Internet of Things

Interfacing various devices with Internet Of Things

Bachelor's Thesis for the Attainment of the Degree
Bachelor of Technology at the Integral University, Lucknow

Supervisor:	Dr. Mohd. Atif Siddiqui
Mentor:	Mr. Mohd Zaid
Study Program:	Bachelor of Technology (CSE)
Submitted by:	Abdul Saboor
Address:	164/125c, Haider Mirza Road
Matriculation Number:	2300101433
Submission Date:	20.06.2024

Abstract

JEL Classifications: XXX, XXX, XXX, XXX . . .
Keywords: KEYWORD1, KEYWORD2, . . .

Contents

List of Figures

List of Tables

1. Arduino UNO

1.1 Interfacing Devices With Arduino UNO

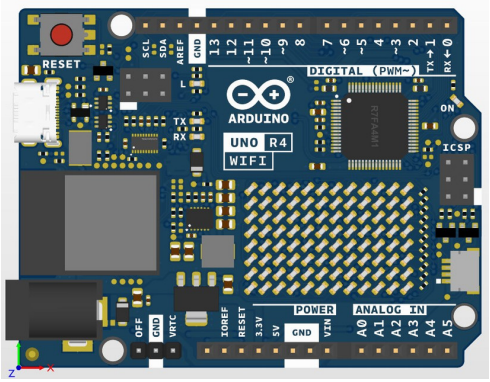

Figure 1: Arduino UNO

Arduino UNO is a peripheral device with a microcontroller inside. This device has 14 digital pins(0-13) and 6 Analog PINs(A0-A5). THe device also has PWM pins that can be used for PULSE Modulation. Arduino UNO can be powered by a USB cable or a 9V battery. C++ Language is used to code . Usually the program has Two Methods:void setup() and void loop(). loop() is the repeating code and setup() is done only initially.

Microcontroller	ATmega328P
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input Voltage Limits	6-20 V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (0.5 KB is used for Bootloader)
SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 MHz

1.1.1 Interface LED Blinking

An LED can be interfaced with ArduinoUNO to perform basic blinking functions.This is done by using the digital output of the ARduino UNO. A series nresistance of 1kohms is also added to protect LED from over current damage.

Technical Specifications:

Selection Guide

Part No.	Dice	Lens Type	Iv (mcd) [2] @ 20mA		Viewing Angle [1]
			Min.	Typ.	2θ1/2
WP7113SRD/D	Super Bright Red (GaAlAs)	RED DIFFUSED	180	250	30°

Figure 2: Extended Specification

Electrical / Optical Characteristics at TA=25°C

Symbol	Parameter	Device	Typ.	Max.	Units	Test Conditions
λpeak	Peak Wavelength	Super Bright Red	660		nm	I_F=20mA
λD [1]	Dominant Wavelength	Super Bright Red	640		nm	I_F=20mA
Δλ1/2	Spectral Line Half-width	Super Bright Red	20		nm	I_F=20mA
C	Capacitance	Super Bright Red	45		pF	V_F=0V;f=1MHz
V_F [2]	Forward Voltage	Super Bright Red	1.85	2.5	V	I_F=20mA
I_R	Reverse Current	Super Bright Red		10	uA	V_R = 5V

Notes:
1.Wavelength: +/-1nm.
2. Forward Voltage: +/-0.1V.

Figure 3: Extended Specification

CODE:

```
 //GLOBAL VARIABLES
 int LEDpin = 13;// LEDis connected in the port 13
int delayT = 1000;//chosen time interval in ms

void setup() {
pinMode(LEDpin, OUTPUT);// assign PORT 13 as output
}

void loop() {
digitalWrite(LEDpin, HIGH);// turn LED on
delay(delayT);// keep the LED on for 1 seconds
digitalWrite(LEDpin, LOW);// turns LED off
delay(delayT);//keeps LED off for one second
}
```

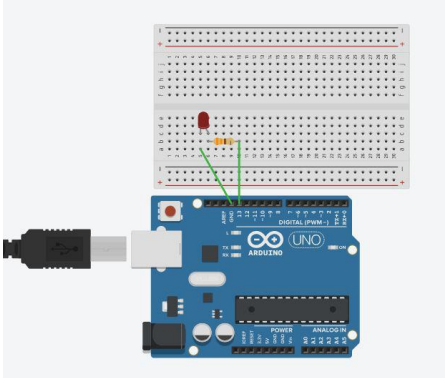

Figure 4: Circuit Diagram for A Blinking LED

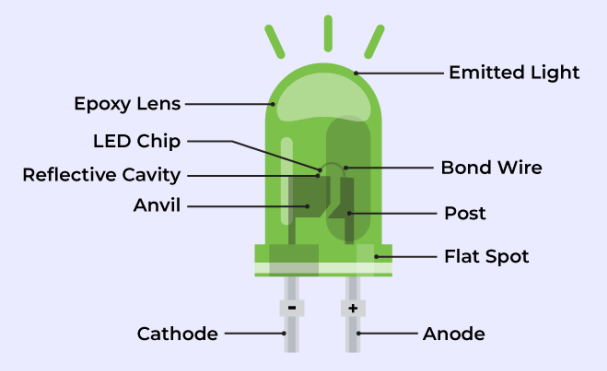

Figure 5: Caption

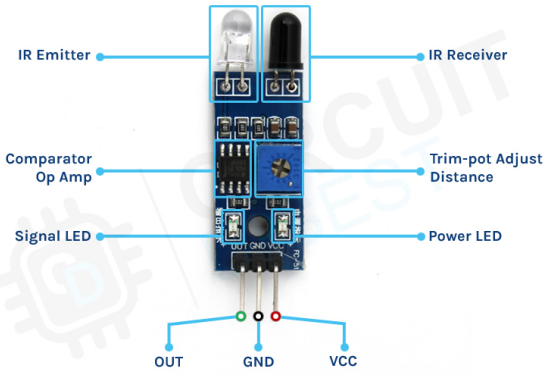

Figure 6: IR Sensor

1.1.2 Interface IR Sensor & Print Result On Serial Monitor

The IRSensor is an obstacle detection sensor. It is often referred as Proximity Sensor. In this project we used IR Sensor with One Infra-red transmitter and one Infrared receiver. The device contains 3 terminal pins i.e. Vcc, GND, Output. The device has the following Specifications

Working Voltage	5V
Working Current	43mA
Wavelength of IR Radiation used	940 nm to 950 nm

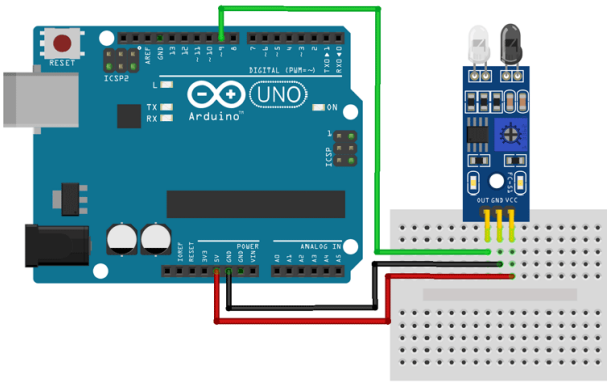

Figure 7: This is the Circuit Diagram for IRsensor

CODE:

```
    // Arduino IR Sensor Code
int IRSensor = 9; // connect ir sensor module to Arduino pin 9

void setup()
{
  Serial.begin(9800);
  pinMode(IRSensor, INPUT); // IR Sensor pin INPUT
  }
void loop()
{
  int sensorStatus = digitalRead(IRSensor);
  if(sensorStatus==1)
  Serial.println("obstacle detected");
  else
  Serial.println("Clear");

  delay(1000);
}
```

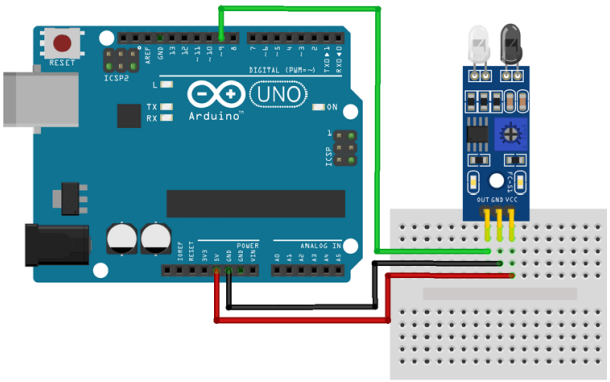

Figure 8: INBUILT LED BLink witb IR Sensor

1.1.3 Control An LED with Arduino and an IR Sensor

CODE:

```
    // Arduino IR Sensor Code
int IRSensor = 9; // connect ir sensor module to Arduino pin 9
int LED = 13; // in bulit arduino LED
void setup()
{
  Serial.begin(9600);
  pinMode(IRSensor, INPUT); // IR Sensor pin INPUT
  pinMode(LED, OUTPUT); // LED Pin Output
}
void loop()
{
  int sensorStatus = digitalRead(IRSensor);
  // Set the GPIO as Input
  if (sensorStatus == 1) // Check if the pin high or not
  {
    digitalWrite(LED, LOW); // LED LOW
    Serial.println("Motion Ended!");
    // print Motion Detected! on the serial monitor window
  }
  else
  {
    digitalWrite(LED, HIGH); // LED High
    Serial.println("Motion Detected!");
    // print Motion Ended! on the serial monitor window
  }
  delay(1000);
```

```
}
```

1.1.4 Interface Potentiometer & Print On Serial Monitor

Figure 9: Potentiometer

CODE:

```
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

void loop() {
  // read the input on analog pin A0:
  int analogValue = analogRead(A0);
  // Rescale to potentiometer's voltage (from 0V to 5V):
  float voltage = 5*analogValue/1023;

  // print out the value you read:
  Serial.print("Analog: ");
  Serial.print(analogValue);
  Serial.print(", Voltage: ");
  Serial.println(voltage);
  delay(1000);
}
```

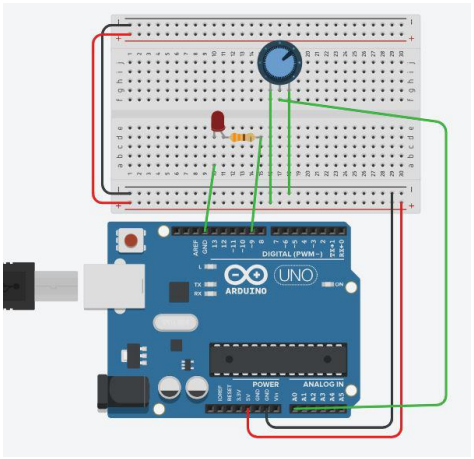

Figure 10: LED Intensity Control With Potentiometer

1.1.5 Control LED intensity With Potentiometer and Arduino

CODE:

```
int ledPin=9;
int potPin=A0;
int potValue;
int pinValue;
int DT=1000;
void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(potPin, INPUT);
  Serial.begin(9600);
}

void loop()
{
  potValue=analogRead(potPin);
  Serial.println(potValue);
  delay(DT);
  pinValue=(255.0/1023.0)*potValue;
  analogWrite(ledPin,pinValue);
}
```

1.1.6 Interface UltrasonicSensor With Arduino

CODE:

```
const int trigPin = 9;
const int echoPin = 10;
// defines variables
long duration;
int distance;
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
}
void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;
  // Prints the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.println(distance);
}
```

1.1.7 Interface DH-11 Sensor with Arduino

CODE:

```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#define DHTTYPE    DHT11     // DHT 11
#define DHTPIN 2
DHT_Unified dht(DHTPIN, DHTTYPE);
uint32_t delayMS;
void setup() {
  Serial.begin(9600);
```

```
  dht.begin();
  sensor_t sensor;
  delayMS = sensor.min_delay / 1000;
}
void loop()
{
  sensors_event_t event;
  dht.temperature().getEvent(&event);
  Serial.print(F("Temperature: "));
  Serial.print(event.temperature);
  Serial.println(F("°C"));
  dht.humidity().getEvent(&event);
  Serial.print(F("Humidity: "));
  Serial.print(event.relative_humidity);
  Serial.println(F("%"));
  delay(delayMS);
}
```

1.1.8 Interface DC motor with Arduino UNO

CODE:

```
const int ENA_PIN = 9; // the Arduino pin connected to the EN1 pin L298N
const int IN1_PIN = 6; // the Arduino pin connected to the IN1 pin L298N
const int IN2_PIN = 5; // the Arduino pin connected to the IN2 pin L298N

void setup() {
  // initialize digital pins as outputs.
  pinMode(ENA_PIN, OUTPUT);
  pinMode(IN1_PIN, OUTPUT);
  pinMode(IN2_PIN, OUTPUT);
}

void loop() {
  digitalWrite(IN1_PIN, HIGH); // control motor A spins clockwise
  digitalWrite(IN2_PIN, LOW);  // control motor A spins clockwise
  analogWrite(ENA_PIN, speed); // control the speed
  delay(1000); // rotate at maximum speed 1 seconds in in clockwise direction

  // change direction
```

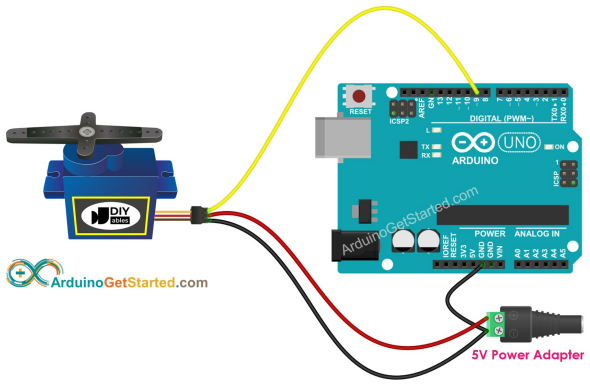

Figure 11: Control DC Motor With Arduino UNO

```
  digitalWrite(IN1_PIN, LOW);   // control motor A spins anti-clockwise
  digitalWrite(IN2_PIN, HIGH);  // control motor A spins anti-clockwise
delay(1000); // rotate at maximum speed 1 seconds in in anti-clockwise direction

 digitalWrite(IN1_PIN, LOW);   // control motor A stop
  digitalWrite(IN2_PIN, LOW);  // control motor A stop
  delay(1000); // stop motor 1 second
}
```

1.1.9 Interface Brushless DC motor with Arduino UNO

CODE:

```
#include <Servo.h>

Servo ESC;     // create servo object to control the ESC

int potValue;  // value from the analog pin

void setup() {
  // Attach the ESC on pin 9
  ESC.attach(11,1000,2000);
}

void loop() {
  potValue = analogRead(A0);
  // reads the value of the potentiometer (value between 0 and 1023)
  potValue = map(potValue, 0, 1023, 0, 180);
```

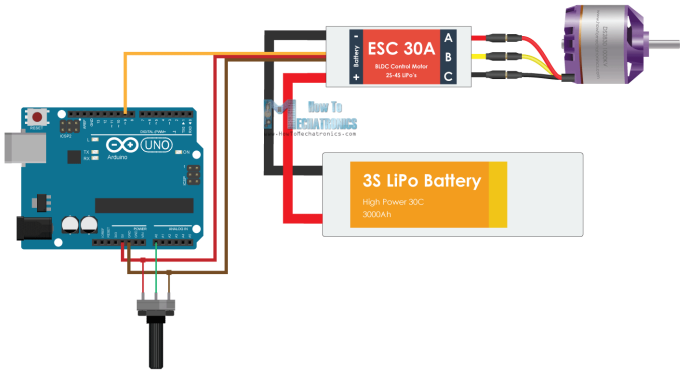

Figure 12: controlling speed of BLDC with Potentiometer

```
  // scale it to use it with the servo library (value between 0 and 180)
  ESC.write(potValue);
  // Send the signal to the ESC
}
```

1.1.10 Interface Servo DCMotor with Arduino UNO

CODE:

```
#include <Servo.h>
Servo servo;  // create servo object to control a servo

void setup() {
  servo.attach(9);  // attaches the servo on pin 9 to the servo object
  servo.write(0);   // rotate slowly servo to 0 degrees immediately
}

void loop() {
  for (int pos = 0; pos <= 180; pos += 1)
  {  // rotate slowly from 0 degrees to 180 degrees, one by one degree
    // in steps of 1 degree
    servo.write(pos);
    // control servo to go to position in variable 'pos'
    delay(10);
    // waits 10ms for the servo to reach the position
  }

  for (int pos = 180; pos >= 0; pos -= 1)
  {
```

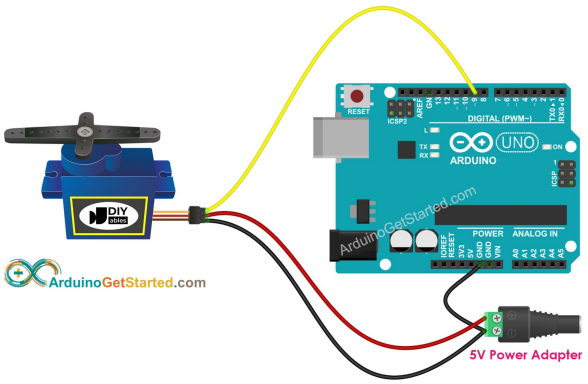

Figure 13: Servo Motor With Arduino UNO

```
  // rotate from 180 degrees to 0 degrees, one by one degree
    servo.write(pos);
    // control servo to go to position in variable 'pos'
    delay(10);
    // waits 10ms for the servo to reach the position
  }
}
```

2. Node MCU

2.1 Controlling various devices via Node MCU

2.1.1 Interface LED blinking With Node MCU

2.1.2 interface IR sensor With NODE MCU

2.1.3 Control LED Blink with IRsensor and Node MCU

2.1.4 Interface Potentiometer with Node MCU

2.1.5 Intensity control of LED with Node MCU and Potentiometer

2.1.6 Interface Ultrasonic Sensor With Node MCU

2.1.7 Interface DH-11 Sensor with Node MCU

2.1.8 Interface DC motor with Node MCU

2.1.9 Interface Brushless DC motor with Node MCU

2.1.10 Interface Servo DCMotor with Node MCU

2.2 Connectivity Of NodeMCU with Mobile Board

3. ThingsBoard

3.1 Write and Rewrite Data on thingsboard

3.2 Read data from Thingsboard

3.3 DH-11 sensor Application

3.3.1 Display Humidity And Temperature On thingsBoard with the help of DH-11 sensor and NODE MCU

3.4 Using widgets in ThingsBoard

3.5 Setting Up ALARMS on Thingsboard

A. Second Appendix

A.1 Detailed Appendix 1

A.2 Detailed Appendix 2

Declaration of Authorship

I hereby declare that I have written the present thesis with the title XXX independently and without the use of other than the indicated aids. I affirm that I have not used any sources other than those indicated and that all passages taken verbatim or in spirit from published and unpublished writings are identified as such. Furthermore, I assure that the work has not yet been submitted in the same or similar form in the context of another examination. I am aware that in the event of deception, the thesis will be graded as ”failed”.

date, place	signature