LAB REPORT: 2

Name: Arghya Roy

Roll Number: 2021115008

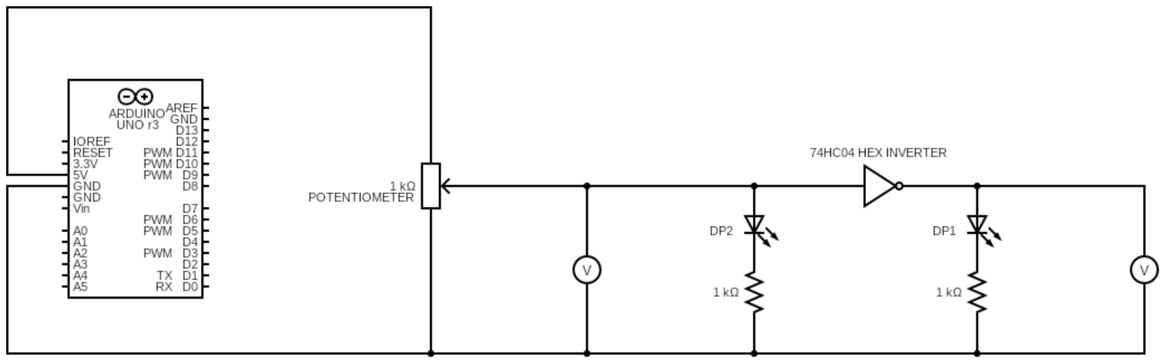
Group: 8

Part A: Logic Levels

Aim/Objective of the experiment: To find the tipping point voltage for the Integrated Circuit (IC)

Electronic components used: 1 Arduino board, two 1 kilo ohm resistors, 2 LEDs, 1 breadboard, 2 multimeters, 1 hex inverter(74HC04), 1 potentiometer, wires

Reference Circuit:



Procedure:

1. The circuit is set up, as shown in the reference figure above, on the breadboard
2. The potentiometer shaft is turned to one end so that the multimeter reads 0V.

3. The LEDs DP1 and DP2 are connected with appropriate resistors. DP2 glows.
4. The potentiometer shaft is gradually rotated up to the other end and the transitions in DP1 and DP2 are tabulated.

Conclusion:

The following is observed.

	Low	High
Input	2.5mV to 2.44V	2.44V to 5V
Output	1.44V to 5V	0V to 2.5V

So, we can conclude that 2.44V is the tipping point of the IC.

TinderCAD simulation: https://www.tinkercad.com/things/9W1VOuhvDhW-lab-2-a/editel?sharecode=bW0ZshfTIpyGoKpM7CrUUtsWwyTODBDd-Dg77t73trc

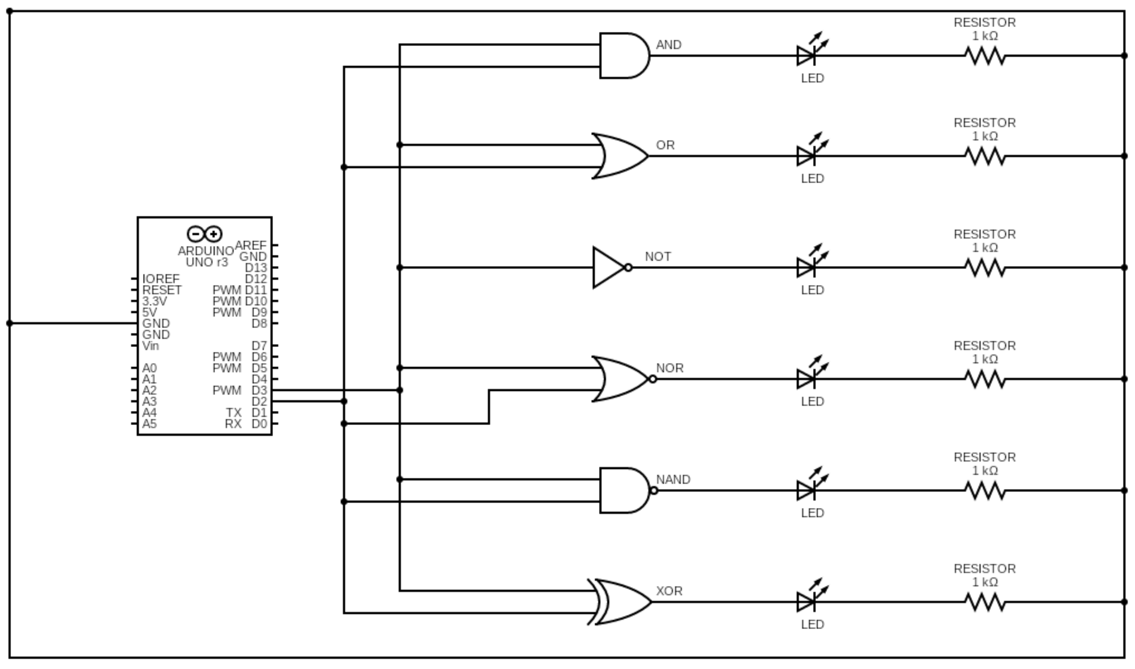
---

Part B: Verifying Truth Tables of Logic Gates

Aim/Objective of the experiment: To verify the truth tables of logic gates

Electronic components used: 1 Arduino board, six 1 kilo ohm resistors, 6 LEDs, 1 breadboard, 1 quad AND gate(74HC08), 1 quad OR gate(74HC32), 1 hex inverter(74HC04), 1 quad NOR gate(74HC02), 1 quad NAND gate(74HC00), 1 quad XOR gate(74HC86), wires

Reference Circuit:



<u>Procedure</u>:

1. The IC is placed on the breadboard and V_{cc} and *Gnd* connections are given to it.
2. The inputs for values of A and B are taken from the serial monitor and are routed to the input pins of the IC.
3. An LED is connected with the appropriate resistor to the output of the respective gate.
4. The output of the respective gate is noted for different values of input in a truth table.
5. Steps 3 and 4 are repeated for each gate.
6. The code used:

```
int x, y, chance;

void setup()
{
  pinMode(3, OUTPUT);
  pinMode(2, OUTPUT);
  Serial.begin(9600);
  chance=1;
}

void loop()
```

```
{
  if(Serial.available()>0 && chance==1)
  {
    x=Serial.read();
    x=x-'0';
    chance=2;
    digitalWrite(3,x);
  }

  if(Serial.available()>0 && chance==2)
  {
    y=Serial.read();
    y=y-'0';
    chance=1;
    digitalWrite(2,y);
  }

  delay(100);
}
```

<u>Conclusion</u>:

The truth tables:

A	B	AND	OR	NOT (A')	NOR	NAND	XOR
0	0	0	0	1	1	1	0
0	1	0	1	1	0	1	1
1	0	0	1	0	0	1	1
1	1	1	1	0	0	0	0

Thus, the truth tables of all the six gates are verified.

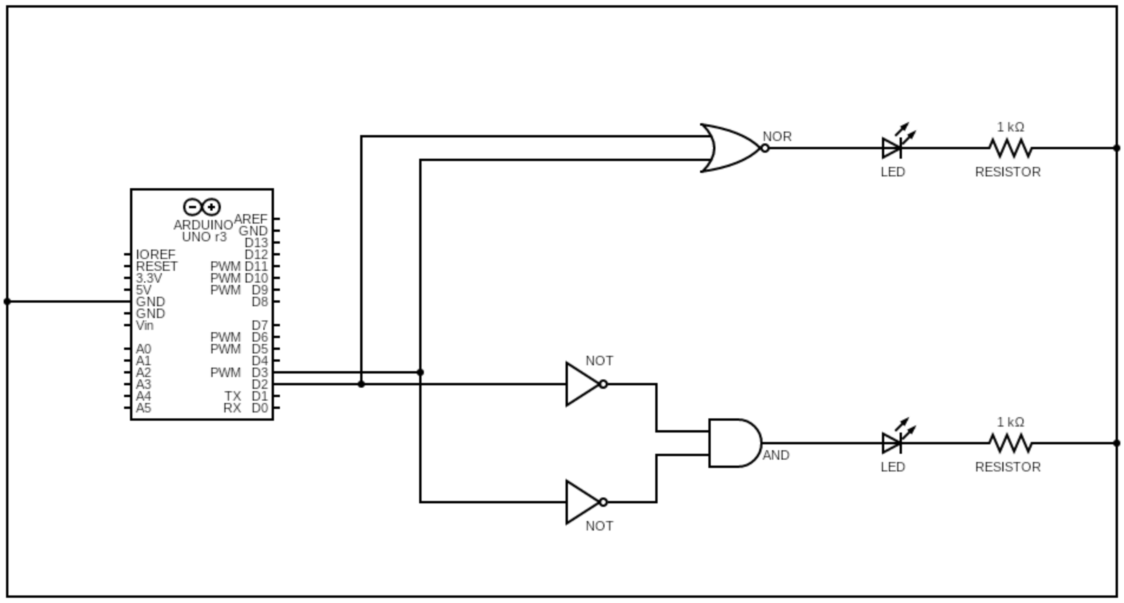
<u>TinkerCAD Simulation</u>: https://www.tinkercad.com/things/1RlCjbUv6Aj-stunning-kasi/editel?sharecode=5-FYFQ3np_FeciURfFBcz4CZs-ORCJ7q3ZgBaQoKNc0

Part C: De Morgan's Law

Aim/Objective of the experiment: To verify De Morgan's Law

Electronic components used: 1 Arduino board, two 1 kilo ohm resistors, 2 LEDs, 1 breadboard, 1 quad AND gate(74HC08), 1 hex inverter(74HC04), 1 quad NOR gate(74HC02), wires

Reference Circuit:



Procedure:

To verify (A+B)' = A'·B'

1. A circuit is set up consisting of two NOT gates and one AND gate to perform the function Y = A'·B'
2. The truth table of the circuit is obtained by noting the output of the function for different values of A and B. It is verified that the output of the function is same as that of the NOR gate.
3. The code used:

```
int x,y,chance;
```

```
void setup()
{
  pinMode(3, OUTPUT);
  pinMode(2, OUTPUT);
  Serial.begin(9600);
  chance=1;
}

void loop()
{
  if(Serial.available()>0 && chance==1)
  {
    x=Serial.read();
    x=x-'0';
    chance=2;
    digitalWrite(3,x);
  }

  if(Serial.available()>0 && chance==2)
  {
    y=Serial.read();
    y=y-'0';
    chance=1;
    digitalWrite(2,y);
  }

  delay(100);
}
```

Conclusion:

The truth tables:

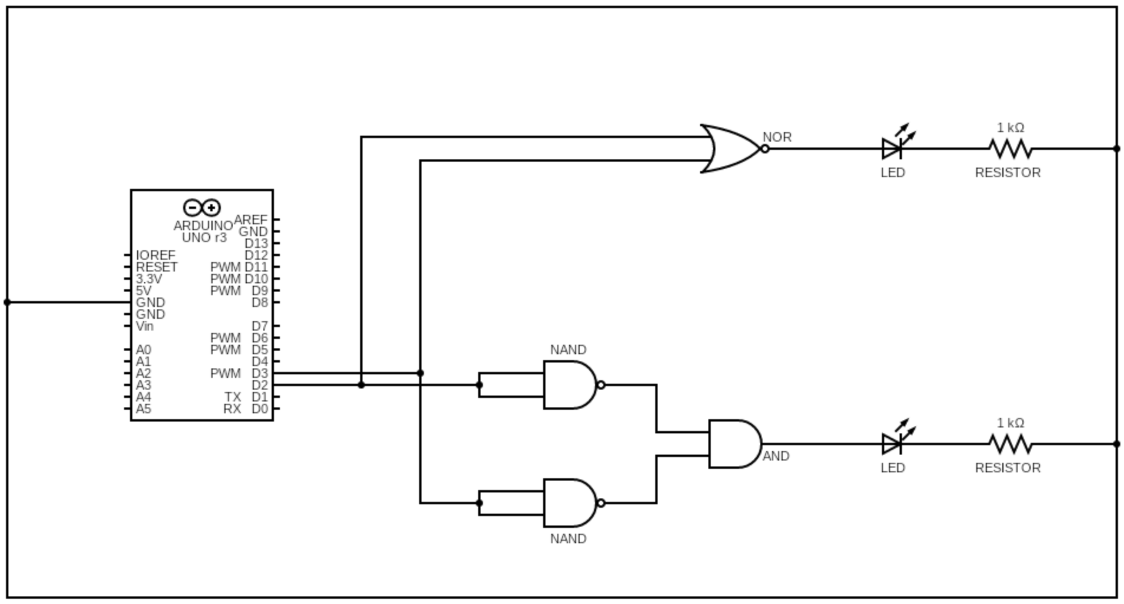
A	B	A’·B’	(A+B)’
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Thus, De Morgan’s first law is verified.

TinkerCAD Simulation: https://www.tinkercad.com/things/0uMZQM3cFWh-lab-2c-de-morgans-law/editel?sharecode=9MgFQ3TsyE8bViq9meUC_z3Qh_qe3f2_AzR0l_DMPNM

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?

Ans. If the same variable is passed as both the inputs of the NAND gate, we will get the complement of the variable and thus, the function of the NOT gate is performed.

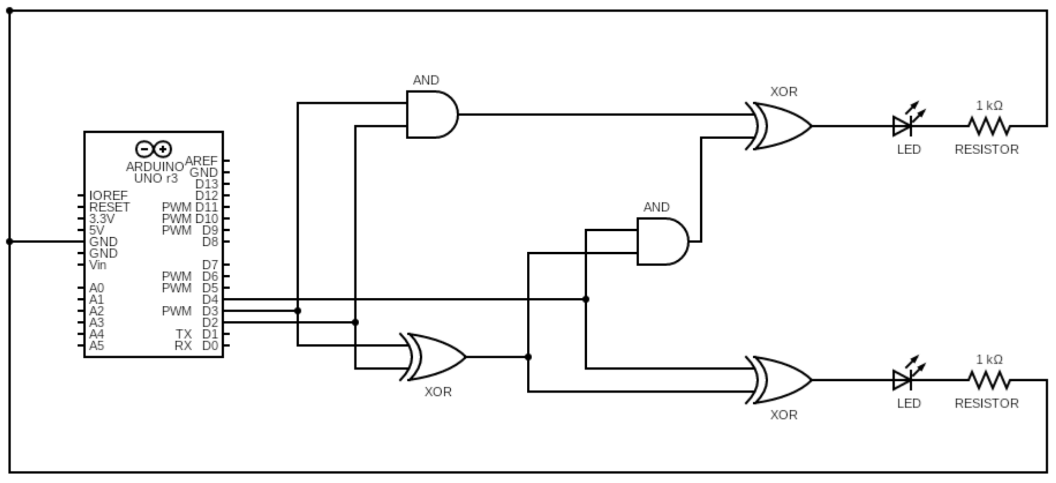


Part D: Binary Full Adder

Aim/Objective of the experiment: To verify the truth table of a full adder

Electronic components used: 1 Arduino board, 1 74HC86 XOR gate, 1 74HC08 AND gate, five 1 kilo ohm resistors, 5 LEDs, wires

Reference Circuit:



Procedure:

1. The circuit of a half-adder is set up using an XOR gate and an AND gate. The inputs A and B are applied from two input pins and the outputs S1 and C1 are observed on two LED displays for all combinations of the inputs. These values are tabulated, and the operation of the half adder is verified.
2. Another half adder is set up using another XOR gate and another AND gate out of the same ICs used in Step 1, and the C input and the S1 input generated by the first half adder are connected as its inputs to generate the final SUM output and the C2 output.
3. The final carry output is generated from the intermediate carry outputs C1 and C2, using the unused gates in the XOR and AND ICs deployed so far.
4. The truth table is verified experimentally by applying the inputs A, B and C through three input pins and displaying the S1, C1, C2, SUM and CARRY outputs.

5. The code used:

```
int x,y,z;

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

void loop()
{

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

    digitalWrite(3,x);
  }

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

    digitalWrite(2,y);
  }

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

    digitalWrite(4,z);
  }

  delay(100);

}
```

Conclusion:

The truth table:

A	B	C	C1	C2	S1	CARRY	SUM
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1
0	1	0	0	0	1	0	1
0	1	1	0	1	1	1	0
1	0	0	0	0	1	0	1
1	0	1	0	1	1	1	0
1	1	0	1	0	0	1	0
1	1	1	1	0	0	1	1

Thus, the truth table of binary full adder is verified.

TinkerCAD Simulation: https://www.tinkercad.com/things/eyjnZbIusWH-lab-2d-full-adder/editel?sharecode=tpRQOXIHKTqP4WLrqmL60OiRKluhz8wjHGsGyy4Y-mw