DELIVERABLE

Experiments Manual of Internet of Things Laboratory

Written by	**Responsibility**
Eng Samiha Alfalahat - AHU	
Dr Saud Althunibat - AHU	
Dr Abdullah Alhasanat -AHU	
Edited by	
Dr Moath Alsafasfeh - AHU	
Dr Ziyad Altarawneh - MU	
Approved by	
Saud Althunibat (AHU)	Project Coordinator

This publication was produced with the financial support of the European Union. Its contents are the sole responsibility of the partners of IREEDER project and do not necessarily reflect the views of the European Union'

Contents

1. Introduction

The Erasmus+ Capacity Building Project for Introducing Recent Electrical Engineering Developments into undERgraduate curriculum (hereinafter called as "the IREEDER Project") is implemented from November 2019 to November 2022 (3-year project).

The main objective of the IREEDER project is to improve the capacities of high quality education in Jordan, using state of the art technology and training staff on improving the quality of the courses taught by making the best use of these technologies. Specifically, IREEDER aims at introducing the recent developments in Electrical Engineering to the undergraduate curricula, where three subjects in Renewable Energy (RE), Internet of Things (IoT) and Cyber Security (CS) will be developed. Also, three laboratories for training the students in the selected topics will be established in three different Jordanian partners (Universities).

The IREEDER Project is expected to produce three main outputs by the end of the project period, such as:

- Output 1: Teaching materials about the project topics (IoT, CS, RE) accompanied by experimental activities
- Output 2: Establishment of three labs (in three Jordanian universities) related to the project topics, accompanied by a server for a remote lab with virtual lab software at each university of the Jordanian partners
- Output 3: Training workshops in Europe and Jordan

In order to complete the outputs, IREEDER Project has endeavored to conduct various activities for its staff since November 2019. This deliverable contains the capacity building plan to be developed in the process of conducting the above-mentioned output activities. It is expected that they can be also used by those who will conduct the training courses in the long-term, after the end of the project.

2. IoT lab Experiments

IoT Experience Lab Sheets

Exp No.1
Arduino-Based RGB LED

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.1. Experiment No. 1 : RGB LED

Introduction

An RGB LED is actually three small LEDs — one red, one green and one blue — inside a normal LED housing. This RGB LED has all t he internal LEDs share the same ground wire, so there are four legs in total. To turn on one color, ensure ground is connected, and then power one of the legs just as you would a regular LED. Don't forget the current-limiting resistors. If you turn on more than one color at a time, you will see the colors start to blend together to form a new color.

ANALOG OUTPUT (PULSE-WIDTH MODULATION): The digitalWrite() command can turn pins on (5V) or off (0V), but what if you want to output 2.5V? The analogWrite() command can output 2.5 volts by quickly switching a pin on and off so that it is only on 50 percent of the time (50% of 5V is 2.5V). By doing this, any voltage between 0 and 5V can be produced. This is what is known as Pulse-Width Modulation (PWM). It can create many different colors on the RGB LED.

Objectives

- Learn how to use Arduino IDE by write, verify, and upload the code
- Learn about the working concept of RGB Led
- Connect the project component to blink RGB LED

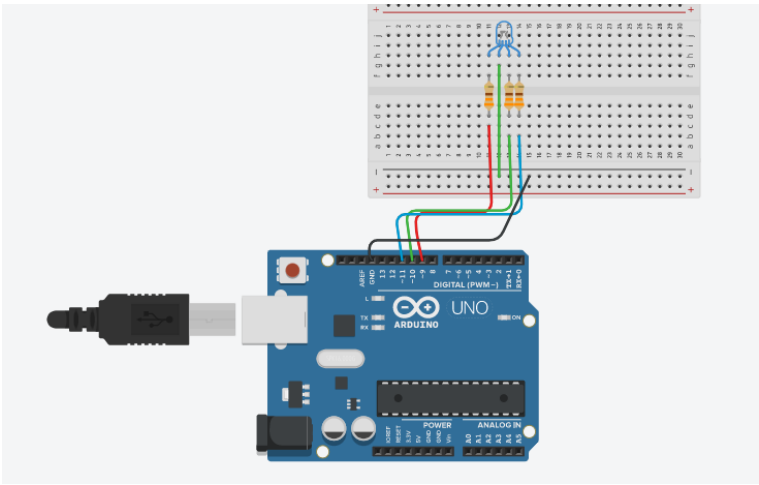
Experiment Components

- Arduino Microcontroller
- RGB LED
- Breadboard
- Resistors (330Ohm)
- Wires

Experiment Procedure

- The RGD Led has three terminal for Red,Blue, and green colours,we need to connect these terminals to digital pins in Arduino, the COM terminal connection to gound. The circuit Diagrams below show two ways to control the RGB led. The RGB Mixed colors and RGD Night light.

1. RGB LED-Mixed Colors with delay



Circuit Diagram

Code Program

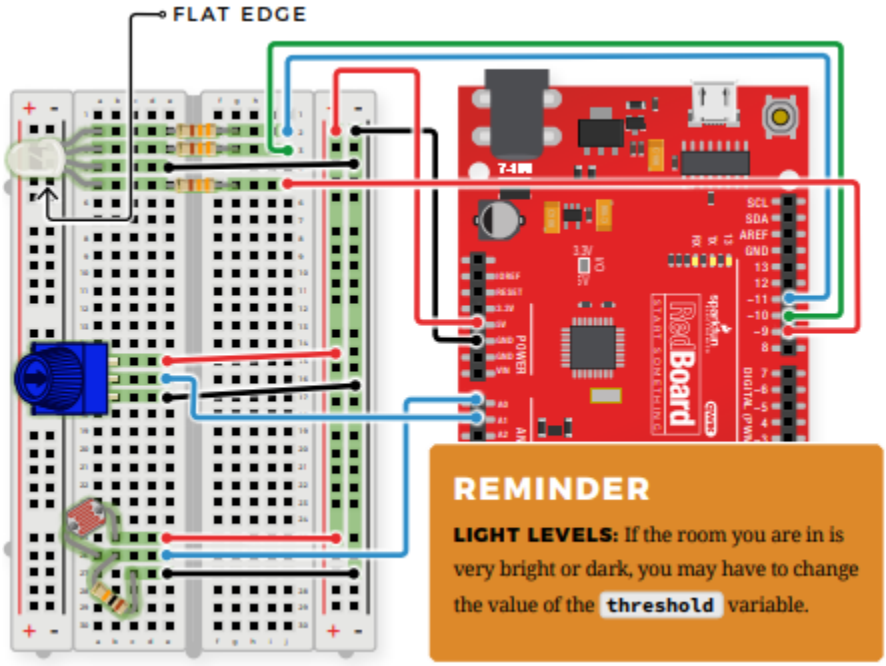
```
/*
  SparkFun Inventor's Kit
  Circuit 1D-RGB */
//LEDs are connected to these pins
int RedPin = 9;
int GreenPin = 10;
int BluePin = 11;
void setup() {
  Serial.begin(9600);         //start a serial connection with the computer
```

```
 //set the LED pins to output
 pinMode(RedPin, OUTPUT);
 pinMode(GreenPin, OUTPUT);
 pinMode(BluePin, OUTPUT);
}

void loop() {

   red();
  delay(300);
  blue();
 delay(300);   //short delay so that the printout is easier to read
 green();
 delay(300);
 orange();
 delay(300);
 yellow ();
 delay(300);
  cyan ();
 delay(300);
 void magenta();
 delay(300);
 turnOff ();
 delay(1000);

}
void red () {
 //set the LED pins to values that make red
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 0);
}
void orange () {
 //set the LED pins to values that make orange
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 50);
 analogWrite(BluePin, 0);
}
void yellow () {
 //set the LED pins to values that make yellow
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 100);
 analogWrite(BluePin, 0);
}
```

```
Void green () {

 //set the LED pins to values that make green
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 100);
 analogWrite(BluePin, 0);
}
void cyan () {

 //set the LED pins to values that make cyan
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 100);
 analogWrite(BluePin, 100);
}
void blue () {

 //set the LED pins to values that make blue
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 100);
}
void magenta () {
 //set the LED pins to values that make magenta
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 100);
}
void turnOff () {

 //set all three LED pins to 0 or OFF
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 0);
}
```

2. RGB Night-Light

Circuit Diagram



Code Program

```
/*Turns an RGB LED on or off based on the light level read by a photoresistor.
  Change colors by turning the potentiometer.*/
int photoresistor = A0;        //variable for storing the photoresistor value
int potentiometer = A1;        //this variable will hold a value based on the position of the knob
int threshold = 700;          //if the photoresistor reading is lower than this value the light will turn on

//LEDs are connected to these pins
int RedPin = 9;
int GreenPin = 10;
int BluePin = 11;

void setup() {
  Serial.begin(9600);         //start a serial connection with the computer

  //set the LED pins to output
  pinMode(RedPin, OUTPUT);
  pinMode(GreenPin, OUTPUT);
  pinMode(BluePin, OUTPUT);
}
```

```
void loop() {

 photoresistor = analogRead(A0);       //read the value of the photoresistor
 potentiometer = analogRead(A1);

 Serial.print("Photoresistor value:");
 Serial.print(photoresistor);        //print the photoresistor value to the serial monitor
 Serial.print("  Potentiometer value:");
 Serial.println(potentiometer);        //print the potentiometer value to the serial monitor

 if (photoresistor < threshold) {      //if it's dark (the photoresistor value is below the threshold) turn the LED
on
   //These nested if statements check for a variety of ranges and
   //call different functions based on the current potentiometer value.
   //Those functions are found at the bottom of the sketch.
   if (potentiometer > 0 && potentiometer <= 150)
     red();
   if (potentiometer > 150 && potentiometer <= 300)
     orange();
   if (potentiometer > 300 && potentiometer <= 450)
     yellow();
   if (potentiometer > 450 && potentiometer <= 600)
     green();
   if (potentiometer > 600 && potentiometer <= 750)
     cyan();
   if (potentiometer > 750 && potentiometer <= 900)
     blue();
   if (potentiometer > 900)
     magenta();
 }
 else {                        //if it isn't dark turn the LED off

   turnOff();                    //call the turn off function

 }

 delay(100);                     //short delay so that the printout is easier to read
}

void red () {

 //set the LED pins to values that make red
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 0);
```

```
}
void orange () {

 //set the LED pins to values that make orange
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 50);
 analogWrite(BluePin, 0);
}
void yellow () {

 //set the LED pins to values that make yellow
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 100);
 analogWrite(BluePin, 0);
}
void green () {

 //set the LED pins to values that make green
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 100);
 analogWrite(BluePin, 0);
}
void cyan () {

 //set the LED pins to values that make cyan
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 100);
 analogWrite(BluePin, 100);
}
void blue () {

 //set the LED pins to values that make blue
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 100);
}
void magenta () {

 //set the LED pins to values that make magenta
 analogWrite(RedPin, 100);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 100);
}
void turnOff () {
```

```
 //set all three LED pins to 0 or OFF
 analogWrite(RedPin, 0);
 analogWrite(GreenPin, 0);
 analogWrite(BluePin, 0);
}
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- YOUR Report must include:
    - Introduction: A brief about the experiment
    - Material and Methods/ Procedure
    - Analysis
    - Results and Discussion
    - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

IoT Experience Lab Sheets

Exp No.2
Buzzer and digital trumpet

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunbat

Experiment No. 2: Buzzer and digital trumpet

Introduction

BUZZER: The buzzer uses a small magnetic coil to vibrate a metal disc inside a plastic housing. By pulsing electricity through the coil at different rates, different frequencies (pitches) of sound can be produced. Attaching a potentiometer to the output allows you to limit the amount of current moving through the buzzer and lower its volume.

TONE FUNCTION: To control the buzzer, you will use the tone() function. This function is similar to PWM in that it generates a wave that is of a certain frequency on the specified pin. The frequency and duration can both be passed to the tone() function when calling it. To turn the tone off, you need to call noTone() or pass a duration of time for it to play and then stop. Unlike PWM, tone() can be used on any digital pin.
Each note has frequency please refer to below table

MUSICAL NOTE	FREQUENCY (HZ)	USING VARIABLES	USING AN ARRAY
A	220	A_FREQUENCY	FREQUENCY[0]
B	247	B_FREQUENCY	FREQUENCY[1]
C	261	C_FREQUENCY	FREQUENCY[2]
D	294	D_FREQUENCY	FREQUENCY[3]
E	330	E_FREQUENCY	FREQUENCY[4]
F	349	F_FREQUENCY	FREQUENCY[5]
G	392	G_FREQUENCY	FREQUENCY[6]

Objectives:

1. In this circuit, you'll use the Red Board and a small buzzer to make music, and you'll learn how to program your own songs using arrays.

Experiment Components

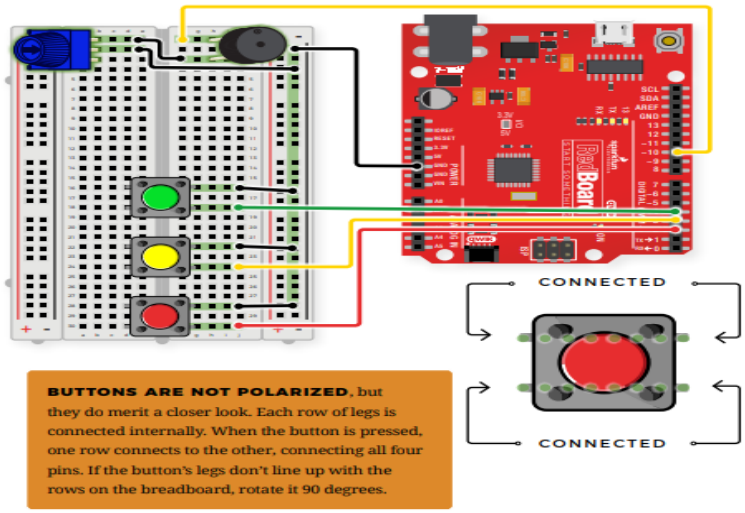
1. Arduino Microntroller
2. Buzzer
3. Potentiometer

4. resistor
5. Wires

Experiment Procedure

The Buzzer has positive terminal that must be connect to arduino through resistor, and negative terminal which connected to the gournd. The buzzer will act by produce a tone based on the potentiometer reading. the Diagram below show the connection.

Circuit Diagram



Code Program

```
//set the pins for the button and buzzer
int firstKeyPin = 2;
int secondKeyPin = 3;
int thirdKeyPin = 4;

int buzzerPin = 10;


void setup() {
```

```
 //set the button pins as inputs
 pinMode(firstKeyPin, INPUT_PULLUP);
 pinMode(secondKeyPin, INPUT_PULLUP);
 pinMode(thirdKeyPin, INPUT_PULLUP);

 //set the buzzer pin as an output
 pinMode(buzzerPin, OUTPUT);
}

void loop() {

 if (digitalRead(firstKeyPin) == LOW) {     //if the first key is pressed
  tone(buzzerPin, 262);                 //play the frequency for c
 }
 else if (digitalRead(secondKeyPin) == LOW) { //if the second key is pressed
  tone(buzzerPin, 330);                 //play the frequency for e
 }
 else if (digitalRead(thirdKeyPin) == LOW) { //if the third key is pressed
  tone(buzzerPin, 392);                 //play the frequency for g
 }
 else {
  noTone(buzzerPin);                    //if no key is pressed turn the buzzer off
 }
}
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
  - Introduction: A brief about the experiment
  - Material and Methods/ Procedure
  - Analysis
  - Results and Discussion
  - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

IoT Experience Lab Sheets

Exp No.3: Motion Alarm Using Arduino

Prepared By: Eng. Samiha Alfalaht
Reviewed by Dr. Saud Althunibat

Exp No.3: Motion Alarm Using Arduino

Introduction:

In this experiment we are going to control the motion of physical object connected to a servo motor. Based on readings come from an Ultrasonic sensor (Ultrasonic sensor: detect an object within specific area by using sound wave echo mechanism) an action taken by moving the servo motor with proper angle. And make a LED light turn ON or OFF accordingly. A sound alert raised from buzzer the there are any object motion detected.

Objectives

1. To demonstrate the procedure of reading value and take physical action according to these reading

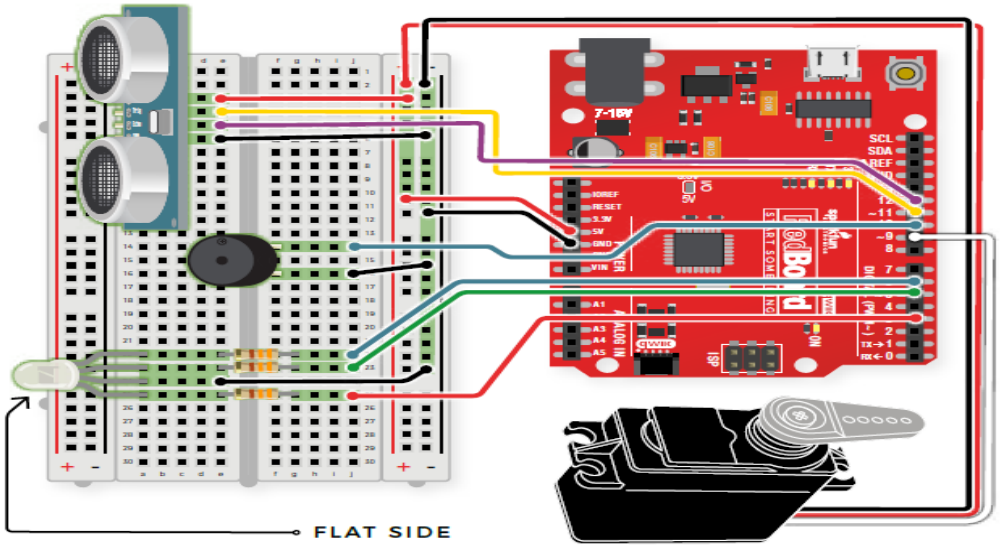
Experiment Components:

1. Arduino Microcontroller
2. Ultrasonic sensor
3. Servo motor
4. Breadboard
5. Buzzer
6. Led
7. Resistors
8. Wires

Experiment Procedure

The Diagram below show the connection of this project

Circuit Diagram



Code Program

```
#include <Servo.h>              //include the servo library

const int trigPin = 11;         //connects to the trigger pin on the distance sensor
const int echoPin = 12;         //connects to the echo pin on the distance sensor

const int redPin = 3;           //pin to control the red LED inside the RGB LED
const int greenPin = 5;         //pin to control the green LED inside the RGB LED
const int bluePin = 6;          //pin to control the blue LED inside the RGB LED

const int buzzerPin = 10;       //pin that will drive the buzzer

float distance = 0;             //stores the distance measured by the distance sensor

Servo myservo;                  //create a servo object

void setup()
{
  Serial.begin (9600);       //set up a serial connection with the computer

  pinMode(trigPin, OUTPUT);   //the trigger pin will output pulses of electricity
  pinMode(echoPin, INPUT);    //the echo pin will measure the duration of pulses coming back from the distance sensor

  //set the RGB LED pins to output
```

```
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);

  pinMode(buzzerPin, OUTPUT);   //set the buzzer pin to output

  myservo.attach(9);          //use pin 9 to control the servo

}

void loop() {
  distance = getDistance();   //variable to store the distance measured by the sensor

  Serial.print(distance);     //print the distance that was measured
  Serial.println(" in");      //print units after the distance

  if (distance <= 10) {                  //if the object is close

    //make the RGB LED red
    analogWrite(redPin, 255);
    analogWrite(greenPin, 0);
    analogWrite(bluePin, 0);

    //this code wiggles the servo and beeps the buzzer
    tone(buzzerPin, 272);        //buzz the buzzer pin
    myservo.write(10);          //move the servo to 45 degrees
    delay(100);                 //wait 100 milliseconds
```

```
  noTone(buzzerPin);          //turn the buzzer off
  myservo.write(150);         //move the servo to 135 degrees
  delay(100);                 //wait 100 milliseconds


 } else if (10 < distance && distance < 20) { //if the object is a medium distance

  //make the RGB LED yellow
  analogWrite(redPin, 255);
  analogWrite(greenPin, 50);
  analogWrite(bluePin, 0);

 } else {                     //if the object is far away

  //make the RGB LED green
  analogWrite(redPin, 0);
  analogWrite(greenPin, 255);
  analogWrite(bluePin, 0);
 }

 delay(50);     //delay 50ms between each reading
}

//-----------------FUNCTIONS-----------------------------

//RETURNS THE DISTANCE MEASURED BY THE HC-SR04 DISTANCE SENSOR
```

```
float getDistance()
{
  float echoTime;                 //variable to store the time it takes for a ping to bounce off an object
  float calculatedDistance;       //variable to store the distance calculated from the echo time

  //send out an ultrasonic pulse that's 10ms long
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  echoTime = pulseIn(echoPin, HIGH);      //use the pulsein command to see how long it takes for the
                                          //pulse to bounce back to the sensor

  calculatedDistance = echoTime / 148.0;  //calculate the distance of the object that reflected the pulse (half the bounce time multiplied by the speed of sound)

  return calculatedDistance;              //send back the distance that was calculated
}
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
    - Introduction: A brief about the experiment

- Material and Methods/ Procedure
- Analysis
- Results and Discussion
- Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

IoT Experience Lab Sheets

Exp No.4: Arduino based Autonomous Robot

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.2. Experiment No. 4

Introduction

This **LAB** describes the Autonomous vehicle smart movement. This Robot navigates the world on its own. When the robot senses an object using the distance sensor, it will back up and change course. This kind of system is used in Mars rovers, autonomous cars and the bots built for all kinds of robotics competitions.

Objectives

2. Build a complete autonomous Robot which be able to detect an obstacle and make a movement based on sensed reading.

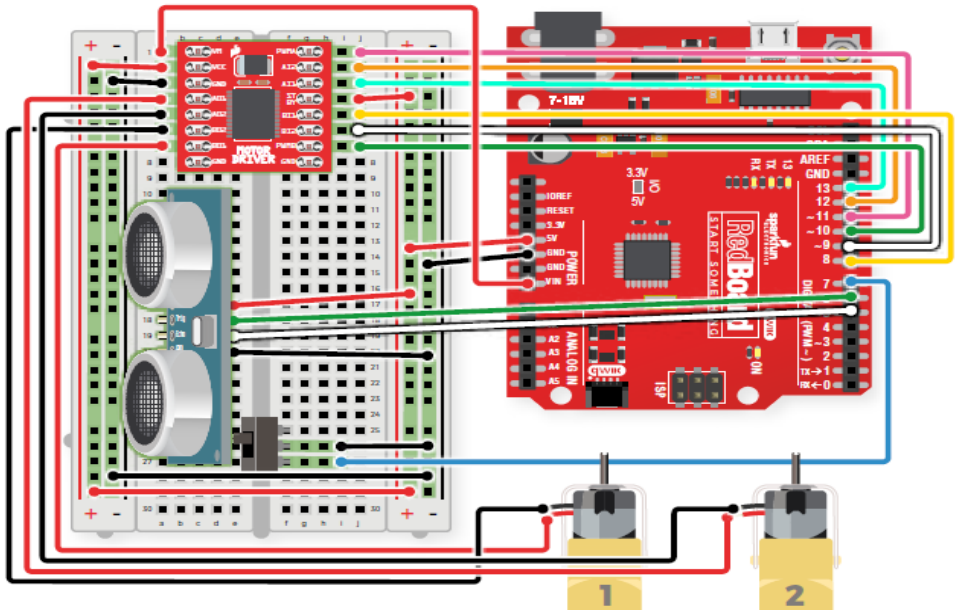
Experiment Components:

1. Arduino Microcontroller
2. Breadboard
3. Ultrasonic sensor
4. Switch or push button
5. Two DC Motors
6. Wheels
7. DC Motor Driver

Experiment Procedure

In this lab the switch will turn On or OFF the robot, If the switch is turned on, If no obstacle is detected, then drive forward. If an obstacle is detected, stop, back up, and turn right. If no obstacle is detected, start driving forward again. The below Diagram show the connections.

Circuit Diagram



Code Program:

```
//the right motor will be controlled by the motor A pins on the motor driver
const int AIN1 = 13;          //control pin 1 on the motor driver for the right motor
const int AIN2 = 12;          //control pin 2 on the motor driver for the right motor
const int PWMA = 11;          //speed control pin on the motor driver for the right motor

//the left motor will be controlled by the motor B pins on the motor driver
const int PWMB = 10;          //speed control pin on the motor driver for the left motor
const int BIN2 = 9;           //control pin 2 on the motor driver for the left motor
const int BIN1 = 8;           //control pin 1 on the motor driver for the left motor


//distance variables
const int trigPin = 6;
const int echoPin = 5;

int switchPin = 7;            //switch to turn the robot on and off

float distance = 0;           //variable to store the distance measured by the distance sensor

//robot behaviour variables
int backupTime = 300;         //amount of time that the robot will back up when it senses an object
int turnTime = 200;           //amount that the robot will turn once it has backed up

/*****************************************************************************/
void setup()
```

```
{
  pinMode(trigPin, OUTPUT);          //this pin will send ultrasonic pulses out from the distance sensor
  pinMode(echoPin, INPUT);        //this pin will sense when the pulses reflect back to the distance sensor

  pinMode(switchPin, INPUT_PULLUP);    //set this as a pullup to sense whether the switch is flipped


  //set the motor control pins as outputs
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);
  pinMode(PWMA, OUTPUT);

  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
  pinMode(PWMB, OUTPUT);

  Serial.begin(9600);                  //begin serial communication with the computer
  Serial.print("To infinity and beyond!");  //test the serial connection
}

/*****************************************************************************/
void loop()
{
  //DETECT THE DISTANCE READ BY THE DISTANCE SENSOR
```

```
distance = getDistance();

Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" in");          // print the units

if (digitalRead(switchPin) == LOW) { //if the on switch is flipped

 if (distance < 10) {          //if an object is detected
  //back up and turn
  Serial.print(" ");
  Serial.print("BACK!");

  //stop for a moment
  rightMotor(0);
  leftMotor(0);
  delay(200);

  //back up
  rightMotor(-255);
  leftMotor(-255);
  delay(backupTime);

  //turn away from obstacle
  rightMotor(255);
  leftMotor(-255);
  delay(turnTime);
```

```
  } else {                  //if no obstacle is detected drive forward
    Serial.print(" ");
    Serial.print("Moving...");


    rightMotor(255);
    leftMotor(255);
  }
 } else {                   //if the switch is off then stop

  //stop the motors
  rightMotor(0);
  leftMotor(0);
 }

 delay(50);                 //wait 50 milliseconds between readings
}

/*****************************************************************************/
void rightMotor(int motorSpeed)                 //function for driving the right motor
{
 if (motorSpeed > 0)                        //if the motor should drive forward (positive speed)
 {
  digitalWrite(AIN1, HIGH);                 //set pin 1 to high
  digitalWrite(AIN2, LOW);                  //set pin 2 to low
 }
```

```
 else if (motorSpeed < 0)                    //if the motor should drive backward (negative speed)
 {
  digitalWrite(AIN1, LOW);                   //set pin 1 to low
  digitalWrite(AIN2, HIGH);                  //set pin 2 to high
 }
 else                                        //if the motor should stop
 {
  digitalWrite(AIN1, LOW);                   //set pin 1 to low
  digitalWrite(AIN2, LOW);                   //set pin 2 to low
 }
 analogWrite(PWMA, abs(motorSpeed));         //now that the motor direction is set, drive it at
the entered speed
}

/*****************************************************************************/
void leftMotor(int motorSpeed)               //function for driving the left motor
{
 if (motorSpeed > 0)                         //if the motor should drive forward (positive speed)
 {
  digitalWrite(BIN1, HIGH);                  //set pin 1 to high
  digitalWrite(BIN2, LOW);                   //set pin 2 to low
 }
 else if (motorSpeed < 0)                    //if the motor should drive backward (negative speed)
 {
  digitalWrite(BIN1, LOW);                   //set pin 1 to low
  digitalWrite(BIN2, HIGH);                  //set pin 2 to high
 }
```

```
 else                                //if the motor should stop
 {
  digitalWrite(BIN1, LOW);                  //set pin 1 to low
  digitalWrite(BIN2, LOW);                  //set pin 2 to low
 }
 analogWrite(PWMB, abs(motorSpeed));              //now that the motor direction is set, drive it at
the entered speed
}

/*****************************************************************************/
//RETURNS THE DISTANCE MEASURED BY THE HC-SR04 DISTANCE SENSOR
float getDistance()
{
 float echoTime;              //variable to store the time it takes for a ping to bounce off an object
 float calculatedDistance;       //variable to store the distance calculated from the echo time

 //send out an ultrasonic pulse that's 10ms long
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);

 echoTime = pulseIn(echoPin, HIGH);      //use the pulsein command to see how long it takes for
the
                          //pulse to bounce back to the sensor

 calculatedDistance = echoTime / 148.0;  //calculate the distance of the object that reflected the
pulse (half the bounce time multiplied by the speed of sound)
```

```
  return calculatedDistance;          //send back the distance that was calculated
}
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
    - Introduction: A brief about the experiment
    - Material and Methods/ Procedure
    - Analysis
    - Results and Discussion
    - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

IoT Experience Lab Sheets

Exp No.5: Introduction RASLinux OS

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.3. Experiment No. 5

Introduction

Raspberry Pi OS is a free operating system based on Debian, optimised for the Raspberry Pi hardware, and is the recommended operating system for normal use on a Raspberry Pi. The OS comes with over 35,000 packages: precompiled software bundled in a nice format for easy installation on your Raspberry Pi.

Raspberry Pi OS is under active development, with an emphasis on improving the stability and performance of as many Debian packages as possible on Raspberry Pi.

Objectives

1. How to work with Raspain operating system of raspberry Pi
2. Use the terminal to execute commands

Experiment Components:

1. ETS IoT Kit
2. SD card to install the OS

Experiment Procedure

It's important to keep your Raspberry Pi up to date. The first and probably the most important reason is security. A device running Raspberry Pi OS contains millions of lines of code that you rely on. Over time, these millions of lines of code will expose well-known vulnerabilities, which are documented in publicly available databases meaning that they are easy to exploit. The only way to mitigate these exploits as a user of Raspberry Pi OS is to keep your software up to date, as the upstream repositories track CVEs closely and try to mitigate them quickly.

Keeping your Operating System up to Date

APT keeps a list of software sources on your Raspberry Pi in a file at `/etc/apt/sources.list`. Before installing software, you should update your package list with `apt update`. Go ahead and open a Terminal window and type:

>>sudo apt update

Next, upgrade all your installed packages to their latest versions with the following command:

>>sudo apt full-upgrade

General Linux Commands using Terminal (BASH)

File Commands		
1.	ls	Directory listing
2.	ls -al	Formatted listing with hidden files
3.	ls -lt	Sorting the Formatted listing by time modification
4.	cd dir	Change directory to dir
5.	cd	Change to home directory
6.	pwd	Show current working directory
7.	mkdir dir	Creating a directory dir
8.	cat >file	Places the standard input into the file
9.	more file	Output the contents of the file
10.	head file	Output the first 10 lines of the file
11.	tail file	Output the last 10 lines of the file
12.	tail -f file	Output the contents of file as it grows,starting with the last 10 lines
13.	touch file	Create or update file
14.	rm file	Deleting the file
15.	rm -r dir	Deleting the directory
16.	rm -f file	Force to remove the file
17.	rm -rf dir	Force to remove the directory dir
18.	cp file1 file2	Copy the contents of file1 to file2
19.	cp -r dir1 dir2	Copy dir1 to dir2;create dir2 if not present
20.	mv file1 file2	Rename or move file1 to file2,if file2 is an existing directory
21.	ln -s file link	Create symbolic link link to file

File permission		
1.	chmod octal file	Change the permission of file to octal,which can be found separately for user,group,world by adding, • 4-read(r) • 2-write(w) • 1-execute(x)

Installing Python using command line

you can start the installation of Python 3.X with the command:

```
>>sudo apt install python3.x
>>python –version
```

Open and write python code file

You can use any file editor available in your OS we are going to use (NANO editor)

```
>> nano example.py
```

After wrint the python code press(CTRL+X), press Y to save your changes, you can execute your code by writing the following command on shell

```
>>sudo python example.py
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
  - Introduction: A brief about the experiment
  - Material and Methods/ Procedure
  - Analysis
  - Results and Discussion
  - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

Exp No.6: LED blincking using ETS IoT Kit

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.4. Experiment No. 6

Introduction

Red, blue, and green LEDs are referred to as RGB LEDs. RGB LED products combine three colors to create over 16 million different light hues. Furthermore, pigment colors like brown or pink are difficult, if not impossible, to achieve. RGB LED controllers operate on a much simpler premise. To create a specific color mix, they change the power on each of the three channels (red, green, and blue). To generate a purple color, for example, the red and blue channels would be turned on, while the green channel would be completely turned off.

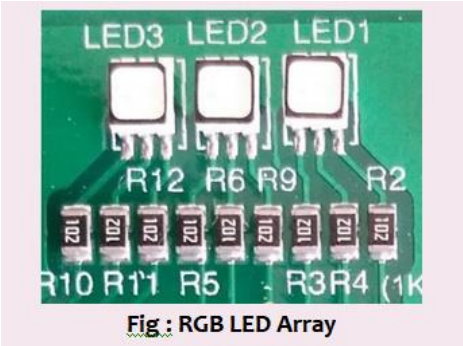


Fig : RGB LED Array

Objectives

1. This Experiment is used to blink the RGB LEDs in the ETS IoT Trainer kit

Experiment Components:

1. ETS IoT Trainer Kit
2. Power adaptor
3. Monitor
4. HDMI to VGI converter
5. Mouse and Keyboard

Experiment Procedure

Here, we're working with RGB LEDs, which come in a variety of colors and can be combined with red, green, and blue. The blinking program below uses LED 1 as RED, LED 2 as Green, and LED 3 as Blue.

Circuit Diagram

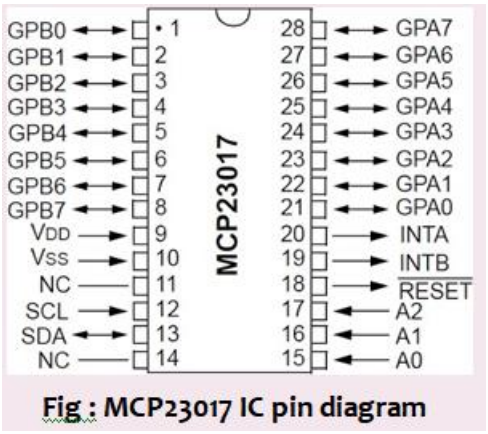
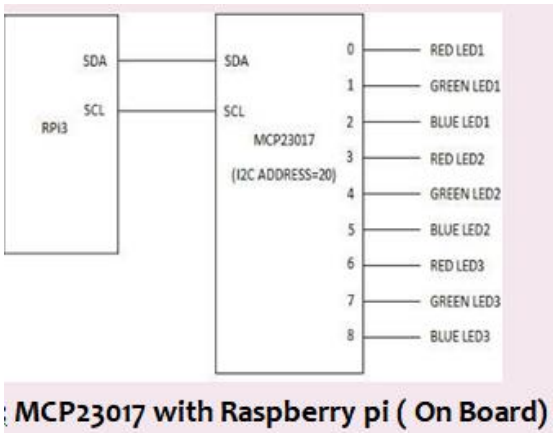


Fig : MCP23017 IC pin diagram



MCP23017 with Raspberry pi (On Board)

Code Program

```
#mcp23017 library
pathimport sys
sys.path.append('/home/pi/Adafruit-Raspberry-Pi-Python-Code- legacy/Adafruit_MCP230xx')
from Adafruit_MCP230XX import
Adafruit_MCP230XXimport time

#mcp IC configuration
mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16) # MCP23017

#mcp input/output configuration
mcp.config(0,mcp.OUPUT) mcp.config(1,
mcp.OUTPUT) mcp.config(2,
mcp.OUTPUT) mcp.config(3,
mcp.OUTPUT) mcp.config(4,
mcp.OUTPUT) mcp.config(5,
mcp.OUTPUT) mcp.config(6,
mcp.OUTPUT) mcp.config(7,
mcp.OUTPUT) mcp.config(8,
mcp.OUTPUT)

try:
 while True:
```

```
        #RGB LED blink
        mcp.output(0, 1)
        mcp.output(4, 1)
        mcp.output(8, 1)
        time.sleep(1)
        mcp.output(0, 0)
        mcp.output(4, 0)
        mcp.output(8, 0)time.sleep(1)

    except KeyboardInterrupt:
        #Ctrl+C and stop the
        programmcp.output(0,
        0)
        mcp.output(1, 0)
        mcp.output(2, 0)
        mcp.output(3, 0)
        mcp.output(4, 0)
        mcp.output(5, 0)
        mcp.output(6, 0)
        mcp.output(7, 0)
        mcp.output(8, 0)
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
    - Introduction: A brief about the experiment
    - Material and Methods/ Procedure
    - Analysis
    - Results and Discussion

- Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

Exp No.7 Push Button based LED control using ETS IoT Kit

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.5. Experiment No. 7

Introduction

This **LAB** describes the Push-Buttons are normally-open tactile switches. Push buttons allow us to power the circuit or make any particular connection only when we press the button. Simply, it makes the circuit connected when pressed and breaks when released. A push button is also used for triggering of the SCR by gate terminal

Objectives

1. In this Experiment the is study about control the RGB LEDs by pressing the Pushbutton

Experiment Components:

1. ETS IoT Trainer Kit
2. Power adaptor
3. Monitor
4. HDMI to VGI converter
5. Mouse and Keyboard

Experiment Procedure

We used the push button to turn on and off the LEDs in the ETS IoT Kit. In the following program, if we press Switch 1, all three RGB LEDs will glow in that LED 1 will be RED, LED 2 will be GREEN, and LED 3 will be BLUE. The MCP23017 IC is used to connect all of the LEDs and switches in an I2C fashion.

Circuit Diagram

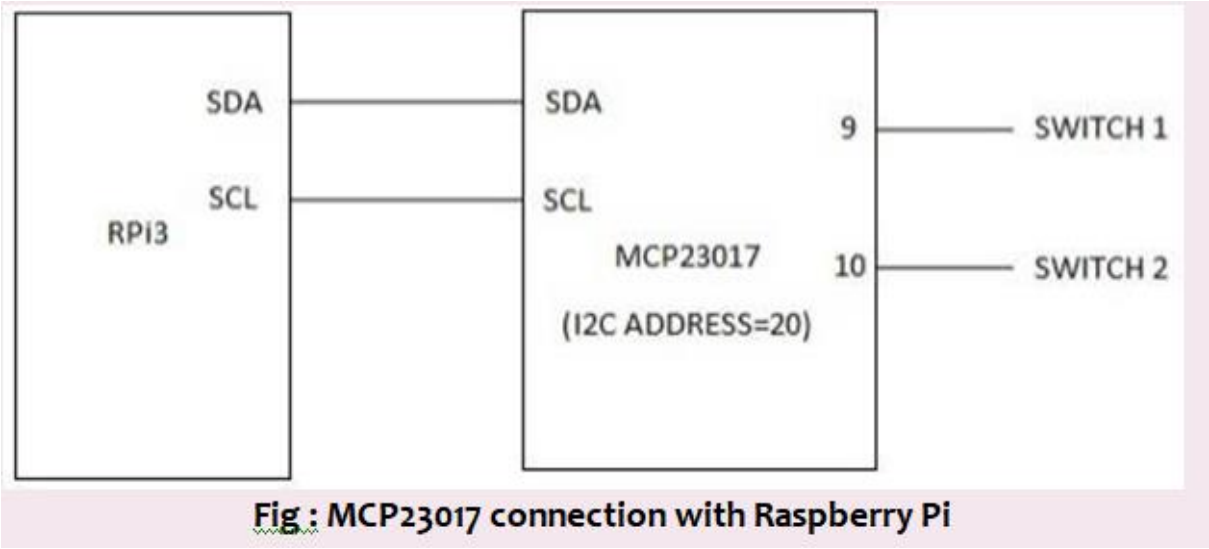


Fig : MCP23017 connection with Raspberry Pi

Code Program

```
import sys
sys.path.append('/home/pi/Adafruit-Raspberry-Pi-Python-Code- legacy/Adafruit_MCP230xx')

from Adafruit_MCP230XX import
Adafruit_MCP230XXimport time

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

 # MCP23017# Set pin 3 to input with the pullup resistor enabled
mcp.config(9, mcp.INPUT)
mcp.pullup(9, 1)
mcp.config(10, mcp.INPUT)
mcp.pullup(10, 1)
mcp.config(0, mcp.OUTPUT)
mcp.config(1, mcp.OUTPUT)
mcp.config(2, mcp.OUTPUT)
mcp.config(3, mcp.OUTPUT)
mcp.config(4, mcp.OUTPUT)
mcp.config(5, mcp.OUTPUT)
mcp.config(6, mcp.OUTPUT)
```

```
mcp.config(7, mcp.OUTPUT)
mcp.config(8, mcp.OUTPUT)

while (True):
  print "Pin 9 = %d" % (mcp.input(9))
  print "Pin 10 = %d" % (mcp.input(10))
  pin9=mcp.input(9) pin10=mcp.input(10)
  #time.sleep(2)
  if (pin9==0):
    mcp.output(0,1)
    elif (pin10==0):
    mcp.output(4,1)
    else:
    mcp.output(0,0)
    mcp.output(4,0)
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
  - Introduction: A brief about the experiment
  - Material and Methods/ Procedure
  - Analysis
  - Results and Discussion
  - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

Exp No.8: OLED Display using ETS IoT Kit

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.6. Experiment No. 8

Introduction

The acronym 'OLED' stands for Organic Light-Emitting Diode, which is a technology that uses LEDs to produce light from organic molecules.

These organic LEDs are used to make what are regarded as the best display panels in the world. Aside from the previously mentioned advantages of OLED displays, some disadvantages include: It has a shorter lifespan than some other display technologies. This shorter lifetime is primarily due to the blue organic material, but it improves over time due to moisture migration. Poor readability in direct sunlight

Objectives

1. study about the OLED Display and its functionality

Experiment Components:

1. ETS IoT Trainer Kit
2. Power adaptor
3. Monitor
4. HDMI to VGI converter
5. Mouse and Keyboard

Experiment Procedure

The experiment is designed to work with the ETS IoT Kit's OLED Display.
The Raspberry Pi is connected to the OLED display via I2C.The ETS IoT Kit's OLED displays the temperature, humidity, and pressure of the room using the BME 280 sensor. we will investigate the BME 280 in relation to the following experiment.

Circuit Diagram

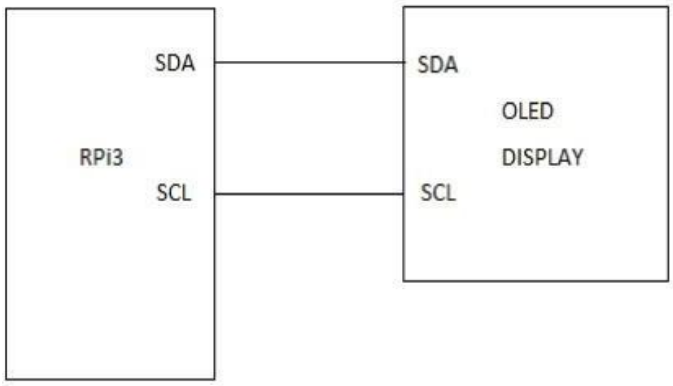


Fig : OLED connection with Raspberry Pi

Code Program

```
import sys

sys.path.append('/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/BME280') import
BME280 as BME
import time


import Adafruit_SSD1306
```

```
from PIL import
Image from PIL
import ImageDraw
from PIL import
ImageFont
# Raspberry Pi pin configuration:
RST = 24


# Note you can change the I2C address by passing an i2c_address parameter
like:
disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST, 2c_address=0x3C)


# Initialize library.
disp.begin()
ImageDraw
# Clear display.
disp.clear()
disp.display()


#Create blank image for drawing.
#Make sure to create image with mode '1' for 1-bit
color.
width = disp.width
height = disp.height
image = Image.new('1', (width, height))


# Get drawing object to draw on image.
draw = imageDraw.Draw(image)


# Draw a black filled box to clear the image.
draw.rectangle((0,0,width,height), outline=0, fill=0)


# Draw some shapes.
### First define some constants to allow easy resizing of shapes.
```

```
padding = 2
top = padding

# Move left to right keeping track of the current x position for drawing shapes.
x = padding


# Load default font.
font = ImageFont.load_default()


temperature,pressure,humidity = BME.readBME280All()#Write
two lines of text.
draw.text((x, top),          "Temperature: "+str(temperature),  font=font, fill=1)
draw.text((x, top+20), "Pressure: "+str(pressure), font=font, fill=1) draw.text((x,
top+40), "Humidity: "+str(humidity), font=font, fill=1)
#Display image.
disp.image(image)
disp.display()
time.sleep(5)
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
    - Introduction: A brief about the experiment
    - Material and Methods/ Procedure
    - Analysis
    - Results and Discussion
    - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

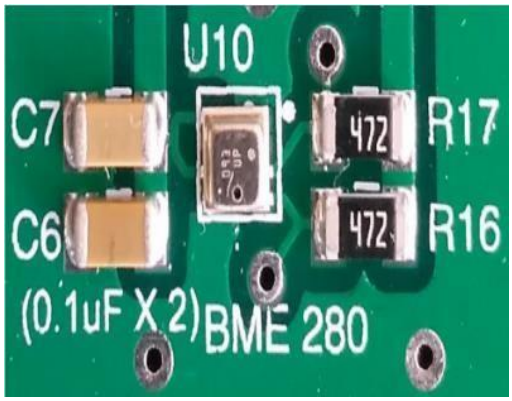
Exp No.9: Tempreture,Humidity,pressure monotiring using ETS IoT Kit

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.7. Experiment No. 9

Introduction

The BME280 is a humidity sensor designed specifically for mobile applications and wearables where small size and low power consumption are important design parameters. The unit combines high linearity and high accuracy sensors, making it ideal for low current consumption, long-term stability, and EMC robustness. This precision sensor can measure relative humidity from 0 to 100% with 3% accuracy, barometric pressure from 300Pa to 1100 hPa with 1% accuracy, and temperature from -40°C to 85°C with 1.0°C accuracy. The sensor module BME280 measures barometric pressure, temperature, and humidity. Because pressure varies with altitude, you can estimate altitude as well. To communicate with a microcontroller, the BME280 sensor employs the I2C or SPI communication protocols.



BME280 Sensor on board

Objectives

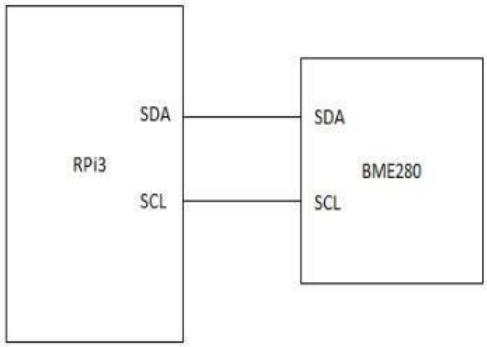
1.This experiment is used to monitor the temperature, humidity and pressure in OLEdisplay screen as well as ThingSpeak cloud platform

Experiment Components:

1. ETS IoT Trainer Kit
2. Power adaptor
3. Monitor
4. HDMI to VGI converter
5. Mouse and Keyboard

Experiment Procedure

Circuit Diagram



BME280 sensor connection with Raspberry Pi

Code Program

BME280 Data Uploading to Thingspeak

```
import sys
sys.path.append('/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/BME280')
import BME280 as BME
import time
 import urllib
while True:
(chip_id,       chip_version)       =
BME.readBME280ID() print "Chip
ID   :", chip_id
```

```
print "Version        :", chip_version temperature,pressure,humidity =
BME.readBME280All()print "Temperature : ", temperature, "C"
print "Pressure : ", pressure, "hPa"
print "Humidity : ", humidity, "%"

urllib.urlopen("https://api.thingspeak.com/update?api_key=JECTKF4RJR44LGVY&field1="
+str(temperature))time.sleep(2)
```

BME280 data on OLED

Refer to the experiment (8) code

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
  - Introduction: A brief about the experiment
  - Material and Methods/ Procedure
  - Analysis
  - Results and Discussion
  - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

Exp No.10 Relay control of ETS IoT using cloud

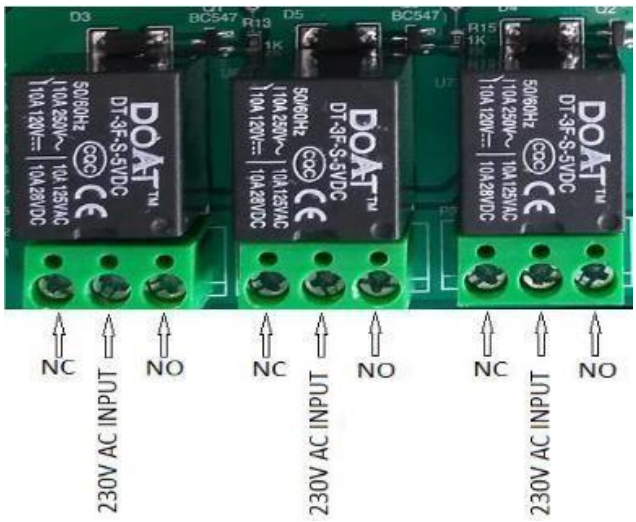
Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.8. Experiment No. 10

Introduction

A relay, like a switch, connects or disconnects two circuits. Instead of manual operation, an electrical signal is applied to a relay, which connects or disconnects another circuit. Different types of relays exist, such as electromechanical and solid state. Electromechanical relays are commonly employed. It is made up of an input terminal for a single or multiple control signals and an operating contact terminal. The switch can have an unlimited number of contacts in various contact forms, such as make contacts, break contacts, or combinations.

They are also used for automatic switching. Relays are used by microprocessors to control a large electrical load. Overload relays protect motors from overload and electrical failure.



Relay on board

Objectives

1. In this Experiment used to control the relay using ThingSpeak cloud Platform

Experiment Components:

1. ETS IoT Trainer Kit
2. Power adaptor
3. Monitor
4. HDMI to VGI converter
5. Mouse and Keyboard

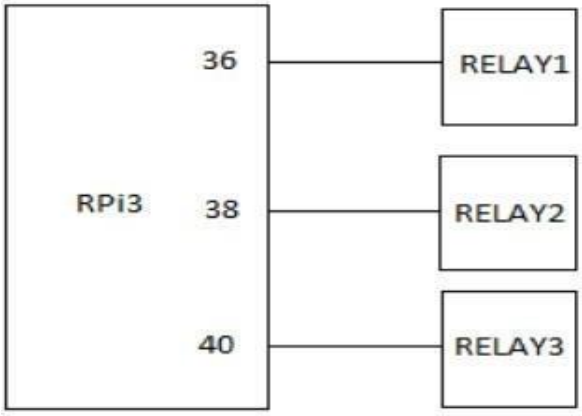
Experiment Procedure

The ThingSpeak Cloud was used to control the relays in the ETS IoT Kit.

Three 12V Relays are included in the ETS IoT Kit and are internally connected to the Raspberry Pi. The relays are controlled by the ThingSpeak Cloud, which sends commands from the cloud to turn on and off the relays.

If the data sent from the cloud is a '1,' the relay will be ON; if the data is a '0,' the relay will be OFF.

Circuit Diagram



Relay connection with Raspberry pi

Code Program

```
import RPi.GPIO as GPIOimport
time
import time
import urllib
CHANNEL_ID=”14456”
READ_API_KEY=”AC123F123456”
```

```
#RPi input/output configuration
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(36, GPIO.OUT)

while True:
        conn = urllib.urlopen("http://api.thingspeak.com/channels/%s/feeds/last.json?
        api_key=%s" \% (CHANNEL_ID,READ_API_KEY))
        response = conn.read()

        if( response == '1'):
                #Relay1 ON
                GPIO.output(36, 1)
                print "Relay ON"
                time.sleep(2)

        else if( response == '0'):
                #Relay1 OFF
                GPIO.output(36, 0)
                print "Relay OFF"
                time.sleep(2)
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
    - Introduction: A brief about the experiment
    - Material and Methods/ Procedure
    - Analysis

- Results and Discussion
- Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

Exp No.11: Monitoring three-axis accelometer sensor Using thingspeak cloud and ETS IoT

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.9. Experiment No. 11

Introduction

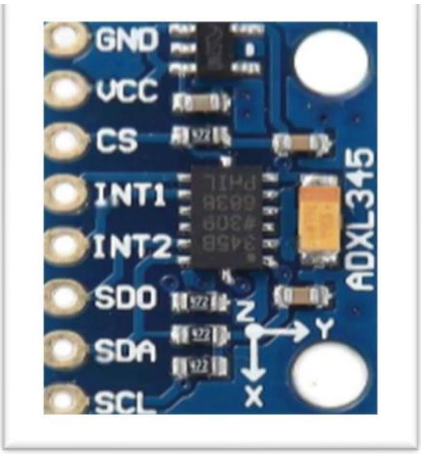
The ADXL345 digital 3-axis accelerometer module is used to detect motion and acceleration.

This module includes an ADXL345 accelerometer with digital data output formatted as 16-bit two's complement and accessible via I2C or SPI.

It measures both static gravity acceleration and dynamic acceleration caused by motion or shock.

It also has flexible interrupt modes that can be assigned to one of its two interrupt pins.

It has free-fall detection, tap detection, and other features. It can also detect motionlessness by comparing acceleration values to user-defined thresholds. The ADXL345 has four sensitivity ranges ranging from +/- 2G to +/- 16G. It also supports data rates ranging from 10Hz to 3200Hz.



Objectives

1. this Experiment used to determine the 3-axis accelerometer values, X,Y and Z axis to detect the motion and acceleration sensor

Experiment Components:

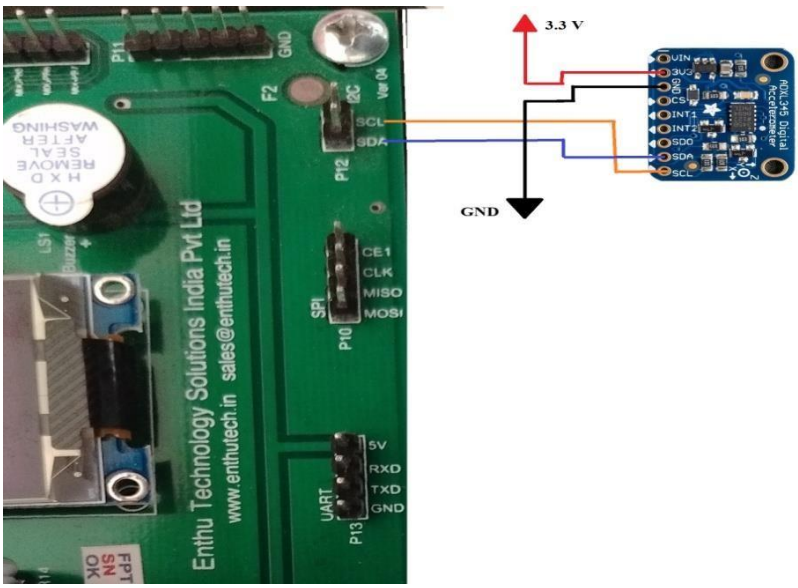
1. ETS IoT Trainer Kit

2. ADXL345 sensor
3. Power Adaptor

Experiment Procedure

We use the external Three Axis sensor in this experiment to determine and monitor the acceleration and gyroscopic values in the ThingSpeak cloud.
The external ADXL 345 sensor is linked to the ETS IoT kit to determine acceleration and gyroscopic values, which are then monitored in the ThingSpeak cloud. Connect the ADXL 345 sensor to the ETS IoT kit as shown in the connection diagram.

Circuit Diagram



Code Program

```
import smbus
import time
 import urllib

# Get I2C bus
bus = smbus.SMBus(1)
```

```
# ADXL345 address, 0x53(83)
# Select bandwidth rate register, 0x2C(44)
#0x0A(10)        Normal mode, Output data rate = 100 Hz
bus.write_byte_data(0x53, 0x2C, 0x0A)
# ADXL345 address, 0x53(83)
# Select power control register, 0x2D(45)
#0x08(08) Auto Sleep disable
bus.write_byte_data(0x53, 0x2D, 0x08)
# ADXL345 address, 0x53(83)
# Select data format register, 0x31(49)
#0x08(08)Self test disabled, 4-wire interface
#Full resolution, Range = +/-2g
bus.write_byte_data(0x53, 0x31,

0x08)time.sleep(0.5)

# ADXL345 address, 0x53(83)
# Read data back from 0x32(50), 2
bytes# X-Axis LSB, X-Axis MSB
while True:


    data0 = bus.read_byte_data(0x53, 0x32)
    data1 = bus.read_byte_data(0x53, 0x33)


# Convert the data to 10-bits
    xAccl = ((data1 & 0x03) * 256) + data0
    if xAccl > 511 :
        xAccl -= 1024


# ADXL345 address, 0x53(83)
# Read data back from 0x34(52), 2
bytes# Y-Axis LSB, Y-Axis MSB
```

```
data0 = bus.read_byte_data(0x53, 0x34)
 data1 = bus.read_byte_data(0x53, 0x35)


# Convert the data to 10-bits
    yAccl = ((data1 & 0x03) * 256) + data0
    if yAccl > 511 :
        yAccl -= 1024

# ADXL345 address, 0x53(83)
# Read data back from 0x36(54), 2
bytes# Z-Axis LSB, Z-Axis MSB
    data0 = bus.read_byte_data(0x53, 0x36)
    data1 = bus.read_byte_data(0x53, 0x37)

# Convert the data to 10-bits
    zAccl = ((data1 & 0x03) * 256) + data0
    if zAccl > 511 :
        zAccl -= 1024

# Output data to screen
    print "Acceleration in X-Axis : %d" %xAccl
    print "Acceleration in Y-Axis : %d" %yAccl
    print "Acceleration in Z-Axis : %d" %zAccl
    time.sleep(5)

    urllib.urlopen("https://api.thingspeak.com/update?api_key=JECTKF4RJR44LGVY&field1=+
    str(xAccl,yAccl,zAccl)")
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.

- **YOUR Report** must include:
  - Introduction: A brief about the experiment
  - Material and Methods/ Procedure
  - Analysis
  - Results and Discussion
  - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet

The End

Exp No.12: Monitoring LDR Data Using Thingspeak and ETS IoT Kit

Prepared By: Eng. Samiha Alfalahat
Reviewed by Dr. Saud Althunibat

1.10. Experiment No. 11

Introduction

An LDR, also known as a photo resistor, photocell, or photoconductor, is a light dependent resistor. It is a resistor whose resistance varies with the amount of light falling on its surface. When light shines on the resistor, the resistance changes. The sensitivity of light dependent resistors or photoresistors varies with incident light wavelength. LDRs are made of semiconductor materials in order to have light sensitive properties. The resistance of the LDR increases as the light level decreases. As this resistance rises in comparison to the other resistor, which has a fixed resistance, the voltage dropped across the LDR rises as well.

Objectives

1. this Experiment used to determine the varying the resistance with the help of lightby using the Light Dependent Resistor

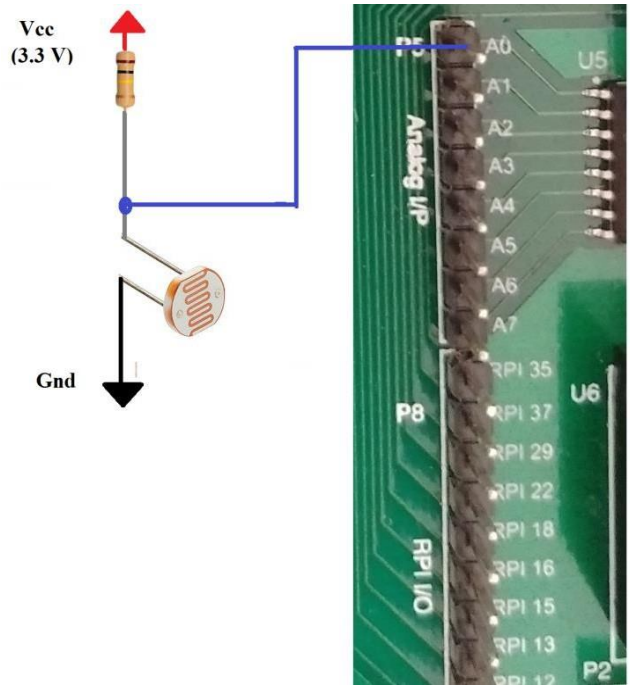
Experiment Components:

1. ETS IoT Trainer Kit
2. Light Dependent Resistor Module
3. Power Adaptor
4. HDMI to VGA Converter
5. Monitor

Experiment Procedure

Using the LDR module and the ETS IoT Kit, we determine the light dependence of the atmosphere. Here, we measure the intensity of the light and vary the module's resistance. To analyze the value light dependence in the room, we read the analog value from the LDR module and publish the data to the ThingSpeak cloud.

Circuit Diagram



Code Program

```
import time

import Adafruit_GPIO.SPI as SPIimport
Adafruit_MCP3008
 import urllib
CLK  = 23
MISO = 21

MOSI = 19

CS  = 24

mcp = Adafruit_MCP3008.MCP3008(clk=CLK, cs=CS, miso=MISO, mosi=MOSI)SPI_PORT =
0
SPI_DEVICE = 0
mcp = Adafruit_MCP3008.MCP3008(spi=SPI.SpiDev(SPI_PORT, SPI_DEVICE))
```

```
# Main program loop.
while True:
 i=mcp.read_adc(0)
 print "LDR Value:",  i
  urllib.urlopen("https://api.thingspeak.com/update?api_key=8NGROYPVG3A1X3JE
&field1="+str(i))
  time.sleep(5)
```

Instructions for Preparation of Lab Report

- Before preparing your report, complete the tasks and answer questions throughout the lab sheet.
- **YOUR Report** must include:
  - Introduction: A brief about the experiment
  - Material and Methods/ Procedure
  - Analysis
  - Results and Discussion
  - Conclusion

Additional Exercises:

will be selected by Lab instructor to provide students with different PV Module datasheet