# PROJECT REPORT

## MINI PROJECT [2]

*Problem Statement*: Create a burglar alarm – detect motion of a person (using a PIR sensor). If there is motion, ring the buzzer.
(Using Arduino UNO)

***Course Name***: INTERNET OF THINGS [IoT]

***Project Submitted by:*** Bhakti Raut (3rd year student at Yeshwantrao Chavan College of Engineering)

***Guided by:*** Swati Chitta (IoT Trainer – Acme grade)

# PROBLEM DEFINITION

***The project title is —*** Create a burglar alarm – detect motion of a person (using a PIR sensor). If there is motion, ring the buzzer (using Arduino UNO).

PIR sensor (Passive Infrared Sensor) is a special type of sensor which is usually used for security purposes. It detects the objects by reading the Infrared radiations emitted by the objects. Any object whose temperature is above absolute zero, emits radiation. This radiation is not visible to human eyes. The PIR sensor is designed to detect this Infrared radiation.

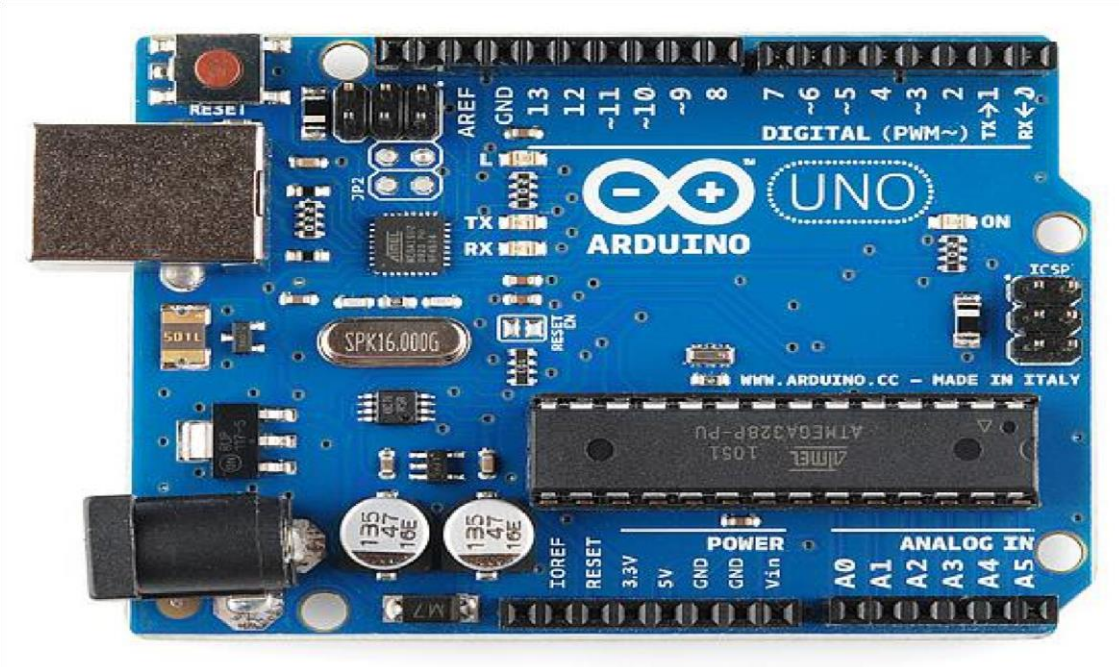
A PIR motion sensor has a pair of pyro electric sensors to detect heat energy from the surrounding environment. It helps generate an electrical signal when they are heated or cooled.

Before starting the actual project let us take a look at the Hardware and Pin Configurations of Arduino UNO and the PIR Sensor in detail.

# HARDWARE AND PIN CONFIGURATION

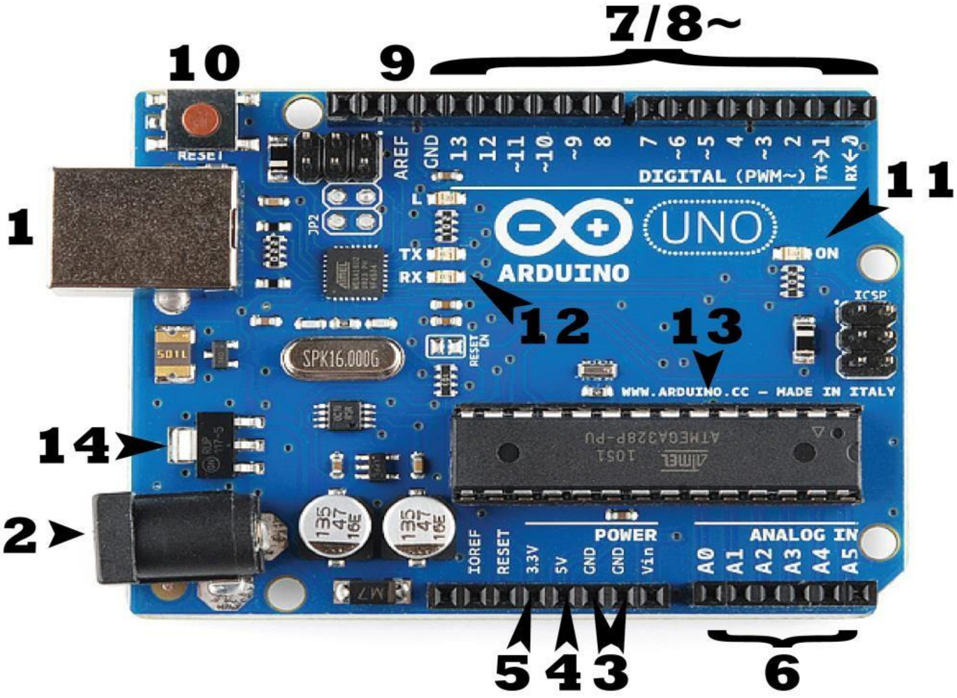
Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.



## WHAT IS ON THE BOARD? ARDUINO PIN CONFIGUARATIONS

There are many varieties of Arduino boards that can be used for different purposes. Some boards look a bit different from the one below, but most Arduinos have the majority of these components in common



- **Power (USB / Barrel Jack)**

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply (like this) that is terminated in a barrel jack. In the picture above the **USB connection is labelled (1)** and the **barrel jack is labelled (2)**. The USB connection is also how you will load code onto your Arduino board.

**NOTE:** Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) you're Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

- **Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)**

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjunction with a breadboard and some wire. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labelled on the board and used for different functions.

- **GND (3):** Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- **5V (4) & 3.3V (5):** As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- **Analog (6):** The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- **Digital (7):** Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
- **PWM (8):** You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). It is a technique for getting analog results by digital means. For example: Results involving DC motors, LED brightness adjustment, etc.
- **AREF (9):** Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

## • Reset Button

Just like the original Nintendo, the Arduino has **a reset button (10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times.

## • Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' (11). This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

- **TX RX LEDs**

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino

UNO where TX and RX appear -- once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs (12). These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

- **Main IC**

The black thing with all the metal legs is an IC, or Integrated Circuit (13). Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATMEGA line of IC's from the ATMEL Company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software.

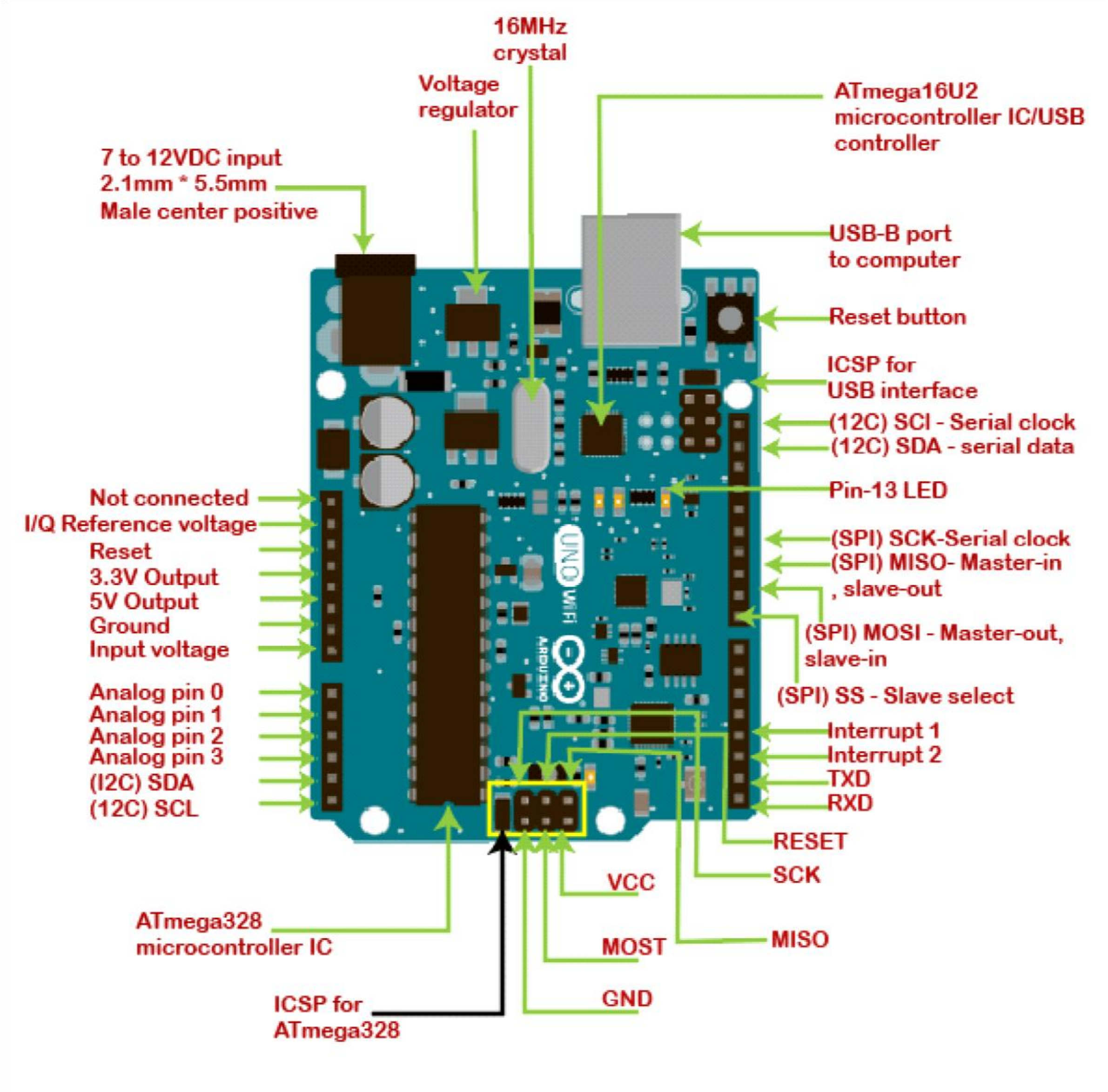
- **Voltage Regulator**

The voltage regulator (14) is not actually something you can (or should) interact with on the Arduino.

But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says -- it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

## HARDWARE CONFIGUARTIONS OF ARDUINO

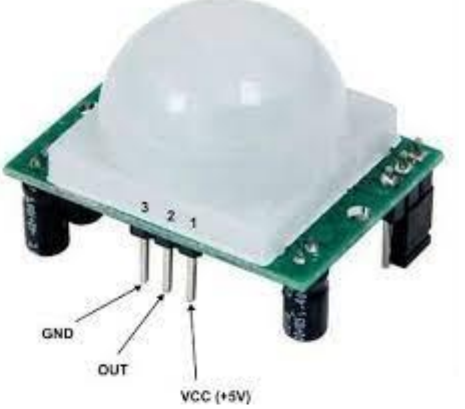
| *In-built components* | *Configurations* |
|---|---|
| SRAM | 2 KB (ATmega328P) |
| EPROM | 1 KB |
| CLOCK SPEED | 16 KHz |
| FLASH MEMORY | 32 KB, 0.5 KB boot loader |
| ATmega16U2 | Used as an USB to serial converter in UNO |
| ICSP for Atmega328 | In-Circuit Serial Programming, Arduino board's firmware |

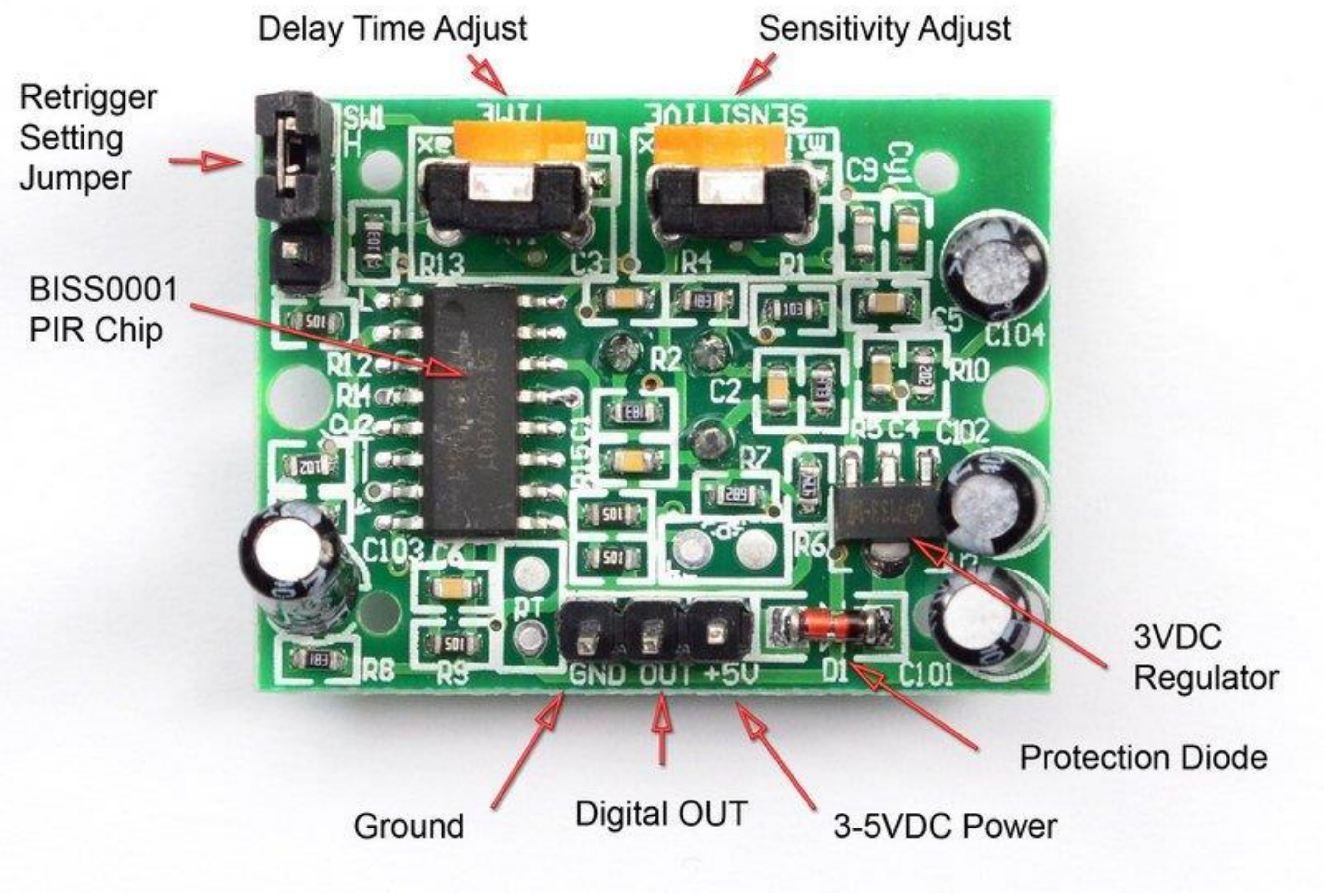

16MHz
crystal
Voltage
regulator
ATmega16U2
microcontroller IC/USB
controller
7 to 12VDC input
2.1mm * 5.5mm
Male center positive
USB-B port
to computer
Reset button
ICSP for
USB interface
(12C) SCI - Serial clock
(12C) SDA - serial data
Pin-13 LED
Not connected
I/Q Reference voltage
Reset
3.3V Output
5V Output
Ground
Input voltage
(SPI) SCK-Serial clock
(SPI) MISO- Master-in
, slave-out
(SPI) MOSI - Master-out,
slave-in
(SPI) SS - Slave select
Analog pin 0
Analog pin 1
Analog pin 2
Analog pin 3
(I2C) SDA
(12C) SCL
Interrupt 1
Interrupt 2
TXD
RXD
RESET
SCK
VCC
ATmega328
microcontroller IC
MOST
MISO
ICSP for
ATmega328
GND

# PIR SENSOR

A passive infrared sensor (PIR sensor) is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view. They are most often used in PIR-based motion detectors. PIR sensors are commonly used in security alarms and automatic lighting applications.

PIR sensors detect general movement, but do not give information on who or what moved. For that purpose, an imaging IR sensor is required.





The above images explain the pin configurations of the PIR Sensor. The PIR sensor circuit consists of three pins, power supply pin, output signal pin, and ground pin.

## SUMMARY

In the above section we discussed about the interfacing and pin configurations of Arduino [Uno] board and the PIR sensor. Thus this describes about the components selection and the different software used for implementing our system. It briefly described about the specifications of the components selected and the description about the software and hardware used.

Now let’s look at the given problem statement and the required connections for the detector to work accurately……

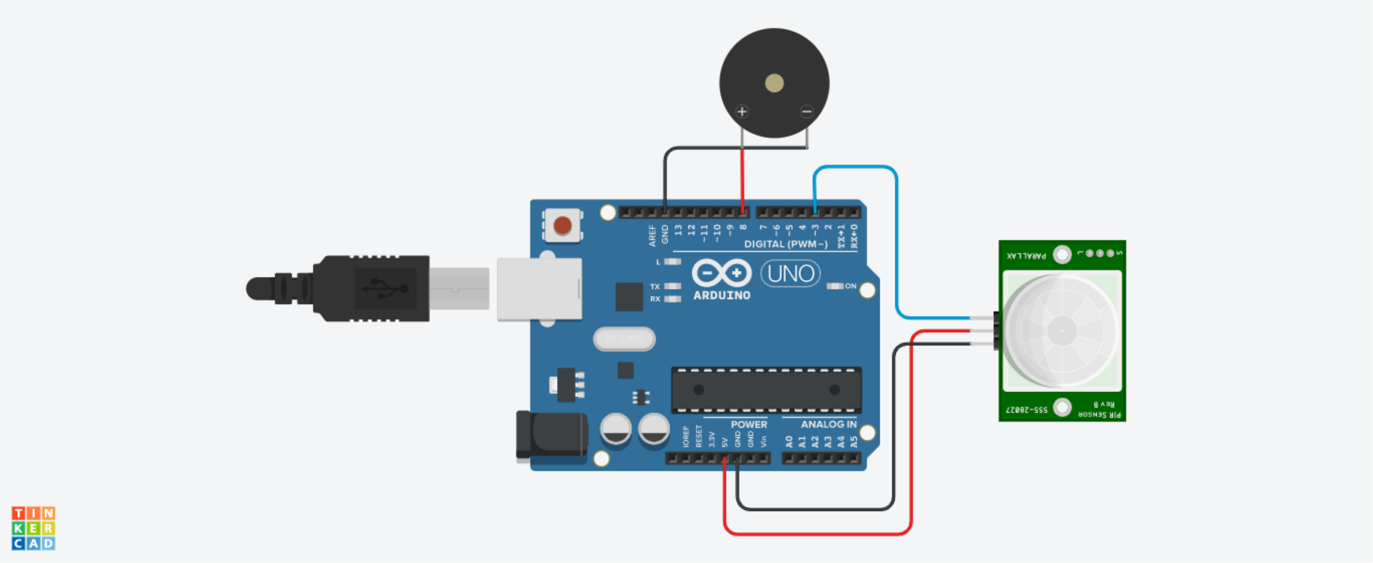
## PROBLEM STATEMENT

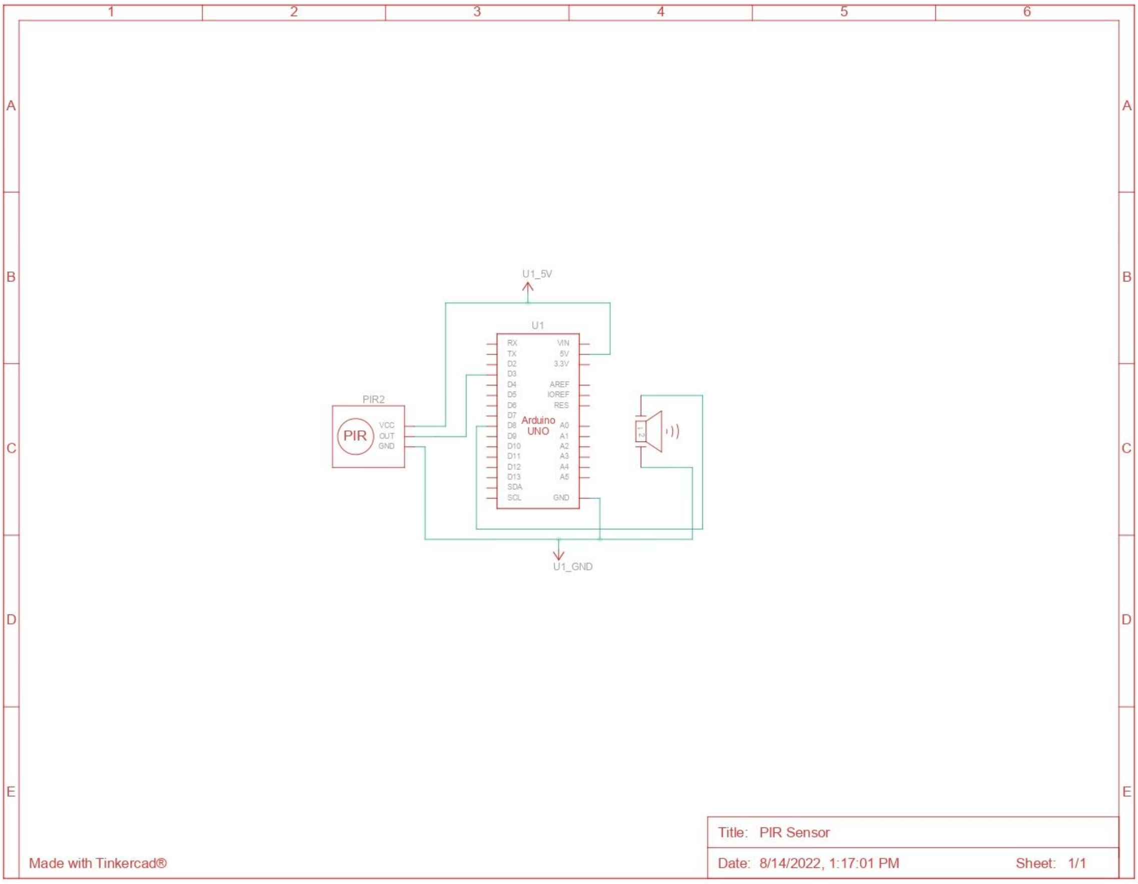
*Create a burglar alarm – detect motion of a person (using a PIR sensor). If there is motion, ring the buzzer (using Arduino UNO).*

## COMPONENTS USED/REQUIRED FOR THE PROJECT

| Name | Quantity | Component |
|---|---|---|
| UNO | 1 | Arduino Uno R3 |
| PIEZO1 | 1 | Piezo /Buzzer |
| PIR Sensor | 1 | PIR Sensor |

## CIRCUIT DIAGRAM OF CONNECTIONS AND DESCRIPTION





The above circuit diagram and connections are developed and simulated over Tinker cad. For designing an accurately working model of a Burglar alarm we will be using the PIR sensor which specifically detects variations in motion in the field view area. We will connect and configure this sensor with a buzzer so that when the motion or fluctuations in the region are detected then it triggers the sensor and sets the buzzer ON. Detecting the presence of a burglar quickly alarming us to take actions immediately.

PIR to Arduino connections are as follows: Connect the Vcc of PIR to 5V on Arduino, the Ground pin of PIR to the ground pin on Arduino and the OUTPUT pin of PIR to Digital pin D3 on Arduino.

One pin of buzzer (positive) is connected to the digital pin D8 on Arduino and the other pin of buzzer to Ground pin on Arduino.

These are the basic connections required for the proper functioning of the Alarm system.

Let's now look into the software details for the project.

## CODE FOR THE ALARM CIRCUIT OVER ARDUINO IDE

// Burglar alarm – detect motion of a person (using a PIR sensor)

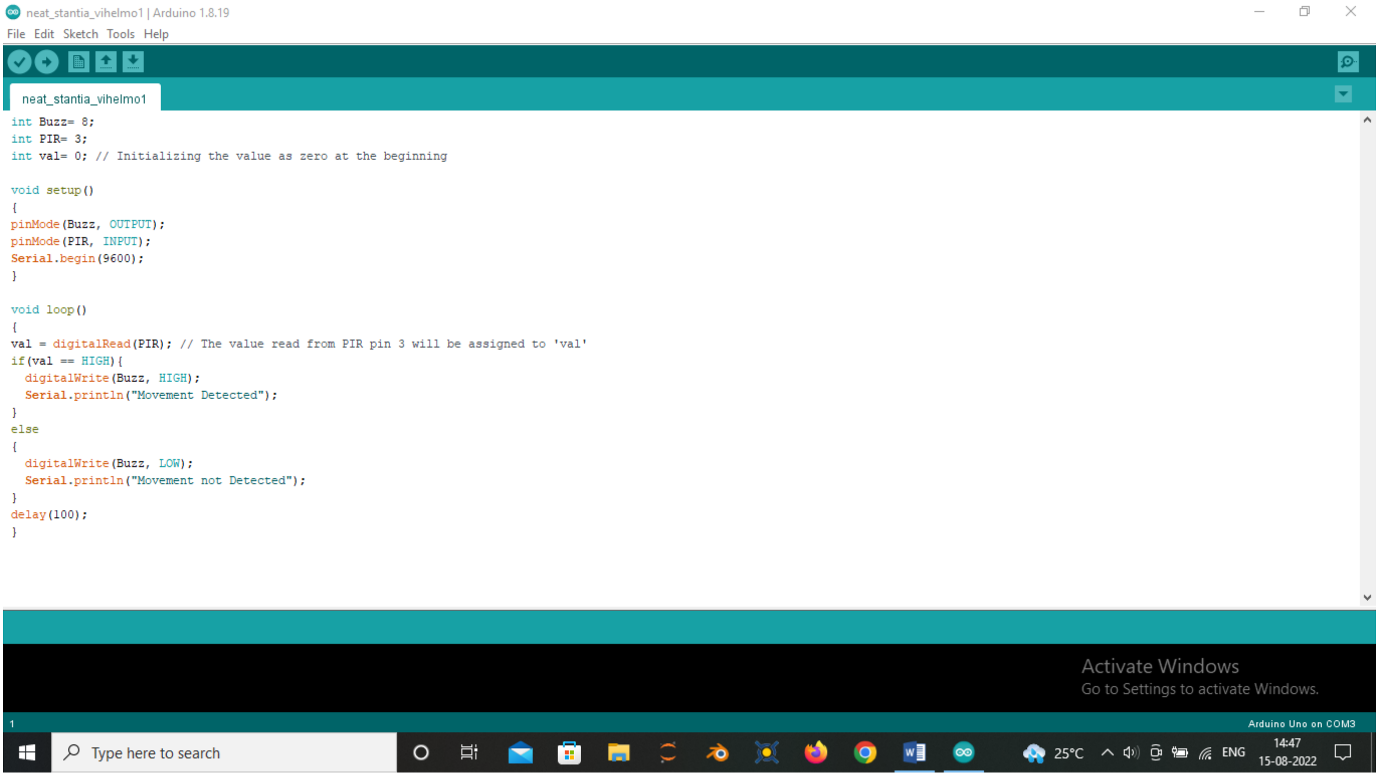
/* Name- Bhakti Raut

IOT Mini Project /*

```
int Buzz= 8;
int PIR= 3;
int val= 0; // Initializing the value as zero at the beginning

void setup()
{
pinMode(Buzz, OUTPUT);
pinMode(PIR, INPUT);
Serial.begin(9600);
}

void loop()
{
val = digitalRead(PIR); // The value read from PIR pin 3 will be assigned to 'val'
if(val == HIGH)
{
digitalWrite(Buzz, HIGH);
Serial.println("Movement Detected");
}
else
{
digitalWrite(Buzz, LOW);
Serial.println("Movement not Detected");
}
delay(100);
}
```

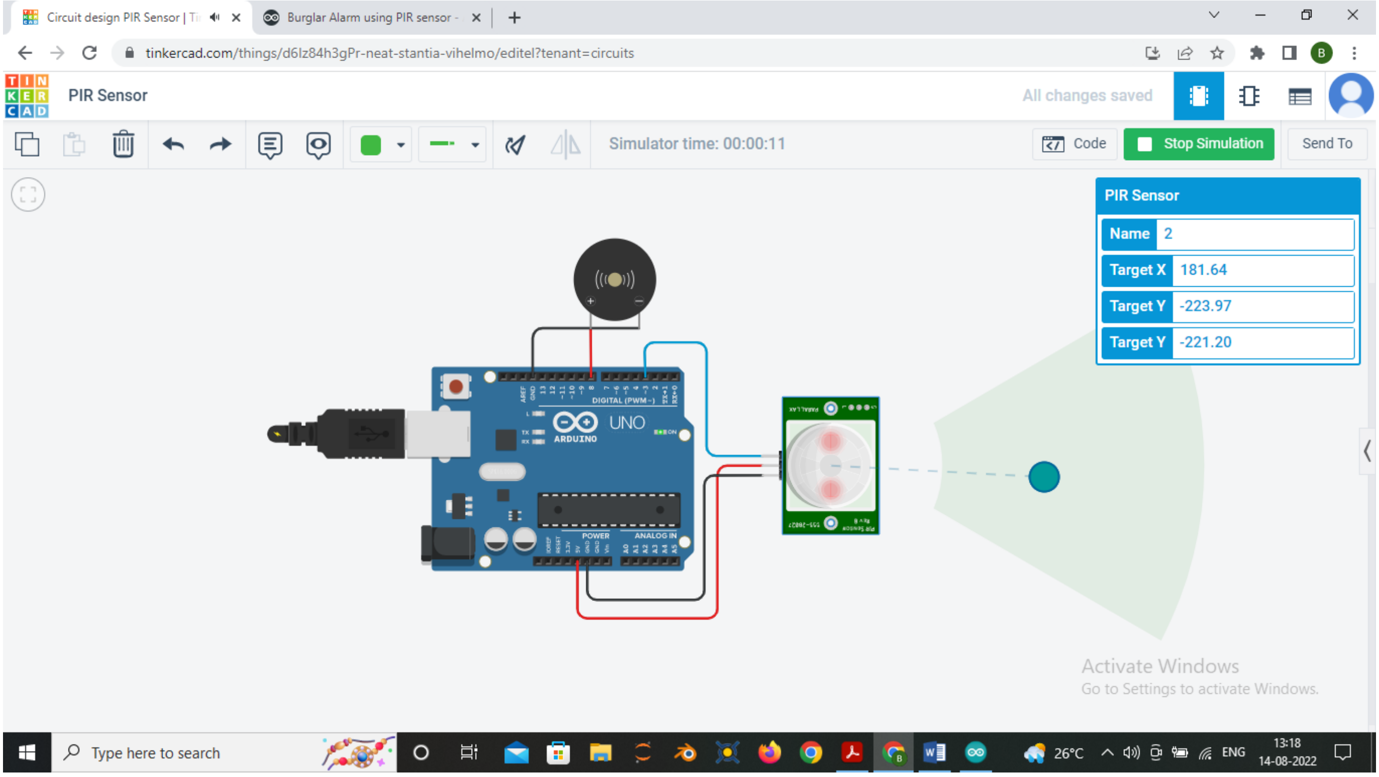

neat_stantia_vihelmo1 | Arduino 1.8.19
File Edit Sketch Tools Help
neat_stantia_vihelmo1
Activate Windows
Go to Settings to activate Windows.
1
Arduino Uno on COM3
Type here to search
25°C
ENG
14:47
15-08-2022

```
int Buzz= 8;
int PIR= 3;
int val= 0; // Initializing the value as zero at the beginning

void setup()
{
pinMode(Buzz, OUTPUT);
pinMode(PIR, INPUT);
Serial.begin(9600);
}

void loop()
{
val = digitalRead(PIR); // The value read from PIR pin 3 will be assigned to 'val'
if(val == HIGH){
  digitalWrite(Buzz, HIGH);
  Serial.println("Movement Detected");
}
else
{
  digitalWrite(Buzz, LOW);
  Serial.println("Movement not Detected");
}
delay(100);
}
```

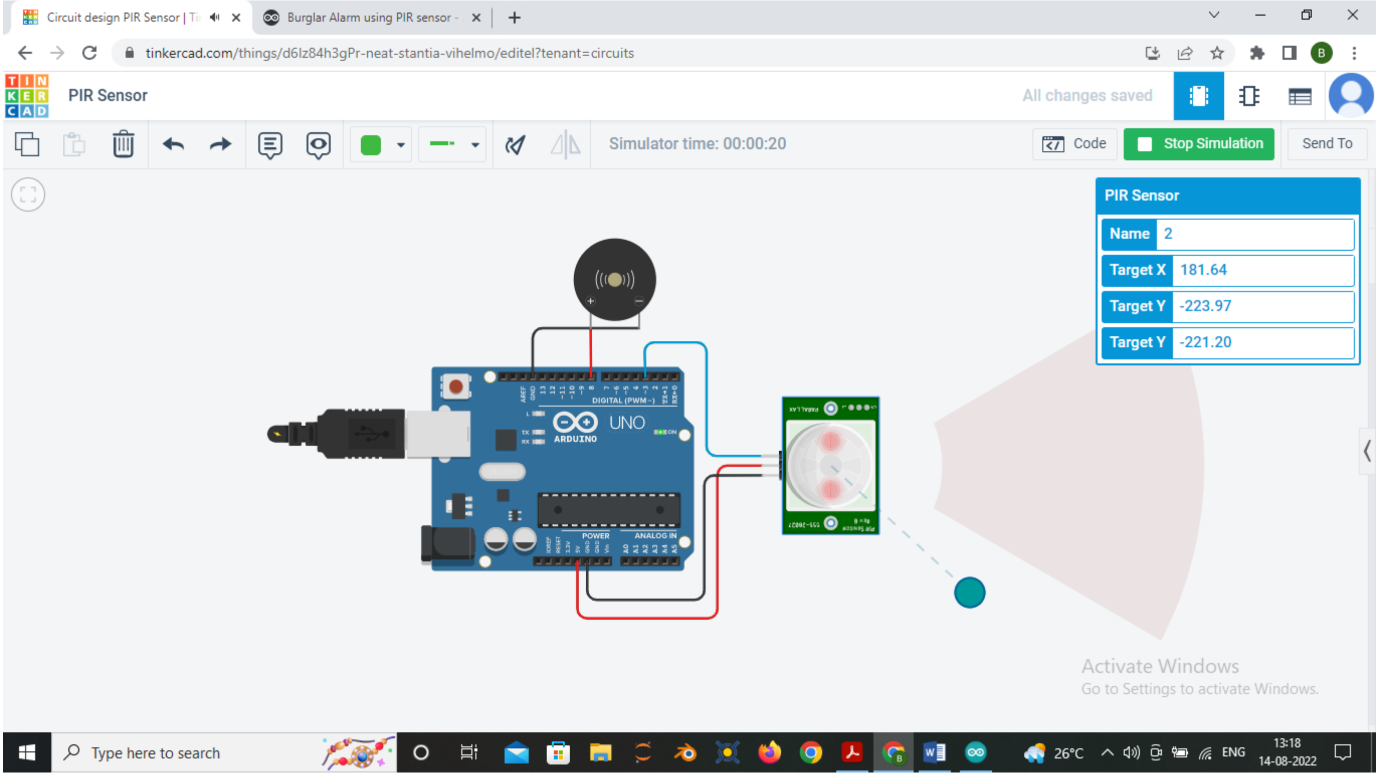
In the above code we have used the if/else condition to trigger the sensor. If the motion is detected within the range of the PIR sensor those fluctuations in IR will be detected and then a HIGH will be sent to the sensor and the buzzer will be triggered, alarming us of the unnecessary movement in restricted areas. A delay of 100 ms is introduced between the functioning.


Circuit design PIR Sensor | Ti
Burglar Alarm using PIR sensor -
tinkercad.com/things/d6lz84h3gPr-neat-stantia-vihelmo/editel?tenant=circuits
PIR Sensor
All changes saved
Code
Stop Simulation
Send To
UNO
ARDUINO
DIGITAL (PWM~)
POWER
ANALOG IN
Activate Windows
Go to Settings to activate Windows.
Type here to search
26°C
ENG
13:18
14-08-2022

Simulator time: 00:00:11

PIR Sensor

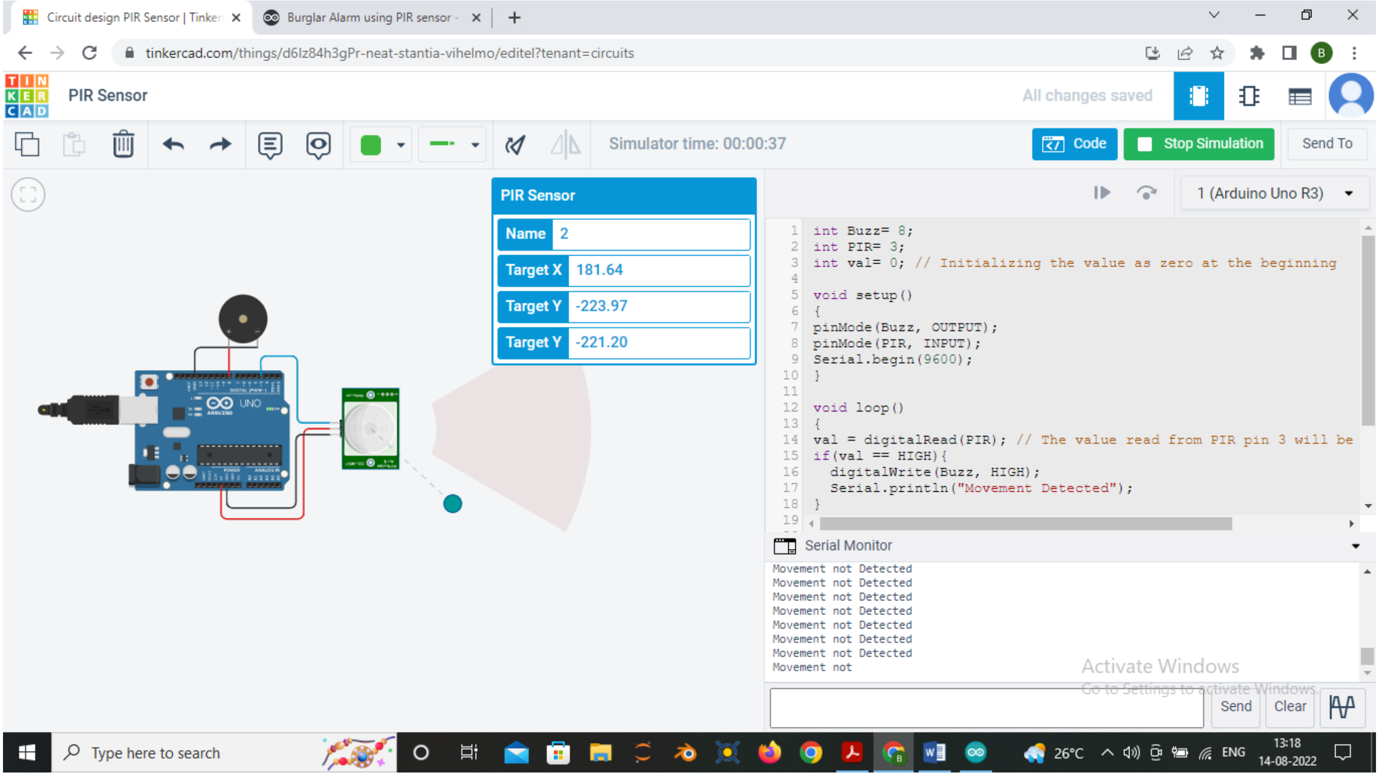
| | |
|---|---|
| Name | 2 |
| Target X | 181.64 |
| Target Y | -223.97 |
| Target Y | -221.20 |


Circuit design PIR Sensor | Ti
Burglar Alarm using PIR sensor -
tinkercad.com/things/d6lz84h3gPr-neat-stantia-vihelmo/editel?tenant=circuits
PIR Sensor
All changes saved
Code
Stop Simulation
Send To
UNO
ARDUINO
DIGITAL (PWM~)
POWER
ANALOG IN
Activate Windows
Go to Settings to activate Windows.
Type here to search
26°C
ENG
13:18
14-08-2022

Simulator time: 00:00:20

PIR Sensor

| | |
|---|---|
| Name | 2 |
| Target X | 181.64 |
| Target Y | -223.97 |
| Target Y | -221.20 |



Circuit design PIR Sensor | Tinker
Burglar Alarm using PIR sensor -
tinkercad.com/things/d6lz84h3gPr-neat-stantia-vihelmo/editel?tenant=circuits
PIR Sensor
All changes saved
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
Serial Monitor
Send
Clear
Type here to search
26°C
ENG
13:18
14-08-2022

Simulator time: 00:00:37

PIR Sensor

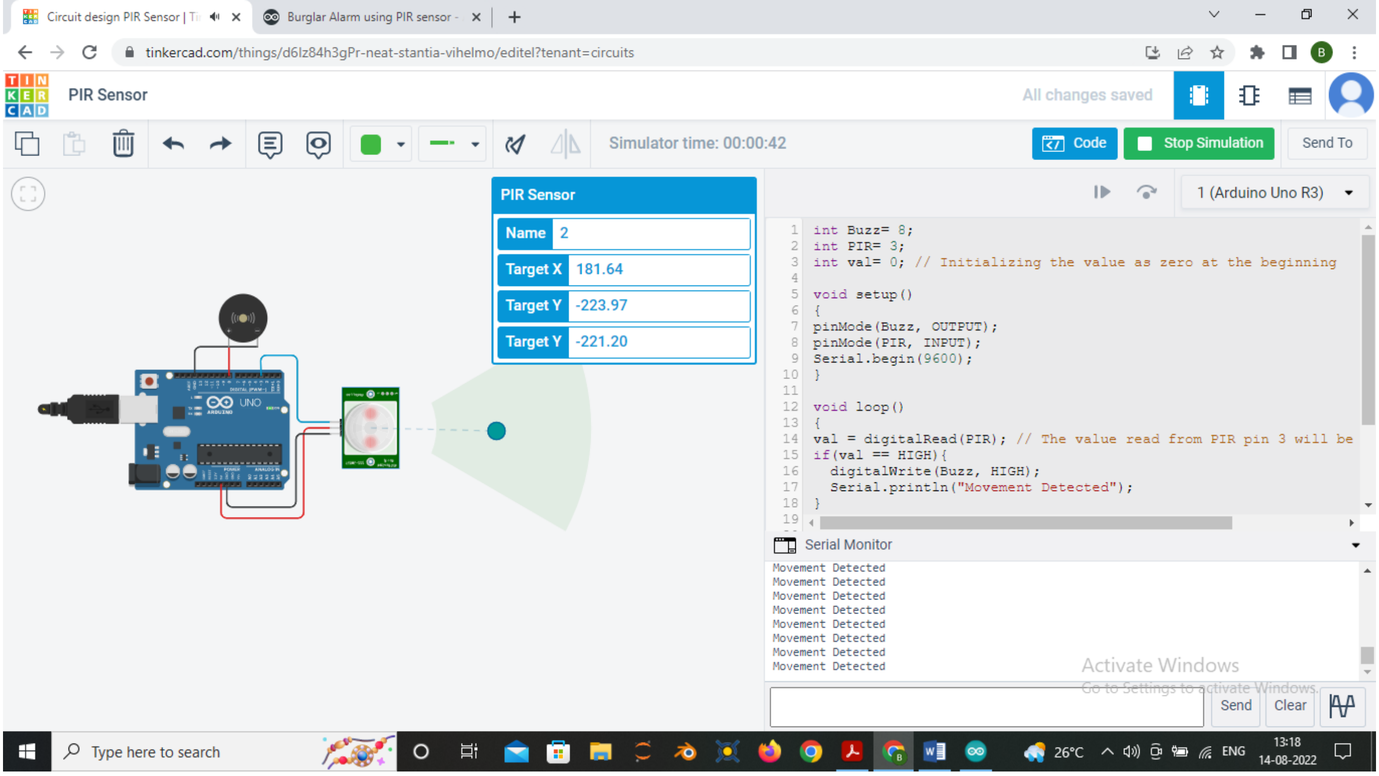
| | |
|---|---|
| Name | 2 |
| Target X | 181.64 |
| Target Y | -223.97 |
| Target Y | -221.20 |

```
int Buzz= 8;
int PIR= 3;
int val= 0; // Initializing the value as zero at the beginning

void setup()
{
pinMode(Buzz, OUTPUT);
pinMode(PIR, INPUT);
Serial.begin(9600);
}

void loop()
{
val = digitalRead(PIR); // The value read from PIR pin 3 will be
if(val == HIGH){
  digitalWrite(Buzz, HIGH);
  Serial.println("Movement Detected");
}
```

```
Movement not Detected
Movement not Detected
Movement not Detected
Movement not Detected
Movement not Detected
Movement not Detected
Movement not Detected
Movement not
```



Circuit design PIR Sensor | Ti
Burglar Alarm using PIR sensor -
tinkercad.com/things/d6lz84h3gPr-neat-stantia-vihelmo/editel?tenant=circuits
PIR Sensor
All changes saved
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
Serial Monitor
Send
Clear
Type here to search
26°C
ENG
13:18
14-08-2022

Simulator time: 00:00:42

PIR Sensor

| | |
|---|---|
| Name | 2 |
| Target X | 181.64 |
| Target Y | -223.97 |
| Target Y | -221.20 |

```
int Buzz= 8;
int PIR= 3;
int val= 0; // Initializing the value as zero at the beginning

void setup()
{
pinMode(Buzz, OUTPUT);
pinMode(PIR, INPUT);
Serial.begin(9600);
}

void loop()
{
val = digitalRead(PIR); // The value read from PIR pin 3 will be
if(val == HIGH){
  digitalWrite(Buzz, HIGH);
  Serial.println("Movement Detected");
}
```

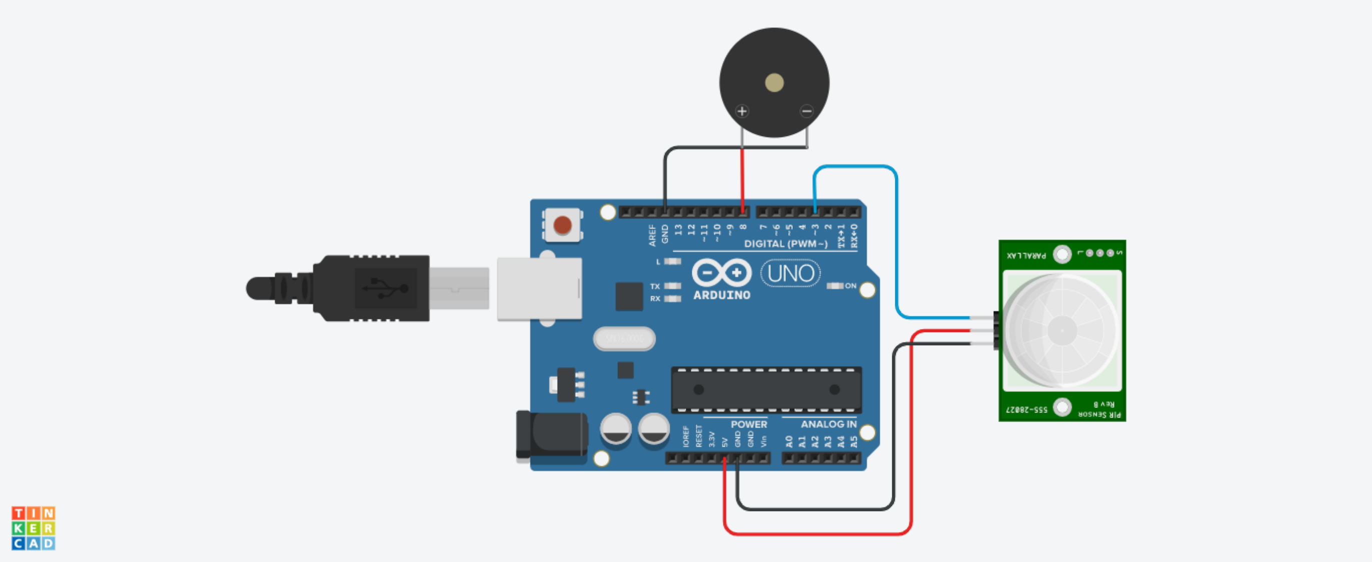
```
Movement Detected
Movement Detected
Movement Detected
Movement Detected
Movement Detected
Movement Detected
Movement Detected
Movement Detected
```

# CONCLUSION

We can observe from the simulations above that when the object is moved or some motion is generated then quickly the PIR sensor detects it and sets the buzzer on, alarming us of the motion detected. There is a delay of 100 ms in between the movement being detected and the next movement. This is the basic prototype of how large scale security devices work and can be used in our day to day households as a Burglar alarm system.

We learnt the basic hardware and pin configurations of Arduino UNO and how it can be configured with sensors, etc. We gained a sufficient knowledge of how to develop the code over Arduino IDE.

We have successfully designed a burglar alarm system using PIR sensor and Arduino UNO.



DIGITAL (PWM~)
UNO
ARDUINO
TX
RX
ON
POWER
ANALOG IN
PARALLAX
555-28027
PIR SENSOR Rev B
TIN
KER
CAD