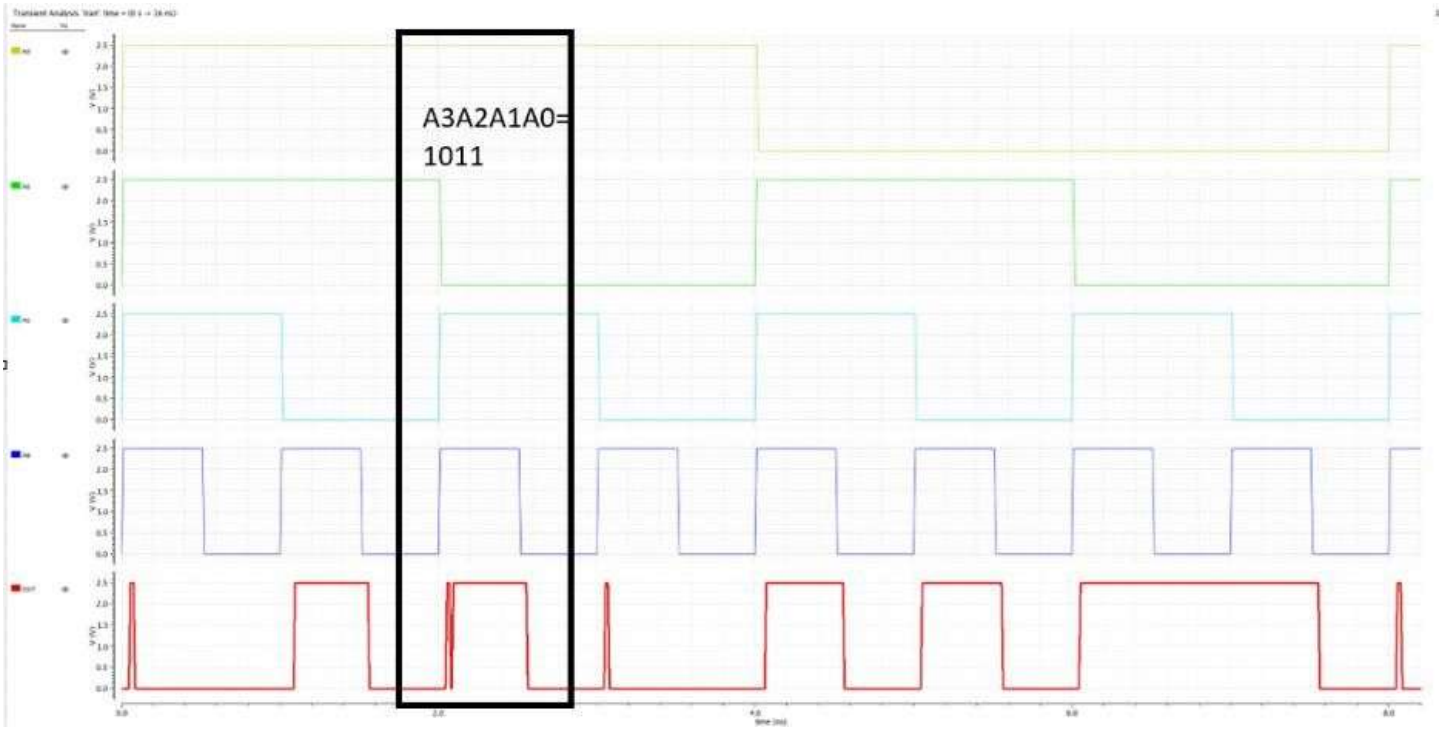


Result Waveform Analysis:

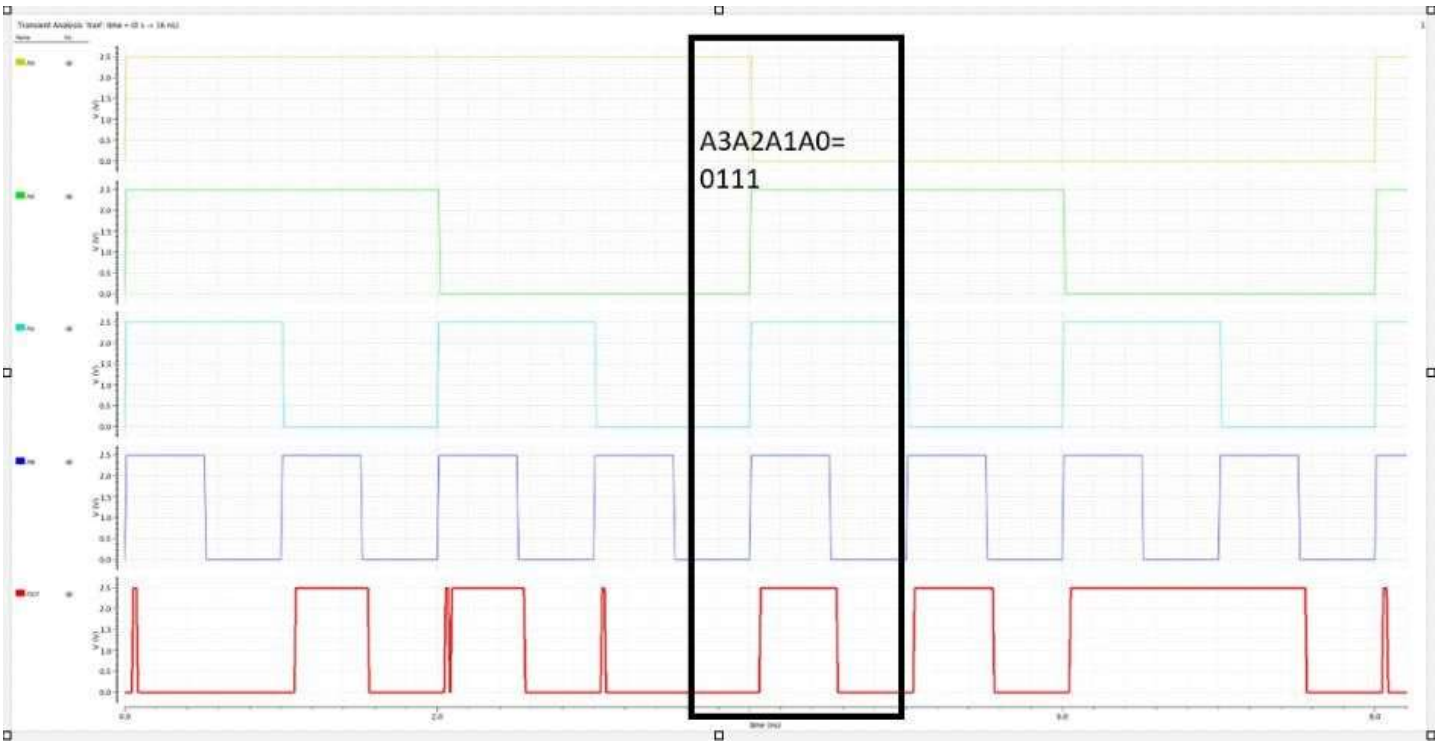
As we can see that in the screenshot when A3, A2, A1, A0 = 1101 respectively as per the truth table it is 13. And at the output waveform we can see that it is in 1 position as 13 is considered as a prime number.



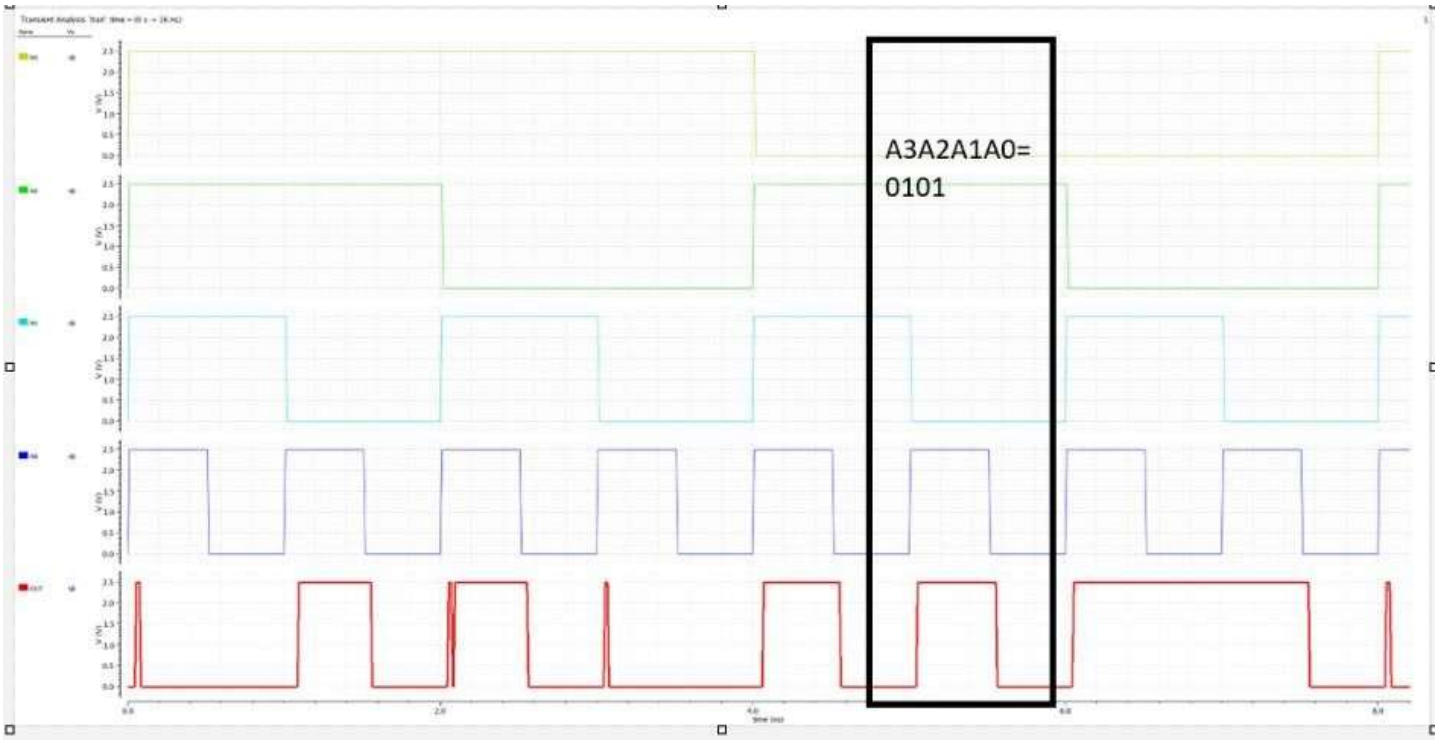
When the inputs A3, A2, A1, A0 = 1011 respectively as per the truth table it is 11. As 11 is a prime number at the output waveform it is highlighted.



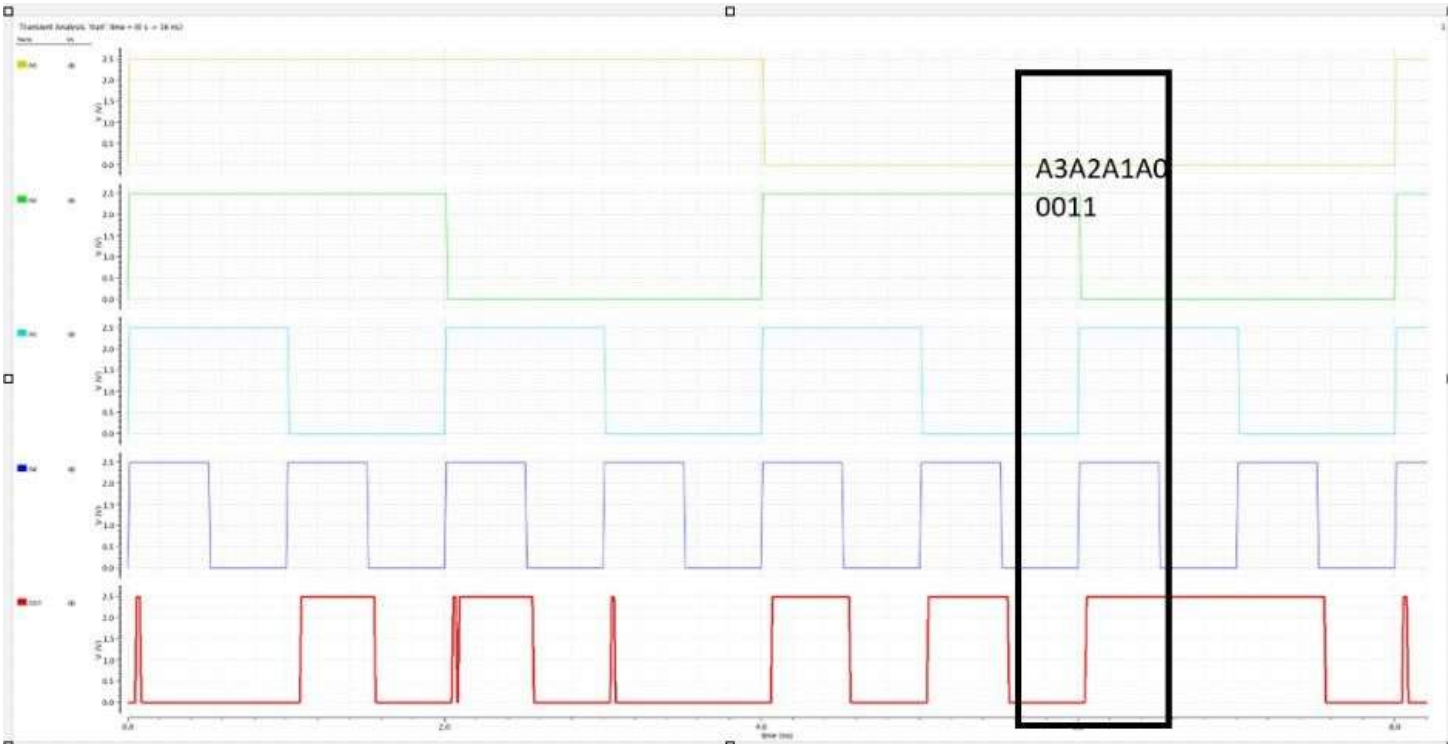
When the inputs A3, A2, A1, A0 = 0111 respectively as per the truth table it is 7. As 7 is a prime number at the output waveform it is highlighted.



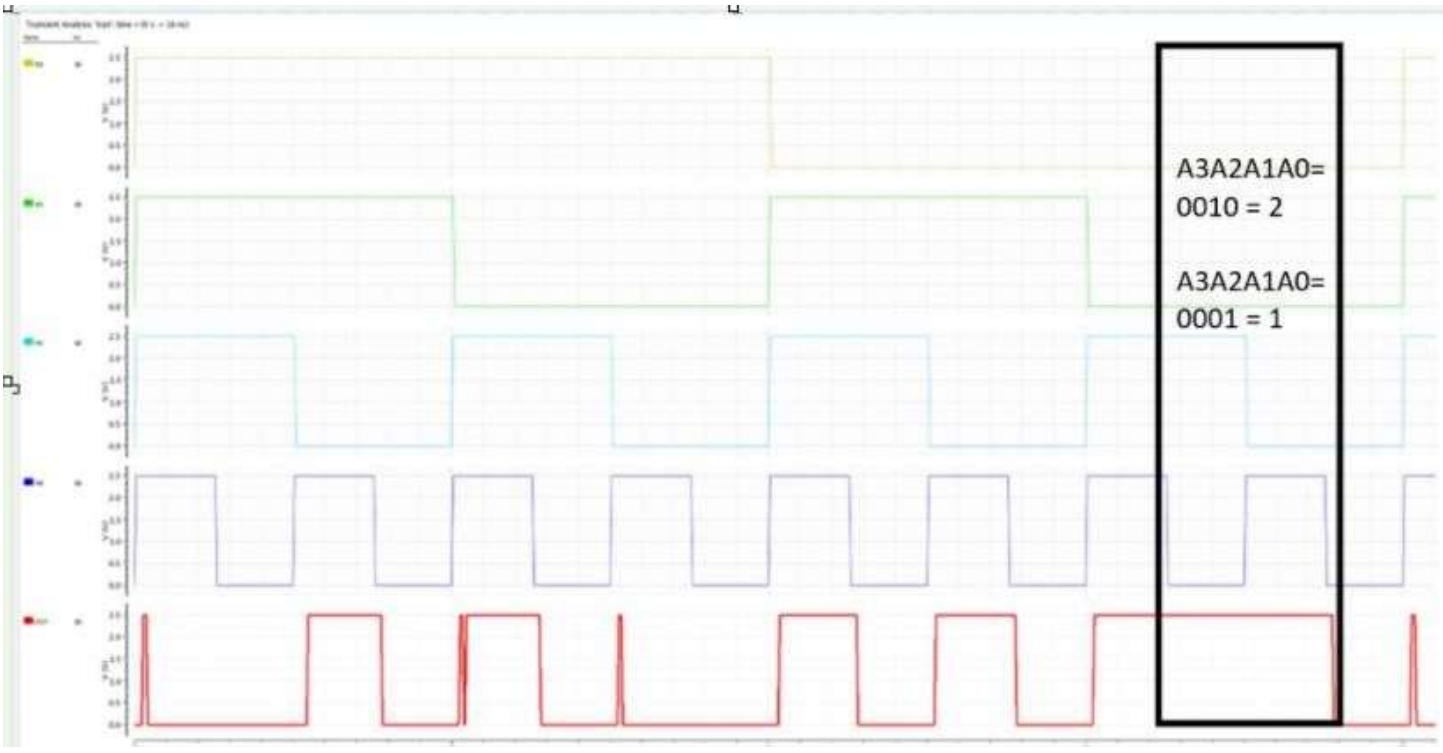
When the inputs A3, A2, A1, A0 = 0101 respectively as per the truth table it is 5. As 5 is a prime number at the output waveform it is highlighted.



When the inputs A3, A2, A1, A0 = 0011 respectively as per the truth table it is. As 3 is a prime number at the output waveform it is highlighted.



In this waveform, the 4-bit prime detector correctly highlighted prime numbers 2 and 1 in the output waveform.



So, in this way the 4-bit prime detector is working fine and it's highlighting the exact results.

Conclusion:

4 – Bit prime detector is successfully implemented in the cadence virtuoso tool and the results are appropriate and the output waveforms are correctly highlighted when there is a prime number. But if we use logic gates like NAND or NOR that might require fewer transistors. As a result, it can operate faster.