# Acknowledgement

We are heartily thankful to our course instructor Md. Shahriar Hussain whose encouragement, supervision and support from the preliminary to the concluding level enabled us to develop an understanding of the subject.

We are thankful to our lab instructor Fatema Zahra for his updates and kind support in developing our project.

We are thankful to the group members and mates who have helped in selecting and collecting information about our project. We greatly appreciate their sincere cooperation.

Lastly, we offer our regards and blessings to all of those who supported us in any respect during the completion of the study.

Thanks to our creator, He blessed us all in this project without which it would not have been possible.

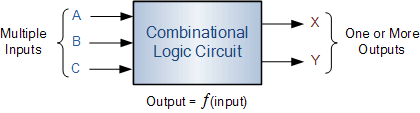
# Abstract

The purpose of this report was to know about the contents of CSE 231: Digital Logic Design. WE worked in groups consisting four members, so we got to do a team work. We implemented what we learnt in the theory part. From this project we got to face many unexpected problems and found ways to solve them. We got the chance to understand each step in designing a combinational as well as a sequential part. We could know why we do each step and which work leads to another thoroughly. From the project we also got to know about different types of ICs and their configurations. We could compare them and choose the better one to serve the purpose. We had to calculate the budget of the project. So, doing so helped us in learning to work within budget and made us think about efficient ways to implement a design with minimum cost. We, all the group members tried to contribute to the project as much as we could and tried to use all the knowledge we earned from our course.

# Introduction:

The purpose of this project was to understand and implement the basic concept of digital logic design and work as a team. We were assigned to design and implement a project with a combinational and sequential circuit which will display some specific letters or digits in a seven-segment display.

Combinational circuit is a circuit in which we combine the different gates in the circuit, for example encoder, decoder, multiplexer and demultiplexer. Some of the characteristics of combinational circuits are following -

* The output of combinational circuit at any instant of time, depends only on the levels present at input terminals.
* The combinational circuit do not use any memory. The previous state of input does not have any effect on the present state of the circuit.
* A combinational circuit can have an n number of inputs and m number of outputs.

***Figure 1: Block diagram of combinational circuit***

1. **Specifications & Design Tools:**

To implement the combinational part, we could use basic gates like AND, OR, X-OR ICs. We could use universal gates such as, NOT, NAND, NOR gates. We could use Encoders or Decoders. We could also use Multiplexers (MUX) for example, 16:1 or 8:1. We had to design for all of the mentioned methods but we implemented our design only using 8:1 multiplexers (MUX).

We decided to implement using the following design according to the cost for each implementation. Minimum cost implementation was the goal of our project.

For each simulation of the designs Logisim-win-2.7.1 was used.

The list of equipment’s we needed to implement the project is given below -

* IC 74151 - 8:1 Multiplexers
* Connecting Wires
* Bread board
* 9 Volts Battery
* Resistors
* Breadboard power supply stick (Step Down)

1. **Design Methodology:**

In this section, the steps taken to design the circuits are explained. The truth tables, circuit pictures, state diagram and state tables are attached. Equations for each part is mentioned.

1. **Truth Table:**

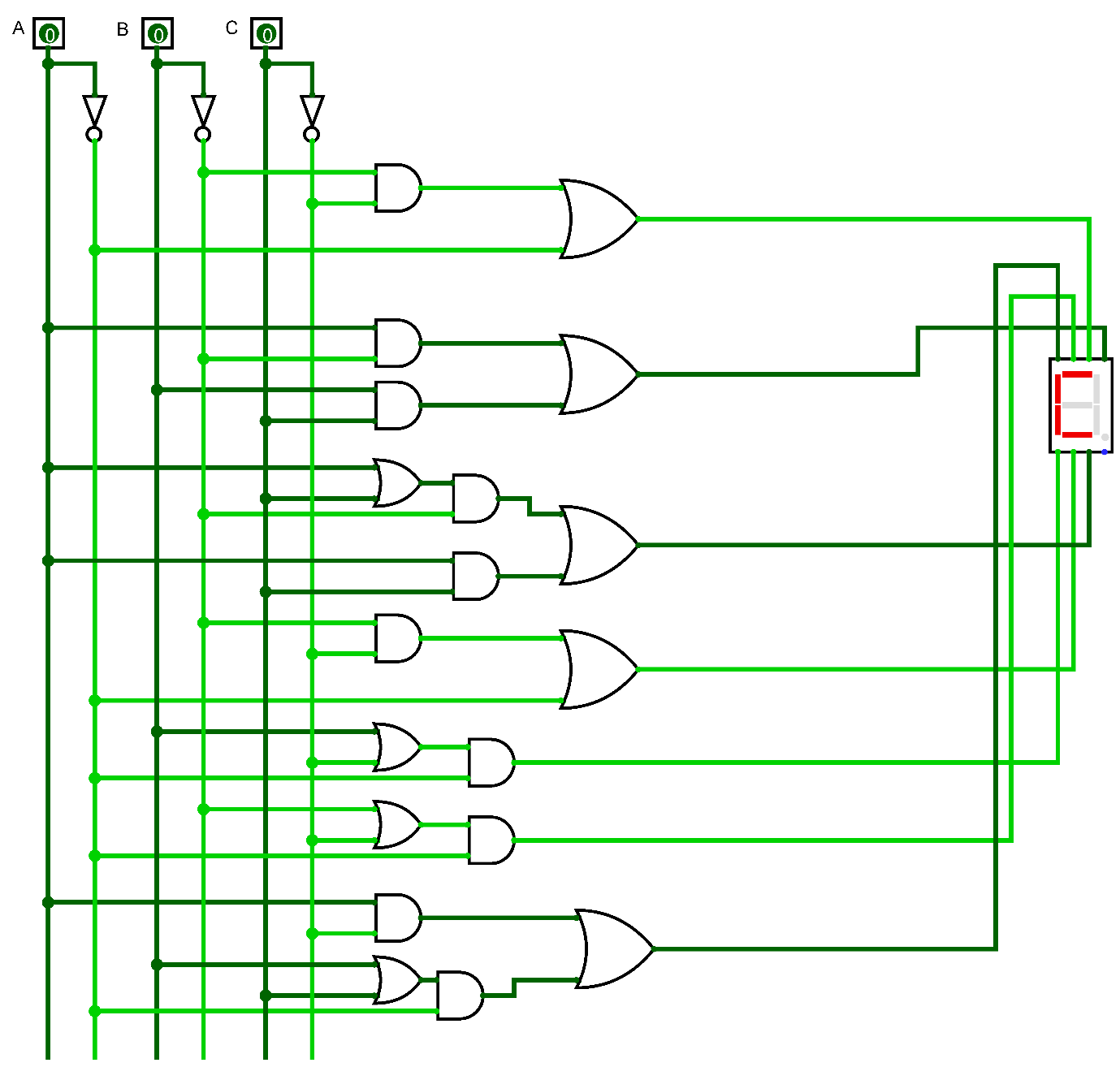
In our project, switching on the circuit results in the string of characters “CSE231-1” being displayed on the seven-segment display. Only one display can be use. To solve the circuit, at first, we derived a truth table in which

A binary value is assigned to express each of the characters. The table is following –

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | a | b | c | d | e | f | g |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |

1. **Combinational (Basic gates):**

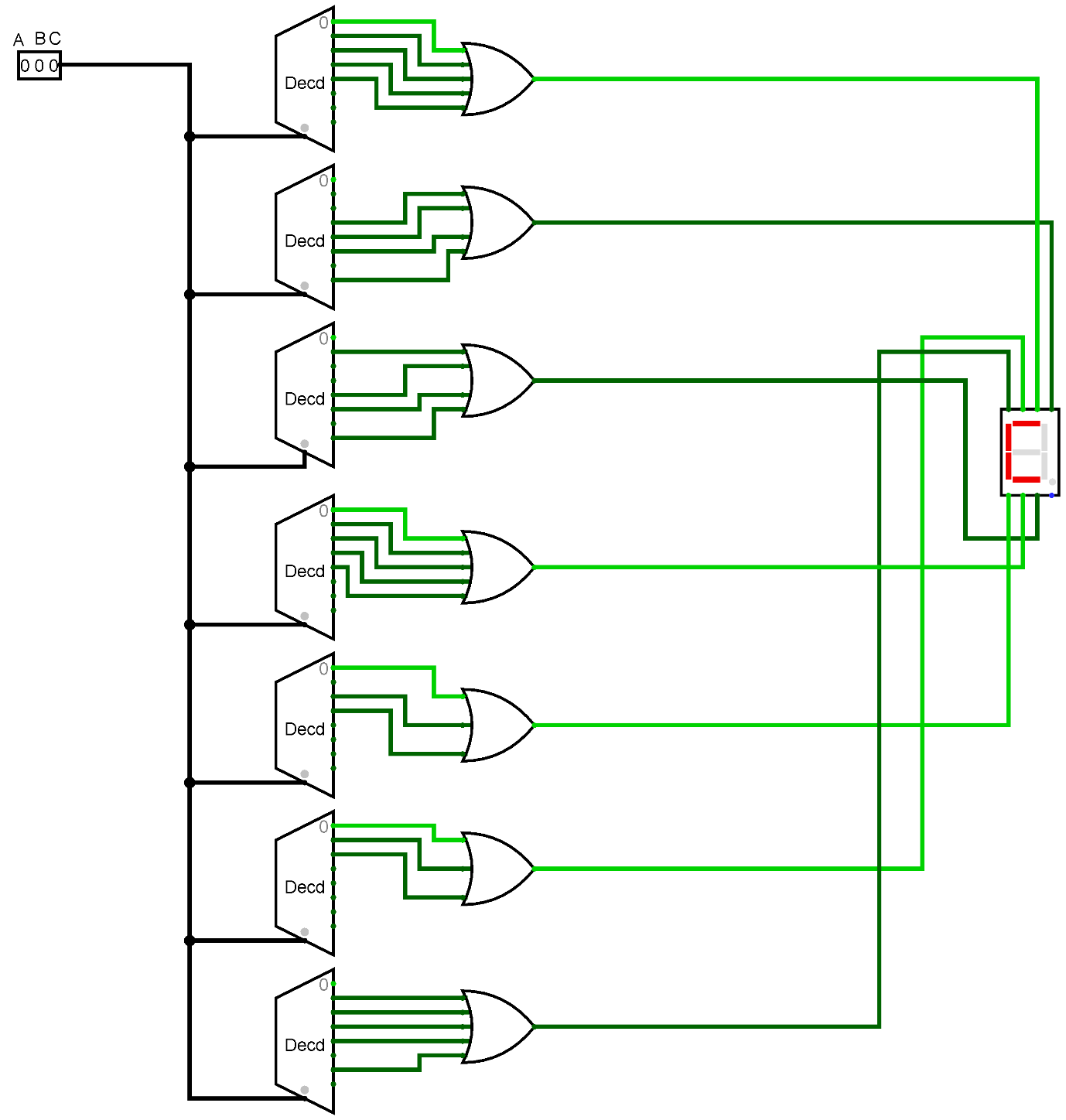
Logic gates are the basic building blocks of any digital system. It is an electronic circuit having one or more than one input and only one output. The relationship between the input and the output is based on a certain logic. Based on this, logic gates are named as AND gate, OR gate, NOT gate etc.



***Figure: Logisim of Combinational part using Basic gates.***

1. **Combinational (Decoder):**

A binary decoder is a combinational logic circuit that converts binary information from the n coded inputs to a maximum of 2n unique outputs.



***Figure: Logisim of Combinational part using Decoder.***

MUX 1:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
| Output | 1 | 1 | D’ | 0 | 0 | 0 | 0 | 0 |

MUX 2:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
| Output | 0 | D | 1 | D | 0 | 0 | 0 | 0 |

MUX 3:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
|  | D | 0 | 1 | D | 0 | 0 | 0 | 0 |

MUX 4:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
|  | 1 | 1 | D’ | 0 | 0 | 0 | 0 | 0 |

MUX 5:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
|  | D’ | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

MUX 6:

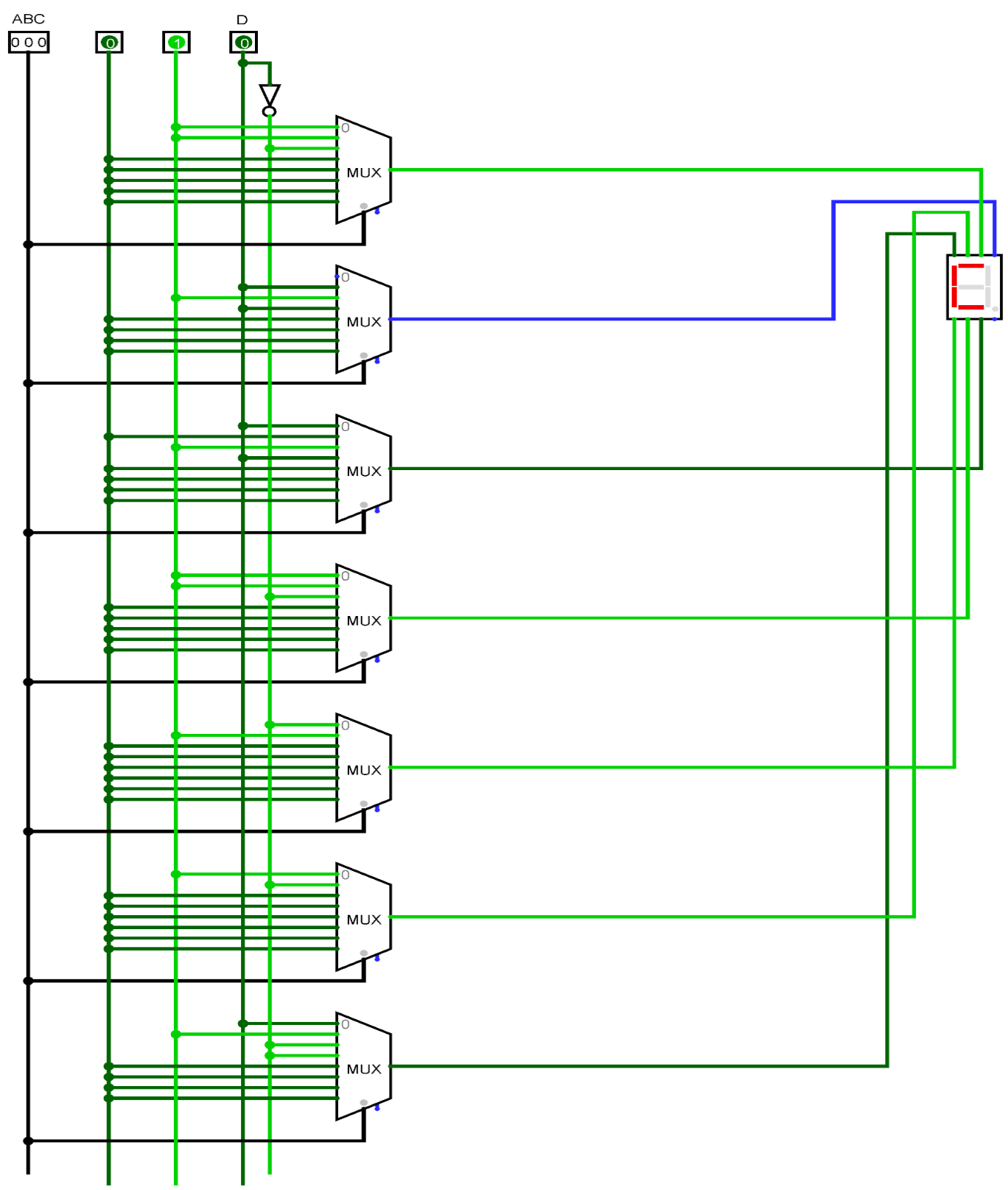
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
|  | 1 | D’ | 0 | 0 | 0 | 0 | 0 | 0 |

MUX 7:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 |
| D | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
| D’ | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
|  | D | 1 | D’ | D’ | 0 | 0 | 0 | 0 |

1. **Combinational (Multiplexer):**

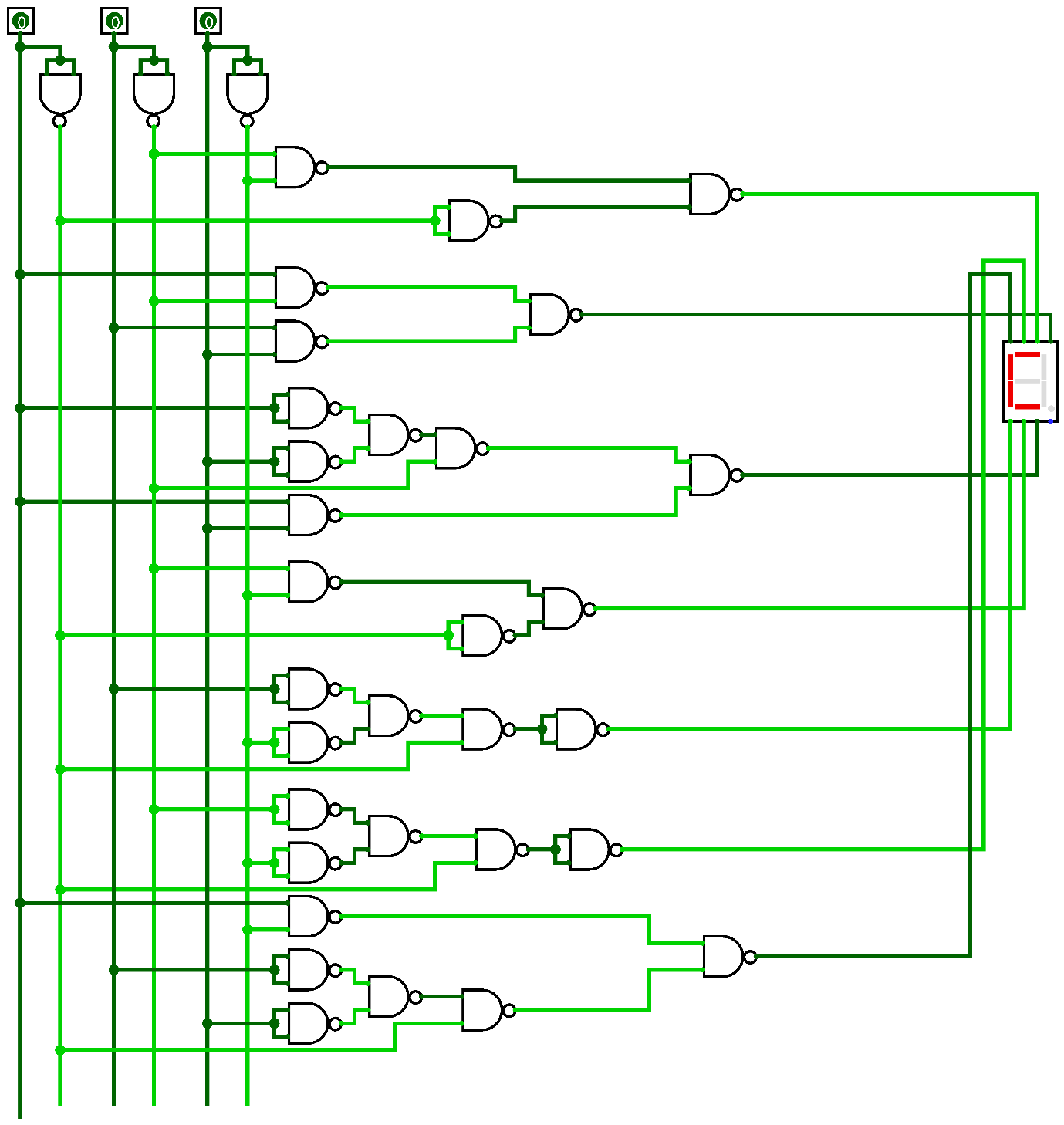
A multiplexer (or mux) is a device that selects between several analog or digital input signals and forwards it to a single output line. A multiplexer of inputs has selected lines, which are used to select which input line to send to the output.



***Figure: Logisim of Combinational part using MUX***

1. **Universal 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.



***Figure: Logisim of Combinational part using Universal gate***

1. **Sequential (JK Flip Flop):**

The flip flop is a basic building block of sequential logic circuits. It is a circuit that has two stable states and can store one bit of state information. Here we are going to use JK flip flop for our sequential part. We Use counter and JK-Flipflop to solve sequential part.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B(t) | C(t) | D(t) | B(t+1) | C(t+1) | D(t+1) | JB | KB | JC | KC | JD | KD |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | X | 0 | x | 1 | x |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | X | 1 | X | X | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | X | X | 0 | 1 | X |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | X | X | 1 | X | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | x | 0 | 0 | X | 1 | X |
| 1 | 0 | 1 | 1 | 1 | 0 | X | 0 | 1 | X | X | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | X | 0 | X | 0 | 1 | X |
| 1 | 1 | 1 | 0 | 0 | 0 | X | 1 | x | 1 | x | 1 |

***Truth table for sequential circuit using counter and JK Flipflop.***

After completing the truth table, we calculated the K-Map to built an equation and from there we built this circuit diagram for JK-Flipflop.

For JB:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 0 | 1 | 0 |
| X | X | X | X |

**JB = C(t) D(t)**

For kB:

|  |  |  |  |
| --- | --- | --- | --- |
| X | X | X | X |
| 0 | 0 | 1 | 0 |

**kB = C(t) D(t)**  
For JC:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | X | X |
| 0 | 1 | X | X |

**JC = D(t)**  
For KC:

|  |  |  |  |
| --- | --- | --- | --- |
| X | X | 1 | 0 |
| X | X | 1 | 0 |

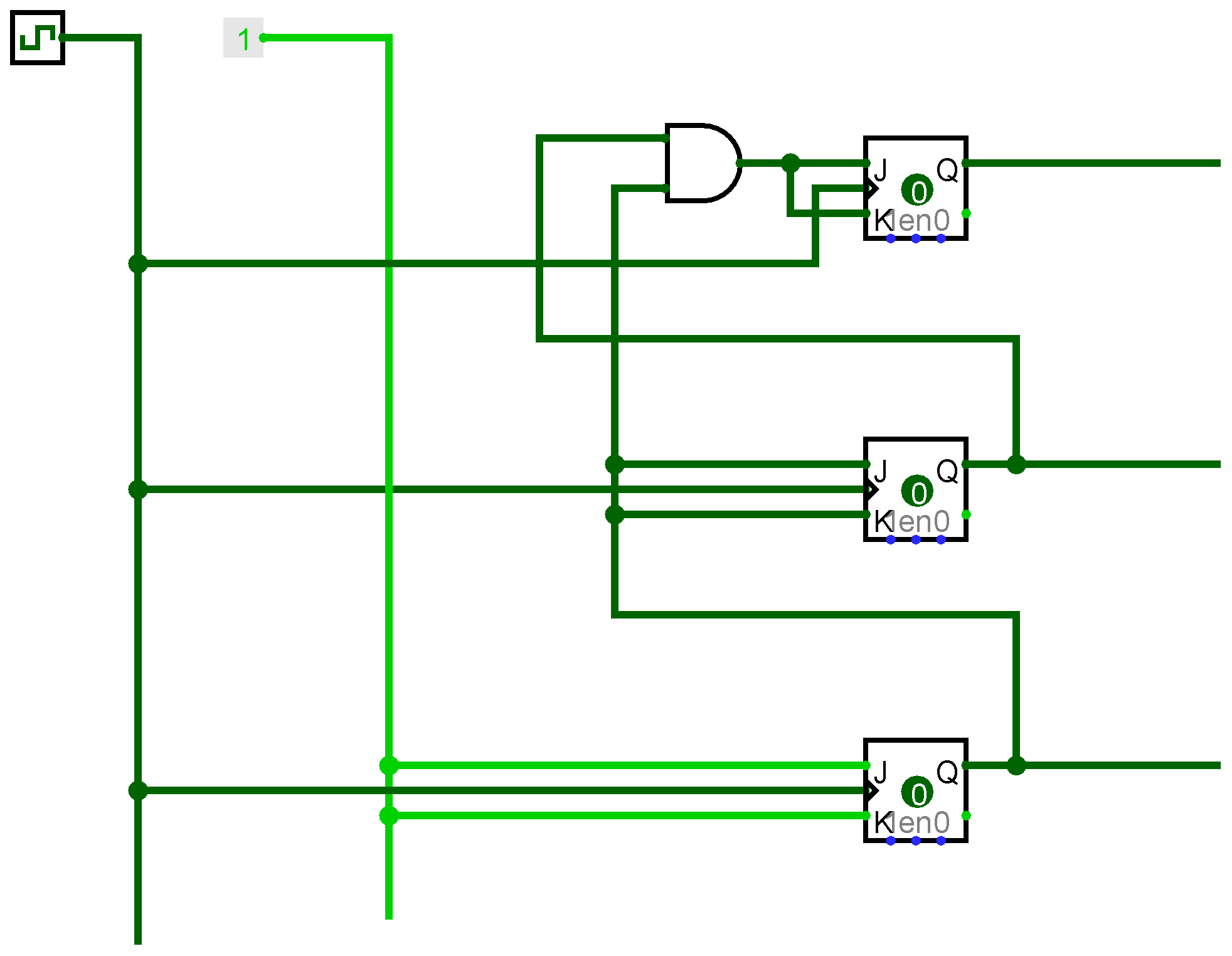
**KC = D(t)**  
For JD:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | X | X | 1 |
| 1 | X | X | 1 |

**JD = 1**  
For JD:

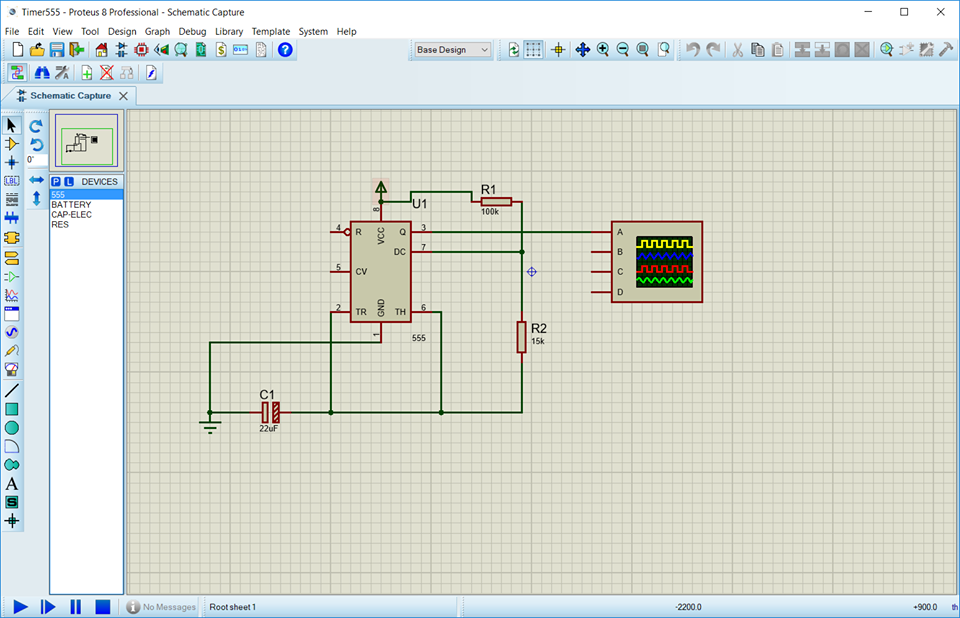
|  |  |  |  |
| --- | --- | --- | --- |
| X | 1 | 1 | X |
| X | 1 | 1 | X |

**JD = 1**



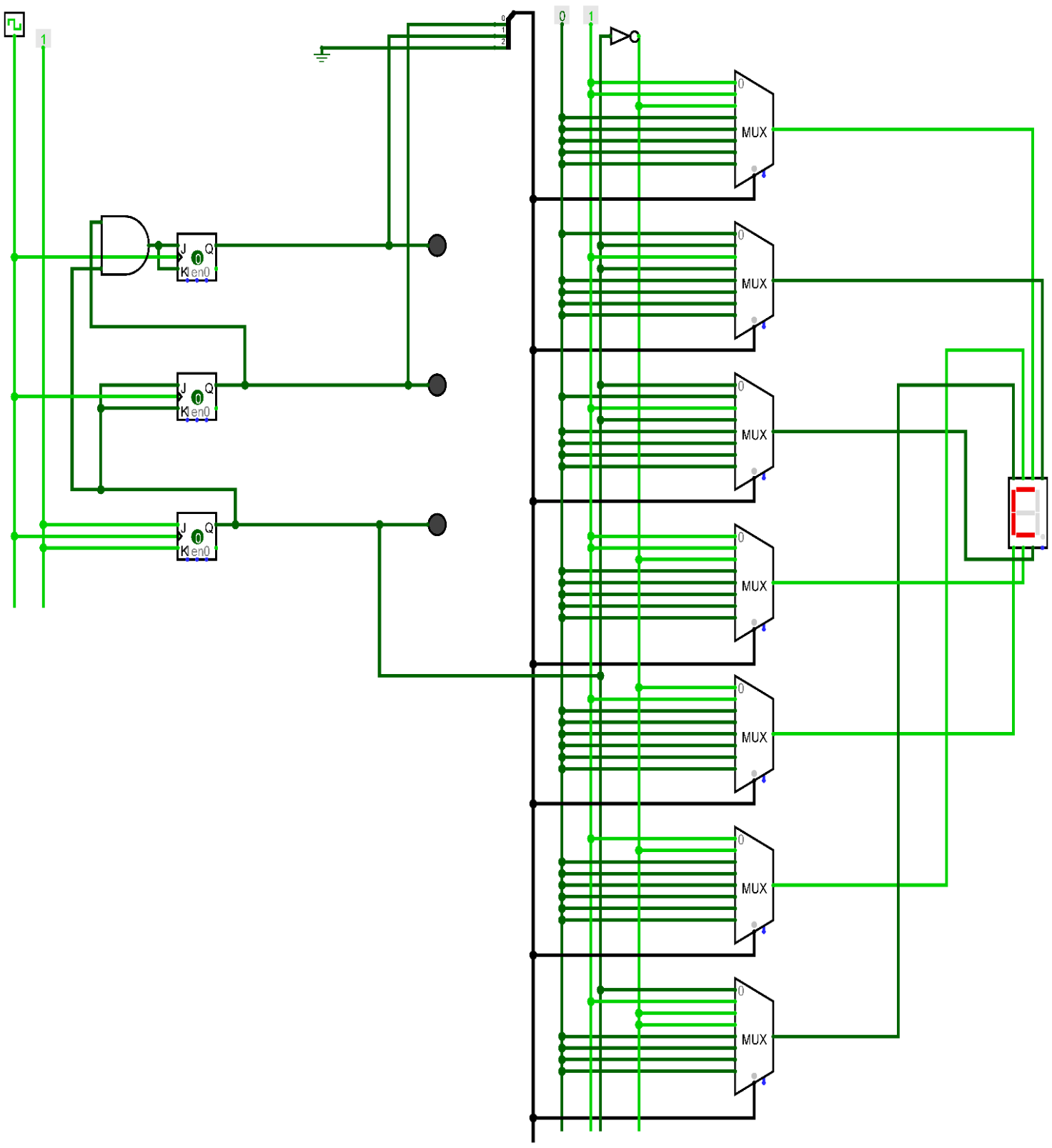
***Figure: Logisim of Sequential part using JK Flipflop***

1. **Clock:** We uses the following circuit to make clock pulse.



1. **Sequential Circuit:**

After adding the clock and the flip-flop with the combinational part, we got a whole sequential circuit.



***Figure: Logisim of combinational with Sequential part using JK Flipflop & Mux***