LAB REPORT-4

Name: Snigdha stp

Roll number: 2023102036

Group number: G6

Objective: Using 1-bit operands, this experiment uses an ALU to accomplish 4 logical operations and 4 arithmetic operations.

Electronic components:

1)Arduino Uno R3

2)One NOT Gate (IC – 7404 (OR) Hex Inverter)

3)Two QUAD OR Gate (IC - 7432)

4)Two QUAD AND Gate (IC - 7411)

5)Two 8-input multiplexers (74LS151)

6)One XOR Gate (IC - 7486)

Procedure:

1) A circuit with two 8:1 multiplexers is provided. Her Y1 stands for (carry or borrow) and Y0 stands for the sum. For varying values of X0 and X1, all of the functions on A, B, and C are carried out using a 2:1 multiplexer. i.e., A.B, A+B, A’.B , A’+B functions are produced using a 2:1 multiplexer, and A+B+C and A-B-C functions are implemented using XOR gate, AND gate, etc.

2) Connect GND and VCC, then connect all selected inputs F0, F1, and F2.

3) Take the inputs A, B, C, F0, F1, and F2 from Arduino.

4) Connect the enable pin to the respective terminals.

5) In the Arduino IDE, execute the following code

void setup ()

{

Serial.begin(9600);

pinMode (2, INPUT);

pinMode (3, INPUT);

pinMode (4, INPUT);

pinMode (5, INPUT);

pinMode (6, INPUT);

pinMode (7, INPUT);

}

int F0, F1, F2, A, B, C;

void loop ()

{

Serial.println("Enter the values in the order F2, F1, F0, A, B, C : ");

while (Serial.available() == 0)

{}

F2=Serial.read()-48;

while (Serial.available () == 0)

{}

F1=Serial.read()-48;

while (Serial.available () == 0)

{}

while (Serial.available () == 0)

{}

A=Serial.read()-48;

while (Serial.available () == 0)

{}

B=Serial.read()-48;

while (Serial.available() == 0)

{}

C=Serial.read()-48;

digitalWrite (7, F2);

Serial.print(F2);

digitalWrite (6, F1);

Serial.print(F1);

digitalWrite (5, F0);

Serial.print(F0);

digitalWrite (2, A);

Serial.print(A);

digitalWrite (3, B);

Serial.print(B);

digitalWrite (4, C);

Serial.print(C);

Serial.println("\n");

delay(1000);

}

Truth Table:



Observation:

After applying combinations of A,B,C; the output values are

tabulated below for each combination of select lines F0,F1,F2.

**(0,0,0)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0/1 | 0/1 | 0/1 | - | 0 |

**Y1 = “-”**

**Y0= 0**

**(0,0,1)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0/1 | - | 0 |
| 0 | 1 | 0/1 | - | 1 |
| 1 | 0 | 0/1 | - | 1 |
| 1 | 1 | 0/1 | - | 1 |

**Y1=”-”**

**Y0 = A+B**

**(0,1,0)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0/1 | - | 0 |
| 0 | 1 | 0/1 | - | 0 |
| 1 | 0 | 0/1 | - | 0 |
| 1 | 1 | 0/1 | - | 1 |

**Y1 = “-”**

**Y0 = A.B**

**(0,1,1)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0/1 | - | 0 |
| 0 | 1 | 0/1 | - | 1 |
| 1 | 0 | 0/1 | - | 1 |
| 1 | 1 | 0/1 | - | 0 |

**Y1 = “-”**

**Y0 = A^B**

**(1,0,0)**

**A+B**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0/1 | 0 | 0 |
| 0 | 1 | 0/1 | 0 | 1 |
| 1 | 0 | 0/1 | 0 | 1 |
| 1 | 1 | 0/1 | 1 | 0 |

**Y1 = Carry**

**Y0 = Sum**

**(1,0,1) A - B**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0/1 | 0 | 0 |
| 0 | 1 | 0/1 | 1 | 1 |
| 1 | 0 | 0/1 | 0 | 1 |
| 1 | 1 | 0/1 | 0 | 0 |

**Y1 = Borrow**

**Y0 = Difference**

**(1,1,0)**

**A + B + C**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

**Y1 = Carry**

**Y0 = Sum**

**(1,1,1)**

**A - B - C**

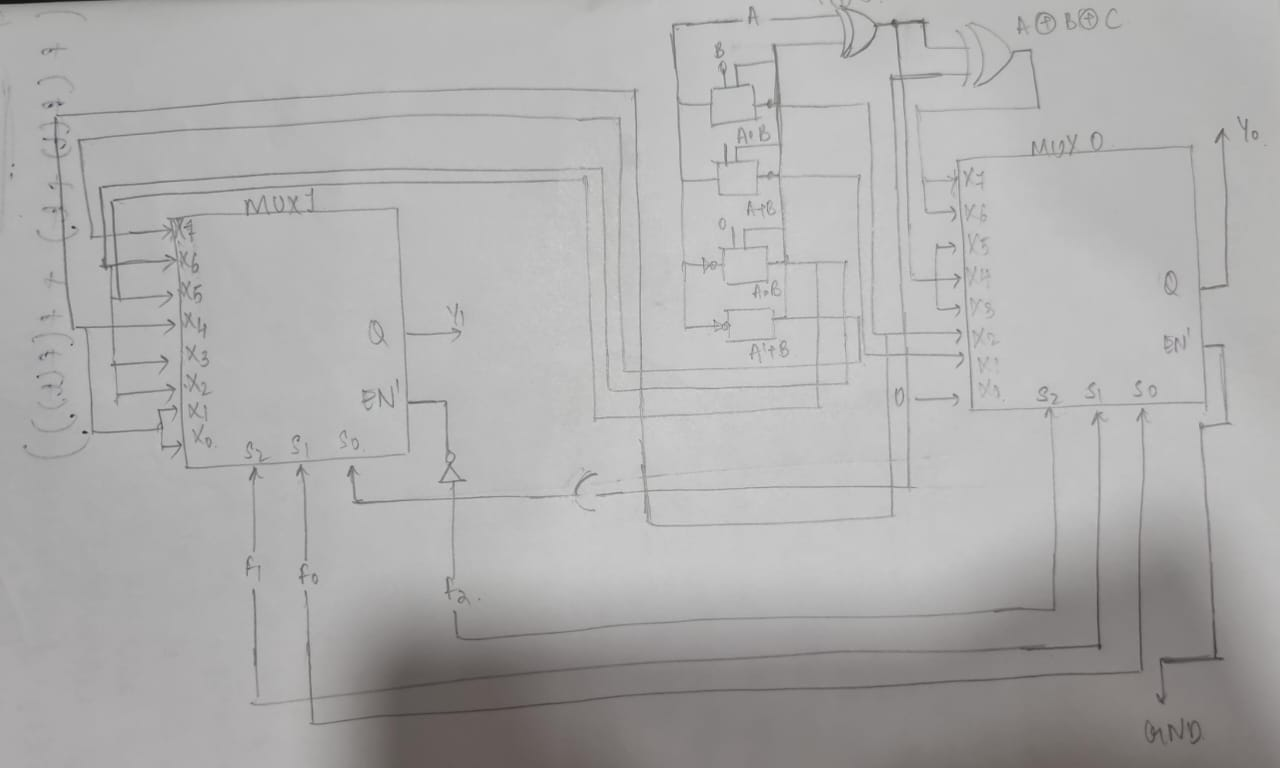
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Y1 | Y0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

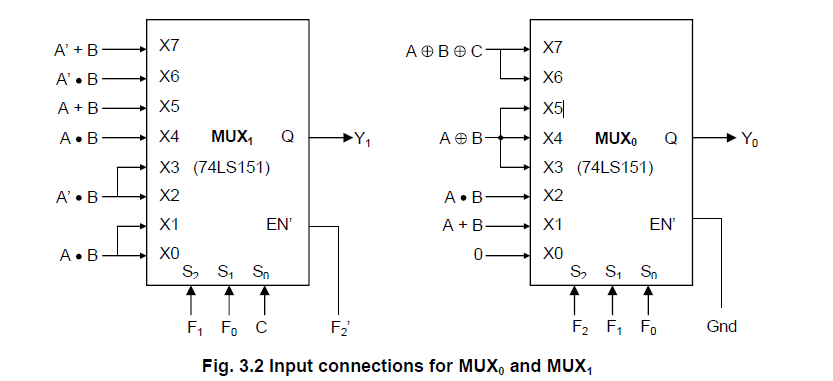
**Y1 = Borrow**

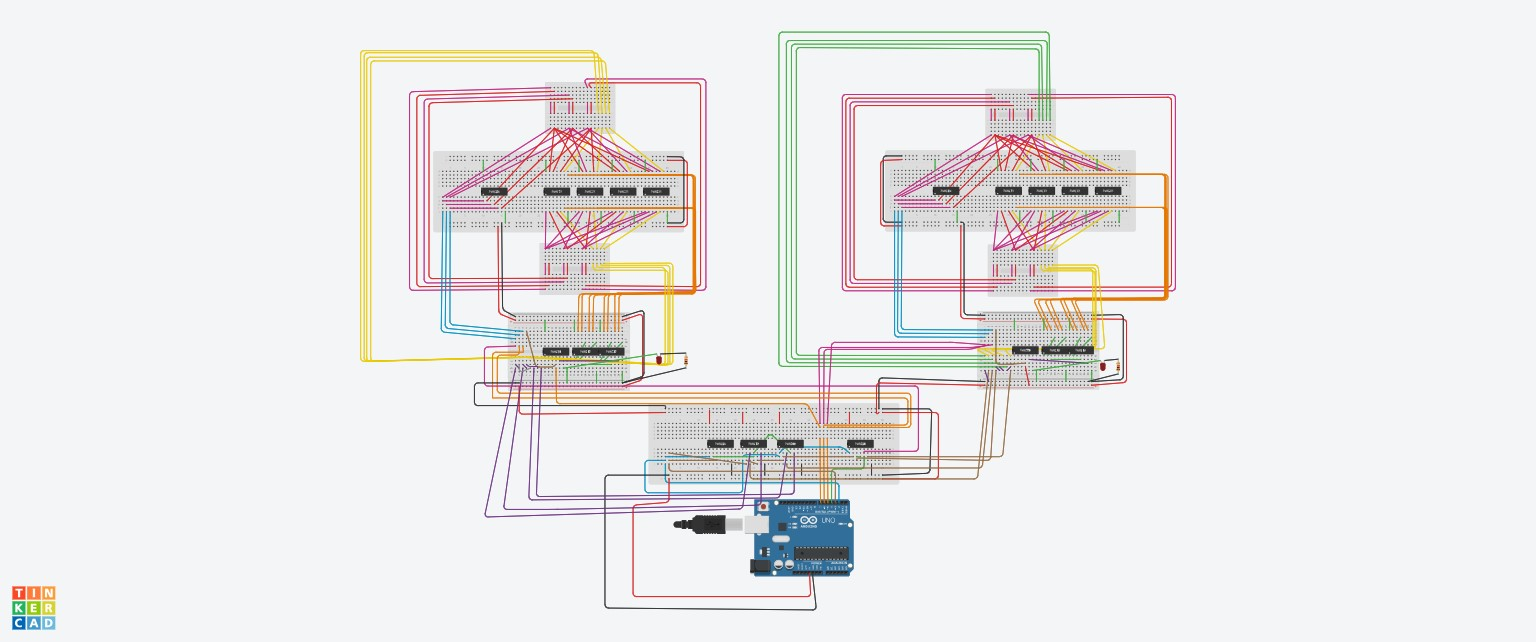
**Y0 = Difference**

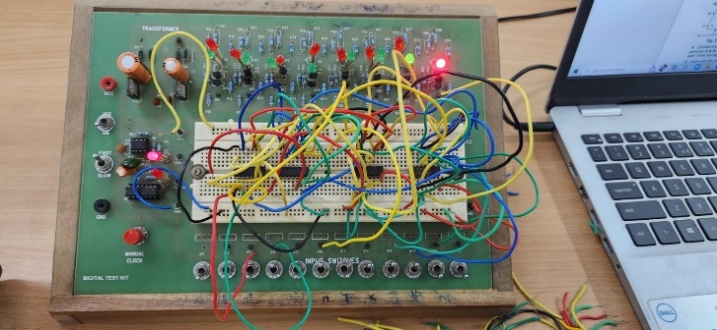
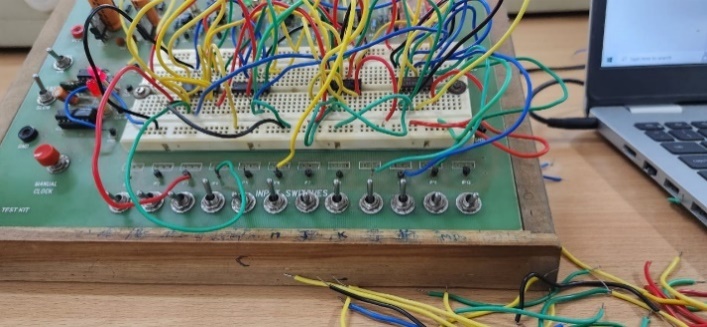
CONCLUSION: This experiment revealed how ALU, the heart of DSM, works. We created a 1-bit binary functions output ALU, and we may create other bits of ALU for more sophisticated circuits. The ALU has storage for input operands, adding operands, the accumulated result, and shifting results. Gated circuits govern the flow of bits and the operations performed on them in the ALU's subunits. This experiment taught us about the internal complexity of an ALU circuit.

Reference Circuit:









Tinkercad link:

<https://www.tinkercad.com/things/dk1vvC20XQT-copy-of-2-81-mux/editel?sharecode=WP9m7h-QGLVGOg4icK0DN0J29DkDOZWARL1y7yflf5s>