IR_arduino

```
int SensorPin = 2;
int OutputPin = 13;
void setup() {
pinMode(OutputPin, OUTPUT);
pinMode(SensorPin, INPUT);
Serial.begin(9600);
}
void loop() {
int SensorValue = digitalRead(SensorPin);
Serial.print("SensorPin Value: ");
Serial.println(SensorValue);
delay(1000);
if (SensorValue==LOW){ // LOW MEANS Object Detected
digitalWrite(OutputPin, HIGH);
}
else
{
digitalWrite(OutputPin, LOW);
}
}
```

IR_raspberry

```
import RPi.GPIO as GPIO #GPIO library
import time #library for sleep
import board
import digitalio
import adafruit_character_lcd.character_lcd as characterlcd
GPIO.setmode(GPIO.BCM)
lcd_columns = 16
lcd_rows = 2
lcd_rs = digitalio.DigitalInOut(board.D5)
lcd_en = digitalio.DigitalInOut(board.D6)
lcd_d4 = digitalio.DigitalInOut(board.D12)
lcd_d5 = digitalio.DigitalInOut(board.D13)
lcd_d6 = digitalio.DigitalInOut(board.D16)
lcd_d7 = digitalio.DigitalInOut(board.D17)
lcd = characterlcd.Character_LCD_Mono(
  lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns, lcd_rows)
IR_OUT = 21
BUZ = 22
GPIO.setup(IR_OUT, GPIO.IN)
GPIO.setup(BUZ, GPIO.OUT)
def destroy():
  GPIO.output (BUZ, GPIO.LOW)
  GPIO.cleanup()
if __name__ =='__main__':
  try:
    while True:
      IR_State = GPIO.input(IR_OUT)
      if (IR_State == True):
        print ("OBJECT DETECTED")
        lcd.clear()
        lcd.message ="OBJECT DETECTED"
        GPIO.output (BUZ, GPIO.HIGH)
        time.sleep(0.5)
        GPIO.output (BUZ, GPIO.LOW)
      else:
        lcd.clear()
        lcd.message ="NO OBJECT"
        time.sleep(0.5)
        print ("NO OBJECT")
  except KeyboardInterrupt:
    destroy()
```

DHT11_arduino

```
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
void setup() {
Serial.begin(9600);
dht.begin();
}
void loop() {
delay(2000);
float temperature = dht.readTemperature();
float humidity = dht.readHumidity();
if (isnan(temperature) || isnan(humidity)) {
Serial.println("Failed to read from DHT sensor");
return;
}
Serial.print("Temperature: ");
Serial.print(temperature);
Serial.println(" °C");
Serial.print("Humidity: ");
Serial.print(humidity);
Serial.println(" %");
}
```

DHT11_raspberry

```
#sudo pip3 install adafruit-circuitpython-dht
#sudo apt-get install libgpiod2
import time
import board
import digitalio
import adafruit_character_lcd.character_lcd as characterlcd
import adafruit_dht
import RPi.GPIO as GPIO
dhtDevice = adafruit_dht.DHT11(board.D19)
lcd_columns = 16
lcd_rows = 2
lcd_rs = digitalio.DigitalInOut(board.D5)
lcd_en = digitalio.DigitalInOut(board.D6)
lcd_d4 = digitalio.DigitalInOut(board.D12)
lcd_d5 = digitalio.DigitalInOut(board.D13)
lcd_d6 = digitalio.DigitalInOut(board.D16)
lcd_d7 = digitalio.DigitalInOut(board.D17)
lcd = characterlcd.Character_LCD_Mono(
  lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns, lcd_rows)
if __name__ == '__main__':
  while True:
    try:
      temperature_c = dhtDevice.temperature
      temperature_f = temperature_c * (9 / 5) + 32
      humidity = dhtDevice.humidity
      print("Temp: {:.1f} F / {:.1f} C    Humidity: {}% "
          .format(temperature_f, temperature_c, humidity))
      lcd.clear()
      #lcd_line_1 = "Temperature:" + str(temperature_c) + " C"
      #lcd_line_2 = "\nHumidity:"+ str(humidity) + " %"
      #lcd.message = lcd_line_1 + lcd_line_2;
      lcd.message = ("Temper:%.1f C " %temperature_c)
      lcd.message = ("\nHumidity:%.1F " %humidity)
      time.sleep(2.0)
    except RuntimeError as error:
      print(error.args[0])
      time.sleep(2.0)
      continue
  except KeyboardInterrupt:
    GPIO.cleanup()
    print ('Exiting Program')
    exit()
```

Temperature_arduino

```
#define TRIG_PIN 9
#define ECHO_PIN 10
void setup() {
Serial.begin(9600);
pinMode(TRIG_PIN, OUTPUT);
pinMode(ECHO_PIN, INPUT);
}
void loop() {
digitalWrite(TRIG_PIN, LOW);
delayMicroseconds(2);
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
long duration = pulseIn(ECHO_PIN, HIGH);
loat distance_cm = (duration * 0.0343) / 2;
Serial.print("Distance: ");
Serial.print(distance_cm);
Serial.println(" cm");
delay(1000); // Wait for a second before taking the next measurement
}
```

Temperature_raspberry

```
import RPi.GPIO as GPIO #library for Raspberry Pi GPIOs
import time #library to use sleep function
import board
import digitalio
import adafruit_character_lcd.character_lcd as characterlcd
GPIO.setmode(GPIO.BCM)
lcd_columns = 16
lcd_rows = 2
lcd_rs = digitalio.DigitalInOut(board.D5)
lcd_en = digitalio.DigitalInOut(board.D6)
lcd_d4 = digitalio.DigitalInOut(board.D12)
lcd_d5 = digitalio.DigitalInOut(board.D13)
lcd_d6 = digitalio.DigitalInOut(board.D16)
lcd_d7 = digitalio.DigitalInOut(board.D17)
lcd = characterlcd.Character_LCD_Mono(
  lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns, lcd_rows)
TRIGGER = 19
ECHO = 20
 GPIO.setup(TRIGGER, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)
lcd.clear()
 def distance():
  GPIO.output(TRIGGER, True)
   time.sleep(0.00001)
  GPIO.output(TRIGGER, False)
   StartTime = time.time()
  StopTime = time.time()
   while GPIO.input(ECHO) == 0:
     StartTime = time.time()
  while GPIO.input(ECHO) == 1:
     StopTime = time.time()
   TimeElapsed = StopTime - StartTime
  distance = (TimeElapsed * 34300) / 2
   return distance
if __name__ == '__main__':
  try:
     while True:
       dist = distance()
       print ("Measured Distance = %.1f cm" % dist)
       lcd.clear()
       lcd.message = ("Dist.:%.1f cm" % dist)
       time.sleep(2)
    except KeyboardInterrupt:
      print("Measurement stopped by User")
      GPIO.cleanup()
```

PIR_arduino

```
int ledPin = 13;
int inputPin = 2;
int pirState = LOW;
int val = 0;
void setup() {
pinMode(ledPin, OUTPUT);
pinMode(inputPin, INPUT);
Serial.begin(9600);
}
void loop(){
val = digitalRead(inputPin);
if (val == HIGH) {
digitalWrite(ledPin, HIGH);
if (pirState == LOW) {
Serial.println("Motion detected!");
pirState = HIGH;
}
} else {
digitalWrite(ledPin, LOW);
if (pirState == HIGH){
Serial.println("Motion ended!");
pirState = LOW;
}
}
}
```

PIR_raspberry

```
import RPi.GPIO as GPIO #GPIO library
import time #library for sleep
import board
import digitalio
import adafruit_character_lcd.character_lcd as characterlcd
GPIO.setmode(GPIO.BCM)
lcd_columns = 16
lcd_rows = 2
lcd_rs = digitalio.DigitalInOut(board.D5)
lcd_en = digitalio.DigitalInOut(board.D6)
lcd_d4 = digitalio.DigitalInOut(board.D12)
lcd_d5 = digitalio.DigitalInOut(board.D13)
lcd_d6 = digitalio.DigitalInOut(board.D16)
lcd_d7 = digitalio.DigitalInOut(board.D17)
lcd = characterlcd.Character_LCD_Mono(
  lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns, lcd_rows)
PIR = 21
BUZ = 22
GPIO.setup(PIR, GPIO.IN)
GPIO.setup(BUZ, GPIO.OUT)
if __name__ =='__main__':
  try:
    while True:
      PIR_State = GPIO.input(PIR)
      if (PIR_State == True):
        print ("Motion Detected")
        lcd.clear()
        lcd.message = "Motion Detected"
        GPIO.output (BUZ, GPIO.HIGH)
        time.sleep(0.5)
        GPIO.output (BUZ, GPIO.LOW)
        time.sleep(0.5)
      else:
        lcd.clear()
        lcd.message = "NO Motion"
        print ("No Motion")
        time.sleep(0.5)
  except KeyboardInterrupt:
    GPIO.cleanup()
```

LED_arduino

```
int LEDpin = 13;
int delayT = 1000;
void setup() {
pinMode(LEDpin, OUTPUT);
}
void loop() {
digitalWrite(LEDpin, HIGH);
delay(delayT);
digitalWrite(LEDpin, LOW);
delay(delayT);
}
```

LED_raspberry

```
import RPi.GPIO as GPIO
import time
LED = 5
GPIO.setmode (GPIO.BCM)
GPIO.setup(LED, GPIO.OUT)
GPIO.output(LED, GPIO.HIGH)
def blink():
  GPIO.output(LED, GPIO.HIGH)
  time.sleep(1)
  GPIO.output(LED, GPIO.LOW)
  time.sleep(1)
def destroy():
  GPIO.output(LED, GPIO.LOW)
  GPIO.cleanup()
if __name__ == '__main__':
  try:
    while True:
      blink()
  except KeyboardInterrupt:
    destroy()
```

Zigbee_arduino

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX, TX
void setup() {
Serial.begin(57600);
while (!Serial) {
;
}
Serial.println("Goodnight moon!");
mySerial.begin(4800);
mySerial.println("Hello, world?");
}
void loop() { // run over and over
if (mySerial.available()) {
Serial.write(mySerial.read());
}
if (Serial.available()) {
mySerial.write(Serial.read());
}
}
```

Zigbee_raspberry

```
# Install Python Serial Package
# sudo pip3 install pyserial
# Check the COM PORT Number
# sudo dmesg|grep tty
# Use ttyS0 for on-board Serial
# Use ttyUSB0  or ttyUSB1 for USB to serial converter
# after checking the com port number
import time
import serial
ser = serial.Serial(
   port='/dev/ttyUSB2',
   baudrate = 9600,
   parity=serial.PARITY_NONE,
   stopbits=serial.STOPBITS_ONE,
   bytesize=serial.EIGHTBITS,
   timeout=1
)
counter=0
if __name__ == "__main__":
  try:
    while True:
      ser.write(str.encode('Write counter: %d \n'%(counter)))
      time.sleep(1)
      counter += 1
      x=ser.readline().strip()
      if len(x) != 0 :
        print(x)
  except KeyboardInterrupt:
    ser.close()
    print ('Exiting Program')
```

MQTT

```
# Install the MQTT Publisher
# sudo pip3 install paho-mqtt
# open the MQTT browser client in web browser
# http://www.hivemq.com/demos/websocket-client/
import os
import sys
import time
import board
import adafruit_dht
import paho.mqtt.client as mqtt
import json
dhtDevice = adafruit_dht.DHT11(board.D19, use_pulseio=False)
sensor_data = {'temperature': 0, 'humidity': 0}
MQTTServer = 'broker.mqttdashboard.com'
client = mqtt.Client()
client.connect(MQTTServer, 1883, 8000)
client.loop_start()
if __name__ == '__main__':
  while True:
    try:
      # Print the values to the serial port
      temperature = dhtDevice.temperature
      humidity = dhtDevice.humidity
      print("Temp: {:.1f} C    Humidity: {}% ".format( temperature, humidity))
      time.sleep(2.0)
      sensor_data['temperature'] = temperature
      sensor_data['humidity'] = humidity
      client.publish('RPI4_MQTT', json.dumps(sensor_data), 1)
      time.sleep(5)
    except RuntimeError as error:
      print(error.args[0])
      time.sleep(2.0)
      continue
    except KeyboardInterrupt:
      client.loop_stop()
      client.disconnect()
      print ('Exiting Program')
      exit()
```

Motor_arduino

```
int DCMOTOR = 13;
int delayT = 1000;
void setup() {
// put your setup code here, to run once:
pinMode(DCMOTOR, OUTPUT);
}
void loop() {
// put your main code here, to run repeatedly:
digitalWrite(DCMOTOR, HIGH);
delay(delayT);
digitalWrite(LEDpin, LOW);
delay(delayT);
```

Motor_raspberry

```
import RPi.GPIO as GPIO
import time
EN1 = 25
IN1 = 26
IN2 = 27
GPIO.setmode(GPIO.BCM)
GPIO.setup(EN1,GPIO.OUT)
GPIO.setup(IN1,GPIO.OUT)
GPIO.setup(IN2, GPIO.OUT)
def destroy():
  GPIO.output(25, False)
  GPIO.output(26, False)
  GPIO.output(27, False)
  GPIO.cleanup()
def Clockwise():
  GPIO.output(25, True)
  GPIO.output(26, True)
  GPIO.output(27, False)
def AntiClockwise():
  GPIO.output(25, True)
  GPIO.output(26, False)
  GPIO.output(27, True)
def Stop():
  GPIO.output(25, False)
  GPIO.output(26, False)
  GPIO.output(27, False)
if __name__ == '__main__':
  try:
    while True:
      Clockwise()
      time.sleep(2)
      Stop()
      time.sleep(1)
      AntiClockwise()
      time.sleep(2)
      Stop()
      time.sleep(1)
  except KeyboardInterrupt:
    destroy()
```

Bluetooth

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX, TX
void setup() {
Serial.begin(57600);
while (!Serial) {
;
}
Serial.println("Goodnight moon!");
mySerial.begin(4800);
mySerial.println("Hello, world?");
}
void loop() { // run over and over
if (mySerial.available()) {
Serial.write(mySerial.read());
}
if (Serial.available()) {
mySerial.write(Serial.read());
}
}
```