**Practical Assignment**

**Subject-Advanced Embedded Programming**

**Subject code-CAT-851**

**Question 1. Create a web application based on .NET framework/Android(Based on Specialization Opted by student) App based on following Theme.**

**Submission Date & Time for Assignment: 02-04-2020 (10.00pm)**

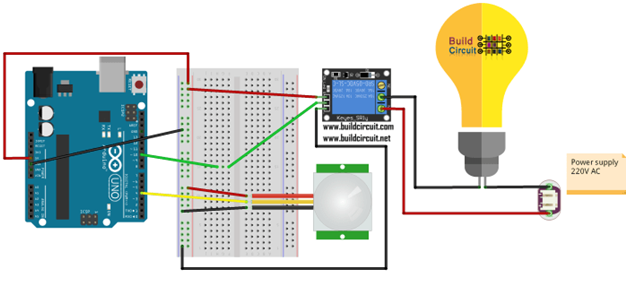
* 1. You can choose one hardware from Raspberry pi and Arduino.
  2. Circuit Diagram is required with clearly visibility of wiring that you have implemented for making the connections.
  3. You should implement any cloud service to send the data from App to Gateway for controlling the sensors.
  4. Live Status of All mentioned Equipment should be clearly visible whether it is ON or OFF with color code green for ON red for OFF.

1. Control all the electrical equipment by using Raspberry pi/Arduino.
2. Consider the following details for designing of the interface.
3. Room is having – 2 Tube Lights,2 Fans, smoke sensor with Alarm
4. Dining room is having- 1 Tv, 2 Fans,2 Tube Lights, Motor controlled windows and doors.

(Note: Motor controlled doors and windows are like automatic when we go closer to door it will be opened automatically either by using PIR sensor,IR sensor,Ultrasonic Sensor.)

1. Kitchen is having- 1 Tube light, 1 Fan, 1 Induction Stove,Gas Sensor.
2. Washroom is having – 1 Light ,1 Water geyser
3. Main Gate is having- 2 Cameras(Pi cameras) which will be controlled from APP.
4. Get the live Level of Water in Water Tank with Alarm at full state.
5. Get the temperature of Room and Take a decision to turn on the AC after getting presence of person inside the room.
6. Auto off features for Lights and Fans on getting No movement in Room for 3 minutes.

**Circuit Diagram:-**



const int negativeRELE = 10;

const int pinPIR = 4;

void setup () {

pinMode (pinRELE, OUT);

pinMode (pinPIR, INPUT);

}

void loop () {

int valued = digitalRead (pinPIR);

if (value == HIGH) {

digitalWrite (pinRELE, HIGH);

delay (5000);

digitalWrite (pinRELE, DOWN);

}

}

The main aim of this practical is to control our home appliances from all over the world at any time.

**Requirements:-**

1. Visual Studio.
2. Arduino IDE.
3. LED Light (1 number).
4. Aurduino Board.

**Step 1-** Programming Arduino

Open Arduino IDE, if already installed or download and install it from this link.

Now, open the IDE and enter the code, given below in it.

char incomingOption;

void setup()

{

pinMode(13, OUTPUT);

Serial.begin(9600);

}

void loop()

{

incomingOption = Serial.read();

if (incomingOption=='1')

{

digitalWrite(13, HIGH);

}

else if (incomingOption=='0')

{

digitalWrite(13, LOW);

}

}

Save this sketch in any location and then verify the code.

**Step 2-** Designing in Visual Studio

Open Visual Studio and create new Windows Form Application with any name, you want.

Once you have created a new form Application, open the designer Window of Form1.

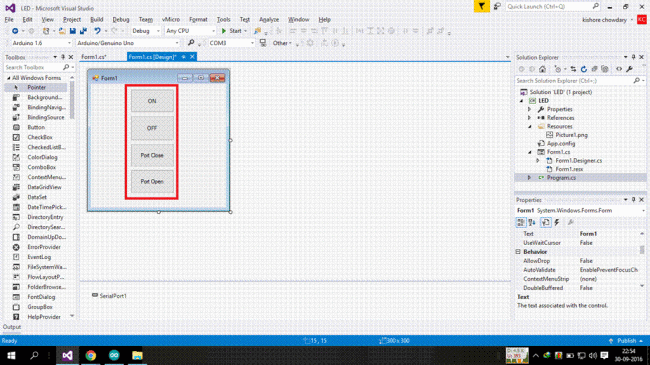
Now, from the tools box, drag and drop the SerialPort tool into Form 1.

This will help you to communicate your program with the Arduino board.

Now, add four buttons to Form 1.

Name the buttons as ON, OFF, Port Open and Port Close.

It will appear as the screenshot, given below-



**Step 3 -** Writing C# code in Visual Studio

You have four buttons. Hence, you need to write a separate code for each button.

Double click on "ON button". You will get a coding page. Paste the code, given below, there-

private void button1\_Click(object sender, EventArgs e)

{

try

{

SerialPort1.Write("1"); //send 1 to Arduino

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

Double click on "OFF button" and paste the code, given below, in it-

private void button2\_Click(object sender, EventArgs e)

{

try

{

SerialPort1.Write("0"); //send 0 to Arduino

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

Now, double click Form1.

You will get a coding page for it.

Paste the code, given below, in it-

private void Form1\_Load(object sender, EventArgs e)

{

SerialPort1.Open(); //open serialPort

}

Now, double click on the Port Close button and paste the code, given below, in it.

private void button3\_Click(object sender, EventArgs e)

{

SerialPort1.Close(); //close serialPort

}

Now, double click on the Port Open button and paste the code, given below, in it-

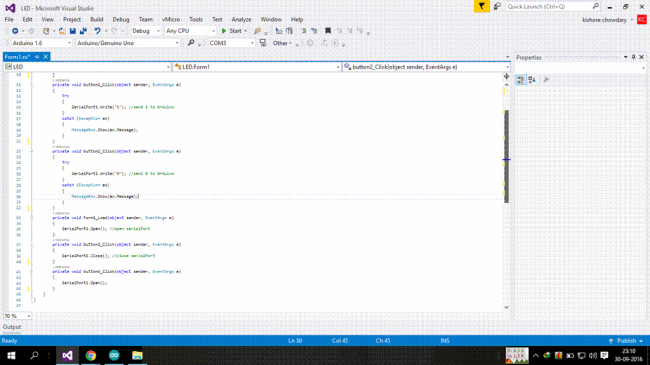
private void button4\_Click(object sender, EventArgs e)

{

SerialPort1.Open();

}

Once you finish all the code, the total view will be like the screen shot, shown below-



The complete set of code is given below. It will show some errors, if you copy and paste the code without double clicking the buttons and the Form. It is good to follow the steps correctly.

using System;

using System.Windows.Forms;

namespace LED

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

SerialPort1.Write("1"); //send 1 to Arduino

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

SerialPort1.Write("0"); //send 0 to Arduino

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void Form1\_Load(object sender, EventArgs e)

{

SerialPort1.Open(); //open serialPort

}

private void button3\_Click(object sender, EventArgs e)

{

SerialPort1.Close(); //close serialPort

}

private void button4\_Click(object sender, EventArgs e)

{

SerialPort1.Open();

}

}

}

**Step 4 -** Connecting the LED to the Arduino and uploading program

Take the positive side of the LED, connect it to the pin 13 and negative side to the GND.

Now, simply verify whether the COM ports are same in the Arduino IDE and the properties of the Form in Visual Studio.

After you verify it, upload the sketch into Arduino board. The code, which you wrote in IDE in the beginning is called as sketch.

Now, you are completely ready with the whole setup.

**Final step -** Working of the project

Deploy the whole setup.

Now, click on "ON button". The LED will glow.

Now, click on the "OFF button". The LED will turn off.

Now; click on the Port Close button and try to turn on the LED. It will show a message, which port is closed.

Now, click on the Port Open button and try to turn on the LED. It will now glow.

Thus you have made your own LED lighting program using the C# language in visual studio.

**Ultrasonic sensor with PIR:-**

**Goals:**

I want to detect motion with the PIR sensor and measure the distance of the target while the PIR detect motion.

One sound alarm can be triggered according to the switch state.

PIR state and Distance to target should be printed via serial port

I use a 220 Ohms resistor connected in between arduino and the piezo buzzer to reduce the volume of the alarm during test.

**Components:**

*1 Arduino Uno*

*2 resistor 220Ohms*

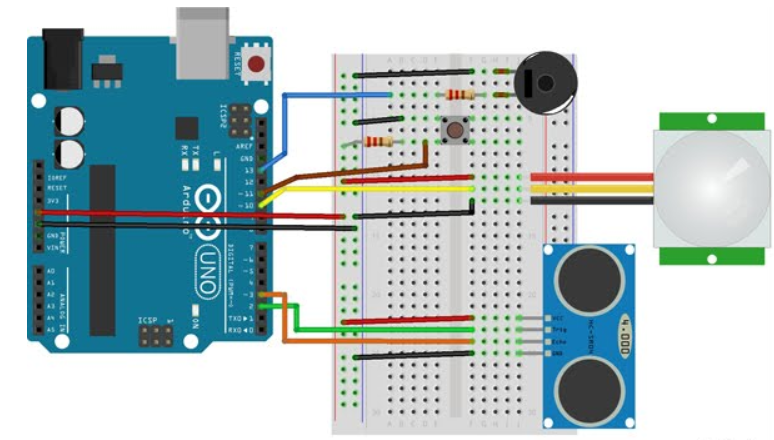
*1*[*HC-SR04*](https://sites.google.com/site/marcobotprojects/data-sheet/ultrasonicsensorhcsr04)

*1*[*PIR D-SUN*](https://sites.google.com/site/marcobotprojects/data-sheet/pird-sundsn-fir800)

*1 Micro Contact.*

*1 Piezo Buzzer*

**Bread Board:**



**Circuit Diagram:**

int switch2 = 11; // Push button to activate or desactivate alarm sound

int echo = 2; // Echo Pin of the ultrasonic sensor

int trigger = 3; // Trigger pin ofthe ultrasonic sensor

int buzzer = 13; // Sound alarm buzzer

int pir = 10; // PIR sensor Pin

int distance = 0; // Default distance measured by ultrasonic sensor

boolean alarm = LOW; // Alarm state LOW Sound trigger, HIGH no sound

// the time when the sensor outputs a low impulse

long unsigned int lowIn;

// the amount of milliseconds the sensor has to be low

// before we assume all motion has stopped

long unsigned int pause = 5000;

boolean lockLow = true;

boolean takeLowTime;

int calibrationTime = 30;

// sinusoid alarm sound variable

float sinVal;

int toneVal;

// Instance creation for the bounce library

Bounce switch2Bounce = Bounce();

// Instance creation for the New ping Library

NewPing sonar(trigger, echo, MAX\_DISTANCE);

void **setup**() {

  Serial.begin(9600);

  pinMode(pir, INPUT);

  pinMode(buzzer, OUTPUT);

  pinMode(switch2,INPUT);

  // Attaching the switch2 to the bounce object with a 100ms bounce intervall

  switch2Bounce.attach(switch2);

  switch2Bounce.interval(100);

  // activation of the pull up resistor for the PIR pin

  digitalWrite(pir, HIGH);

  // give the sensor some time to calibrate

  Serial.print("calibrating sensor ");

  for (int i = 0; i < calibrationTime; i++) {

    Serial.print(".");

    delay(1000);

  }

  Serial.println(" done");

  Serial.println("SENSOR ACTIVE");

  delay(50);

}

////////////////////////////

//LOOP

void loop() {

  readPIR();      // Call the function to Read the PIR sensor

  alarmState();  // Call the function to check the state of the switch2

}

//////////////////

//PIR Funcrion

void readPIR() {

  if (digitalRead(pir) == HIGH) {

    soundAlarm();      //Call the function to sound or not the alarm when motion is detected

    pingDist();        // Call the function to measure distance from th ultrasonic sensor if motion is detected

    if (lockLow) {

      //makes sure we wait for a transition to LOW before any further output is made:

      lockLow = false;

      Serial.println("---");

      Serial.print("motion detected at ");

      Serial.print(millis() / 1000);

      Serial.println(" sec");

      delay(50);

    }

    takeLowTime = true;

  }

  if (digitalRead(pir) == LOW) {

    if (takeLowTime) {

      lowIn = millis(); //save the time of the transition from high to LOW

      takeLowTime = false; //make sure this is only done at the start of a LOW phase

    }

    //if the sensor is low for more than the given pause,

    //we assume that no more motion is going to happen

    if (!lockLow && millis() - lowIn > pause) {

      //makes sure this block of code is only executed again after

      //a new motion sequence has been detected

      lockLow = true;

      Serial.print("motion ended at "); //output

      Serial.print((millis() - pause) / 1000);

      Serial.println(" sec");

      delay(50);

    }

  }

}

//////////////

// Ultrasonic sensor distance reading

void pingDist () {

  delay(50);                      // Wait 50ms between pings (about 20 pings/sec). 29ms should be the shortest delay between pings.

  distance = sonar.convert\_cm(sonar.ping\_median(10));

  Serial.print("Distance");

  Serial.println(distance); // Convert ping time to distance in cm and print result (0 = outside set distance range)

}

////////////////////

// Alarm Sound a sinusiod is generated and outputed on the buzzer pin

void soundAlarm() {

  if (alarm==LOW){

  for (int x = 0; x < 180; x++) {

    // convert degrees to radians then obtain value

    sinVal = (sin(x \* (3.1412 / 180)));

    // generate a frequency from the sin value

    toneVal = 2000 + (int(sinVal \* 1000));

    tone(buzzer, toneVal);

    delay(1);

 }

 noTone(buzzer);

  }

}

/////////////////

//Check the status of the switch and detect the falling edge.

void alarmState() {

 if (switch2Bounce.update()) {

    if (switch2Bounce.fell()) {

      alarm = !alarm;

      Serial.print("Alarm=");

      Serial.println(alarm);

    }

}

}

**To access camera over internet in iot we use the following code:-**

import urllib

import cv2,time

import numpy as np

first\_frame = None

url= 'http://192.168.2.52:8080/shot.jpg’

imgResp = urllib.urlopen(url)

imgNp = np.array(bytearray(imgResp.read()),dtype=np.uint8)

img = cv2.imdecode(imgNp,-1)

cv2.namedWindow(‘image’,cv2.WINDOW\_NORMAL)

cv2.resizeWindow(‘image’,600,600)

i=0

while True:

imgResp=urllib.urlopen(url)

imgNp=np.array(bytearray(imgResp.read()),dtype=n.uint8)

img=cv2.imdecode(imgNp,-1)

cv2.imshow(‘image’,img)

cv2.imwrite(‘test.jpg’’,img)

key=cv2.waitKey(1)

if key==ord(‘q’):

break