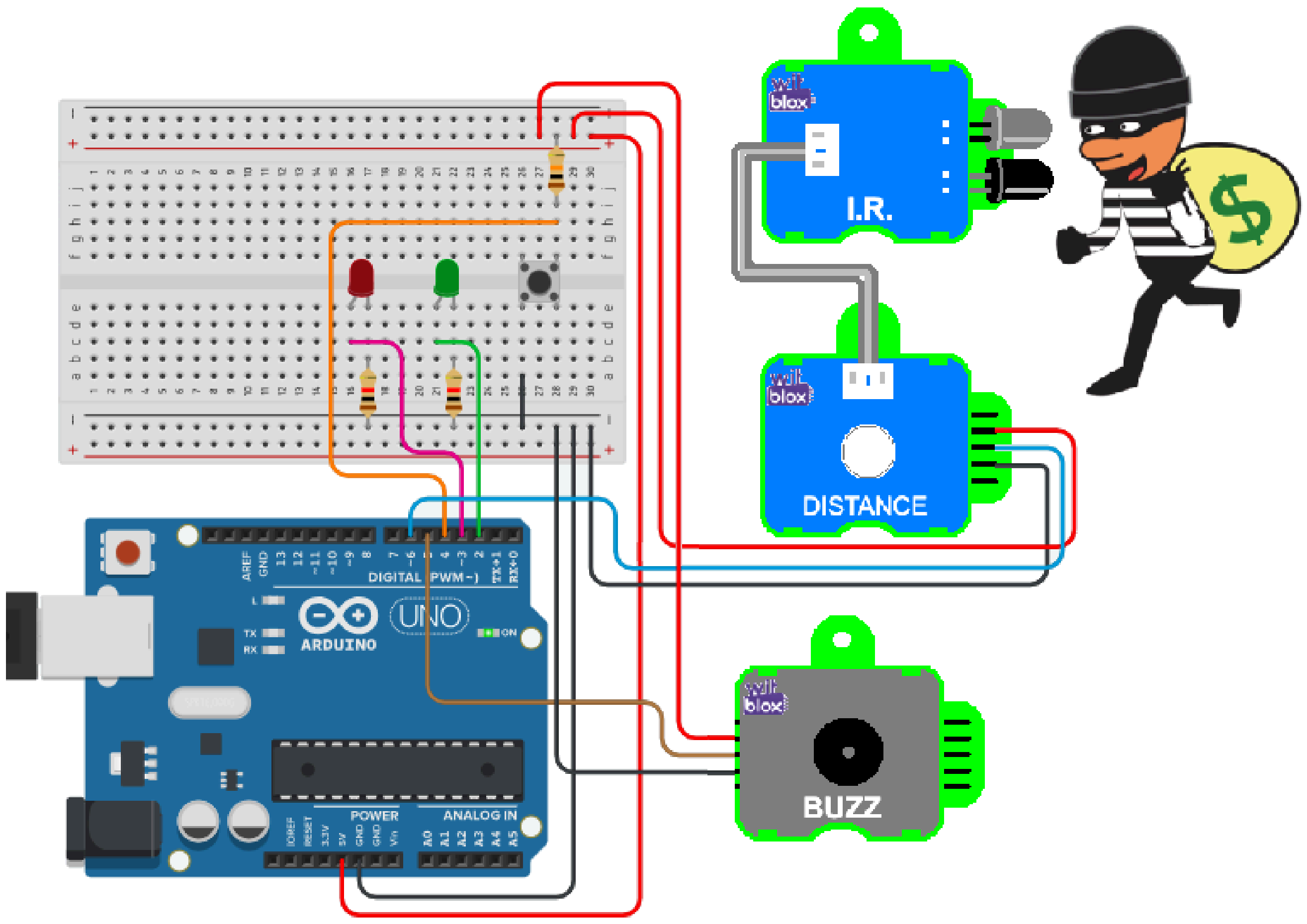
Activity: Make an Automatic Intrusion Alarm System

Objective:

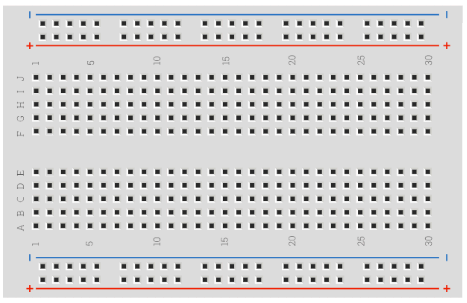
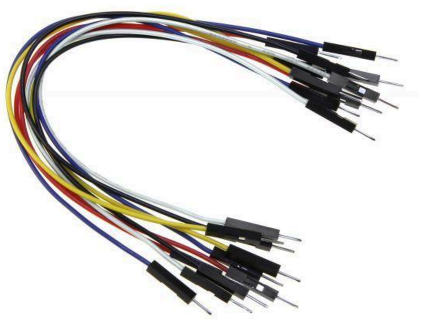


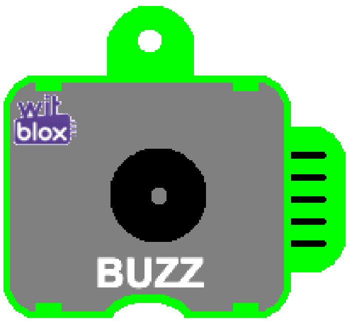
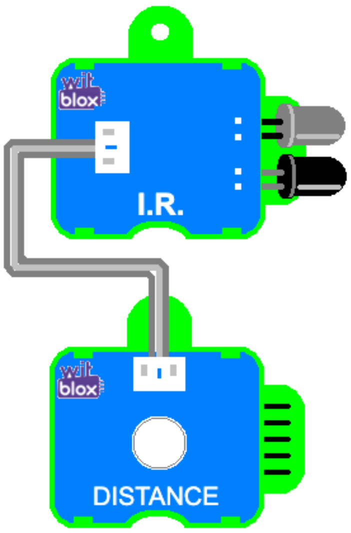
Creating an Automatic Intrusion Alarm system using an IR sensor, Arduino, and a buzzer can be a valuable project to enhance home security and provide basic protection against thefts and burglaries involving break-ins. This project aims to provide an easy, hands-on and educational way to understand what are the different parts for a basic home security system and how an Arduino can be used to implement such a system with minimal coding and hardware.

This system makes use of an IR sensor to detect the presence of any unwanted person who has broken into the house, an Arduino to set a defined logic for the system's operation and finally a Buzzer to notify the user or people about the attempt of intrusion going on. Two status LEDs, one Red and one Green, are also used to visually indicate whether the system is armed for use or not.

So, in today's activity we will learn how to make an automatic intrusion alarm system with arming and disarming settings using an Arduino, a Buzz blox and the IR Distance sensor blox.

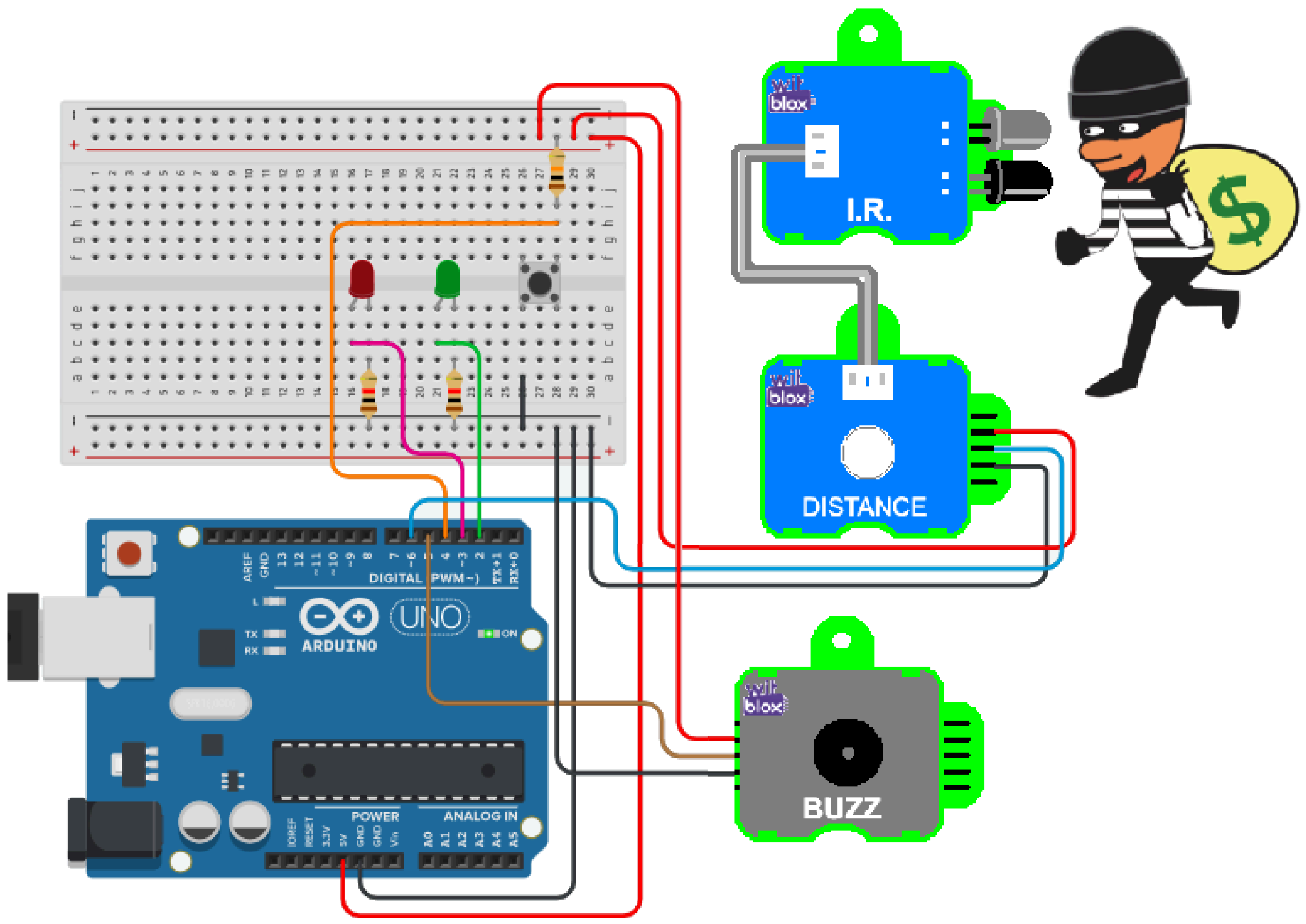
Materials Required:

S.no.	Part	Qty	Image
1	Arduino (Nano / UNO)	1	
2	Breadboard	1	
3	USB cable for Arduino	1	
4	Connection Wires (M - M)	10	

5	Buzz Blox	1	
6	5mm Red LED	1	
7	5mm Green LED	1	
8	1k resistor (Brown Black Red Golden)	2	
9	Distance Sensor Blox	1	

Connection Diagram:

The connection diagram shown in the image below is based on the Arduino UNO.



Explanation:

The connection diagrams shown in both the images show the connections between the Arduino, the Input and the Output. The Distance sensor blox and the Push Button are the Inputs for the Arduino while the Buzzer and the Red and the Green LEDs are the Outputs for the Arduino.

When we carefully observe the connection diagrams, we find that:

1. The Green LED is connected such that its +ve terminal is connected to Arduino digital pin 2 (D2) and its -ve terminal is connected to one side of the 1k resistor. The other side of the 1k resistor is connected to the -ve of the breadboard. The Arduino - Green LED connection is shown by the **green line** in the diagram above.

2. The Red LED is connected such that its +ve terminal is connected to Arduino digital pin 3 (D3) and its -ve terminal is connected to one side of the 1k resistor. The other side of the 1k resistor is connected to the -ve of the breadboard. The Arduino - Red LED connection is shown by the **pink line** in the diagram above.

3. The Push Button is connected such that its top-right pin is connected to the +ve of the breadboard through a 10k pull-up resistor and its bottom-left pin is connected to the -ve of the breadboard. Additionally, its top-right pin is also connected to Arduino digital pin 4 (D4). The Arduino - Button connection is shown by the **orange line** in the diagram above.

4. The Buzz blox is connected such that its +ve power pin (2nd pin from the top) is connected to +ve of the breadboard while its -ve terminal is connected to the -ve of the breadboard. The Data input pin of the Buzz blox is connected to Arduino digital pin 5 (D5). The Arduino - Buzzer connection is shown by the **brown line** in the diagram above.

5. The Distance sensor blox is connected such that its +ve power pin (2nd pin from the top) is connected to the 5v pin of the Arduino and its -ve power pin (4th pin from the top) is connected to the Gnd pin of the Arduino. The Data output pin of the Distance sensor blox (3rd pin from the top) is connected to Arduino digital pin 6 (D6). The Arduino - Distance sensor blox connection is shown by the **blue line** in the diagram above.

6. Finally, the Gnd pin of the Arduino is connected to the -ve of the breadboard. The Arduino is connected to the computer through the USB cable which serves for providing power, program upload and bi-directional serial data communication with the computer.

Arduino Code:

Here is the complete Arduino code for making an Automatic Intrusion Alarm System using Distance sensor blox, Buzzer, Button, LEDs and Arduino.

```
#define Green  2  // Green  for pin 2
#define Red    3  // Red    for pin 3
#define Button 4  // Button for pin 4
#define Buzzer 5  // Buzzer for pin 5
#define Sensor 6  // Sensor for pin 6

int sensor_state; // for sensor store
int button_state; // for button store
int count;        // for count store


void setup()
{
  pinMode(Green, OUTPUT);   // Green as  Output
  pinMode(Red, OUTPUT);     // Red as    Output
  pinMode(Button, INPUT);   // Button as Input
  pinMode(Buzzer, OUTPUT);  // Buzzer as Output
  pinMode(Sensor, INPUT);   // Sensor as Input

  digitalWrite(Red,LOW);    // Turn Off Red LED
  digitalWrite(Green,HIGH); // Turn On Green LED
}


void loop()
{
  button_state = digitalRead(Button); // check Button

  if(button_state == 0)               // if pressed
    {
      count = count + 1;              // increment count
      delay(500);                     // wait for 500ms
    }

  if(count == 1)                      // pressed once
```

```
    {
      digitalWrite(Red,HIGH);          // Red LED On
      digitalWrite(Green,LOW);         // Green LED Off

      sensor_state = digitalRead(Sensor); // check Sensor

      if(sensor_state == 1)            // if sensor active
        {
          digitalWrite(Buzzer,HIGH);   // turn Buzzer On
        }
    }

  if(count == 2)                       // pressed twice
    {
      count = 0 ;                      // reset count
      digitalWrite(Buzzer,LOW);        // turn Buzzer On
      digitalWrite(Red,LOW);           // Red LED Off
      digitalWrite(Green,HIGH);        // Green LED On
    }
}
```

Explanation:

```
#define Green  2  // Green  for pin 2
#define Red    3  // Red    for pin 3
#define Button 4  // Button for pin 4
#define Buzzer 5  // Buzzer for pin 5
#define Sensor 6  // Sensor for pin 6

int sensor_state; // for sensor store
int button_state; // for button store
int count;        // for count store
```

Here, we are declaring that:

1. Arduino digital pin 2 (D2) will be referred to as “Green” as it is used for the Green LED.
2. Arduino digital pin 3 (D3) will be referred to as “Red” as it is used for the Red LED.

3. Arduino digital pin 4 (D4) will be referred to as “Button” as it is used for the Button.
4. Arduino digital pin 5 (D5) will be referred to as “Buzzer” as it is used for the Buzz blox.
5. Arduino digital pin 5 (D6) will be referred to as “Sensor” as it is used for the Distance sensor blox.
6. We are taking three integer type variables and naming them “sensor_state”, “button_state” and “count” as they are used for storing the state of the sensor, the button and the counter.

```
void setup()
{
  pinMode(Green, OUTPUT);    // Green as  Output
  pinMode(Red, OUTPUT);      // Red as    Output
  pinMode(Button, INPUT);    // Button as Input
  pinMode(Buzzer, OUTPUT);   // Buzzer as Output
  pinMode(Sensor, INPUT);    // Sensor as Input

  digitalWrite(Red,LOW);     // Turn Off Red LED
  digitalWrite(Green,HIGH); // Turn On Green LED
}
```

Here, inside the setup function, we are declaring that the “Buzzer” pin, the “Red” pin and the “Green” pin will be used as an Output pin and the “Sensor” pin and the “Button” pin will be used as Input pins. This needs to be done just once. Next, we turn the Green LED On and the Red LED Off.

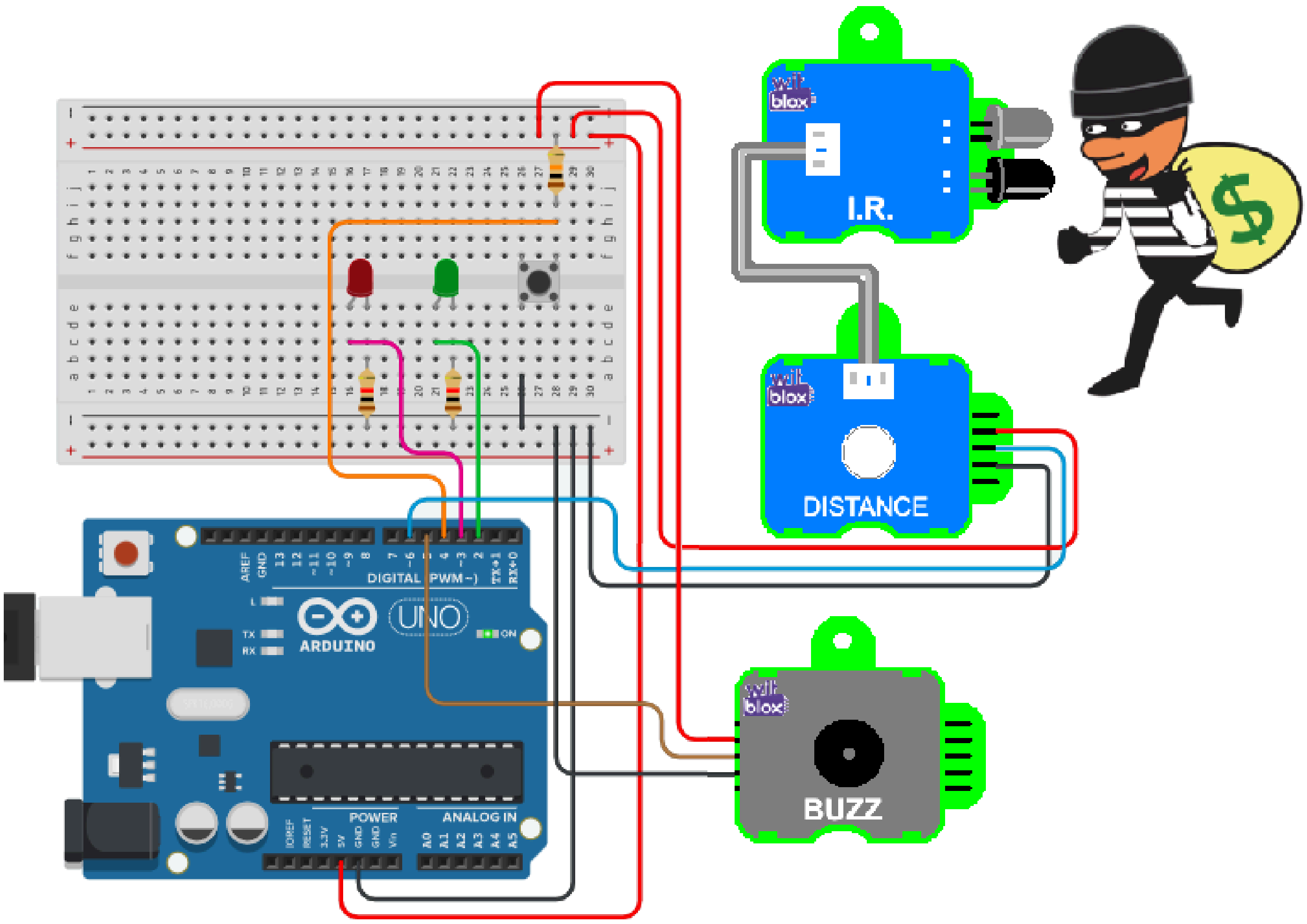
```
void loop()
{
  button_state = digitalRead(Button); // check Button

  if(button_state == 0)               // if pressed
    {
      count = count + 1;              // increment count
      delay(500);                     // wait for 500ms
    }

  if(count == 1)                      // pressed once
    {
      digitalWrite(Red,HIGH);         // Red LED On
      digitalWrite(Green,LOW);         // Green LED Off

      sensor_state = digitalRead(Sensor); // check Sensor

      if(sensor_state == 1)            // if sensor active
        {
          digitalWrite(Buzzer,HIGH);   // turn Buzzer On
        }
    }

  if(count == 2)                       // pressed twice
    {
      count = 0 ;                      // reset count
      digitalWrite(Buzzer,LOW);        // turn Buzzer On
      digitalWrite(Red,LOW);           // Red LED Off
      digitalWrite(Green,HIGH);        // Green LED On
    }
}
```

Here, in the loop function, first the state of the Push button is checked. If the Button is pressed, the counter is incremented by 1. Triggering the IR sensor does not trigger the alarm as long as the button is not pressed once. Pressing the Button once Arms the alarm system and pressing it again disarms the system.

When the system is in the Armed state, triggering the IR sensor triggers the Buzzer which stays On indefinitely as long as the Button is not pressed again or the Arduino is reset or the power is turned Off.

When the Buzzer is On, pressing the button turns the Buzzer off and puts the system back in the disarmed mode. As for the Red and Green LEDs, the Red LED stays On and the Green LED stays Off when the system is in the armed state and the Green LED stays On and the Red LED stays Off when the system is in the disarmed state.

Outcome and Observations:



1. Once the Arduino code is compiled and uploaded, the Buzzer and the Red remain Off while the Green LED remains On.

2. The system is now in the disarmed mode. In this mode, triggering the IR sensor does not trigger the Buzzer.

3. Now, when we press and release the Button once, the Green LED turns Off and the Red LED turns On indicating that the system is now armed. Now, if we trigger the IR sensor, the buzzer turns On and stays On.

4. The Buzzer can only be turned off if:
   a. The Button is pressed again to disarm the system.
   b. Reset button is pressed on the Arduino.
   c. Power is disconnected from the system.