DSM Lab Report

Name: Anirudh Kaushik

Roll No: 2020111015

Group Number: 5

Experiment Part A

Objective

To determine the tipping voltage for a hex inverter (7HC04) using LEDs and a multimeter.

Experiment setup/ procedure

Materials required:

1. 2xLEDs
2. 2xResistors(1kΩ)
3. 2xMultimeters
4. 1xPotentiometer(1kΩ)
5. 1xArduino uno R3
6. 1xBreadBoard

Procedure:

1. Set up the circuit shown in the circuit diagram on the breadboard and turn the potentiometer shaft to one end so that the multimeter reads 0V.
2. DP1 and DP2 are LEDs connected with appropriate resistors. DP2 must be glowing.
3. Now rotate the potentiometer shaft gradually up to the other end and tabulate the transitions in DP1 and DP2.

Code:

```
void setup()
{
  pinMode(13, OUTPUT);
}
```

Arduino used only as source of voltage.

Observations

Observation table

INPUT VOLTAGE (Potentiometer Reading)	HEX INVERTER (7HC04) OUTPUT VOLTAGE	LED 1 (Connected to HEX INVERTER/NOT gate)	LED 2 (Connected normally)
0 to ~ 2.44V	4.87 V	ON	OFF
~2.44V to ~ 2.5V	4.87V	ON(	ON
~2.52V	0.00V	OFF	ON
~2.52V to 5V	0.00V	OFF	ON

NOT gate truth table

INPUT	OUTPUT
1 (HIGH)	0 (LOW)
0 (LOW)	1 (HIGH)

1. The LED connected through the NOT gate remains ON (i.e., NOT gate outputs HIGH voltage) when the input voltage is in the range 0 to 2.52 V (LOW) and the LED connected normally remains OFF.
2. When the input voltage is in the range 2.44V to 2.52V both the LEDs remain ON simultaneously (transition period)
3. The LED connected through the NOT gate remains OFF (i.e., NOT gate outputs LOW voltage) when the input voltage is in the range 2.52 V to 5V (HIGH) and the LED connected normally remains ON.

Conclusion

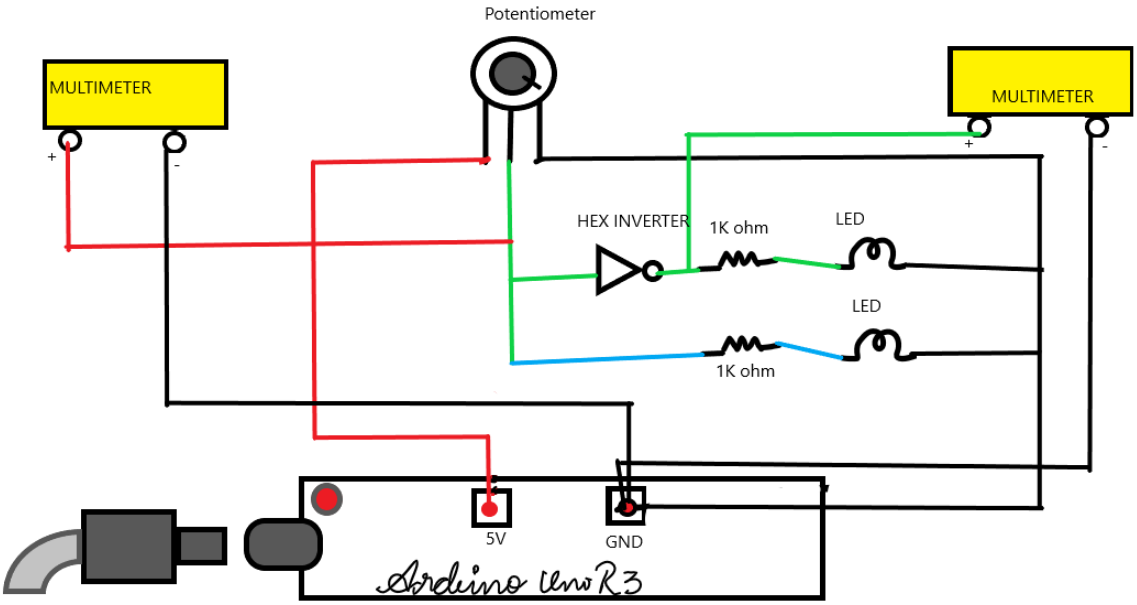
1. Thus it can be concluded that the LOW voltage (corresponding to state 0) for the HEX Inverter is from 0v to 2.52v.
2. 2.52v is the tipping point for the HEX INVERTER, when the Inverter output changes from HIGH(4.87V) to LOW(0V) and vice versa.
3. In the range ~2.44V to ~2.52V we see both the LEDS glowing, hence this can be called the transition period for switching the output voltage from HIGH to LOW.

a) Comparing these voltages with the specifications for binary logic level for a 0-5 V range.
b) $0 \leq VOL \leq 0.4$, $0 \leq VIL \leq 0.8$, $2.0 \leq VIH \leq 5.0$, $2.4 \leq VOH \leq 5.0$
c) We see that out inverter gives:
d) $VOL = 0V$, $0 \leq VIL \leq 2.52$, $2.52 \leq VIH \leq 5.0$, $VOH = 4.87V$
e) Which is within the same range.

Tinkercad link with circuit

A. https://www.tinkercad.com/things/7ZGEHiobeg1-ingenious-robo-snaget/editel?sharecode=taRk4RedQc3U-O5zcjMGS118kep27R3KpHUJFGAZC00

Circuit Diagram



Experiment Part B

Objective

The goal of this part of the experiment is to take input from the serial monitor and verify the truth table of logic gates: (NOT, OR, AND, XOR, NOT, NAND) using TTL 74XX family of ICs.

Experiment setup/ procedure

Materials required:

2	Arduino Uno R3
6	1 kΩ Resistor
2	Red LED
1	Quad NAND gate
1	Quad OR gate
1	Quad NOR gate
1	Quad XOR gate
1	Quad AND gate
1	Hex Inverter
1	Blue LED
1	Green LED
1	White LED
1	Yellow LED

Procedure:

1. Place the IC on breadboard and give Vcc and Gnd connection to it.
2. Take inputs from the Serial Monitor for values of A and B and route them to the input pins of the IC.
3. Connect an LED with appropriate resistor to the output of the GATE.
4. Note the output of the chosen gate for different values of input in a truth table.

Code:

```
int a = 13;
int b = 12;
int A,B;

int x;
void setup() {
  pinMode(a, OUTPUT);
  pinMode(b, OUTPUT);
  digitalWrite(13,HIGH);
  digitalWrite(12,HIGH);
  Serial.begin(9600); //opens serial port, sets data rate to 9600 bps

}
void loop() {
  if(Serial.available() > 0)
  {
    while(!Serial.available())
    {}
    A = Serial.read(); // x would be an integer between 0 and 255
    // depending on the ascii value of the character read
    A = A - '0'; // Subtracting ascii value of 0 from x.
    if (A == 1)
    {
      digitalWrite(13,HIGH);
      Serial.print("A =");
      Serial.println(A);
    }
    if(A == 0)
    {
      digitalWrite(13,LOW);
      Serial.print("A =");
      Serial.println(A);
    }
    while(!Serial.available())
    {}
    B = Serial.read(); // x would be an integer between 0 and 255
```

```
  // depending on the ascii value of the character read
  B = B - '0';
  if(B == 1)
  {
    digitalWrite(12,HIGH);
    Serial.print("B =");
    Serial.println(B);
  }
  if(B == 0)
  {
    digitalWrite(12,LOW);
    Serial.print("B =");
    Serial.println(B);
  }
  Serial.println("====");



  }
//Use the value of x in code.
}
```

Conclusion

The truth tables for the logic gates are as follows:

OR gate truth table

INPUT A	INPUT B	OUTPUT	LED
1	1	1	ON
1	0	1	ON
0	1	1	ON
0	0	0	OFF

AND gate truth table

INPUT A	INPUT B	OUTPUT	LED
1	1	1	ON
1	0	0	OFF
0	1	0	OFF
0	0	0	OFF

NAND gate truth table

INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	1	ON
0	1	1	ON
0	0	1	ON

NOR gate truth table

INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	0	OFF
0	1	0	OFF
0	0	1	ON

XOR gate truth table

INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	1	ON
0	1	1	ON
0	0	0	OFF

NOT gate truth table

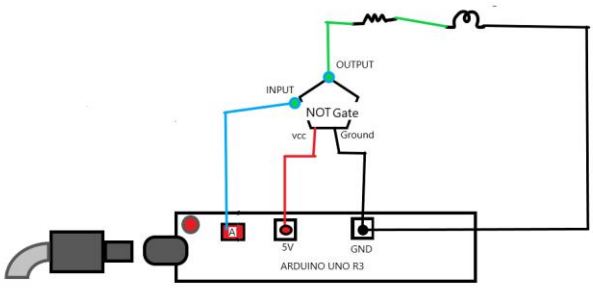
INPUT A	OUTPUT	LED
1	0	OFF
0	1	ON

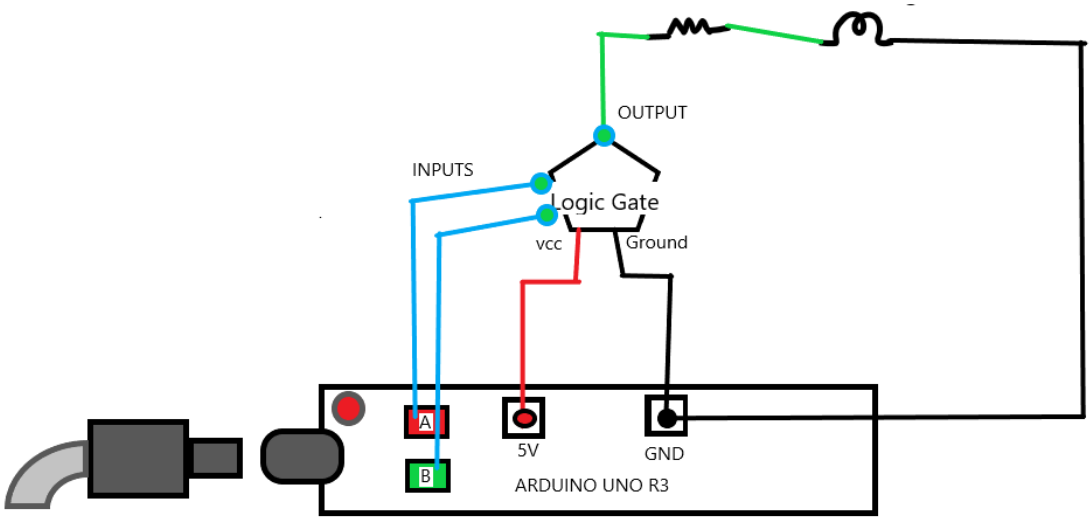
Tinkercad Link with Circuit

https://www.tinkercad.com/things/9WNjs84oLZ2-lab-2-part-b/editel?sharecode=0Y6iRXrrlPG9fjs-Ogs_C9mSFqLJAAXaWHxsdFZ0IYA

Circuit Diagram

This is a general diagram for the experiment, notice that the not gate will take only one input:




OUTPUT
INPUTS
Logic Gate
vcc
Ground
A
B
5V
GND
ARDUINO UNO R3

Experiment Part C

Objective

De Morgan's theorems state that $(A + B)\,0 = A0 \cdot B0$ and $(A \cdot B)\,0 = A0 + B0$.

Our objective is to Verify these theorems by proceeding step by step.

Experiment setup/ procedure

Materials required:

2	Arduino Uno R3
2	Hex Inverter
1	Quad AND gate
2	1 kΩ Resistor
2	Red LED
1	Quad OR gate
1	Voltage Multimeter

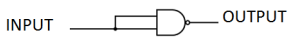
Procedure:

1. Set up a circuit consisting of two NOT gates and one AND gate to perform function Y = (A+B)' = A'.B'
2. Obtain the truth table of this circuit by noting the output of the function for different values of A and B. Verify that the output of the function is same as that of the NOR gate.
3. Repeat steps 1 and 2 using an OR gate instead of an AND gate to verify that the truth table is same as that of the NAND gate.

Q) How would you realise the above circuit if you have only NAND gates instead of NOT gates? i.e How would you use NAND gates to perform function of NOT gates?

To you use NAND gates instead of NOT gate we just have to join the inputs of the NAND gate and the output of the previous gate will serve as the sole input of this gate. Thus our NAND gate will now work as a NOT gate. Now, we just replace the NOT gate in the circuit with this modified NAND gate and our circuit is complete.

NOT gate from NAND gate



```
int a = 13;
int b = 12;
int A,B;

int x;
void setup() {
  pinMode(a, OUTPUT);
  pinMode(b, OUTPUT);
  digitalWrite(13,HIGH);
  digitalWrite(12,HIGH);
  Serial.begin(9600); //opens serial port, sets data rate to 9600 bps

}
void loop() {
  if(Serial.available() > 0)
  {
    while(!Serial.available())
    {}
    A = Serial.read(); // x would be an integer between 0 and 255
    // depending on the ascii value of the character read
    A = A - '0'; // Subtracting ascii value of 0 from x.
    if (A == 1)
    {
      digitalWrite(13,HIGH);
      Serial.print("A =");
      Serial.println(A);
    }
    if(A == 0)
    {
      digitalWrite(13,LOW);
      Serial.print("A =");
```

```
  Serial.println(A);
 }
 while(!Serial.available())
 {}
 B = Serial.read(); // x would be an integer between 0 and 255
 // depending on the ascii value of the character read
 B = B - '0';
 if(B == 1)
 {
  digitalWrite(12,HIGH);
  Serial.print("B =");
  Serial.println(B);
 }
 if(B == 0)
 {
  digitalWrite(12,LOW);
  Serial.print("B =");
  Serial.println(B);
 }
 Serial.println("====");



 }
//Use the value of x in code.
}
```

Conclusion

Successfully verified de morgan's law with the help of an OR gate, a NOT gate and an AND gate.

The truth table for the OR gate + NOT gate:

INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	0	OFF
0	1	0	OFF
0	0	1	ON

Truth table for A`.B`

INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	0	OFF
0	1	0	OFF
0	0	1	ON

Hence First De Morgan law has been verified.

Truth Table for (A' + B')

INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	1	ON
0	1	1	ON
0	0	1	ON

NAND gate truth table

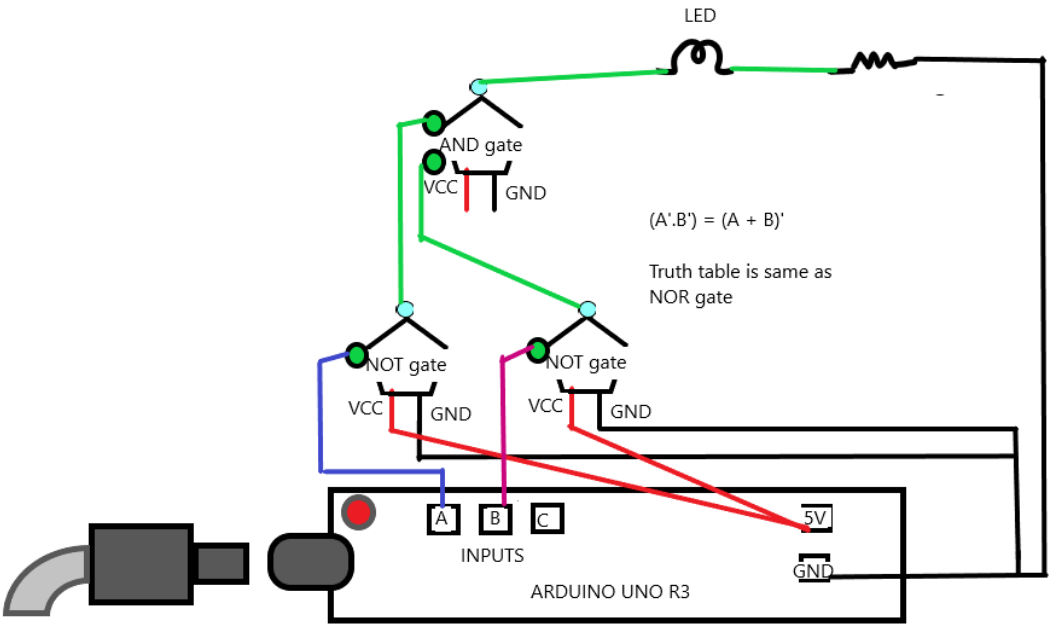
INPUT A	INPUT B	OUTPUT	LED
1	1	0	OFF
1	0	1	ON
0	1	1	ON
0	0	1	ON

Hence second De Morgan law has been verified.

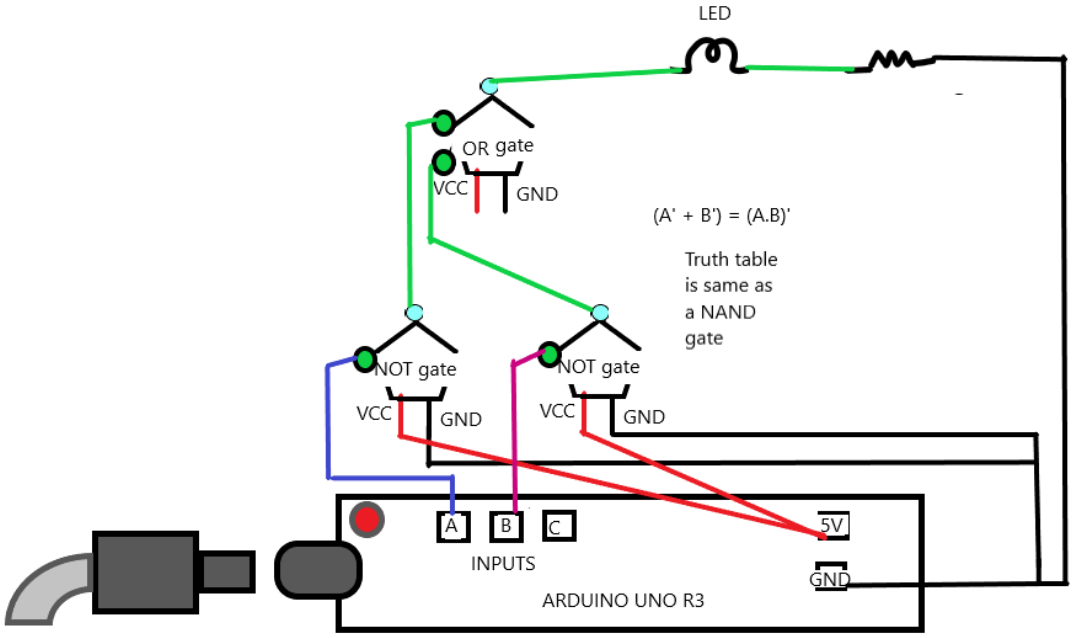
Tinkercad Link with Circuit

https://www.tinkercad.com/things/cWUd12uFbRa-lab-2-part-c/editel?sharecode=tj4JJEYMIz2LQIZCJWOgLqHbhonoi_nNB3hg-XslXSs

Circuit Diagram



==========



Experiment Part D

Objective

To make a Binary Full-Adder Circuit using only XOR and AND gates.

Experiment setup/ procedure

Materials required:

1	Arduino Uno R3
1	Quad XOR gate
1	Quad AND gate
5	1 kΩ Resistor
1	Green LED
1	Blue LED
1	Yellow LED
1	Red LED
1	White LED

Procedure:

A binary Full Adder adds two bits A and B along with a carry in C to generate SUM and CARRY bits as output. The first step to achieve this is to make a binary Half Adder, which adds two binary inputs A and B to give a sum S1 and a carry C1 according to the following Boolean expressions for the outputs S1 and C1: S1 = A 0 · B + A · B 0 = A ⊕ B and C1 = A · B Another Half Adder is then used to generate the final SUM by adding the third binary input C to the S1 bit generated by the first Half Adder: SUM = S1 ⊕ C The carry bit generated by this Half Adder is given by C2 = S1 · C . Write down the complete truth table of a Full Adder, including columns for the intermediate outputs S1, C1 and C2. Find out the logic for generating the final CARRY output from C1 and C2. As XOR and AND gates are going to be used for the Half Adders, try to obtain a logic for CARRY using the same type of gates, so that the complete realisation of the Full Adder is possible without necessitating a third IC.

1. Set up the circuit of a Half Adder using an XOR gate and an AND gate. Apply the inputs A and B from two input pins and observe the outputs S1 and C1 on two LED displays for

all combinations of the inputs. Tabulate these values and verify the operation of the Half Adder.

2. Set up another Half Adder using another XOR and another AND gate out of the same ICs used in step 1, and connect the C input and the S1 output generated by the first Half Adder as its inputs to generate the final SUM output and the C2 output.
3. Generate the final CARRY output from the intermediate carry outputs C1 and C2, using the unused gates in the XOR and AND ICs deployed so far 4. Verify the truth table experimentally by applying the inputs A, B and C through three input pins and displaying the S1, C1, C2, SUM and CARRY outputs.

Code:

```
int a = 13;

int b = 12;

int c = 11;

int A,B,C;


int x;

void setup() {

  pinMode(a, OUTPUT);

  pinMode(b, OUTPUT);

  pinMode(c,OUTPUT);

  digitalWrite(13,HIGH);

  digitalWrite(12,HIGH);

  digitalWrite(11,HIGH);

  Serial.begin(9600); //opens serial port, sets data rate to 9600 bps


}

void loop() {

  if(Serial.available() > 0)

  {

    while(!Serial.available())

    {}

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

    // depending on the ascii value of the character read
```

```
A = A - '0'; // Subtracting ascii value of 0 from x.
if (A == 1)
{
  digitalWrite(13,HIGH);
  Serial.print("A =");
  Serial.println(A);
}
if(A == 0)
{
  digitalWrite(13,LOW);
  Serial.print("A =");
  Serial.println(A);
}
while(!Serial.available())
{}
B = Serial.read(); // x would be an integer between 0 and 255
// depending on the ascii value of the character read
B = B - '0';
if(B == 1)
{
  digitalWrite(12,HIGH);
  Serial.print("B =");
  Serial.println(B);
}
if(B == 0)
{
  digitalWrite(12,LOW);
  Serial.print("B =");
  Serial.println(B);
}
```

```
   while(!Serial.available())
   {}
   C = Serial.read(); // x would be an integer between 0 and 255
   // depending on the ascii value of the character read
   C = C - '0';
   if(C == 1)
   {
     digitalWrite(11,HIGH);
     Serial.print("C =");
     Serial.println(C);
   }
   if(C == 0)
   {
     digitalWrite(11,LOW);
     Serial.print("C =");
     Serial.println(C);
   }
   Serial.println("====");




  }
//Use the value of x in code.
}
```

Conclusion

Successfully made a 2-Bit Full-Adder circuit.

The truth table for it is as follows:

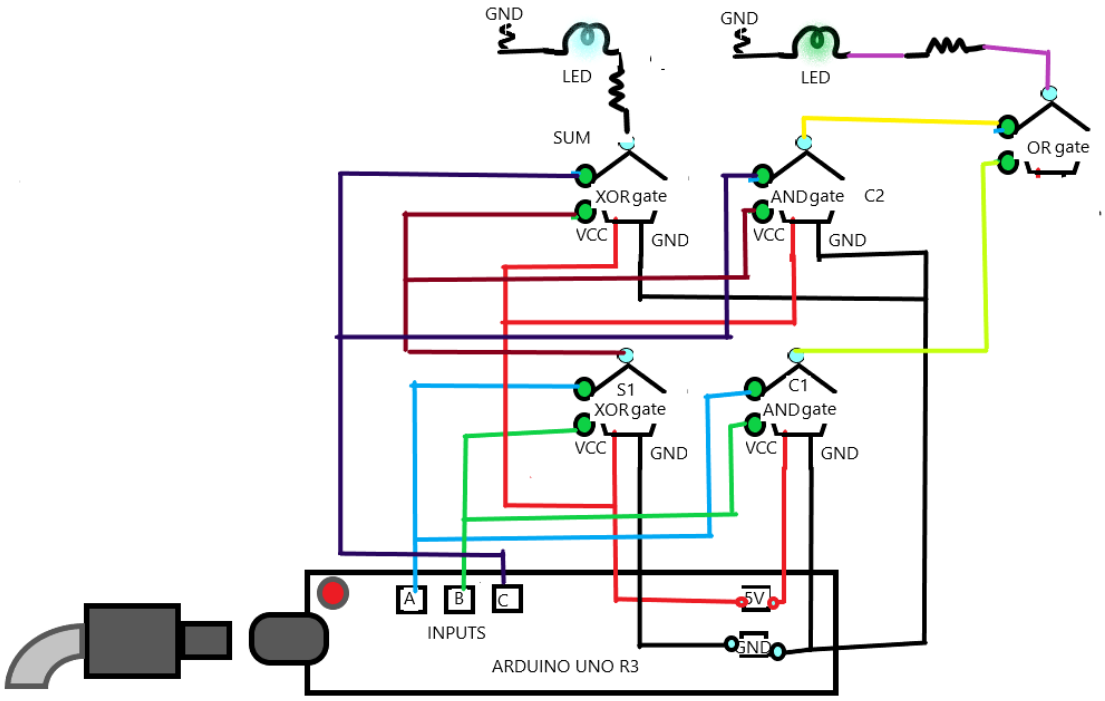
INPUT A	INPUT B	CARRY C-IN	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tinkercad Link with Circuit

https://www.tinkercad.com/things/34l3cK2UHRd-lab-2-part-d/editel?sharecode=9qnxqdDTxrdC0AyvcwErbVscj4nnmzwHtZ7VYnqEkGE

Circuit Diagram

Note that the OR gate is made using XOR gate and AND gate



ALL LINKS TO DSM LAB 2

A. https://www.tinkercad.com/things/7ZGEHiobeg1-ingenious-robo-snaget/editel?sharecode=taRk4RedQc3U-O5zcjMGS118kep27R3KpHUJFGAZC00
B. https://www.tinkercad.com/things/9WNjs84oLZ2-lab-2-part-b/editel?sharecode=0Y6iRXrrlPG9fjs-Ogs_C9mSFqLJAAXaWHxsdFZ0IYA

C. https://www.tinkercad.com/things/cWUd12uFbRa-lab-2-part-c/editel?sharecode=tj4JJEYMIz2LQIZCJWOgLqHbhonoi_nNB3hg-XslXSs
D. https://www.tinkercad.com/things/34l3cK2UHRd-lab-2-part-d/editel?sharecode=9qnxqdDTxrdC0AyvcwErbVscj4nnmzwHtZ7VYnqEkGE