CS225/CS226 PROJECT REPORT

Topic: Multifunctional Bi-Directional Person Counter

Name : YASH SHARMA

Roll No. : 1901CS73

Motivation

In many places, overcrowding could lead to stampede, pickpocketing and other unfortunate things. To monitor the no. of people who are inside a building we could use an Arduino device and it can ring an alarm if no. of people exceed a limit. Especially in these Covid-19 times, there is a need to avoid overcrowding to prevent the spread of the virus.

Introduction

A Bi-Directional person counter is an Arduino device that is useful to monitor the number of persons inside any building. The device maintains the count of people who are inside the building and depending on its value various indicators can be used such as if the count is greater than 0 a led could be turned on and if the count exceeds a limit an alarm can be rung.

Components Used

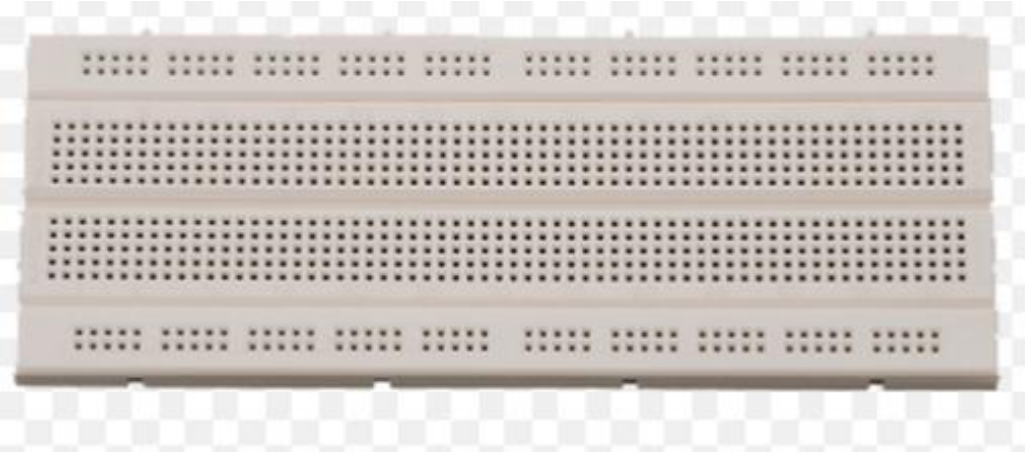
Arduino Uno R3

Arduino Uno is an open-source microcontroller board based on the Microchip ATMega328P microcontroller and developed by Arduino.cc. It is equipped with various digital and analog input/output pins, capable of interfacing with other circuits and is programmable with Arduino IDE.

HC-SR04 Ultrasonic Distance Sensor

It uses SONAR to determine the distance of an object in front of it. It can measure the distance between 2cm to 400 cm with high accuracy. The HC-SR04 Ultrasonic Module has 4 pins, Ground, VCC, Trig, Echo. The Ground and VCC pins of the module needs to be connected to the Ground and 5V pins on the Arduino Board respectively and the trig and echo pins to any digital I/O pin on Arduino Board.

Breadboard

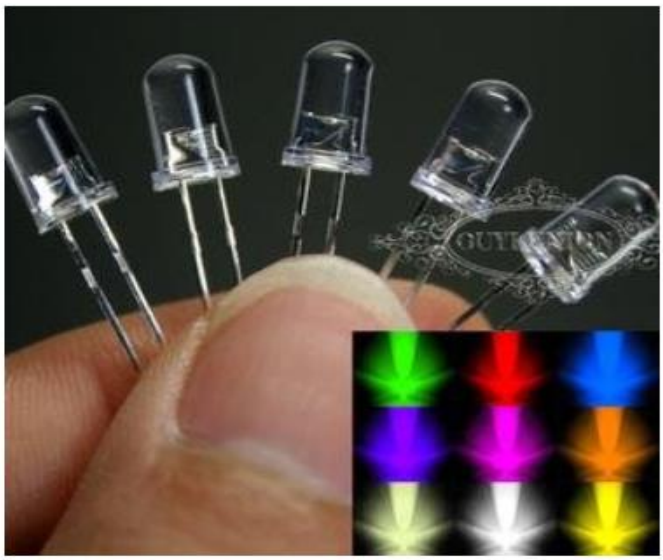


A breadboard is a solderless rectangular plastic board with a bunch of tiny holes in it. These holes let us easily insert electronic components to prototype (meaning to build and test an early version of) an electronic circuit. Most electronic components in electronic circuits can be interconnected by inserting their leads or terminals into the holes and then making connections through wires where appropriate.

Cables

Jumper wires typically come in three versions: male-to-male, male-to-female and female-to-female. The difference between each is in the endpoint of the wire. Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into.

Red LED



This LED glows depending upon the value of the count variable.

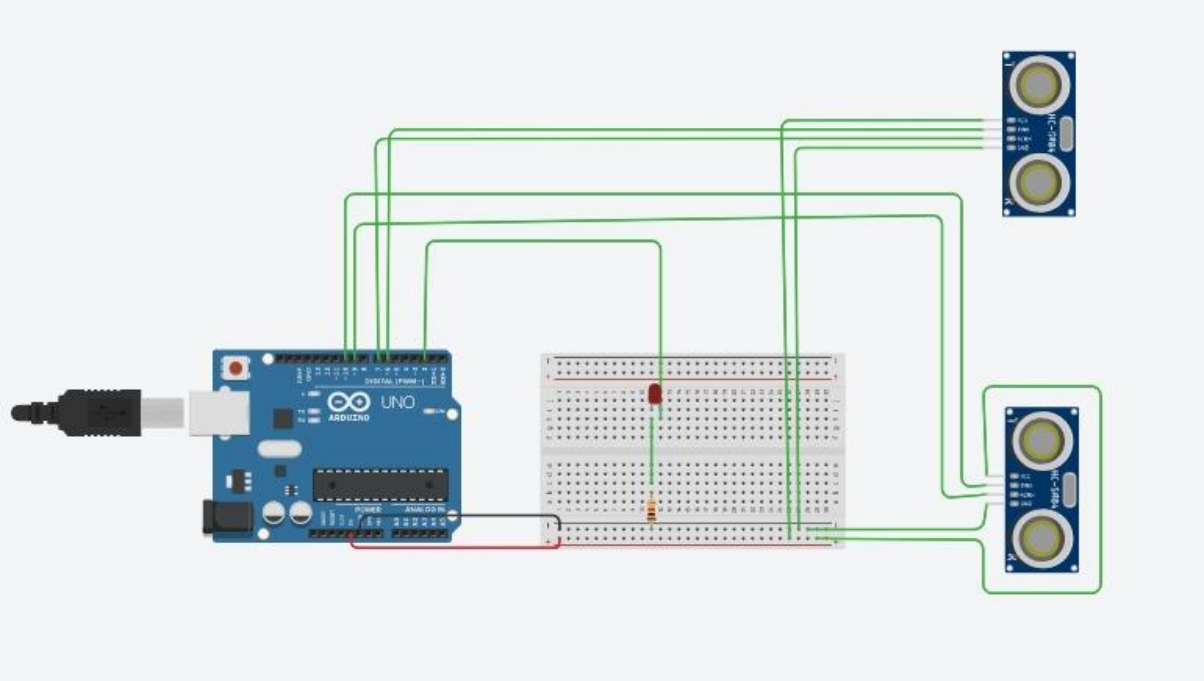
10k ohm Resistor

The resistor provides a voltage drop and thus helps to ensure that the LED doesn’t fuse due to excessive voltage.

App/IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Circuit Diagram



Working

A variable ‘count’ initialized to 0 is used to store the current number of persons present inside a building. When a person enters the building, the front ultrasonic sensor senses the object first followed by the rear sensor and the count is increased by 1 whereas when a person exits the building, the rear ultrasonic sensor senses the object first followed by the front sensor and the count is decreased by 1. An appropriate greeting message and the value of count is displayed on the Serial Monitor. When a person enters the building “Welcome” greeting message is shown and when a person leaves the building “Thank you for visiting” is shown. Also if the value of count is greater than 0 a LED glows.

Code Analysis

Defining global variables

```
/******************** Defining Global Variables ********************/
// count stores the number of people inside a building
int count = 0;

int last_count = 0;

// distance_rear and distance_front store the distance of person from front
// and rear sensor respectively
float distance, distance_rear, distance_front;

float time_taken;

// declare variables for the Arduino pins to which the sensor is connected
int echo1 = 9, trigger1 = 10; // for front ultrasonic sensor
int echo2 = 7, trigger2 = 6;  // for rear ultrasonic sensor
int led = 2;
```

The global variables and all the Arduino pins that are declared in this code snippet.

Setting up the pins of Arduino

```
/************************* Initial setup *************************/
void setup()
{
  Serial.begin(9600);

  // initializing the pin to which echo terminal is connected as input
  // and trig terminal as output
  pinMode(echo1, INPUT);
  pinMode(trigger1, OUTPUT);
  pinMode(echo2, INPUT);
  pinMode(trigger2, OUTPUT);

  // initializing the pin for led as output
  pinMode(led,OUTPUT);
}
```

All the echo, trigger and led pins are initialized as either input or output accordingly.

Function to calculate distance

```
/************************ Helper function ************************/
//function to calculate the distance of an object placed
// in front of the ultrasonic sensor
void calculate_distance(int trigger, int echo)
{
  digitalWrite(trigger, LOW);
  delayMicroseconds(10);        // delay for 10 microseconds
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);        // delay for 10 microseconds
  digitalWrite(trigger, LOW);

  // get the time taken between the instant when waves are emitted and received
  time_taken = pulseIn(echo, HIGH);

  // using the time taken and speed of sound waves i.e. 340 m/s,
  // distance is calculated
  distance = time_taken * 0.017;

  // if distance > 50, we consider it as 50 only
  if(distance > 50)
    distance = 50;
}
```

The “calculate_distance” function is used to calculate the distance of an object placed in front of the ultrasonic sensor. Using the time taken between the instant when waves are emitted and received and the formula distance = (time * speed)/2, distance is calculated. If no object is detected or the object is far away, the distance is assigned as 50 cm.

Main loop

```
/*************************** Main Loop ***************************/
void loop()
{
  // if count > 0, led is turned on otherwise off
  if(count > 0) {
    digitalWrite(led,HIGH);
  }
  else {
    digitalWrite(led, LOW);
  }

  // get distance of object from front sensor
  calculate_distance(trigger1,echo1);
  distance_front = distance;

  if(distance < 50) {
    for(int i=0; i<5;i++) {
      calculate_distance(trigger2, echo2);
      distance_rear = distance;
      if(distance < 50) {
        count++;
        break;
      }
      delay(200);         // delay for 200 milliseconds
    }
  }
```

If the value of count is greater than zero, LED is turned on otherwise it remains off. The distance of the object from the front sensor is calculated and if it is less than 50 cm, then we calculate the distance from the rear sensor. If the object is detected in front of the rear sensor then the count is increased by 1.

```
calculate_distance(trigger2,echo2);
distance_rear = distance; //get distance of rear sensor

if(distance_rear < 50) {
  for(int i=0; i<5;i++) {
    calculate_distance(trigger1, echo1);
    distance_front = distance;
    if(distance < 50) {
      count--;
      break;
    }
    delay(200);         // delay for 200 milliseconds
  }
}

// code to get the distance of an object from both
// front and rear sensor
// Serial.print("Distance from front sensor = ");
// Serial.println(distance_front);
// Serial.print("Distance from rear sensor = ");
// Serial.println(distance_rear);

// since count can't be negative, we set it to 0 if
// somehow it becomes negative
if(count < 0)
  count = 0;

delay(200);         // delay for 200 milliseconds
```

Distance of object from the rear sensor is now calculated and if it is less than 50 cm, then we calculate the distance from the front sensor. If the object is detected in front of the front sensor then the count is decreased by 1. The code for printing the distance of an object from the sensor is commented out and could be used for debugging purposes. If the count somehow becomes negative, it is set to 0.

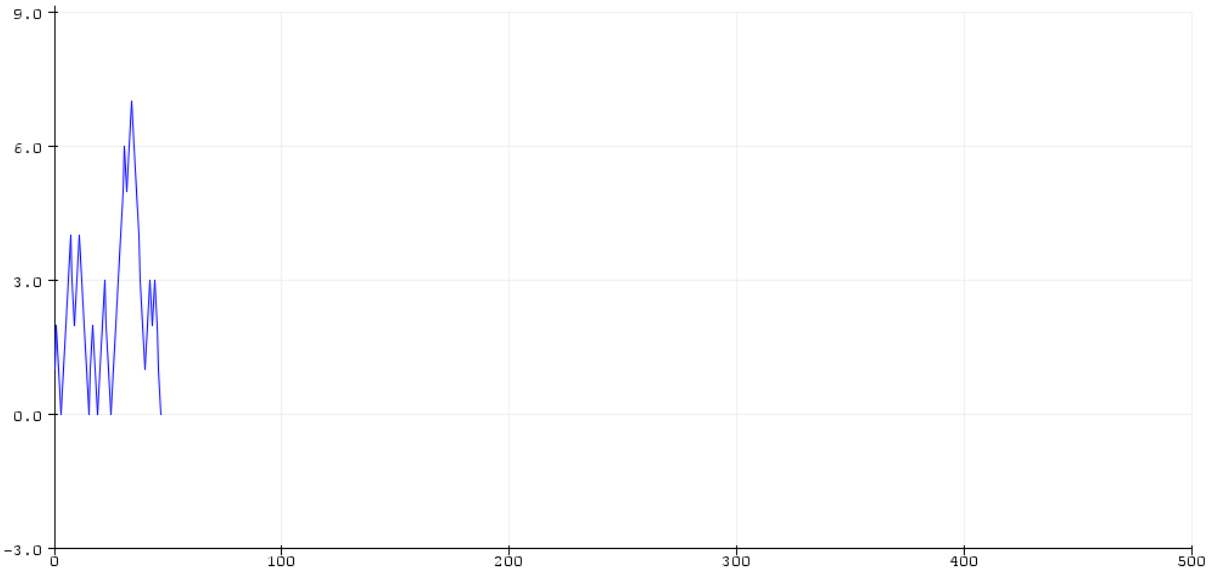
```
  // last_count stores the value of count in previous iteration
  // if count > last_count, it means a person has entered the room
  if(count > last_count)
  {
     Serial.println("Welcome");
     Serial.print("People in room = ");
     Serial.println(count);
  }
  // if count < last_count, it means a person has left the room
  else if(count<last_count)
  {
    //Serial.println("Thank you for visiting");
    //Serial.print("People in room = ");
    Serial.println(count);
  }

  //last_count is set to count
  last_count = count;

}
```

Depending upon if the count is increased or decreased, a greeting message and the value of count is printed.

Serial Plotter



Applications

This Arduino based project can have various applications namely:

1. It can be used in malls and other retail shops to continuously monitor the current number of visitors and find out the number of visitors as a function of time (using Serial Plotter) to know when is the maximum customer count and manage service accordingly.

2. It can be used in a hospital to regulate the number of people waiting outside the doctor's cabin. Using LED, a patient can know if he/she can go inside the cabin or not.

3. Overcrowding of the people can be controlled at busy places, party or rally.

4. An electrical appliance can be switched on or off automatically according to the count of the people. When there is no one inside, all appliances may be automatically switched off to save electricity.

Conclusion

This project is fairly simple to build but has a lot of useful applications. However, while working on the project I found that the ultrasonic sensors weren't very accurate and were sometimes missing a passing person. Thus it shouldn't be used in situations when an exact number is required but it still gives a fairly good estimate.