**DSM Lab Report**

**Name:** Anirudh Kaushik

**Roll No:** 2020111015

**Group Number:** 5

**Experiment Part A**

**Objective**

To design, assemble and test a (1:4) Multiplexer and using basic logic gates (whose select lines and inputs are through Arduino).

**Theory**

A multiplexer (or mux) is a device that selects one of several analogue or digital input signals and forwards the selected input into a single line A multiplexer with 2^n inputs has n select lines, which are used to select which input line to send to the output.

A 4-to-1 multiplexer consists of four data input lines as I0 to I3, two select lines as S0 and S1 and a single output line Y. The select lines S0 and S1 select one of the four input lines (D0 through D3) to connect to the output line. The figure below shows the block diagram of a 4-to-1 multiplexer.

The truth table of a 4-to-1 multiplexer is shown below in which four input combinations 00, 10, 01 and 11 on the select lines respectively switch the inputs I0, I1, I2 and I3 to the output. That means when S1=0 and S0 =0, the output at Y is D0, similarly Y is D1 if the select inputs S1=0 and S0= 1 and so on.

|  |  |  |
| --- | --- | --- |
| **S1** | **S0** | **y** |
| 0 | 0 | I0 |
| 0 | 1 | I1 |
| 1 | 0 | I2 |
| 1 | 1 | I3 |

From the above truth table, we can write the output expressions as

If S1=0 and S0=0 then Y = I0

Therefore, Y = I0 (S1)’ (S0)’

If S1= 0 and S0=1, the Y = I1

Therefore, Y = I1 (S1)’ S0

If S1=1 and S0=0, then Y = I2

Therefore, Y = I2 S1 (S0)’

If S1=1 and S0=1 the Y = I3

Therefore, Y = I3 S1 S0

To get the total data output from the multiplexer, all these product terms are to be summed and then the final Boolean expression of this multiplexer is given as

Y = I0 (S1)’ (S0)’ + I1 (S1)’ S0 + I2 S1 (S0)’ + I3 S1 S0

**Experiment setup/ procedure**

**Materials required:**

|  |  |
| --- | --- |
| 2 | Triple 3-Input AND gate |
| 1 | Quad OR gate |
| 1 | Hex Inverter |
| 1 | Arduino Uno R3 |
| 1 | 1 kΩ Resistor |
| 1 | Red LED |
| 1 | Voltage Multimeter |

**Procedure:**

1. For the IC 74HC04 (Inverter) the data inputs are denoted by 1A, 2A, 3A, 4A and the data outputs by 1Y, 2Y, 3Y, 4Y.
2. For the IC 74HC11 (AND) and 74HC32 (OR), the data inputs are denoted by 1A, 1B, 1C and so on and the data outputs by 1Y, 2Y, 3Y.
3. Write an Arduino code to give different combinations of inputs at input and select lines and view them using LED at the output line.
4. Verify the multiplexer function by tabulating the values of the output(s) for all input combinations.

|  |
| --- |
| Digital Circuits - Multiplexers - Tutorialspoint |

Refer to circuit diagram:

**Code:**

int I1,I2,I3,I4;

int S1,S2;

void setup()

{

pinMode(13, OUTPUT);

pinMode(12, OUTPUT);

pinMode(11, OUTPUT);

pinMode(10, OUTPUT);

pinMode(9, OUTPUT);

pinMode(8, OUTPUT);

digitalWrite(13, LOW);

digitalWrite(12, LOW);

digitalWrite(11, LOW);

digitalWrite(10, LOW);

digitalWrite(9, LOW);

digitalWrite(8, LOW);

Serial.begin(9600);

}

void loop()

{

if(Serial.available() > 0)

{

while(!Serial.available())

{}

I1 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I1 = I1 - '0'; // Subtracting ascii value of 0 from x.

if (I1 == 1)

{

digitalWrite(10,HIGH);

Serial.print("I0 = ");

Serial.println(I1);

}

if(I1 == 0)

{

digitalWrite(10,LOW);

Serial.print("I0 = ");

Serial.println(I1);

}

while(!Serial.available())

{}

I2 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I2 = I2 - '0'; // Subtracting ascii value of 0 from x.

if (I2 == 1)

{

digitalWrite(11,HIGH);

Serial.print("I1 = ");

Serial.println(I2);

}

if(I2 == 0)

{

digitalWrite(11,LOW);

Serial.print("I1 = ");

Serial.println(I2);

}

while(!Serial.available())

{}

I3 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I3 = I3 - '0'; // Subtracting ascii value of 0 from x.

if (I3 == 1)

{

digitalWrite(12,HIGH);

Serial.print("I2 = ");

Serial.println(I3);

}

if(I3 == 0)

{

digitalWrite(12,LOW);

Serial.print("I2 = ");

Serial.println(I3);

}

while(!Serial.available())

{}

I4 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I4 = I4 - '0'; // Subtracting ascii value of 0 from x.

if (I4 == 1)

{

digitalWrite(13,HIGH);

Serial.print("I3 = ");

Serial.println(I4);

}

if(I4 == 0)

{

digitalWrite(13,LOW);

Serial.print("I3 = ");

Serial.println(I4);

}

while(!Serial.available())

{}

S1 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

S1 = S1 - '0'; // Subtracting ascii value of 0 from x.

if (S1 == 1)

{

digitalWrite(9,HIGH);

Serial.print("S0 = ");

Serial.println(S1);

}

if(S1 == 0)

{

digitalWrite(9,LOW);

Serial.print("S0 = ");

Serial.println(S1);

}

while(!Serial.available())

{}

S2 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

S2 = S2 - '0'; // Subtracting ascii value of 0 from x.

if (S2 == 1)

{

digitalWrite(8,HIGH);

Serial.print("S1 = ");

Serial.println(S2);

}

if(S2 == 0)

{

digitalWrite(8,LOW);

Serial.print("S1 = ");

Serial.println(S2);

}

Serial.println("======");

}

}

**Observations**

|  |  |  |
| --- | --- | --- |
| **S1** | **S0** | **y** |
| 0 | 0 | I0 |
| 0 | 1 | I1 |
| 1 | 0 | I2 |
| 1 | 1 | I3 |

The truth table for the multiplexer is as expected. The values for Ii , i = 0,1,2,3 can be varied between 0 and 1. The LED corresponding to Ii will only glow when Ii is set to 1 and the output through **y** is Ii. The LED corresponding to Ii remains OFF for all other combinations.

**Conclusion**

Successfully constructed a 4-to-1 multiplexer circuit and verified its functionality by matching input and output values with the truth-table.

**Tinkercad link with circuit**

1. <https://www.tinkercad.com/things/9h3ogggdynO-smooth-migelo/editel?sharecode=o7vm27DHP5N7jh6IrQht30MxxD-55K79LW7e0OUtkEg>

**Circuit Diagram**

|  |
| --- |
|  |

**Experiment Part B**

**Objective**

To design, assemble and test a (4:1) De-multiplexer using basic logic gates (whose select lines and inputs are through Arduino).

**Theory**

A 1-to-4 demultiplexer consists of one data input line as i, two select lines as S0 and S1 and four output lines as y0, y1, y2 and y3. The select lines S0 and S1 select one of the four output lines (y0 through y3) to connect to the input line. The figure below shows the block diagram of a 1-to-4 demultiplexer.

The truth table of a 1-to-4 demultiplexer is shown below in which four input combinations 00, 10, 01 and 11 on the select lines respectively switch the input to the output lines y0, y1, y2 and y3. That means when S1=0 and S0 =0, the output at Y is y0 = i, similarly y1 is i if the select inputs S1=0 and S0= 1 and so on.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 |

From the above truth table, we can write the output expressions as

If S1=0 and S0=0 then y0 = i

Therefore, y0 = i (S1)’ (S0)’

If S1= 0 and S0=1, the y1 = i

Therefore, y1 = i (S1)’ S0

If S1=1 and S0=0, then y2 = i

Therefore, y2 = i S1 (S0)’

If S1=1 and S0=1 the y3 = i

Therefore, y3 = i S1 S0

From the above expressions of the outputs, a 1-to-4 demultiplexer can be implemented by using basic logic gates.

The below figure shows the logic circuit of 1:4 DEMUX which is implemented by four 3- inputs AND gates and two 1-input NOT gate.

|  |
| --- |
| Digital Circuits - De-Multiplexers - Tutorialspoint |

**Experiment setup/ procedure**

**Materials required:**

|  |  |
| --- | --- |
| 2 | Triple 3-Input AND gate |
| 1 | Hex Inverter |
| 1 | Arduino Uno R3 |
| 4 | 1 kΩ Resistor |
| 1 | Red LED |
| 1 | Blue LED |
| 1 | Green LED |
| 1 | Yellow LED |
| 1 | Voltage Multimeter |

**Procedure:**

1. Write an Arduino code to give different combinations of inputs and select lines and view them using LEDs at the output lines.
2. Verify the demultiplexer function by tabulating the values of the output(s) for all input combinations.

**Code:**

int I,S0,S1;

void setup()

{

pinMode(7, OUTPUT);//I

pinMode(6, OUTPUT);//S0

pinMode(5, OUTPUT);//S1

digitalWrite(7, LOW);

digitalWrite(6, LOW);

digitalWrite(5, LOW);

Serial.begin(9600);

}

void loop()

{

if(Serial.available() > 0)

{

while(!Serial.available())

{}

I = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I = I - '0'; // Subtracting ascii value of 0 from x.

if (I == 1)

{

digitalWrite(7,HIGH);

Serial.print("I = ");

Serial.println(I);

}

if(I == 0)

{

digitalWrite(7,LOW);

Serial.print("I = ");

Serial.println(I);

}

while(!Serial.available())

{}

S0 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

S0 = S0 - '0'; // Subtracting ascii value of 0 from x.

if (S0 == 1)

{

digitalWrite(6,HIGH);

Serial.print("S0 = ");

Serial.println(S0);

}

if(S0 == 0)

{

digitalWrite(6,LOW);

Serial.print("S0 = ");

Serial.println(S0);

}

while(!Serial.available())

{}

S1 = Serial.read(); // x would be an integer between 0 and 255

S1 = S1 - '0'; // Subtracting ascii value of 0 from x.

if (S1 == 1)

{

digitalWrite(5,HIGH);

Serial.print("S1 = ");

Serial.println(S1);

}

if(S1 == 0)

{

digitalWrite(5,LOW);

Serial.print("S1 = ");

Serial.println(S1);

}

Serial.println("======");

}

}

**Observations**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 |

The truth table for the 1-to-4 Demultiplexer has been successfully verified. The input I (or i) can be varied between 0 and 1. For I = 1 the corresponding LED selected by the select lines glows and remains OFF in all other cases.

**Conclusion**

Successfully verified the truth table of a 1-to-4 Demultiplexer experimentally by inputting required code in the Arduino and varying inputs for S1, S0 and I.

**Tinkercad Link with Circuit**

1. <https://www.tinkercad.com/things/3G3EXvXVkv6-smashing-turing/editel?sharecode=sxedBbYjOvlY3TnQmyEasDuhrxt3_9heSoYjLhVIsag>

|  |
| --- |
| **CIRCUIT DIAGRAM ON NEXT PAGE:** |

**Circuit Diagram**

|  |
| --- |
|  |

**Experiment Part C**

**Objective**

Assemble and test circuits designed in Parts A and B

**Experiment setup/ procedure**

**Materials required:**

|  |  |
| --- | --- |
| 4 | Triple 3-Input AND gate |
| 2 | Hex Inverter |
| 5 | 1 kΩ Resistor |
| 2 | Red LED |
| 1 | Blue LED |
| 1 | Green LED |
| 1 | Yellow LED |
| 1 | Voltage Multimeter |
| 1 | Arduino Uno R3 |
| 1 | Quad XOR gate |

**Procedure:**

1. Write a program to give different combinations of inputs at MUX and view them at DEMUX output using LEDs.
2. Connect VCC and GND of both mux and demux to 5V and GND pin of Arduino. Connect 4 pins of Arduino to the mux as input pins. The output of the mux should be given as an input to the demux, and the select lines of both the mux and the demux must also be taken from Arduino. Connect the output of the demux to 4 LEDs. The block diagram and Tinkercad circuit diagram of the final circuit are given below.

**Code:**

int I1,I2,I3,I4;

int S1,S2;

void setup()

{

pinMode(13, OUTPUT);

pinMode(12, OUTPUT);

pinMode(11, OUTPUT);

pinMode(10, OUTPUT);

pinMode(9, OUTPUT);

pinMode(8, OUTPUT);

digitalWrite(13, LOW);

digitalWrite(12, LOW);

digitalWrite(11, LOW);

digitalWrite(10, LOW);

digitalWrite(9, LOW);

digitalWrite(8, LOW);

Serial.begin(9600);

}

void loop()

{

if(Serial.available() > 0)

{

while(!Serial.available())

{}

I1 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I1 = I1 - '0'; // Subtracting ascii value of 0 from x.

if (I1 == 1)

{

digitalWrite(13,HIGH);

Serial.print("I0 = ");

Serial.println(I1);

}

if(I1 == 0)

{

digitalWrite(13,LOW);

Serial.print("I0 = ");

Serial.println(I1);

}

while(!Serial.available())

{}

I2 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I2 = I2 - '0'; // Subtracting ascii value of 0 from x.

if (I2 == 1)

{

digitalWrite(12,HIGH);

Serial.print("I1 = ");

Serial.println(I2);

}

if(I2 == 0)

{

digitalWrite(12,LOW);

Serial.print("I1 = ");

Serial.println(I2);

}

while(!Serial.available())

{}

I3 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I3 = I3 - '0'; // Subtracting ascii value of 0 from x.

if (I3 == 1)

{

digitalWrite(11,HIGH);

Serial.print("I2 = ");

Serial.println(I3);

}

if(I3 == 0)

{

digitalWrite(11,LOW);

Serial.print("I2 = ");

Serial.println(I3);

}

while(!Serial.available())

{}

I4 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

I4 = I4 - '0'; // Subtracting ascii value of 0 from x.

if (I4 == 1)

{

digitalWrite(10,HIGH);

Serial.print("I3 = ");

Serial.println(I4);

}

if(I4 == 0)

{

digitalWrite(10,LOW);

Serial.print("I3 = ");

Serial.println(I4);

}

while(!Serial.available())

{}

S1 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

S1 = S1 - '0'; // Subtracting ascii value of 0 from x.

if (S1 == 1)

{

digitalWrite(9,HIGH);

Serial.print("S0 = ");

Serial.println(S1);

}

if(S1 == 0)

{

digitalWrite(9,LOW);

Serial.print("S0 = ");

Serial.println(S1);

}

while(!Serial.available())

{}

S2 = Serial.read(); // x would be an integer between 0 and 255

// depending on the ascii value of the character read

S2 = S2 - '0'; // Subtracting ascii value of 0 from x.

if (S2 == 1)

{

digitalWrite(8,HIGH);

Serial.print("S1 = ");

Serial.println(S2);

}

if(S2 == 0)

{

digitalWrite(8,LOW);

Serial.print("S1 = ");

Serial.println(S2);

}

Serial.println("======");

}

}

**Observation**

The combined circuit works as expected. Whatever is inputted, i.e., IP1, IP2, IP3, IP4, the corresponding LED selected by the select lines will glow (if the corresponding input was 1) In other words if we input 0 1 1 0 to IP1, IP2, IP3 and IP4 respectively and have S1 = 1 and S0 = 0, the LED selected will be Y3 and since IP3 = 1, Y3 will be 1 and the LED will glow.

The LED remains OFF if:

1. It is not selected
2. The corresponding input IPi is set to 0

**Conclusion**

Successfully created a combination of a 4-to-1 multiplexer and 1-to-4 demultiplexer which takes input from an Arduino Uno R3. The required code for the Arduino has been provided and allows us to vary IP1, IP2, IP3 and IP4 and the select lines S1 and S0. The values that can be entered are 1 and 0 and corresponding output is observed through LEDs labelled Y1, Y2, Y3 and Y4.

**Tinkercad Link with Circuit**

1. <https://www.tinkercad.com/things/6eqOezNtWiX-copy-of-lab-3-part-b/editel?sharecode=YH1CplpDrirQh9gtBKaJmyXahOl0ang0dIHhf3V4VR4>

**Circuit Diagram**

|  |
| --- |
|  |

**All Tinkercad links for the circuits**

1. <https://www.tinkercad.com/things/9h3ogggdynO-smooth-migelo/editel?sharecode=o7vm27DHP5N7jh6IrQht30MxxD-55K79LW7e0OUtkEg>
2. <https://www.tinkercad.com/things/3G3EXvXVkv6-smashing-turing/editel?sharecode=sxedBbYjOvlY3TnQmyEasDuhrxt3_9heSoYjLhVIsag>
3. <https://www.tinkercad.com/things/6eqOezNtWiX-copy-of-lab-3-part-b/editel?sharecode=YH1CplpDrirQh9gtBKaJmyXahOl0ang0dIHhf3V4VR4>