PIR Motion Sensor-Based Home Security System using Arduino

Description:

A PIR Motion sensor-based home security system using an Arduino UNO that allows you to detect the any movement of human beings.
home security system uses a PIR sensor to detect motion, an LCD display to provide information about the status of the system, a buzzer and LED light alert the owner. When the PIR sensor detects motion, The LED light will blink, followed by a buzzer, the LCD will display 'motion detected Alarm triggered' to alert the homeowner. The LCD display shows information about the status of the system, such as whether the Alarm triggered or not.
if logic state=1 Motion detected (Alarm triggered).
logic state =0 motion not detected (Alarm not triggered).

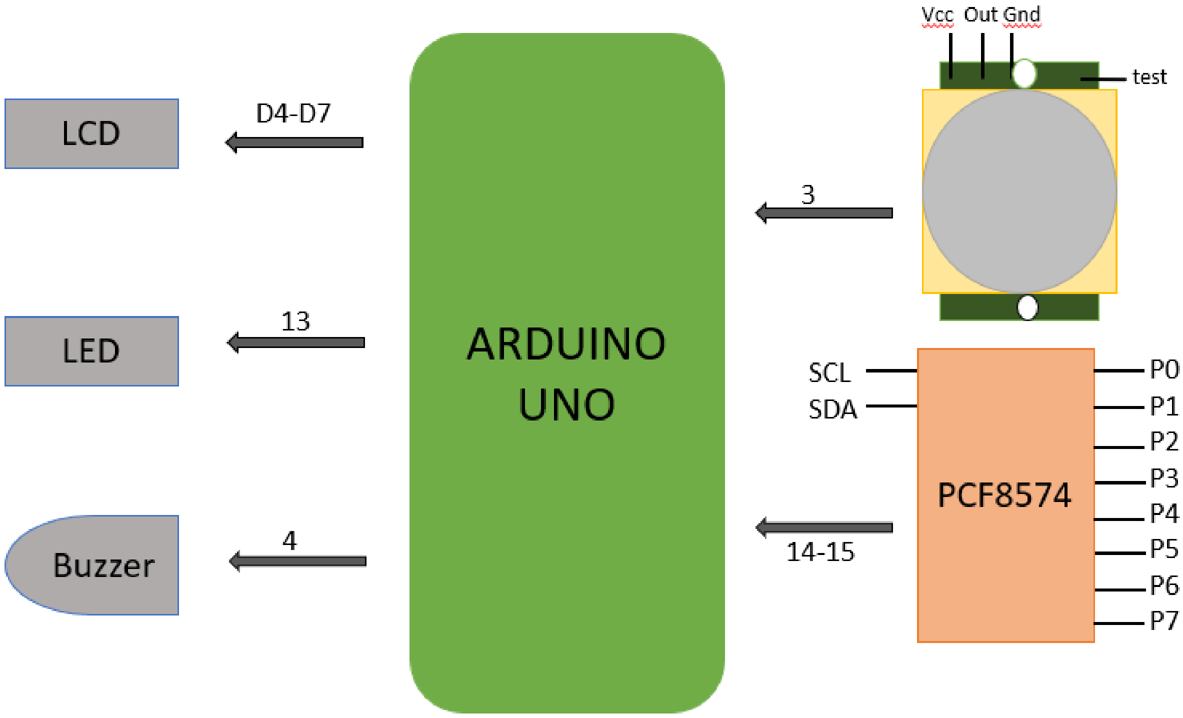
Components Description:

1.PIR Sensor: The PIR sensor detects changes in infrared radiation caused by moving objects within its field of view. When motion is detected, the sensor sends a high signal to the Arduino.

2.Buzzer: A buzzer is a simple device that makes noise that is used to produce an audible alarm when motion is detected. When the Arduino detects motion, it activates the buzzer to alert you.

3.LED: An LED can be used for visual indication. When motion is detected, the LED can be turned on to provide a visual alert.

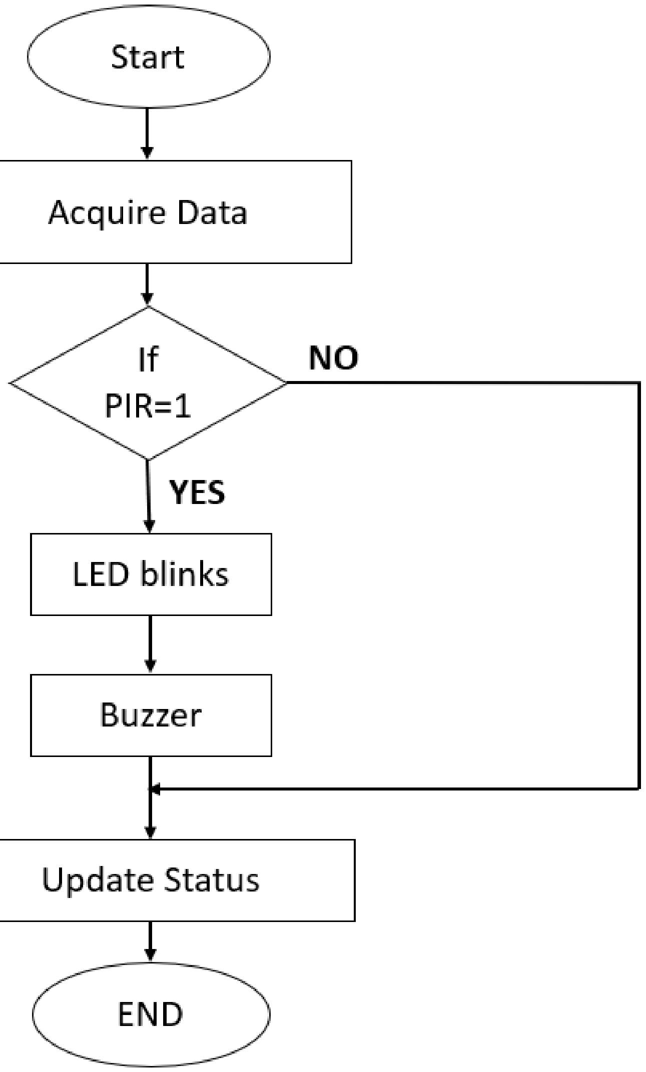
Block Diagram:



Input and Output:

S.No	Description	Name	Type	Data Direction	Specifications	Remarks
1	PIR Sensor VCC	VCC	INP	DI	Digital	Active High
2	PIR Sensor OUT	3	INP	DI	Digital	Active High
3	PIR Sensor GND	GND	INP	DI	Digital	Active High
4	PCF8574 data pin 1	SCL	INP	DI	Digital	Active High
5	PCF8574 data pin 2	SDA	INP/OUT	D(I/0)	Digital	Active High
6	LCD data pin 1	D4	OUT	DO	Digital	Active High
7	LCD data pin 2	D5	OUT	DO	Digital	Active High
8	LCD data pin 3	D6	OUT	DO	Digital	Active High
9	LCD data pin 4	D7	OUT	DO	Digital	Active High
10	LCD RST	RST	OUT	DO	Digital	Active High
11	LCD EN	EN	OUT	DO	Digital	Active High
12	Buzzer	4	OUT	DO	Digital	Active High
13	LED	13	OUT	DO	Digital	Active High
14	PCF8574 data pin 3	P0	INP/OUT	D(I/0)	Digital	Active High
15	PCF8574 data pin 4	P1	INP/OUT	D(I/0)	Digital	Active High
16	PCF8574 data pin 5	P2	INP/OUT	D(I/0)	Digital	Active High
17	PCF8574 data pin 6	P4	INP/OUT	D(I/0)	Digital	Active High
18	PCF8574 data pin 7	P5	INP/OUT	D(I/0)	Digital	Active High
19	PCF8574 data pin 8	P6	INP/OUT	D(I/0)	Digital	Active High
20	PCF8574 data pin 9	P7	INP/OUT	D(I/0)	Digital	Active high

Flow Chart:

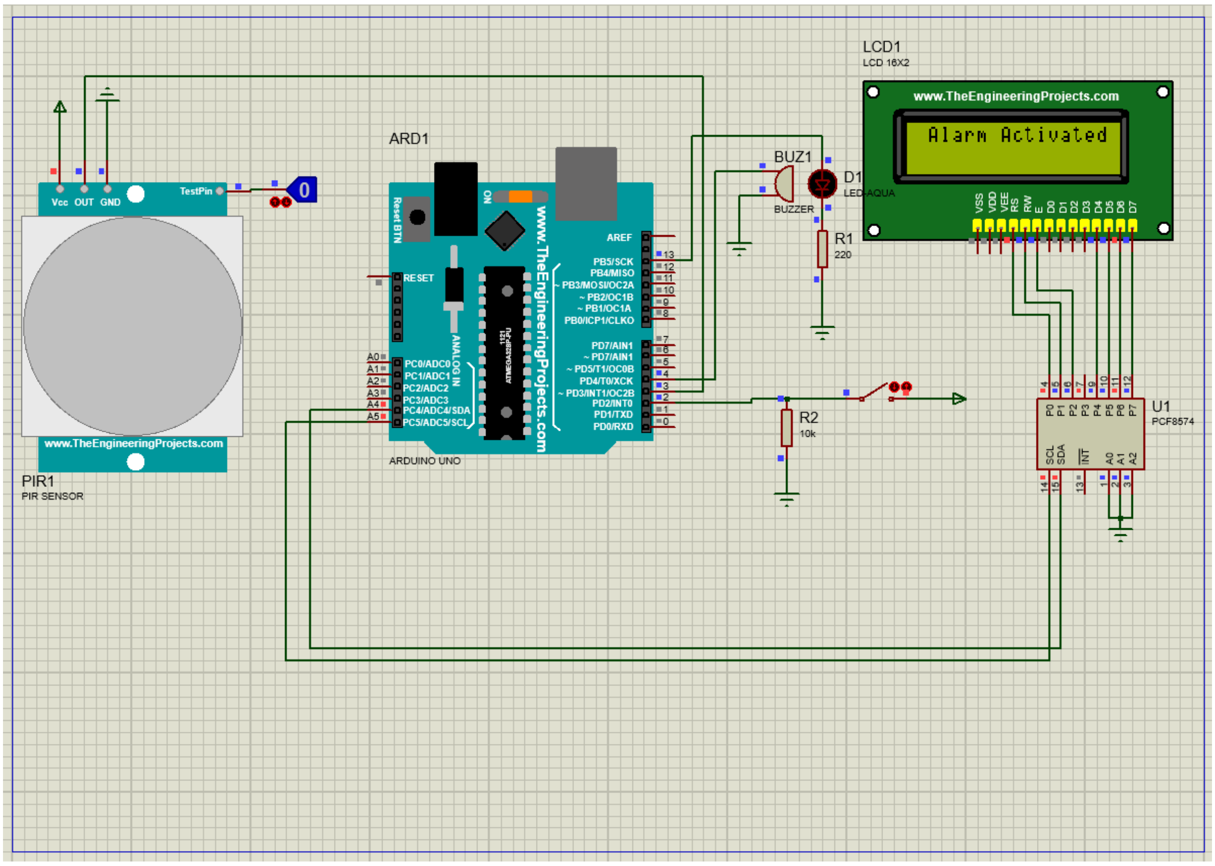


Source Code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

int calibrationTime = 10;     //the time when the sensor outputs a low impulse
int buttonState;              // the current reading from the input pin
int lastButtonState = LOW;    // the previous reading from the input pin
int ledState = LOW;           // ledState used to set the LED
boolean lockLow = true;

const int buttonPin = 2;   //pushbutton attached to pin 2
const int pirPin = 3;      //PIR motion sensor output attached to pin 3
const int buzPin = 4;      //buzzer attached to pin 4
const int ledPin = 13;     //led attached to pin 13
```

```
unsigned long lastDebounceTime = 0;   // the last time the output pin was toggled
unsigned long debounceDelay = 50;     // the debounce time; increase if the output
flickers
unsigned long previousMillis = 0;     // will store last time LED was updated
const long interval = 1000;           // interval at which to blink (milliseconds)

LiquidCrystal_I2C lcd(0x20,16,2);

void setup()
{

  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(buzPin, OUTPUT);
  pinMode(buttonPin, INPUT);
  digitalWrite(pirPin, LOW);

  lcd.init();
  lcd.backlight();
  lcd.setCursor(1,0);
  lcd.print("Alarm Security");
  lcd.setCursor(5,1);
  lcd.print("System");
  delay (1000);
  lcd.clear();
  lcd.setCursor(5,0);
  lcd.print("LOADING");

  //give the sensor some time to calibrate
  for(int i = 0; i < calibrationTime; i++)
  {
    lcd.setCursor(i+3,1);
    lcd.print("*");
    delay(100);
  }
  lcd.clear();
  lcd.setCursor(1,0);
  lcd.print("Alarm Activated");
}

void loop()
{
  if (digitalRead (pirPin)==HIGH)
  {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Motion Detected");
    lcd.setCursor(0,1);
    lcd.print("Alarm Triggered");

    while (lockLow==true)
    {
      unsigned long currentMillis = millis();
      if (currentMillis - previousMillis >= interval)
      {
        previousMillis = currentMillis;     // save the last time you blinked the LED
                                            // if the LED is off turn it on and vice-
versa
      if (ledState == LOW)
      {
        ledState = HIGH;
      }

      else
      {
        ledState = LOW;
      }

      digitalWrite(ledPin, ledState);     // set the LED with the ledState of the
variable
      digitalWrite(buzPin, ledState);     // set the buzzer with the ledState of the
variable
    }

    int reading = digitalRead(buttonPin);
    // check to see if you just pressed the button
    // (i.e. the input went from LOW to HIGH), and you've waited long enough
    // since the last press to ignore any noise:
    // If the switch changed, due to noise or pressing:

    if (reading != lastButtonState)
    {// reset the debouncing timer
      lastDebounceTime = millis();
    }

    if ((millis() - lastDebounceTime) > debounceDelay)
    {
      if (reading != buttonState)
      {
        buttonState = reading;

        if (buttonState == HIGH)
        {
          lockLow=false;
          digitalWrite(ledPin, LOW);
          digitalWrite(buzPin, LOW);
          delay(50);
        }
      }
    }

    lastButtonState = reading;
   }
  }
  if (digitalRead(buttonPin)==HIGH && digitalRead(pirPin)==LOW)
  {
    lcd.clear();
    lcd.setCursor(3,0);
    lcd.print("Alarm Reset");
    delay(1000);
    lcd.clear();
    lcd.setCursor(1,0);
    lcd.print("Alarm Activated");
    lockLow = true;
}
}
```

Schematic:



LCD1
LCD 16X2
www.TheEngineeringProjects.com
Alarm Activated
VSS VDD VEE RS RW E D0 D1 D2 D3 D4 D5 D6 D7
ARD1
BUZ1
BUZZER
D1
LED-AQUA
R1
220
R2
10k
U1
PCF8574
P0 P1 P2 P3 P4 P5 P6 P7
SCL SDA INT A0 A1 A2
Vcc OUT GND
TestPin
PIR1
PIR SENSOR
www.TheEngineeringProjects.com
RESET
Reset BTN
AREF
PB5/SCK
PB4/MISO
~ PB3/MOSI/OC2A
~ PB2/OC1B
~ PB1/OC1A
PB0/ICP1/CLKO
PD7/AIN1
~ PD7/AIN1
~ PD5/T1/OC0B
PD4/T0/XCK
~ PD3/INT1/OC2B
PD2/INT0
PD1/TXD
PD0/RXD
A0 A1 A2 A3 A4 A5
PC0/ADC0
PC1/ADC1
PC2/ADC2
PC3/ADC3
PC4/ADC4/SDA
PC5/ADC5/SCL
ANALOG IN
ATMEGA328P-PU
www.TheEngineeringProjects.com
ARDUINO UNO
ON
0