LAB REPORT: 3

Name: Arghya Roy

Roll Number: 2021115008

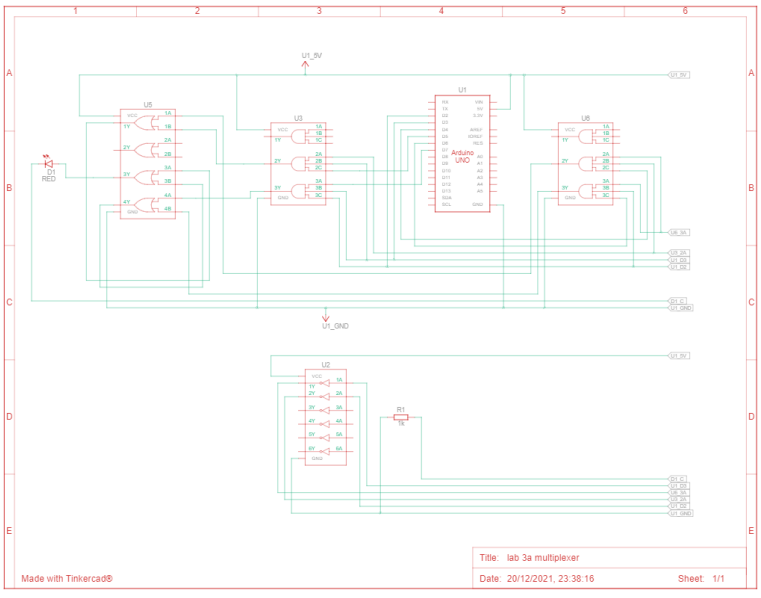
Group: 8

Part A: Designing 4:1 Multiplexer

Aim/Objective of the experiment: To design a 4:1 multiplexer using basic logic gates

Electronic components used: 1 Arduino board, one 1 kilo ohm resistors, 1 LED, 1 breadboard, 1 hex inverter(74HC04), 2 triple 3-input AND gates(74HC11), 1 quad OR gate(74HC32), wires

Reference Circuit:



Procedure:

1. The circuit is set up, as shown in the reference figure above, on the breadboard.
2. An Arduino code is written to give different combinations of inputs as input and select lines and view them using LED at the output line.
3. The multiplexer function is verified by tabulating the values of the output(s) for all input combinations.

   The code:

```
int s1,s0,i0,i1,i2,i3;

void setup()
{
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(7,OUTPUT);
  Serial.begin(9600);
}

void loop()
{

  if(Serial.available()>0)
  {
    s1=Serial.read();
    s1=s1-'0';

    digitalWrite(2,s1);
  }

  if(Serial.available()>0)
  {
    s0=Serial.read();
    s0=s0-'0';

    digitalWrite(3,s0);
  }

  if(Serial.available()>0)
```

```
  {
    i0=Serial.read();
    i0=i0-'0';

    digitalWrite(4,i0);
  }

  if(Serial.available()>0)
  {
    i1=Serial.read();
    i1=i1-'0';

    digitalWrite(5,i1);
  }

  if(Serial.available()>0)
  {
    i2=Serial.read();
    i2=i2-'0';

    digitalWrite(6,i2);
  }

  if(Serial.available()>0)
  {
    i3=Serial.read();
    i3=i3-'0';

    digitalWrite(7,i3);
  }
  delay(100);

}

/*

inputs verified:
001000
010100
100010
110001

*/
```

Conclusion:

S1	S0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

So, the multiplexer truth table is verified.

TinderCAD simulation: https://www.tinkercad.com/things/dWlrJlPdstc-lab-3a-multiplexer/editel?sharecode=jDsKUKQWxWcYQ8GbEGU_aK5y1SdgkEMrHGDanGlRV9o

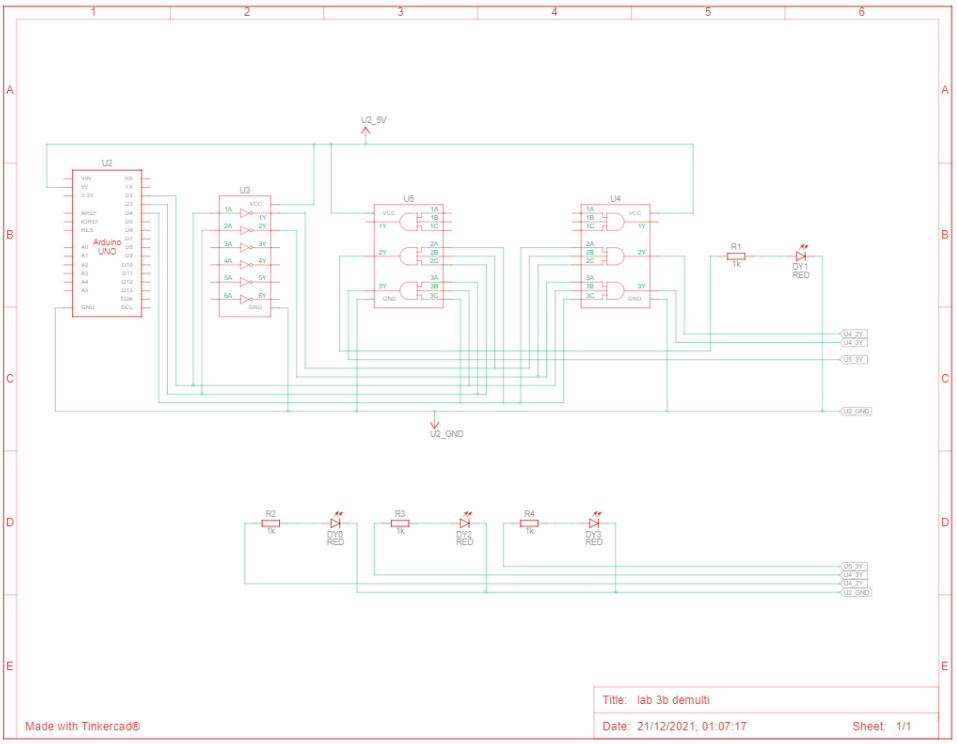
---

Part B: Designing 1:4 Demultiplexer

Aim/Objective of the experiment: To design a 1:4 demultiplexer using basic logic gates

Electronic components used: 1 Arduino board, four 1 kilo ohm resistors, 4 LEDs, 1 breadboard, 2 triple 3-input AND gates(74HC11), 1 hex inverter(74HC04), wires

Reference Circuit:



Procedure:

1. The circuit is set up, as shown in the reference figure above, on the breadboard.
2. An Arduino code is written to give different combinations of inputs and select lines and view them using LEDs at the output lines.
3. The demultiplexer function is verified by tabulating the values of the output(s) for all input combinations.

   The code used:

```
int s1,s0,i;

void setup()
{
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  Serial.begin(9600);
}
```

```
void loop()
{

  if(Serial.available()>0)
  {
    s1=Serial.read();
    s1=s1-'0';

    digitalWrite(2,s1);
  }

  if(Serial.available()>0)
  {
    s0=Serial.read();
    s0=s0-'0';

    digitalWrite(3,s0);
  }

  if(Serial.available()>0)
  {
    i=Serial.read();
    i=i-'0';

    digitalWrite(4,i);
  }

  delay(100);

}

/*

inputs verified:
001
011
101
111

*/
```

Conclusion:

S1	S0	y3	y2	y1	y0
0	0	0	0	0	i
0	1	0	0	i	0
1	0	0	i	0	0
1	1	i	0	0	0

So, the demultiplexer truth table is verified.

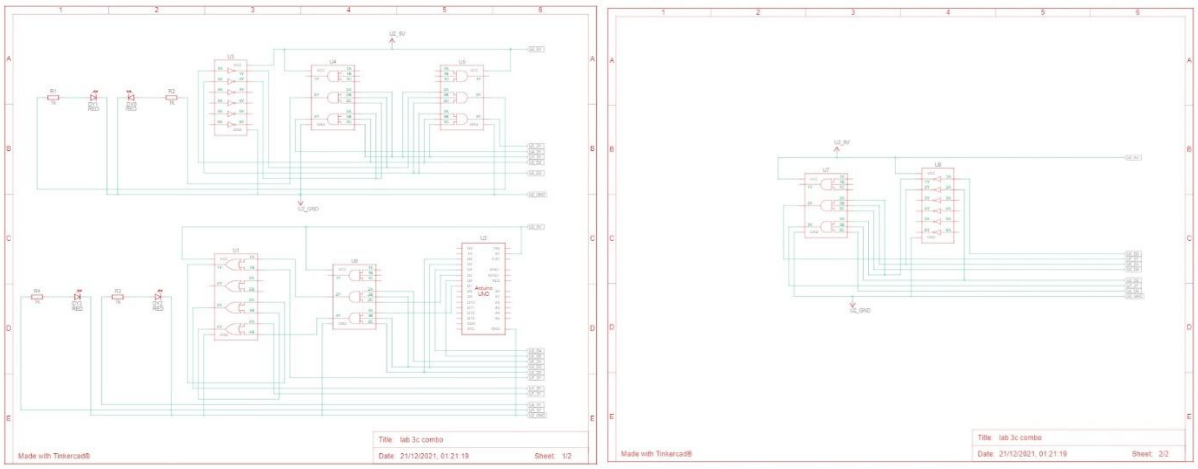
TinkerCAD Simulation: https://www.tinkercad.com/things/866WCErkqLg-lab-3b/editel?sharecode=-X4UItLwS5bqjn5qror4YYG_rjyzS-fMwcWEjr-UYkE

Part C: Assemble both

Aim/Objective of the experiment: To assemble and test the Multiplexer and Demultiplexer circuits

Electronic components used: 2 Arduino boards, four 1 kilo ohm resistors, 4 LEDs, 3 breadboards, 2 hex inverters(74HC04), 4 triple 3-input AND gates(74HC11), 1 quad OR gate(74HC32), wires

Reference Circuit:



Procedure:

1. The circuit is set up, as shown in the reference figure above, on the breadboard.
2. An Arduino code is written to give different combinations of inputs at MUX and then view them at DEMUX output using LEDs.
3. The values of the output(s) for all input combinations are tabulated.

   The code used:

```
int s1,s0,i0,i1,i2,i3;


void setup()
{
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(7,OUTPUT);
  Serial.begin(9600);
}
```

```
void loop()
{

  if(Serial.available()>0)
  {
    s1=Serial.read();
    s1=s1-'0';

    digitalWrite(2,s1);
  }

  if(Serial.available()>0)
  {
    s0=Serial.read();
    s0=s0-'0';

    digitalWrite(3,s0);
  }

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

    digitalWrite(4,i0);
  }

  if(Serial.available()>0)
  {
```

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

    digitalWrite(5,i1);
  }

  if(Serial.available()>0)
  {
    i2=Serial.read();
    i2=i2-'0';

    digitalWrite(6,i2);
  }

  if(Serial.available()>0)
  {
    i3=Serial.read();
    i3=i3-'0';

    digitalWrite(7,i3);
  }
  delay(100);

}

/*

inputs verified:
001000
010100
```

```
100010
110001

*/
```

Conclusion:

S0	S1	Inputs (in MUX)	Outputs (at DEMUX)
0	0	I_0, I_1, I_2, I_3	Output 0
0	1	I_0, I_1, I_2, I_3	Output 1
1	0	I_0, I_1, I_2, I_3	Output 2
1	1	I_0, I_1, I_2, I_3	Output 3

So, it is verified that the DEMUX gives back the outputs corresponding to the input channel.

TinkerCAD Simulation: https://www.tinkercad.com/things/4lxZ50Mbl06-lab-3c/editel?sharecode=xXEbb_fFHChx_nwYmrbnTlrbplrHK5lMTs6B-_W3qQc