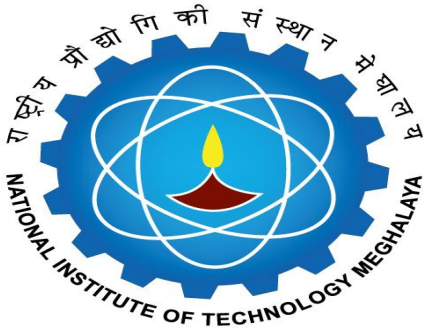


National Institute of Technology,Meghalaya

A PROJECT REPORT ON

Universal IC Tester for Analog and Digital ICs

SUBMITTED TO

Dr. Shubhankar Majumdar

Assistant Professor

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY,MEGHALAYA

SUBMITTED BY

IMON SHAHRIAR (B22EC001)

DEBASHISH NAYAK (B22EC030)

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY,MEGHALAYA

INTRODUCTION:

The main purpose of this project is to check logic gate IC and analog IC, whether the IC is working properly or not. The checking of the IC will be done using Arduino UNO. The main IC that are going to be checked are OR (7432), AND (7408), NAND (7400) , EXOR (7486) and EXNOR (747266) , OPAMP (lm741) , BJT , JFET.

COMPONENTS USED :

- ARDUINO UNO
- RESISTORS
- LEDS (RED, GREEN)
- BREADBOARD
- JUMPER WIRES

• LOGIC GATE (AND,OR,NAND,NOR,EX-OR,EX-NOR, LM741 Operational Amplifier)

ARDUINO UNO

Arduino is a single-board microcontroller meant to make the application more accessible which are interactive objects and its surroundings. The hardware features with an open-source hardware board designed around an 8-bit Atmel AVR microcontroller or a 32-bit Atmel ARM. Current models consists a USB interface, 6 analog input pins and 14 digital I/O pins that allows the user to attach various extension boards.

The Arduino Uno board is a microcontroller based on the ATmega328. It has 14 digital input/output pins in which 6 can be used as PWM outputs, a 16 MHz ceramic resonator, an ICSP header, a USB connection, 6 analog inputs, a power jack and a reset button. The code of the project were install in the Arduino. The arduino sends signals to LED lights when necessary and passes the signals. The connection is done according to the diagram.

LED LIGHTS

A light-emitting diode is a two-lead semiconductor light source. It is a p–n junction diode that emits light when activated. When a suitable voltage is applied to the leads, electrons are able to recombine with electron holes within the device, releasing energy in the form of photons. In our project, we used 4 LED for checking logic gates. If the lights are

on according to the respective truth table then the IC is working. However, if the IC is damaged or different IC which is not in our IC list then all the LED will turn off.

RESISTORS

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. In IC tester, the resisters are protecting LED

BREADBOARD

Breadboard is used as a prototyping tool to facilitate the temporary connection of electronic components without the need for soldering.

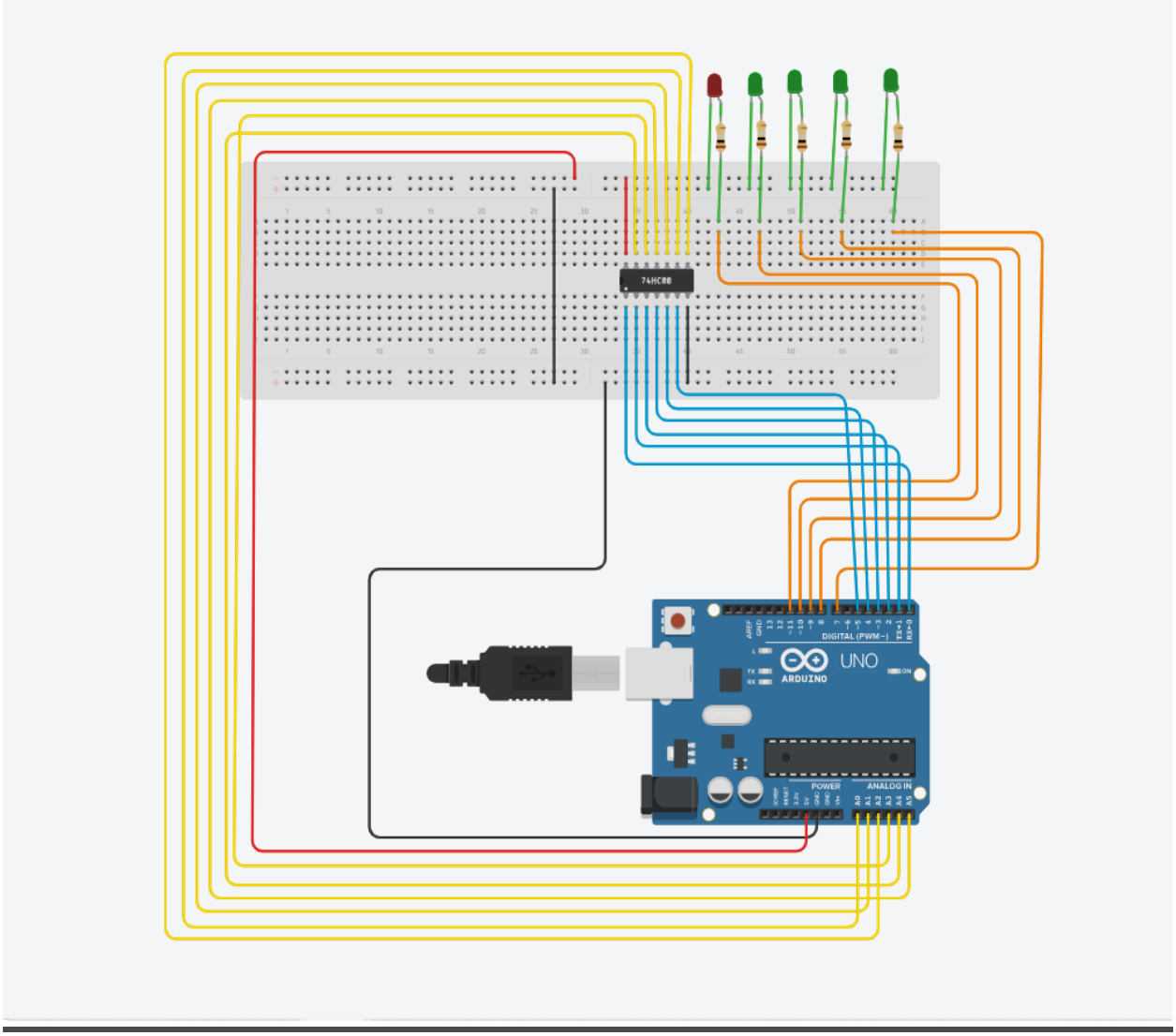
JUMPER WIRES

Jumper wires are used to establish electrical connections between components on the breadboard

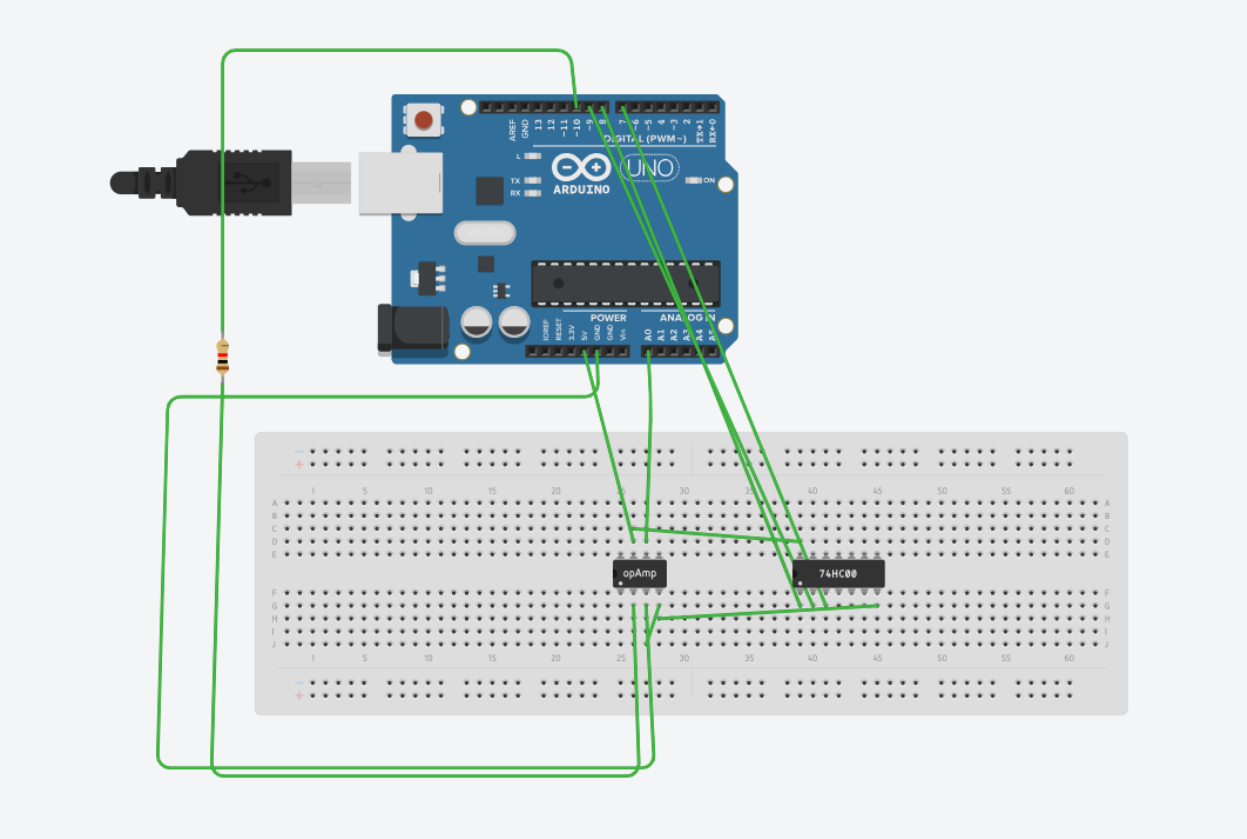
LOGIC GATE

We use here the basic logic gates such as nand, and, nor, or ,not, ex-or,ex-nor to check that it is working or not.

SCHEMATIC DIAGRAM (ONLY FOR DIGITAL IC



SCHEMATIC DIAGRAM (FOR ANALOG AND DIGITAL IC)



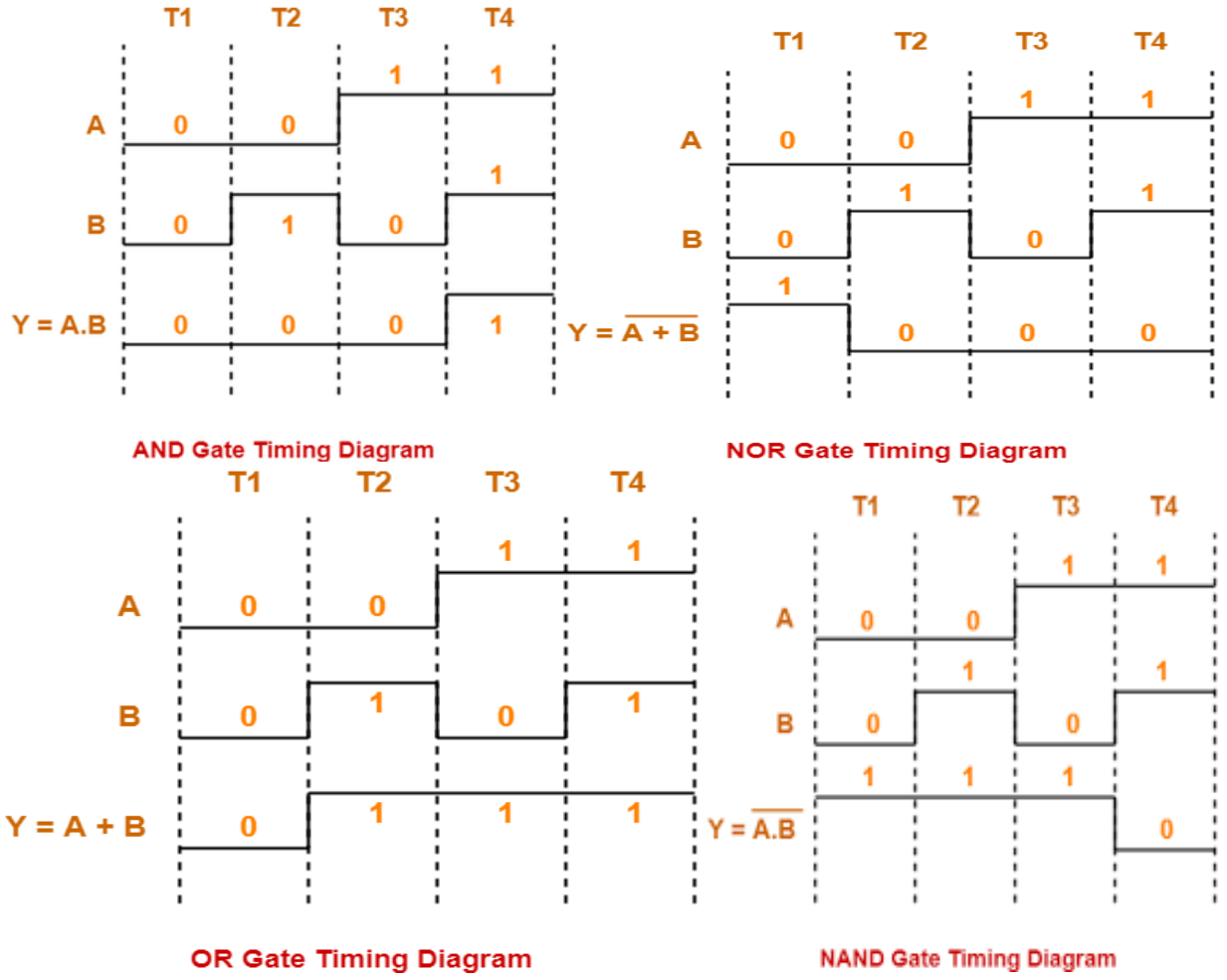
IMPLEMENTATION (ONLY FOR DIGITAL IC):

We connected the arduino uno, Buzzer and LEDs on a breadboard according to the code we have written in the code arduino ide. Different led's are connnected to the various pins of the arduino i.e the green led is connected to pin no. 7,8,9,10 Remaining connections are also done in the breadboard that is ground of the arduino is connected.

IMPLEMENTATION (FOR ANALOG AND DIGITAL IC):

This Arduino IC tester checks Op-Amps and basic logic ICs (NAND, NOR, AND, OR). It uses digital pins 8 and 9 for IC inputs, pin 7 for output, and A0 for Op-Amp analog output. Pin 12 controls an LED to indicate test results.The `setup` function initializes pins, runs tests, and displays results in the Serial Monitor. The `testOpAmp` function checks Op-Amp output voltage, and `testDigitalIC` verifies logic ICs by comparing actual outputs with expected results for all input combinations. The LED lights up for a pass; results are displayed via Serial Monitor. The loop remains empty as tests are only run once.

Timing Diagram



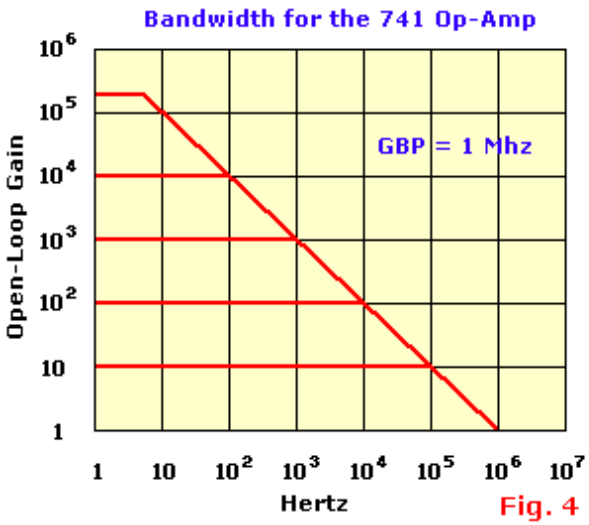


Fig. 4

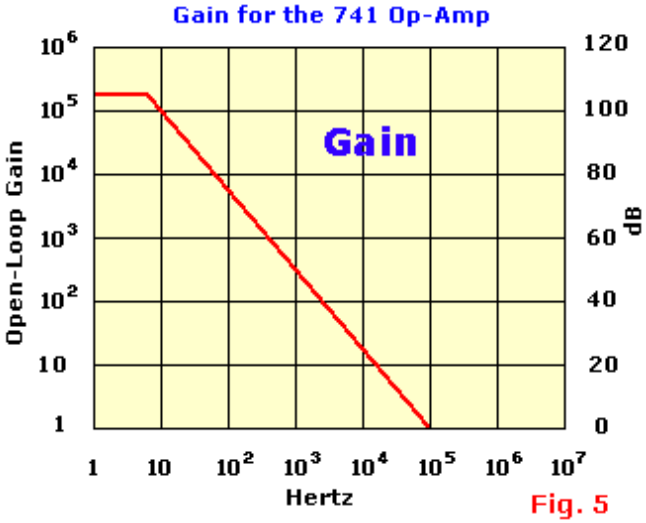


Fig. 5

Code Overview (ONLY FOR DIGITAL IC):

```
#include <LiquidCrystal.h>

const int pin0 = 0;

const int pin1 = 1;

const int pin2 = 2;

const int pin3 = 3;

const int pin4 = 4;

const int pin5 = 5;

const int pin6 = A0;

const int pin7 = A1;

const int pin8 = A2;
```

```
const int pin9 = A3;
const int pin10 = A4;
const int pin11 = A5;
const int led=7,led2=8,led3=9,led4=10, led5=11;
void setup() {
pinMode(pin0,OUTPUT);
pinMode(pin1,OUTPUT);
pinMode(pin2,INPUT);
pinMode(pin3,OUTPUT);
pinMode(pin4,OUTPUT);
pinMode(pin5,INPUT);
pinMode(pin6,OUTPUT);
pinMode(pin7,OUTPUT);
pinMode(pin8,INPUT);
pinMode(pin9,OUTPUT);
pinMode(pin10,OUTPUT);
pinMode(pin11,INPUT);
digitalWrite(pin0,HIGH);
digitalWrite(pin1,HIGH);
```

```
digitalWrite(pin3,HIGH);
digitalWrite(pin4,LOW);
digitalWrite(pin6,LOW);
digitalWrite(pin7,HIGH);
digitalWrite(pin9,LOW);
digitalWrite(pin10,LOW);
}
void loop()
{
//OR GATE
if(digitalRead(pin2)==HIGH)
{
   if(digitalRead(pin5)==HIGH)
  {
   if(digitalRead(pin8)==HIGH)
   {
     if(digitalRead(pin11)==LOW)
     {
digitalWrite(led, LOW); //if gate is in, light is off
```

```
digitalWrite(led2, HIGH); //Truth table values
digitalWrite(led3, HIGH);
digitalWrite(led4, HIGH);
digitalWrite(led5, HIGH);
      }
    }
  }
}
//And gate
if(digitalRead(pin2)==HIGH)
{
       if(digitalRead(pin5)==LOW)
  {
    if(digitalRead(pin8)==LOW)
    {
      if(digitalRead(pin11)==LOW)
      {
        digitalWrite(led, LOW); //if gate is in, light is off
                digitalWrite(led2, LOW); // AND gate truth table outputs are
0, 0, 0, 1 which is how the LEDs are set up
```

```
                digitalWrite(led3, LOW);
                digitalWrite(led4, HIGH);
                digitalWrite(led5, HIGH);
      }
    }
  }
}
//EX-NOR
if(digitalRead(pin2)==LOW)
{
    if(digitalRead(pin5)==HIGH)
  {
    if(digitalRead(pin8)==HIGH)
    {
      if(digitalRead(pin11)==LOW)
      {
digitalWrite(led, HIGH); //if gate is in, light is off
digitalWrite(led2, LOW); //Truth table values
digitalWrite(led3, LOW);
```

```
digitalWrite(led4, HIGH);
digitalWrite(led5, HIGH);
      }
    }
  }
}
//NAND GATE
  if(digitalRead(pin2)==LOW)
{
  if(digitalRead(pin5)==HIGH)
  {
    if(digitalRead(pin8)==HIGH)
    {
      if(digitalRead(pin11)==HIGH)
      {
        digitalWrite(led, HIGH); //if gate is in, light is off
digitalWrite(led2, HIGH); //Truth table values
digitalWrite(led3, HIGH);
digitalWrite(led4, LOW);
```

digitalWrite(led5, HIGH);

} } }}}

Code Overview (FOR ANALOG AND DIGITAL IC):

```
// IC Names
const char opAmpName[] = "LM741 Op-Amp";
const char nandICName[] = "7400 NAND Gate";
const char norICName[] = "7402 NOR Gate";
const char andICName[] = "7408 AND Gate";
const char orICName[] = "7432 OR Gate"; // New OR gate IC name

// Pin assignments for Digital ICs
const int inputPin1 = 8;   // Arduino output to IC input A
const int inputPin2 = 9;   // Arduino output to IC input B
const int outputPin = 7;   // IC output connected to Arduino input
const int ledPin = 12;     // LED to indicate pass/fail

// Pin assignment for Op-Amp
const int opAmpOutputPin = A0;  // Analog pin to read Op-Amp output

void setup() {
  Serial.begin(9600);

  // Configure pins for digital ICs
  pinMode(inputPin1, OUTPUT);
  pinMode(inputPin2, OUTPUT);
  pinMode(outputPin, INPUT);
  pinMode(ledPin, OUTPUT);

  // Initial test message
  Serial.println("Universal IC Tester Starting...");
  delay(1000);

  // Perform tests once
```

```
  testOpAmp();
  testDigitalIC(nandICName, 0);  // Test 7400 NAND gate
  testDigitalIC(norICName, 1);   // Test 7402 NOR gate
  testDigitalIC(andICName, 2);   // Test 7408 AND gate
  testDigitalIC(orICName, 3);    // Test 7432 OR gate
}

void testOpAmp() {
  Serial.print("Testing IC: ");
  Serial.println(opAmpName);

  int opAmpOutput = analogRead(opAmpOutputPin);  // Read analog output of Op-Amp
  float voltage = (opAmpOutput * 5.0) / 1023;    // Convert reading to voltage

  Serial.print("Op-Amp Output Voltage: ");
  Serial.print(voltage);
  Serial.println(" V");

  // Check the output voltage condition

  if (voltage > 1.6) {  // Adjust threshold based on testing
    Serial.println("Op-Amp is working correctly.");
  } else {
    Serial.println("Op-Amp might be faulty.");
  }
  delay(1000);
}

void testDigitalIC(const char *icName, int logicFunction) {
  Serial.print("Testing IC: ");
  Serial.println(icName);

  for (int i = 0; i < 4; i++) {
    // Set input combinations (00, 01, 10, 11)
    int input1 = (i >> 1) & 1;
    int input2 = i & 1;

    digitalWrite(inputPin1, input1);
    digitalWrite(inputPin2, input2);

    delay(100); // Allow time for the output to settle

    int output = digitalRead(outputPin);
    int expectedOutput = 0;
```

```
    // Determine expected output based on logic function
    switch (logicFunction) {
      case 0: // NAND
        expectedOutput = !(input1 && input2);
        break;
      case 1: // NOR
        expectedOutput = !(input1 || input2);
        break;
      case 2: // AND
        expectedOutput = input1 && input2;
        break;
      case 3: // OR
        expectedOutput = input1 || input2;
        break;
    }

    // Print the results to Serial Monitor
    Serial.print("Inputs: ");
    Serial.print(input1);
    Serial.print(", ");
    Serial.print(input2);
    Serial.print(" | Expected Output: ");
    Serial.print(expectedOutput);
    Serial.print(" | Actual Output: ");
    Serial.println(output);

    // Check if output matches expected result
    if (output == expectedOutput) {
      digitalWrite(ledPin, HIGH);  // Turn on LED if output is correct
      Serial.println("Result: IC is working correctly.");
    } else {
      digitalWrite(ledPin, LOW);   // Turn off LED if output is incorrect
      Serial.println("Result: IC might be faulty.");
    }

    delay(500);
  }
}

void loop() {
  // Empty loop since tests are run once in setup()
}
```

\

Advantages:

- Universal compatibility with most ICs.
- Time-efficient and accurate.
- Ideal for R&D, education, and troubleshooting.
- Portable and easy to use.

Applications:

- Educational purposes for understanding circuits.
- R&D labs for IC validation.
- Circuit debugging in engineering projects.

Error analysis :

• The project can deal with the five specific models. Error occurs when any other IC is used. So, if any IC of other model are used then the project will treat it as a damaged one.

• The project can check only one gate. Usually if a gate works, then the IC works. However, in some cases that may not be happen.

Conclusion:

- Universal IC Tester simplifies IC functionality testing.
- Supports both digital and analog ICs with high accuracy.
- Versatile tool for education, R&D, and maintenance.

.