Vinay Omkumar Khilwani, 18BCP126
Dr. Paawan Sharma & Prof. Tolga Kaya
Digital Logic & Design
Pandit Deendayal Petro. Uni. (PDPU)
22 November 2019

Moore FSM for Signal Strength Indicator, Implemented in Logisim

Aim: To program the LCD display related to the signal-strength indicator on the cell-phone's screen.

Abstract: The objective of making this type of Finite State Machine (FSM) is to program the LCD display related to the signal-strength indicator on the cell-phone's screen. The Moore type of FSM has been implemented using three different methods, i.e. Multiplexers, Read-Only Memory (ROM) and Decoders with the use of additional gates, in Logisim.

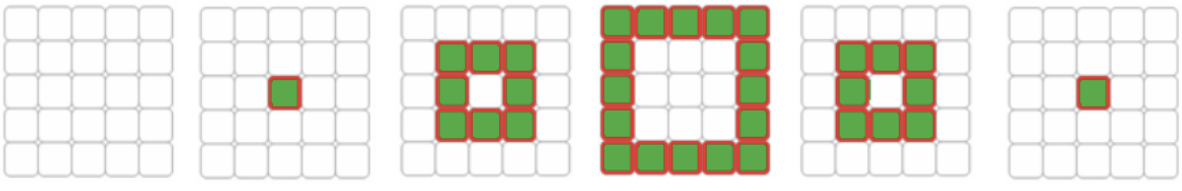
Assumptions:

1. The area on the LCD screen that displays the signal-strength bars, consists of a grid of 5x5 LCD pixels.

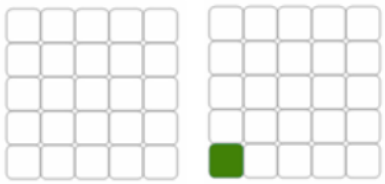
2. The FSM will have a (multi-bit) input (SS) , indicating Signal-Strength.
3. The range of valid values for is: **0 <= SS <= 100**
4. The height of towers increase in ascending manner, i.e. the maximum height of each tower (top-most switched "**ON**" cell of the Matrix), starting from left has its bit value as 0, 6, 12, 18 and 24 respectively.

Basic Functioning:

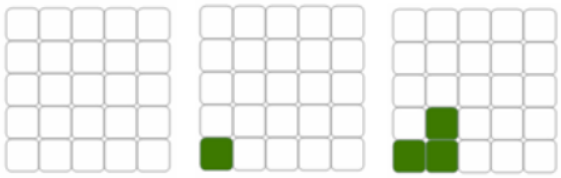
1. If **SS=0**, the FSM will display the following sequence of outputs, in a loop.



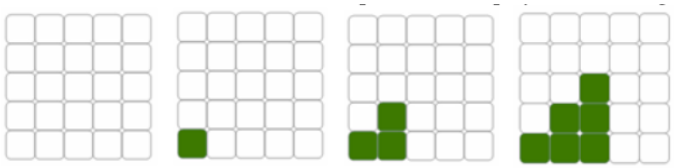
2. If **1 <= SS <= 20**, the FSM will display the following sequence of outputs, in a loop.



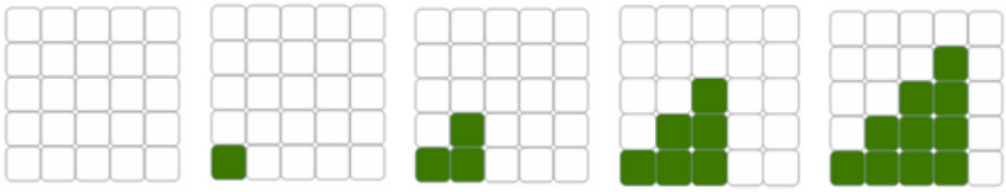
3. If **21 <= SS <= 40**, the FSM will display the following sequence of outputs, in a loop.



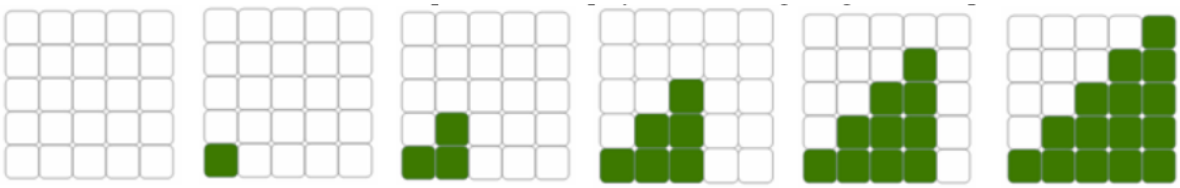
4. If **41 <= SS <= 60**, the FSM will display the following sequence of outputs, in a loop.



5. If **61 <= SS <= 80**, the FSM will display the following sequence of outputs, in a loop.



6. If **81 <= SS <= 100**, the FSM will display the following sequence of outputs, in a loop.



Timeline:

- **November 11-12, 2019:**

    1. Introduction to the Problem Statement
    2. Interaction with the SHU partner for this project - Charles Escott.
    3. Discussed about the topics and softwares we are accustomed to, and started planning accordingly for the project.

- **November 13-14, 2019:**

    1. Discussed and finalised the state Diagram.

- **November 15-16, 2019:**

    1. Started working out different approaches for implementation in Logisim.
    2. Completed the FSM Design in Logisim, using Decoder and Basic (OR & AND) Gates.

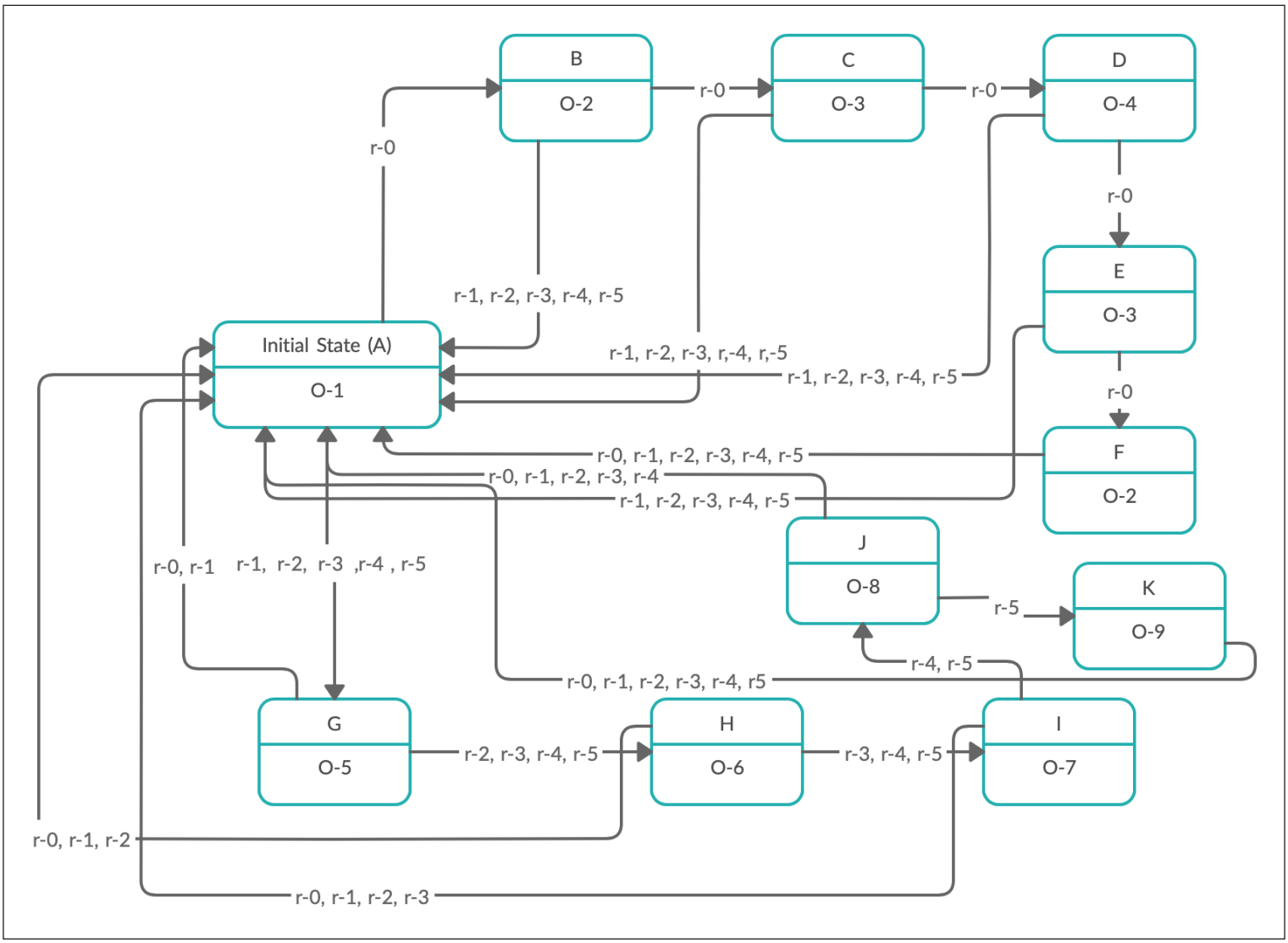
- **November 17-18-19-20, 2019:**

    1. Completed the FSM Design in Logisim, using Multiplexer.
    2. Completed the FSM Design in Logisim, using ROM.

- **November 21, 2019:**

1. Drafted the Final Report

State Diagram (of the FSM):



—> Ranges:

1. r-0: (SS=0)
2. r-1: (1 <= SS <=20)
3. r-2: (21 <= SS <=40)
4. r-3: (41 <= SS <=60)
5. r-4: (61 <= SS <=80)
6. r-5: (81 <= SS <=100)

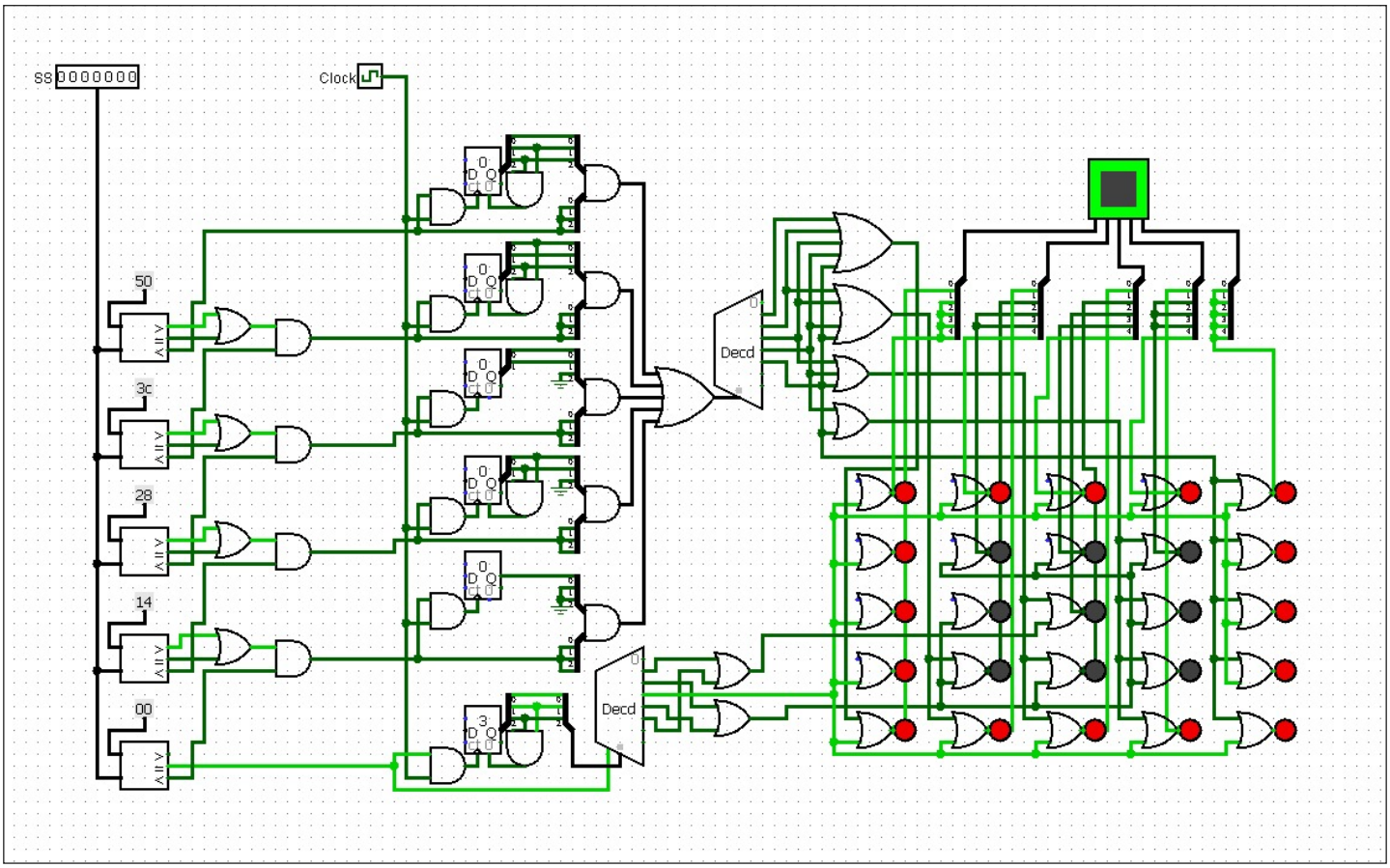
—> Outputs:

1. O-1: Blank LED Matrix (0 0000 0000 0000 0000 0000 0000)
2. O-2: Single Middle Dot (0 0000 0000 0001 0000 0000 0000)
3. O-3: Small Square (0 0000 0111 0010 1001 1100 0000)
4. O-4: Large Square (1 1111 1000 1100 0110 0011 1111)
5. O-5: 1 Signal Bar (0 0000 0000 0000 0000 0000 0001)
6. O-6: 2 Signal Bars (0 0000 0000 0000 0000 0110 0001)
7. O-7: 3 Signal Bars (0 0000 0000 0001 1100 0110 0001)
8. O-8: 4 Signal Bars (0 0000 0111 1001 1100 0110 0001)
9. O-9: 5 Signal Bars (1 1111 0111 1001 1100 0110 0001)

Logisim Implementations:

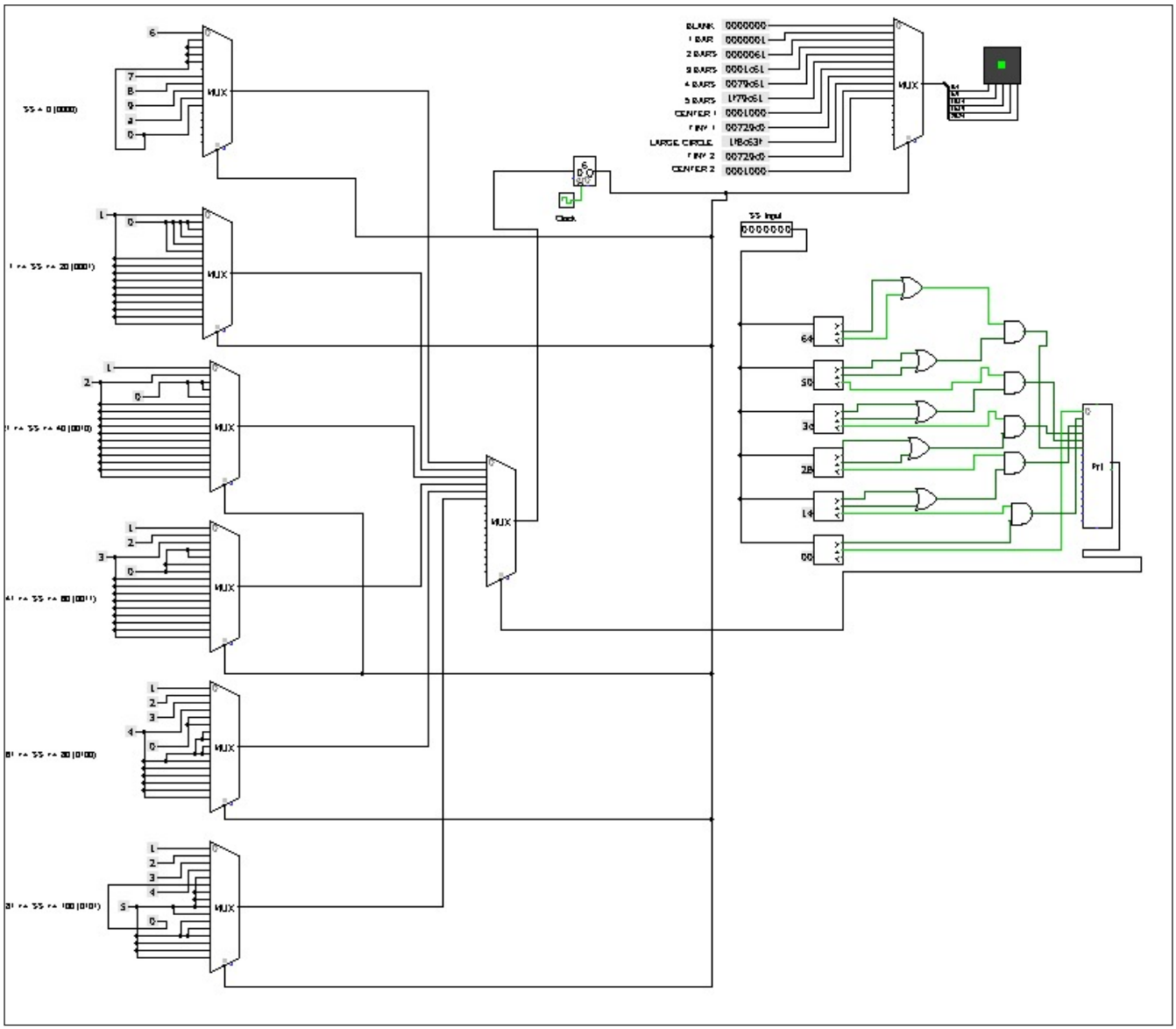
1. Using Decoder & Basic (OR & AND) Gates

- A 7-bit Binary Input (2-bit Decimal Input) is taken. This input is then compared with the comparators and then based on the conditions as stated in the Basic Functioning, the output lines are directed to the counters. After that, the input lines for the decoder are obtained from the different combinations of inputs by the splitter from the values of the comparators. The output lines are then directed to the combinations of OR gates, which are are so as to guide the inputs to the splitter based on the 5x5 LED Matrix values of (0,4) for each of the 5 columns. We have a clock, on the negative edge of which the loop of the output LED Matrix continues.



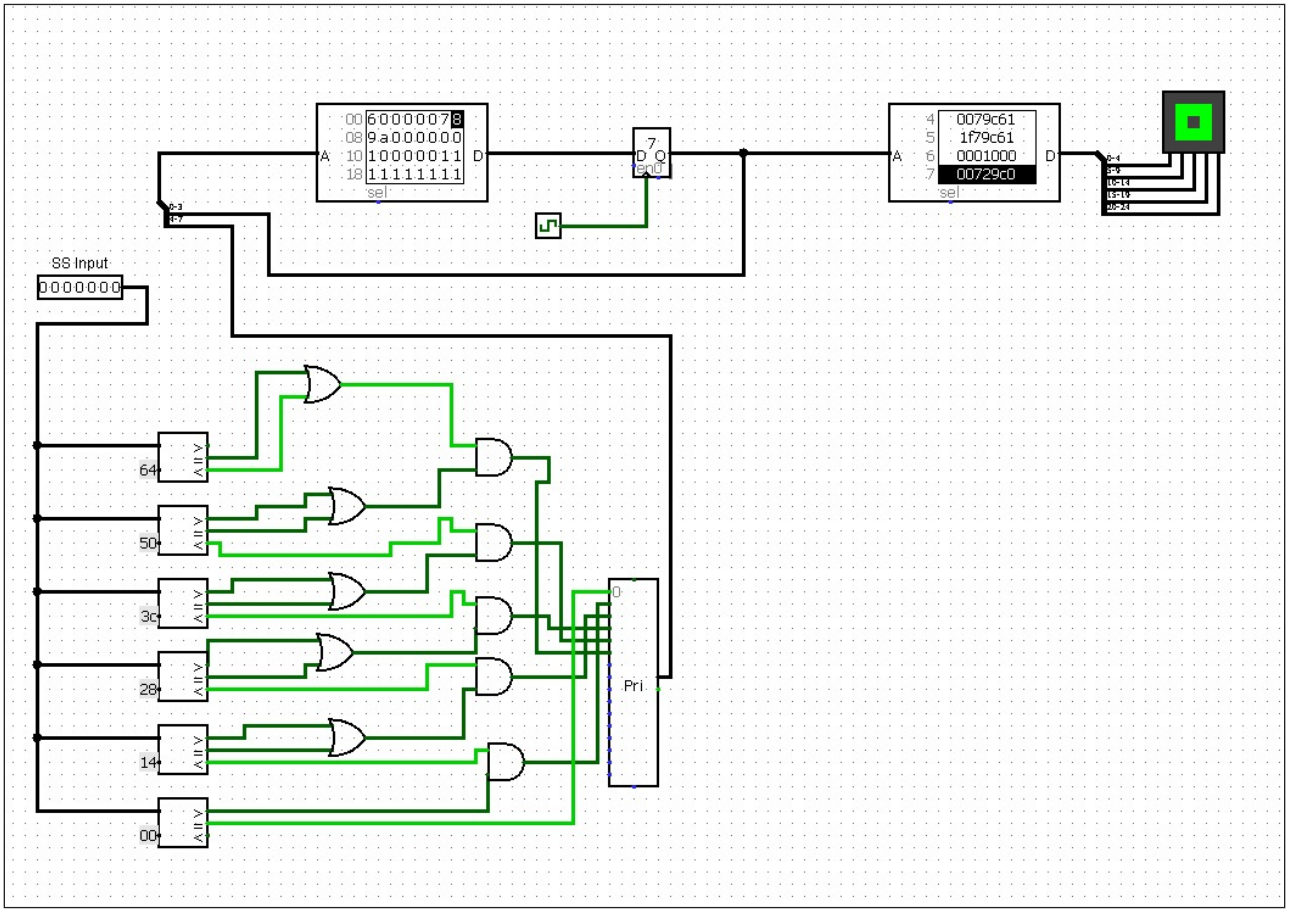
2. Using Multiplexer

- A 7-bit Binary Input (2-bit Decimal Input) is taken. This input is then guided to Multiplexer where the input lines have already come based on Basic Functioning constraints based on the conditions as stated in the Basic Functioning by combinations of numbers from (0,5) and (6,a). Its output line is directed to the counter, with which the clock is attached. After that, based on the hexadecimal values from (0001,1000), the 5X5 LED screen gets its output for the locations of (0,24).



3. Using ROM

- A 7-bit Binary Input (2-bit Decimal Input) is taken. This input is then compared using the comparators, then based on the conditions as stated in the Basic Functioning by converting into Hexadecimal form, its output lines are directed to the combinations of input lines of Priority Encoder. After that, the input lines for the splitter for (0,3) and (4,7) are obtained from the previous OR gates. The two ROMs are programmed according to the hex values and a clock with counter, which is added in-between. The outputs are guided to the splitter based on the 5X5 matrix values of (0,4) to (20,24) for the 5 columns.



My Experience:

It has truly been an amazing experience for me. I enjoyed working with my partner, Charles. I feel privileged to have got such an amazing opportunity for being one of the representatives of PDPU for this collaborative project, for which I would like to thank Dr. Paawan Sharma and Dr. Mazad Zaveri. I would also like to thank Dr. Tolga Kaya and his bunch of students at Sacred Heart University for this collaboration. I look forward to more such amazing future endeavours. Thank you.

Vinay Omkumar Khilwani

18BCP126

Computer Engineering, SOT

Pandit Deendayal Petroleum University (PDPU).

Attachments:

1. .circ Files
2. Simulation Videos
3. Snapshots