LAB REPORT : 8

NAME: ROMICA RAISINGHANI

ROLL NUMBER: 2021101053

GROUP: 1

PART A:

AIM OF THE EXPERIMENT:

To design and simulate a circuit that describes the functioning of the tristate buffer and to get familiar with the working of a tristate buffer and understand data flow control using a tristate buffer.

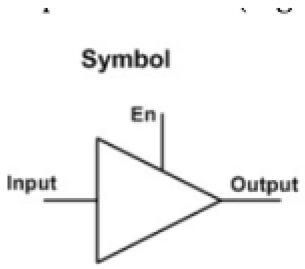


Fig. 1 Tri State Buffer

ELECTRONIC COMPONENTS USED:

1. Arduino Uno R3
2. 1 Quad AND Gate

3. 1 pMOS transistor
4. 1 nMOS transistor
5. 1 orange LED
6. 1 blue LED
7. 1 white LED
8. 1 k ohm resistor
9. 3 200 ohm resistors
10. A bunch of connecting wires

PART B:

AIM OF THE EXPERIMENT:

To design and simulate a circuit that describes . In this experiment, we verify the operation of copying data from one register to another and show the same as output by taking an input number (0-15) and send it to the first register then we write code to enable the tristate buffers in order and correspondingly apply clock pulses to the second shift register such that the content from the first register is transferred.

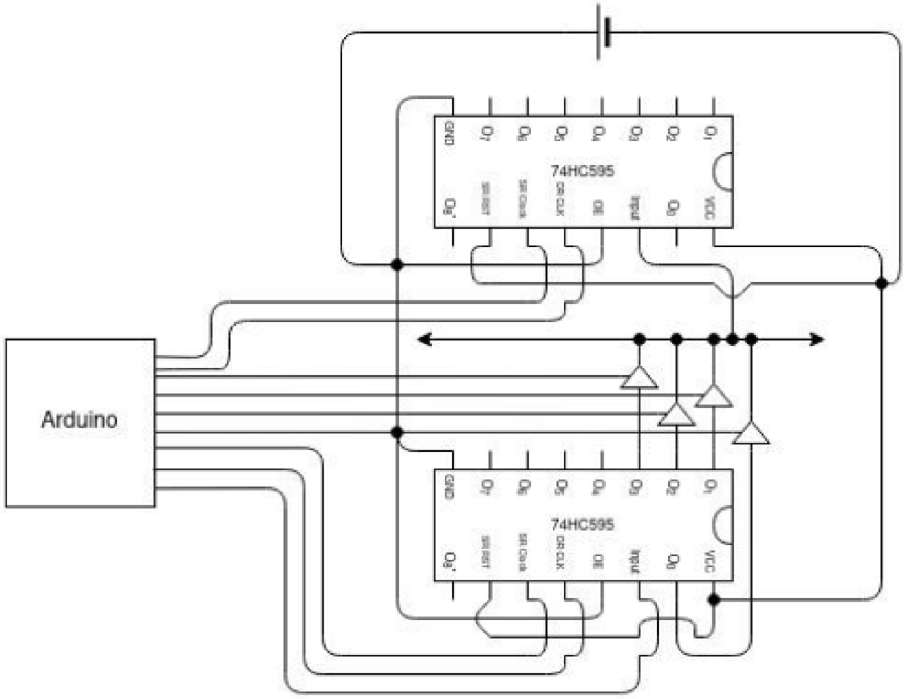


Fig. 4 Circuit Diagram

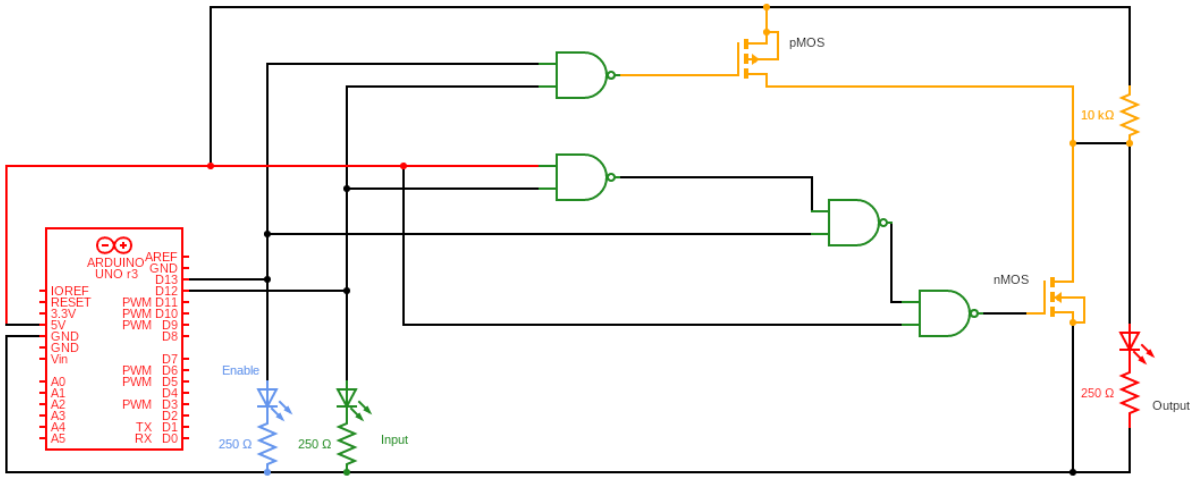
ELECTRONIC COMPONENTS USED:

1. Arduino Uno R3
2. 4 Quad AND Gate
3. 4 pMOS transistors
4. 4 nMOS transistors
5. 8 1k ohm resistors
6. 9 200ohm resistors
7. 4 8k ohm resistors
8. 4 250ohm resistors
9. 2 8-bit shift registers
10. 1 yellow LED
11. 8 blue LEDs

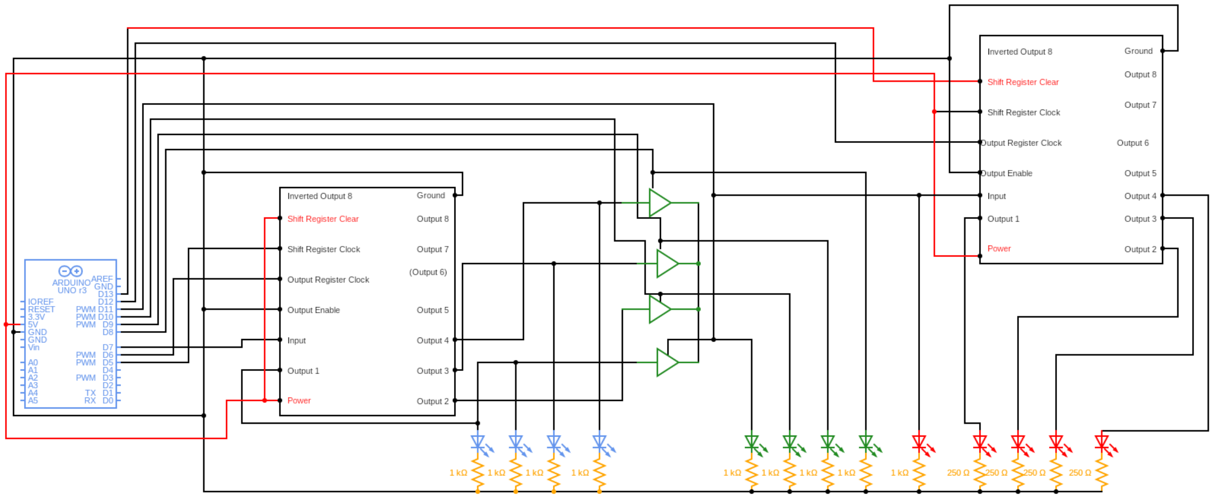
12. 4 white LEDs
13. 8 orange LEDs
14. A bunch of connecting wires

REFERENCE CIRCUITS:

PART A: CIRCUIT DIAGRAM REPRESENTING A TRISTATE BUFFER



PART B: CIRCUIT DIAGRAM TO SHOW DATA FLOW BETWEEN TWO REGISTERS USING TRISTATE BUFFERS



PROCEDURE:

PART A:

1. Provide all the live wire and ground connections from the Arduino to all the breadboards first.

2. Take a wire from the Pin 13,Pin 12of the Arduino and connect it to the anode of Enable LED & Input LED respectively.

3. Take one end of the resistor and connect it to the cathode of the Enable LED & Input LED and ground the other ends.

4. Take a wire from the Pin 12 of the Arduino and connect it as Input 1A into the Quad NAND gate and then connect the output 1as input 2A into the quad NAND gate.

5. Take a wire from the Pin 13 of the Arduino and connect it as input 2B into the Quad NAND Gate and connect output 2 as Input 3A into the Quad NAND Gate.

6. Connect the live wire as input 3B into the Quad NAND and connect output 3 into the gate input of the nMOS.

7. Take a wire from the Pin 13 of the Arduino and connect it as Input 4B into the Quad NAND gate then take a wire from Pin 12 of the Arduino and connect it as Input 4A into the Quad NAND gate and connect the output 4 as gate input into the pMOS.

8. Connect the live wire into the source input of the nMOS.

9. Take a wire from the DRAIN input of the pMOS and connect it to the output of the Output LED.
10. Take one end of the resistor and connect it to the cathode of the Output LED and ground the other end.
11. Take one end of the resistor(10k ohms) and connect it to the cathode of the Output LED and connect the other end to the live wire.
12. Feed an approriate code into the Arduino to make the above circuit work and make sure all the connections are tight by crosschecking the wirings from the given circuit and make the necessary changes.

PART B:

1 . Provide all the live wire and ground connections from the Arduino to all the breadboards first.
2 .Take a wire from pin 7 and connect it as Input into the 8-bit shift register(1).
3 . Take a wire from pin 6 and connect it as Output Register Clock PIn into the 8-bit shift register(1).
4. Take a wire from pin 5 and connect it as Shift Register Clock Pin into the 8-bit shift register(1).
5. Connect the Power and Shift Register Clear Pin to the live wire and the Ground and Output Enable Pin to the ground wire.
7.Take a wire from Output 1,Output 2,Output 3,Output 4 and connect it to the anode of LED A,B,C,D respectively.
8. Take the 4 resistors of 1k ohm each and connect them respectively to the cathode of the respective LEDs and ground the other ends of the resistors.
9. Make 4 buffers taking reference from the above part i.e part A.

10. Take a wire from pin 11,pin 10,pin 9 and pin 8 and connect it as Enable and output 1,output 2,output3 and output 4 as the Input into the Buffer 1,Buffer 2 ,Buffer 3 and Buffer 4 respectively.
11. Take wires from all the outputs from all the buffers and connect them as INPUT into the 8-bit shift register(2).
12. Take wires from all the outputs from all the buffers and connect them to the anode of the LED.

13. Take one end of the resistor and connect it to the cathode of the LED and ground the other end.

14. Take a wire from pin 12 and connect it as Output Register Clock into the 8-bit shift register(2).

15. Take a wire from pin 13 and connect it as Shift Register Clock into the 8-bit shift register(2).

16. Take a wire from Output 1,2,3,4 and connect it to the anode of LED E,F,G and H respectively.

17. Connect the Power and Shift Register Clock Pin to the live wire and the ground and Output Enable Pin to the ground wire.

18. Take the 4 resistors and connect them respectively to the cathode of the respective 4 LEDs.

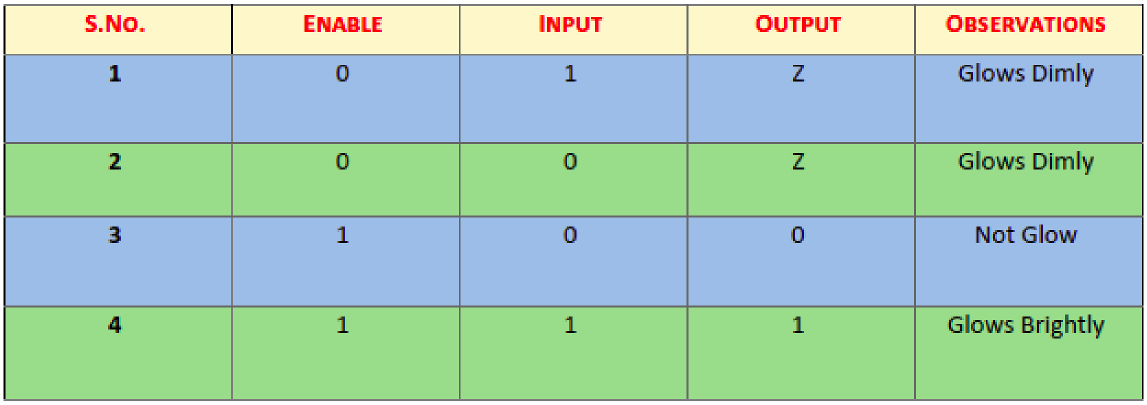
19. Feed an approriate code into the Arduino and make sure all the connections are tight by crosschecking the wirings from the given circuit and make the necessary changes.

CONCLUSION:

PART A:

We have successfully designed and simulated the circuit of a tristate buffer.The tri-state buffer functions just as a regular digital buffer where the value at its input is propagated to its output. But it has an additional capability that allows us to configure its output to a Hi-Z (high impedance) state.

The observation table for the tristate buffer is :



S.No.	Enable	Input	Output	Observations
1	0	1	Z	Glows Dimly
2	0	0	Z	Glows Dimly
3	1	0	0	Not Glow
4	1	1	1	Glows Brightly

PART B:

In the experiment,we have successfully designed and simulate the circuit and also verified the working of a tristate buffer and similar to the example given, use the tri state buffers to transfer contents of one shift register (74HC595 IC) to another shift register. Note that the output of the shift register is parallel while the input is serial.Therefore, we would be connecting the outputs to a single data bus using tristate buffers and reading the input to the second register from that bus.

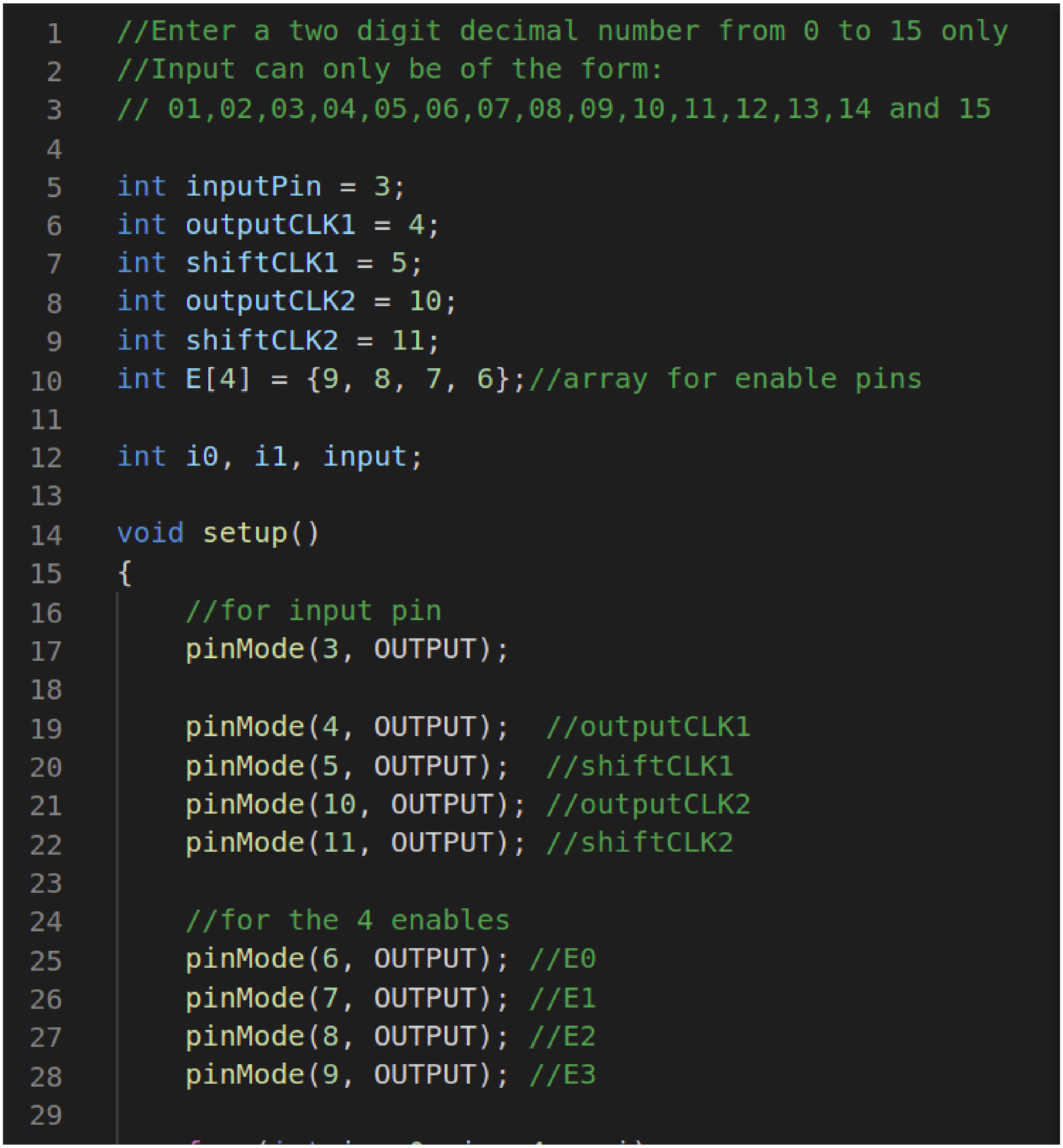
The code used for this simulation is embedded below :

NOTE:

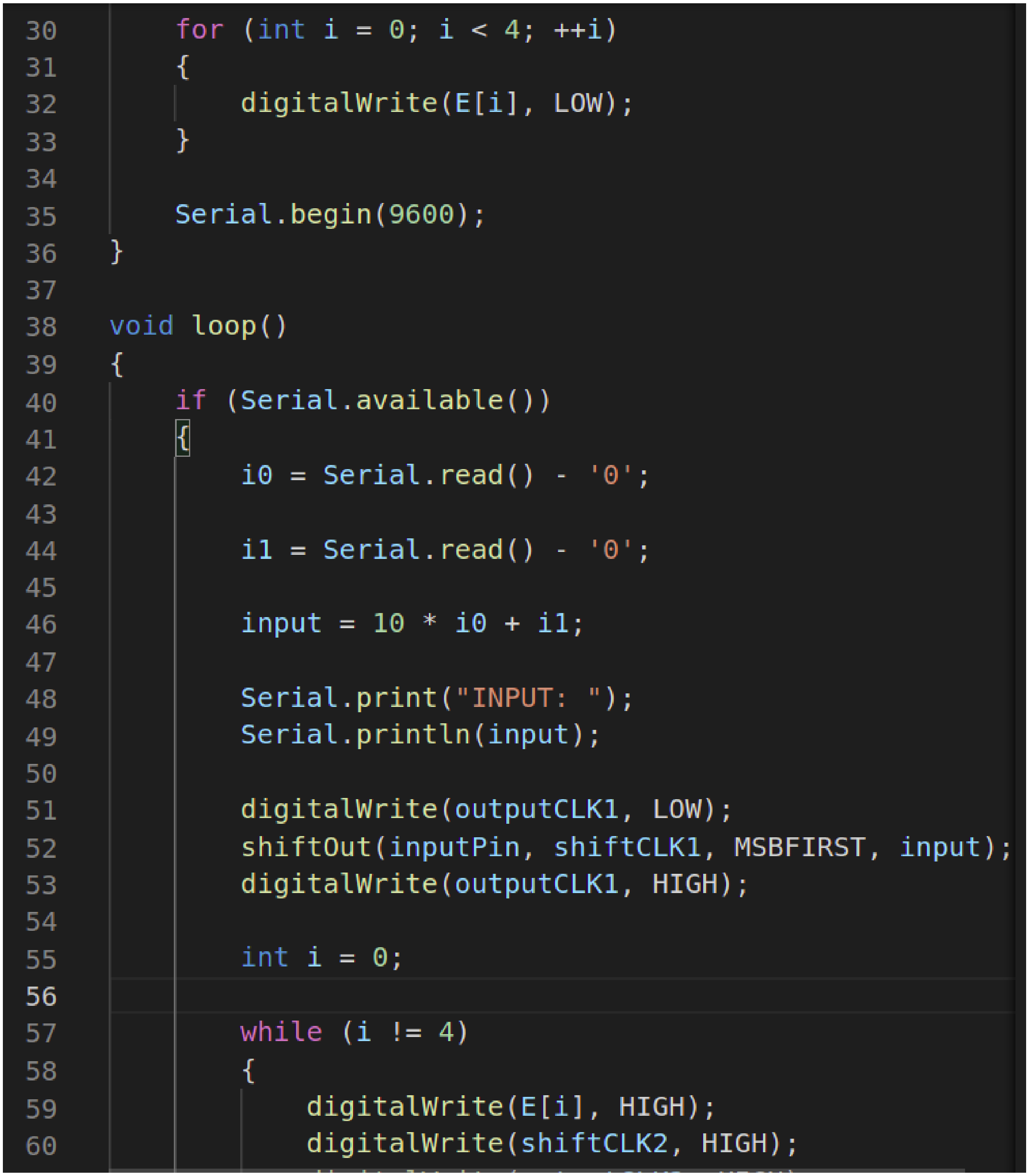
//Enter a two digit decimal number from 0 to 15 only

//Input can only be of the form:

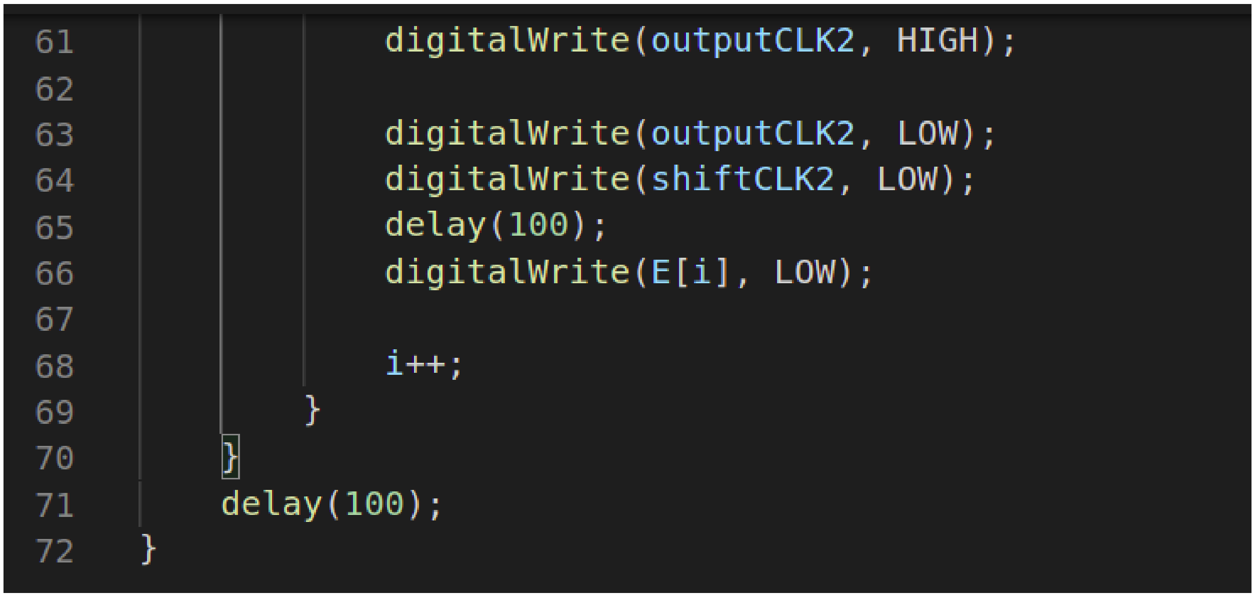
// 01,02,03,04,05,06,07,08,09,10,11,12,13,14 and 15



```
//Enter a two digit decimal number from 0 to 15 only
//Input can only be of the form:
// 01,02,03,04,05,06,07,08,09,10,11,12,13,14 and 15

int inputPin = 3;
int outputCLK1 = 4;
int shiftCLK1 = 5;
int outputCLK2 = 10;
int shiftCLK2 = 11;
int E[4] = {9, 8, 7, 6};//array for enable pins

int i0, i1, input;

void setup()
{
    //for input pin
    pinMode(3, OUTPUT);

    pinMode(4, OUTPUT);  //outputCLK1
    pinMode(5, OUTPUT);  //shiftCLK1
    pinMode(10, OUTPUT); //outputCLK2
    pinMode(11, OUTPUT); //shiftCLK2

    //for the 4 enables
    pinMode(6, OUTPUT); //E0
    pinMode(7, OUTPUT); //E1
    pinMode(8, OUTPUT); //E2
    pinMode(9, OUTPUT); //E3

```



```
    for (int i = 0; i < 4; ++i)
    {
        digitalWrite(E[i], LOW);
    }

    Serial.begin(9600);
}

void loop()
{
    if (Serial.available())
    {
        i0 = Serial.read() - '0';

        i1 = Serial.read() - '0';

        input = 10 * i0 + i1;

        Serial.print("INPUT: ");
        Serial.println(input);

        digitalWrite(outputCLK1, LOW);
        shiftOut(inputPin, shiftCLK1, MSBFIRST, input);
        digitalWrite(outputCLK1, HIGH);

        int i = 0;

        while (i != 4)
        {
            digitalWrite(E[i], HIGH);
            digitalWrite(shiftCLK2, HIGH);
```



```
            digitalWrite(outputCLK2, HIGH);

            digitalWrite(outputCLK2, LOW);
            digitalWrite(shiftCLK2, LOW);
            delay(100);
            digitalWrite(E[i], LOW);

            i++;
        }
    }
    delay(100);
}
```

LINKS FOR TINKERCAD SIMULATION:

PART A:

Tinkercad | From mind to design in minutes

Tinkercad is a free, easy-to-use app for 3D design, electronics, and coding.

https://www.tinkercad.com/things/bWB2dPTT22u-lab8a/editel?sharecode=09kzfpkxJ6am4SUvGJ3ivLghIyK0XCtqQACK7eJK8nM

PART B:

Tinkercad | From mind to design in minutes

Tinkercad is a free, easy-to-use app for 3D design, electronics, and coding.

https://www.tinkercad.com/things/78rdFm5DKLv-lab8b/editel?sharecode=1qyTUV1XEWQmA935ymrOFZiIq2kKJrzQCQhsAlRm_nA

THANK YOU!