LAB REPORT-2

Name:K.Lakshmi Nirmala
Roll No: 2021101126
Group Number: G6

PART-A

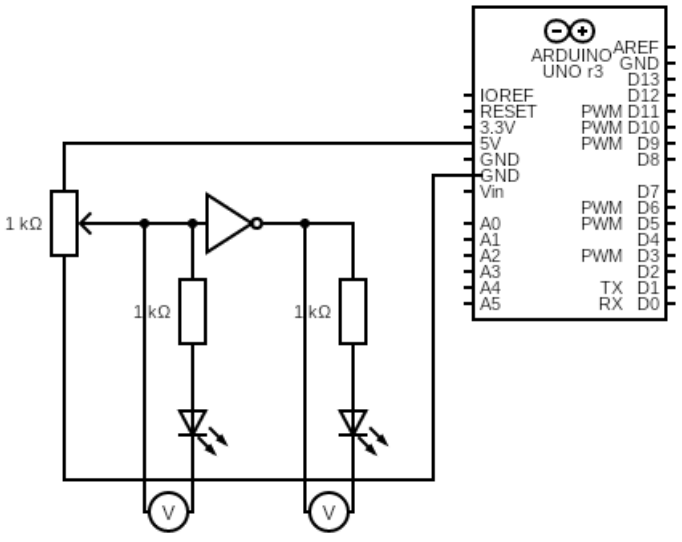
AIM/OBJECTIVE OF THE EXPERIEMENT:

To find the tipping voltage and to check the logic levels of the logic gate in the circuit.

ELECTRONIC COMPONENTS USED:

1) Ardunio UNO3
2) 2 Voltage multimeters
3) 1kΩ potentiometer
4) 2 Resistors
5) 1 Led
6) 74HC04 IC (NOT gate)

REFERENCE CIRCUIT:



PROCEDURE:

1)Place the IC on the breadboard and give Vcc and GND(ground) connections to it.

2)Connect the potentiometer 1st terminal to the vcc connecting wire and connect 2nd terminal to the ground connection wire through the breadboard .

3) Now connect the wiper point of the potentiometer to the any input of the NOT gate IC (say input 1).

4)Now add one multimeter to find input voltage.Positive terminal of the multimeter connect to the input wire ,negative terminal connect to the gnd connecting wire and set up led1(DP1) with connecting to resistor.

5)Now add another multimeter to find output voltage.Positive terminal of the multimeter connects to the output(say ouput1),negative terminal to the gnd connecting wire

6) Setup led2(DP2) from the output connecting wire connect resistor from the cathode as shown in the diagram.

7) Now start simulation note the readings from the both multimeter as input low(IL),input high(IH),output low(OL), outputhigh(OH).

8) Note the tipping voltage where the two leds glown.

CONCLUSION:

1)The working of Not gate was observed. It converted high values to low and low values to high.

1) V(IL)=0.702V | 0.402V | 0.502V and V(IH)=4.04

2) V(OL)= 3.3V | 3.8V | 4.01V and V(OH)=4.87

3)The given range of input and output voltage as 0-5v range as:

$$0 \le V_{OL} \le 0.4;\ 0 \le V_{IL} \le 0.8;\ 2.0 \le V_{IH} \le 5.0;\ 2.4 \le V_{OH} \le 5.0$$

4)Finally,the tipping voltage was found to be **2.36V.**

LINK FOR THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/2TsJa4srnaq-lab2-part1/editel?sharecode=VJae7uMAVSUmWYc3geGImB1jiEJ5t7z3l8QrCtsOEVU

PART-B

VERIFYING THE TRUTH TABLE OF LOGIC GATES

AIM/OBJECTIVE OF THE EXPERIEMENT:

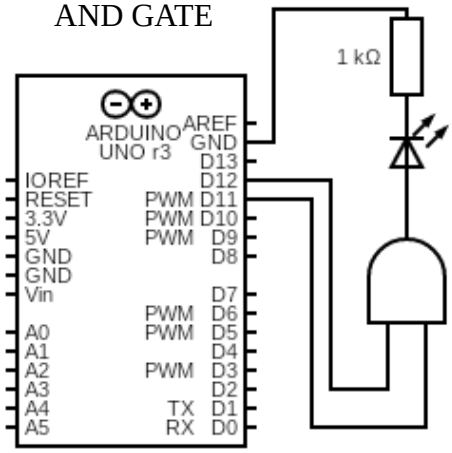
To verify the truth table of logic gates: (NOT, OR, AND, XOR, NOT, NAND) using TTL 74XX family of Ics.

ELECTRONIC COMPONENTS USED:

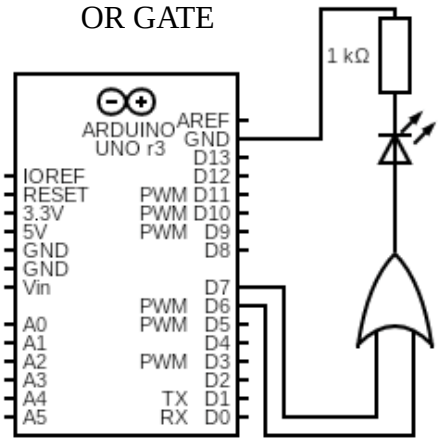
1)Ardunio UNO R3
2)Breadboard
3)TTL 74XX family of IC s for different logic gates
4)LED s
5)Resistors(1kΩ)
6)Connecting wires

REFERENCE CIRCUITS:

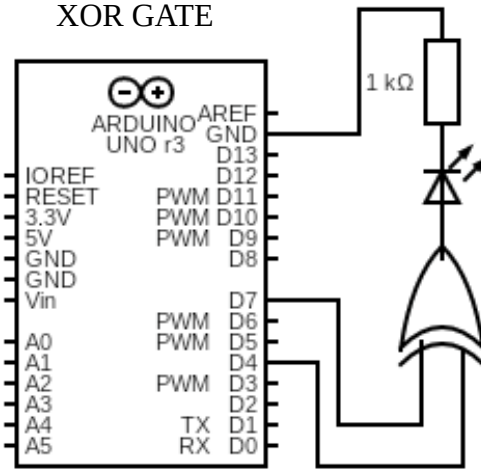
AND GATE



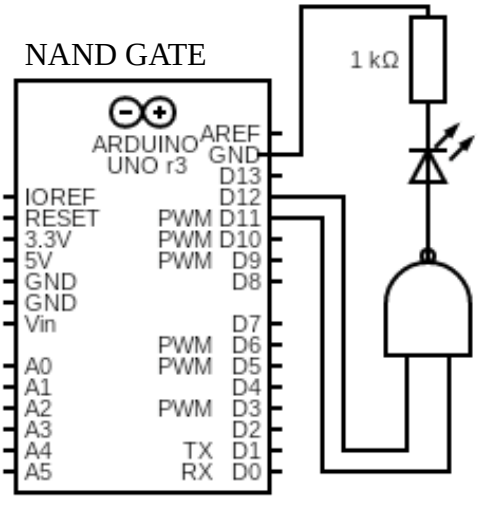
OR GATE



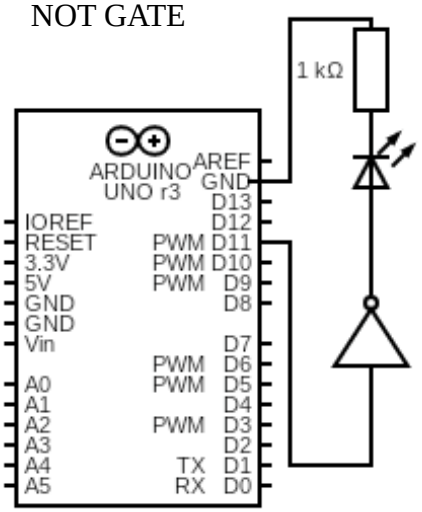
XOR GATE



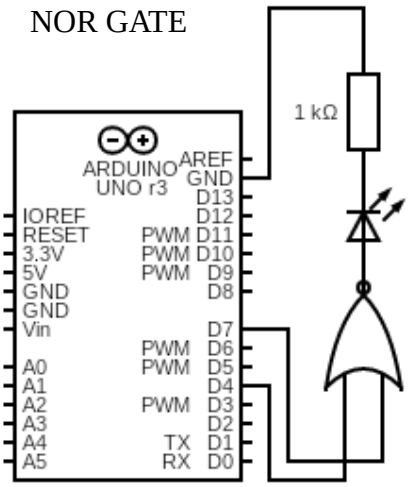
NAND GATE



NOT GATE



NOR GATE



PROCEDURE:

1)For implementing AND gate, place the IC on the breadboard and give Vcc and GND(ground) connections to it.

2)Connect any two digital pins of the Ardunio to any two input pins(say input 1A and input 1B) of the IC through the wires.

3)Connect the LED to the respectuve output pin(here output 1) of the IC with the help of a resistor.

4)Write the code in the code section and start simulation to check for different inputs of the truth table and note their outputs. For the AND gate, we could observe that the LED glows only when both the inputs are in HIGH state.

5)Repeat the same steps to verify the truth table by chnaging the logic gates(OR,NAND,NOR,XOR,NOT). Note that for NOT gate we need to giv0only one input.

6)Now we have to write the code in code section in tinker cad

Code: code for NAND,XOR and NOR gate

```
void setup()
{
        // NAND Gate
        pinMode(11,OUTPUT);
        pinMode(12,OUTPUT);
        // XOR Gate
        pinMode(6,OUTPUT);
        pinMode(7,OUTPUT);
        // NOR GATE
        pinMode(4,OUTPUT);
        pinMode(8,OUTPUT);

}

void loop()
{
                //NAND gate
                digitalWrite(11, LOW);
                digitalWrite(12, HIGH);
                //XOR gate
                digitalWrite(6, LOW);
                digitalWrite(7, HIGH);
                // NOR GATE
                pinMode(4,LOW);
                pinMode(8,HIGH);
                delay(2000); // Wait for 2000 millisecond(s)

                //NAND gate
                digitalWrite(11, HIGH);
                digitalWrite(12, LOW);
                //XOR gate
```

```
digitalWrite(6, HIGH);
digitalWrite(7, LOW);
// NOR GATE
pinMode(4,HIGH);
pinMode(8,LOW);

delay(2000); // Wait for 2000 millisecond(s)

//NAND gate
digitalWrite(11, HIGH);
digitalWrite(12, HIGH);
//XOR gate
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
// NOR GATE
pinMode(4,HIGH);
pinMode(8,HIGH);

delay(2000); // Wait for 2000 millisecond(s)

//NAND gate
digitalWrite(11, LOW);
digitalWrite(12, LOW);
//XOR gate
digitalWrite(6, LOW);
digitalWrite(7, LOW);
// NOR GATE
pinMode(4,LOW);
pinMode(8,LOW);

 delay(2000); // Wait for 2000 millisecond(s)
}
```

CODE: code for AND,OR and NOT gate.

```
void setup()
{
	//AND GATE
	pinMode(12, OUTPUT);
	pinMode(11, OUTPUT);
	//OR GATE
	pinMode(7, OUTPUT);
	pinMode(6, OUTPUT);
	//NOT GATE
	pinMode(2, OUTPUT);
}

void loop()
{
	//AND gate
	digitalWrite(12, LOW);
	digitalWrite(11, HIGH);
	//OR gate
	digitalWrite(7, LOW);
	digitalWrite(6, HIGH);
	//NOT GATE
	digitalWrite(2,LOW);
	delay(2000); // Wait for 2000 millisecond(s)
	//AND gate
	digitalWrite(12, HIGH);
	digitalWrite(11, LOW);
	//OR gate
	digitalWrite(7, HIGH);
	digitalWrite(6, LOW);
	//NOT GATE
	digitalWrite(2,HIGH);
```

```
delay(2000); // Wait for 2000 millisecond(s)
//AND gate
digitalWrite(12, HIGH);
digitalWrite(11, HIGH);
//OR gate
digitalWrite(7, HIGH);
digitalWrite(6, HIGH);
//NOT GATE
digitalWrite(2,LOW);
delay(2000); // Wait for 2000 millisecond(s)
//AND gate
digitalWrite(12, LOW);
digitalWrite(11, LOW);
//OR gate
digitalWrite(7, LOW);
digitalWrite(6, LOW);
//NOT GATE
digitalWrite(2,HIGH);
delay(2000); // Wait for 2000 millisecond(s)



}
```

6)Then give the input values and take output from the code.

7)This is the code we have to enter in code section to verify truth table.

CONCLUSION:

The truth table of the 6 logic gates are observed as follows:

Table 2.1: Truth table for AND,NAND,XOR,NOR logic gates.

INPUT		OUTPUT				
A	B	AND	OR	NAND	XOR	NOR
0	0	0	0	1	0	1
0	1	0	1	1	1	0
1	0	0	1	1	1	0
1	1	1	1	0	0	0

Table 2.2 : Truth table for NOT gate.

INPUT	OUTPUT
1	0
0	1

Therefore, we have successfully verified the truth table of the 6 logic gates.

LINK FOR THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/2MwjCM9GIoq-lab2-partb/editel?sharecode=wXmeyp5eMNJOCclOy7kWLhZzZKGaXBjrVRv6A49xES0

PART-C

De Morgan's Law

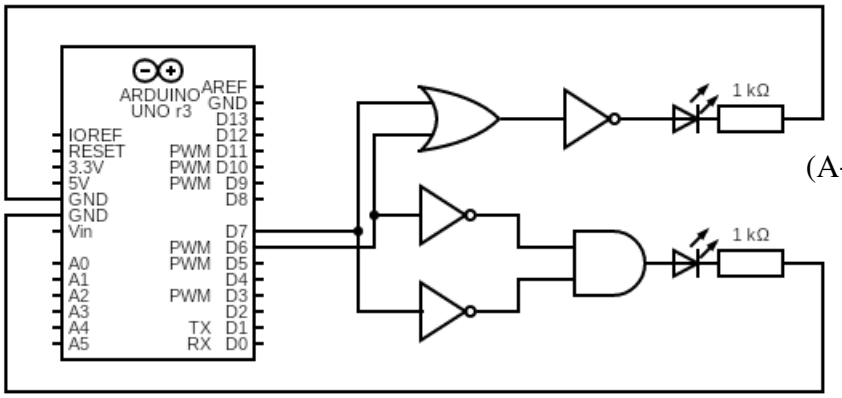
AIM/OBJECTIVE OF THE EXPERIEMENT:

TO verify De Morgan's theorems which state that$(A+B)^1 = A^1+B^1$ and $(A.B)^1 = A^1+B^1$.

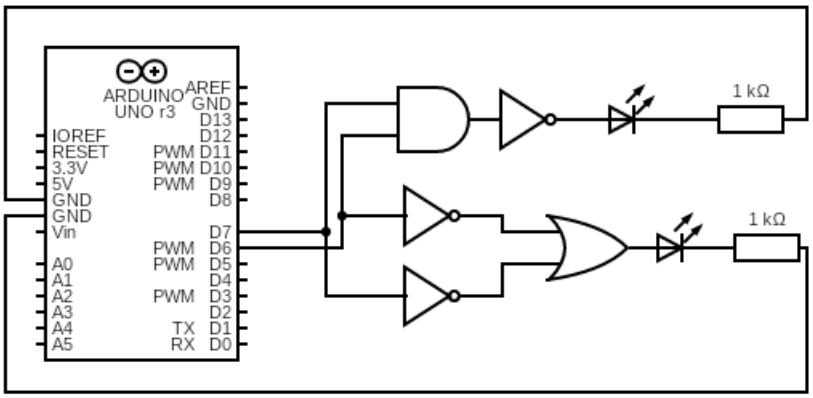
ELECTRONIC COMPONENTS USED:

1) Arduino UNO R3
2) Breadboard
3)LEDs
4)Resistors of 1 kilo ohm resistance
5)Connecting wires
6)TTL 74XX family of ICs for AND,OR and NOT logic gates

REFERENCE CIRCUITS:



$(A+B)^1 = A^1 . B^1$



$(A \cdot B)^1 = A^1 + B^1$

PROCEDURE:

For verifying $(A+B)^1 = A^1 . B^1$

1) Place the rescpective IC chips of OR, AND and NOT gates on the breadboard and give Vcc and GND (ground) connections to the logic gates.

2)Now connect any two digital pins of the Arduino to any two input pins (say input 4A and input 4B) of the OR gate.

3)Connect the output pin (say output 4) to the input pin of the NOT gate. The respective output pin of the NOT gate is connected to the LED with the help of a resistor.

4)This output gives the value of $(A + B)^1$.

(We can also verif this using NOR gate).

5) Now, two input pins of the NOT gate are connected to the same input pins of the Aduino.

6) The two output pins are connected to the AND gate as the inputs and the output pin of the AND gate is connected to the LED with the help of a resistor.This output gives the value of $A^1 . B^1$

7) Write the code and start simulation to check for different inputs of the truth table and note their outputs.

8)We could observe the LEDs blinking at the same time which verifies the above equation.

For verifying $(A \cdot B)^1 = A^1 + B^1$

9)Repeat the above steps using an OR gate instead of an AND gate.
Here $(A \cdot B)^1 = A^1 + B^1$ is verified.Here $(A \cdot B)^1$can also be verified using NAND gate.

10) Now in the code section we have to enter the code and take output values from it and then verify it.

CODE: Code to verify de morgan's law for both ardunios

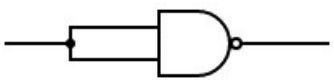
```
void setup()
{
        pinMode(6, OUTPUT);
        pinMode(7, OUTPUT);
}

void loop()
{
        digitalWrite(6, HIGH);
        digitalWrite(7, LOW);
        delay(1000); // Wait for 1000 millisecond(s)


        digitalWrite(6, LOW);
        digitalWrite(7, HIGH);
        delay(1000); // Wait for 1000 millisecond(s)


        digitalWrite(6, LOW);
        digitalWrite(7, LOW);
        delay(1000); // Wait for 1000 millisecond(s)


        digitalWrite(6, HIGH);
        digitalWrite(7, HIGH);
        delay(1000); // Wait for 1000 millisecond(s)
}
```

QUESTION

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?

ANSWER:

We know that logic gate takes 0 or 1 and takes output as per the truth table. As like that taking the inputs of NAND gate as either 0 or 1, it gives the same output and performs as like the NOR gate.



CONCLUSION:

Table:3.1-$(A+B)^1 = A^1 \cdot B^1$

INPUT		OUTPUT			
A	B	NOT A	NOT B	A^1 AND B^1	NOR
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	0	0

Table:3.2- $(A \cdot B)^1 = A^1 + B^1$

INPUT		OUTPUT			
A	B	NOT A	NOT B	A^1 OR B^1	NAND
0	0	1	1	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	1	1

We have been verified the De-morgan's law using truthtable in tinkercad.

LINK OF THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/6hN03460lfa-lab2-part3/editel?sharecode=wn90B-64xwudM33ce0N0_4r8E-fHSZ4MUD5jLd9veI0

PART – D

BINARY FULL ADDER

AIM/OBJECTIVE OF THE EXPERIMENT :

To implement a full adder using half adder circuits (XOR and AND gates)

ELECTRONIC COMPONENTS USED:

1)Arduino UNO R3

2)Breadboard

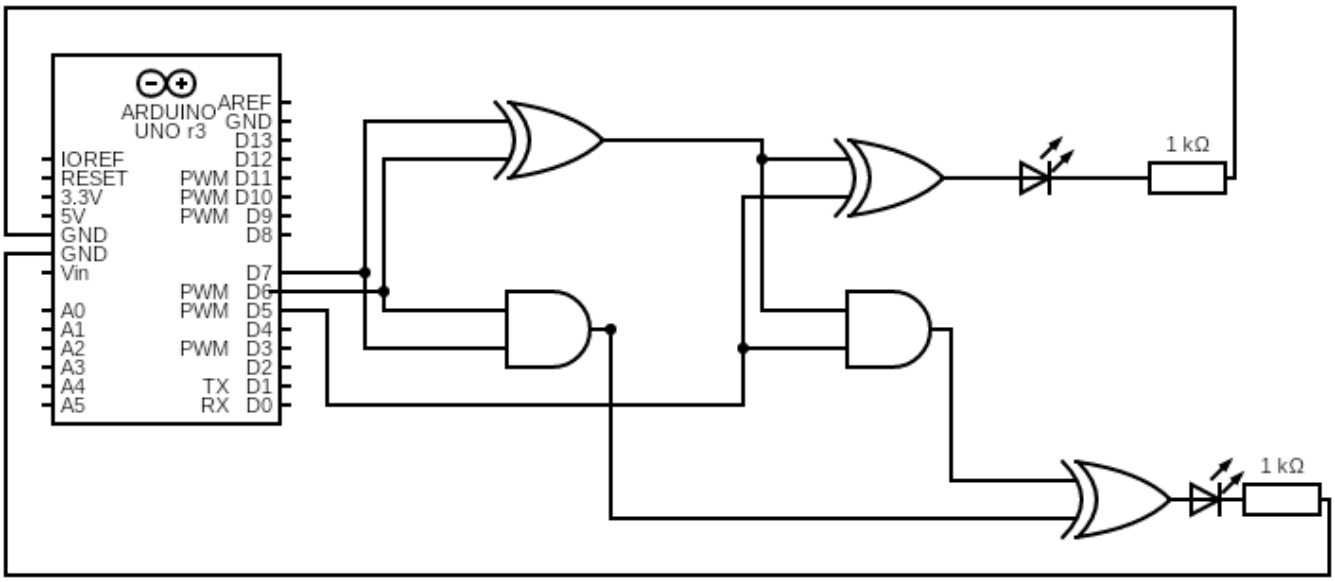
3)LEDs

4)Resistors

5) TTL 74XX family of ICs for XOR and AND logic gates

6)Connecting wires

REFERENCE CIRCUIT:



PROCEDURE:

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 the ouputs are S1 and C1 respectively.

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 ouput generated by the Half adder as its inputs to generate the final SUM output and the C2 output.

3) Connect the outputs C1 and C2 as the inputs to the same XOR gate used in step1. The output generates the final CARRY output. So,that the complete realisation of the Full Adder is possiblewithout necessitating a third IC.

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.

5)Now enter the code in code section.

CODE:

```
void setup()

{

        pinMode(5, OUTPUT);

        pinMode(6, OUTPUT);

        pinMode(7, OUTPUT);

}

void loop()

{

        digitalWrite(5, LOW);

        digitalWrite(6, LOW);

        digitalWrite(7, LOW);

        delay(1000); // Wait for 1000 millisecond(s)

       digitalWrite(5, LOW);

        digitalWrite(6, LOW);

        digitalWrite(7, HIGH);

        delay(1000); // Wait for 1000 millisecond(s)

       digitalWrite(5, LOW);

        digitalWrite(6, HIGH);

        digitalWrite(7, LOW);
```

```
    delay(1000); // Wait for 1000 millisecond(s)

    digitalWrite(5, LOW);
    digitalWrite(6, HIGH);
    digitalWrite(7, HIGH);
    delay(1000); // Wait for 1000 millisecond(s)

    digitalWrite(5, HIGH);
    digitalWrite(6, LOW);
    digitalWrite(7, LOW);
    delay(1000); // Wait for 1000 millisecond(s)

    digitalWrite(5, HIGH);
    digitalWrite(6, LOW);
    digitalWrite(7, HIGH);
    delay(1000); // Wait for 1000 millisecond(s)

    digitalWrite(5, HIGH);
    digitalWrite(6, HIGH);
    digitalWrite(7, LOW);
    delay(1000); // Wait for 1000 millisecond(s)

    digitalWrite(5, HIGH);
    digitalWrite(6, HIGH);
    digitalWrite(7, HIGH);
    delay(1000); // Wait for 1000 millisecond(s)
}
```

CONCLUSION:

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

Therefore , we have successfully implemented a full adder circuit using two half adder circuits and verified it using the truth table. Also the complete realisitation of the full adder became possible without necessitating a third IC.

LINK FOR THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/ayWmhBRH1fp-adder/editel?sharecode=wYB3VTKqrdIWzLWkeCOmkPgtoEHjfB4LXxi4fqQ6GO8