A REPORT OF MINI-PROJECT

ON

“ARDUINO BASED COUNTER”

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

in

Electronics and Communication Engineering

DEPARTMENT OF RADIO PHYSICS AND ELECTRONICS

UNIVERSITY OF CALCUTTA

92 A.P.C. Road, Kolkata, West Bengal 700009

STUDENT DETAILS

NAME	*UNIV. ROLL NO.*	*UNIV. REG. NO.*
Pintu Chakraborty	**T91/ECE/186003**	**053-1121-1389-15**
Rahul Dey	**T91/ECE/186019**	**133-1111-0007-19**
Shambo Roy	**T91/ECE/184061**	**D01-1111-0091-18**
Nabanita Saha	**T91/ECE/184047**	**D01-1212-0077-18**
Ruchishyo Ghosh	**T91/ECE/184049**	**D01-1111-0080-18**

ACKNOWLEDGEMENT

We would like to express our sincere thanks to our professor **Dr. Anjan Kumar Kundu** for his complete guidance and unrelenting support while working on this mini-project. While allowing us to learn doing the project ourselves, he always nudged us in the right direction and did not let us lose our focus.

We would like to thank our fellow group members, who all worked as a team to make this mini-project a success. I would also like to thank each and everyone who helped us in this mini-project and kept us motivated.

CONTENTS

INTRODUCTION

This is an Arduino based embedded system, where we aim to count the number of people present in a certain room at any instance.

In thisCOVID era, social distancing is a very important protocol to be followed amongst others. To ensure such physical distancing, it is necessary to set a strict limit to the number of people present in a closed room. Our project aims to serve this purpose.

Even though the project has been curated for the COVID situation, it still has its utilities in the post-COVID era. This project shall eventually help in preventing excessive gathering at any closed place at any time.

Banks, post offices, shopping malls are some of the very examples of places where we can effectively apply our project.

LITERATURE REVIEW

Arduino Uno

The Arduino Uno is an open source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/P pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.[2] It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

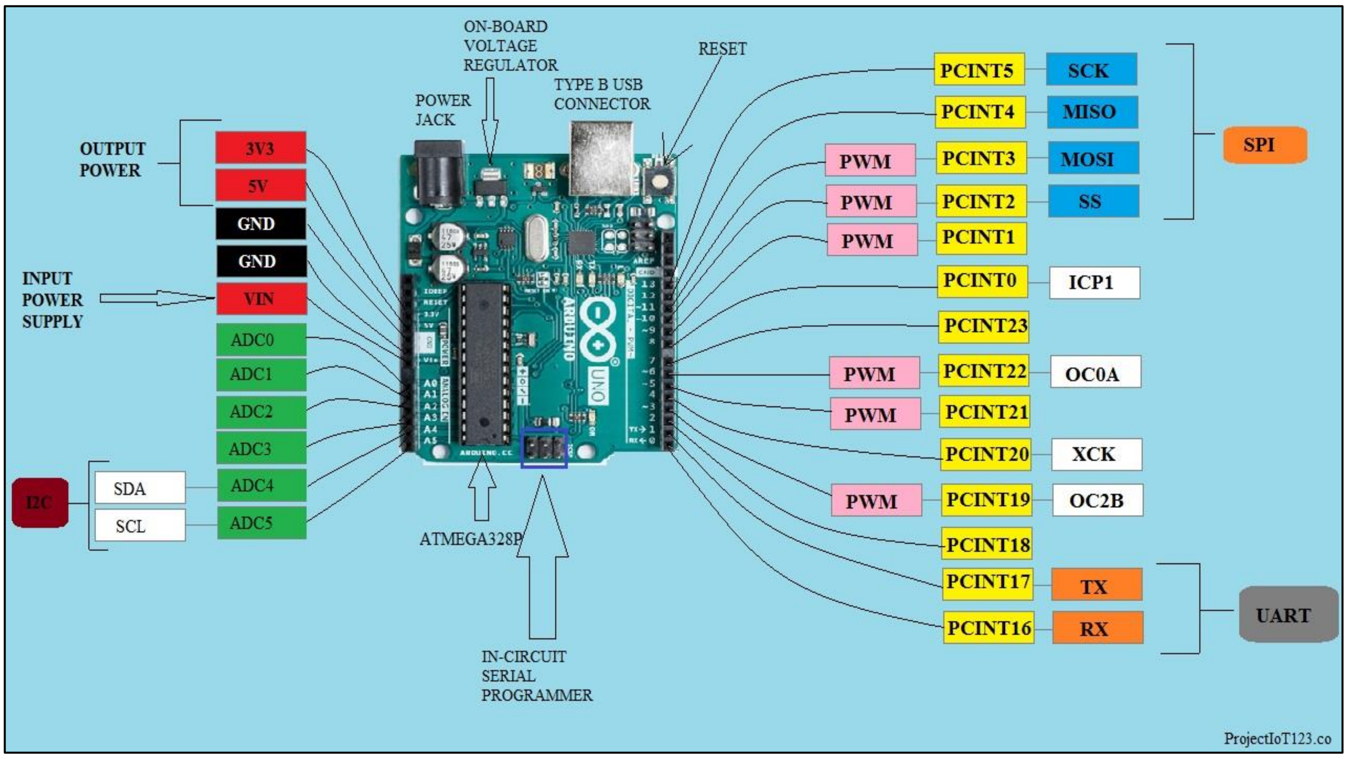
The Uno board is the first in a series of USB-based Arduino boards; it and version 1.0 of the Arduino IDE were the reference versions of Arduino, which have now evolved to newer releases. The ATmega328 on the board comes pre-programmed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.

While the Uno communicates using the original STK500 protocol, it differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Some important details are listed below :

- Microcontroller: Microchip ATmega328P
- Operating Voltage: 5 Volts
- Input Voltage: 7 to 20 Volts
- Digital I/O Pins: 14 (of which 6 can provide PWM output)
- UART: 1
- I2C: 1
- SPPI: 1
- Analog Input Pins: 6
- DC Current per I/O Pin: 20 mA
- DC Current for 3.3V Pin: 50 mA
- Flash Memory: 32 KB of which 0.5 KB is used by bootloader
- SRAM: 2 KB

- EEPROM: 1 KB
- Clock Speed: 16 MHz
- Length: 68.6 mm
- Width: 53.4 mm
- Weight: 25 g

General Pin diagram and their functions :



- **LED** : There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off.
- **VIN** : The input voltage to the Arduino/Genuino board when it is using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V** : This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
- **3V3** : A 3.3volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND** : Ground pins.

- **IOREF** : This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source, or enable voltage translators on the outputs to work with the 5V or 3.3V.
- **Reset** : Typically used to add a reset button to shields that block the one on the board.

Special Pin Functions :

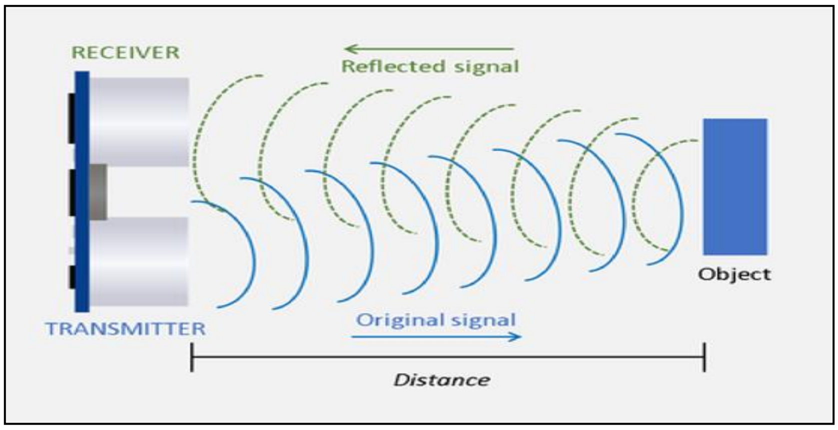
Each of the 14 digital pins and 6 analog pins on the Uno can be used as an input or output, under software control (using pinMode(), digitalWrite(), and digitalRead() functions). They operate at 5 volts. Each pin can provide or receive 20 mA as the recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50K ohm. A maximum of 40mA must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labelled A0 through A5; each provides 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though it is possible to change the upper end of the range using the AREF pin and the analogReference() function.

Additionally, some pins have specialized functions:

- **Serial / UART** : Pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip.
- **External interrupts**: Pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- **PWM** (pulse-width modulation) : Pins 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analogWrite() function.
- **SPI** (Serial Peripheral Interface) : Pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). These pins support SPI communication using the SPI library.
- **TWI** (two-wire interface) / I²C : Pin SDA (A4) and pin SCL (A5). Support TWI communication using the Wire library.
- **AREF** (analog reference): Reference voltage for the analog inputs.

Ultrasonic Distance Sensor

Among various types of ultrasonic sensors, we have used the 3-pin PING sensor (SKU 28015) where a single pin is used for transmission and detection. An ultrasonic sensor works by transmitting an ultrasonic (well above human hearing range) burst and providing an outputpulse that corresponds to the time required for the burst echo to return to thesensor.The Arduino board sends a short pulse to trigger the detection, then “listens” for a pulse on the same pin using the pulseIn() function. The duration of the sensor’s output pulse denotes the time taken by the ultrasound to travel to the object and back to the sensor. By measuring the echo pulse width, the distance to target can be calculated.

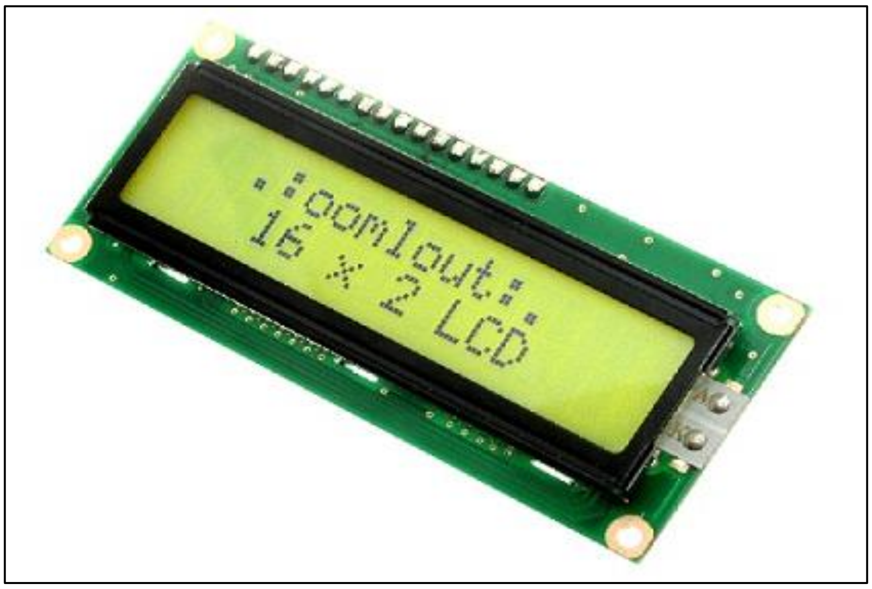


No. of Pins	: 3
Range	: 2 cm to 3 m (0.8 in to 3.3 yd)
Input trigger	: Positive TTL pulse, 2 µs min, 5 µs typ.
Echo pulse	: Positive TTL pulse, 115 µsminimum to 18.5 ms maximum.

Pin Definitions:

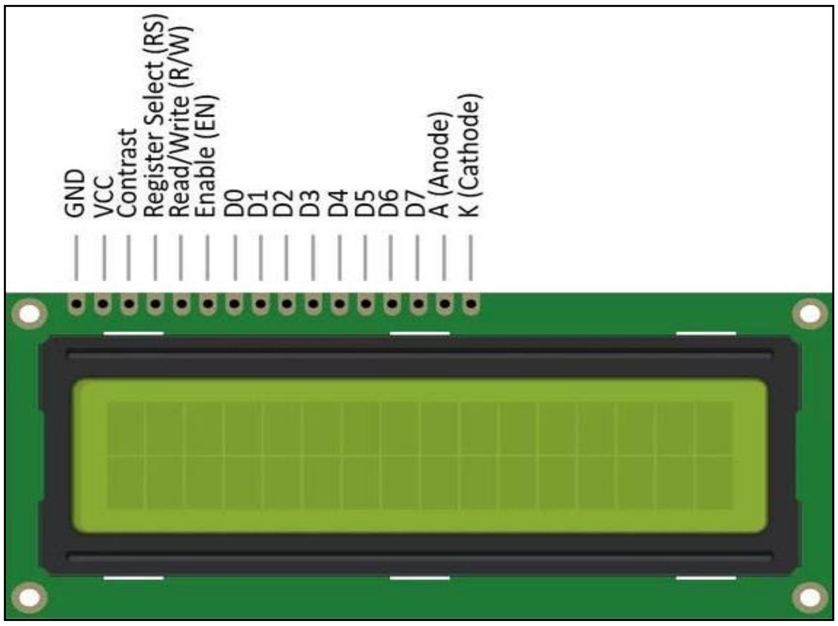
GND	: Ground pin
5V	: 5 Volt DC power input pin
SIG	: Signal pin for I/O operations

LCD Display Module



LCD (Liquid Crystal Display)is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets etc. Here we use LCD 16×2 display module whichhas an LED backlight and can display 32 ASCII characters in two rows with 16 characters on each row. Its works on the principle of blocking the backlight where the characters are being displayed, and hence that particular area will become dark compared to other.The benefits of using this module are that it is inexpensive, is simply programmable and there are no limitations for displaying custom characters and even animations.

LCD 16×2 General Pin diagram and their functions:



- **GND** : Ground pin
- **VCC** : 5 Volt DC power input pin

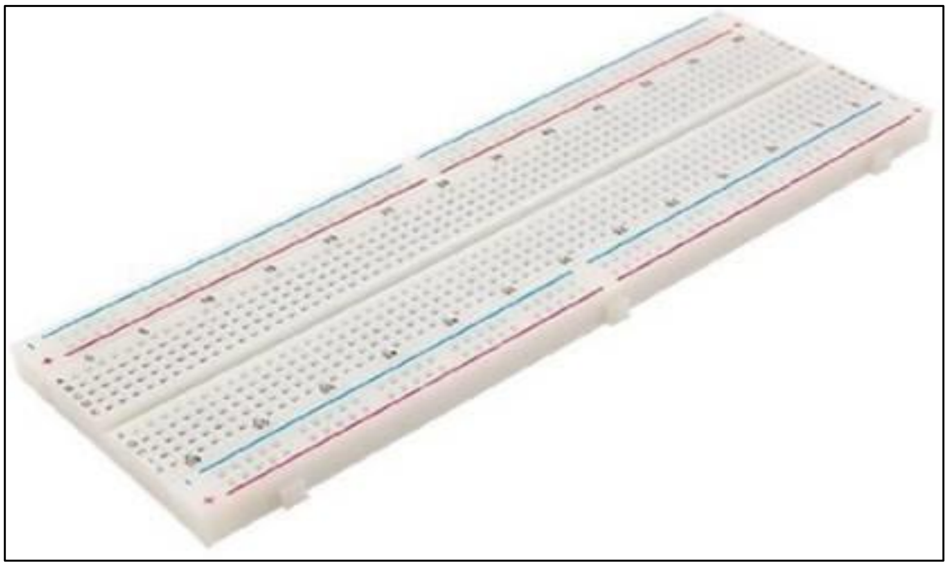
- **Vo (Contrast)** : This pin controls the contrast and brightness of the LCD.
- **RS (Register Select)** : This pin lets the Arduino tell the LCD whether it is sending commands or the data.
- **R/W (Read/Write)** : This pin on the LCD is to control whether data is being read from or being written to the LCD. Since we are using this LCD as an OUTPUT device, this pin is made LOW. This forces it into the WRITE mode.
- **EN (Enable)** : This pin is used to enable the display. When this pin is set to HIGH, the LCD is processing the incoming data.
- **D0-D7 (Data bus)** : These are the pins that carry the 8 bit data we send to the display.
- **A-K (Anode & Cathode)** : These pins are used to supply power and ground respectively, to the backlight of the LCD.

Features of LCD16×2 :

- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- It can work on two modes viz. 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight
- It can display custom generated characters

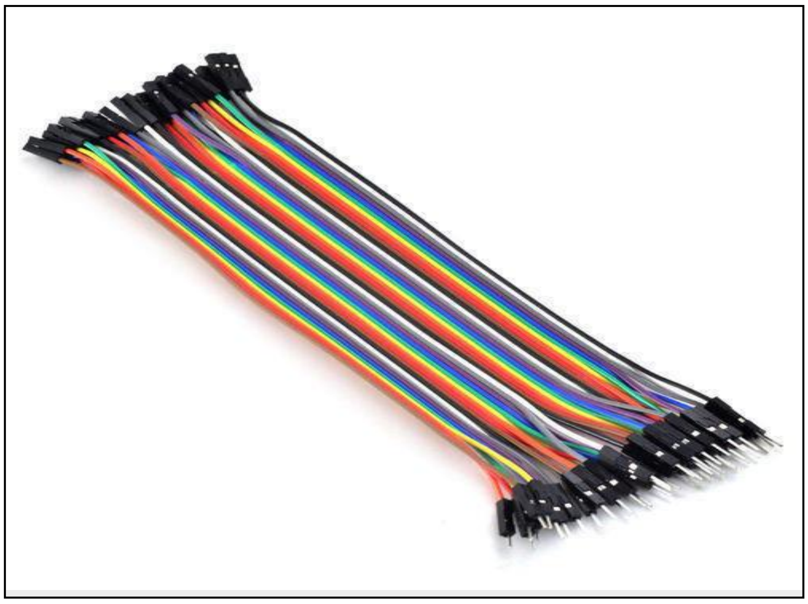
Breadboard

A breadboard is a solderless device for temporary prototype with electronics and test circuit designs. 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.



Jumper wires

Jumper wires are simply, wires with connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.



Battery

LIPO batteryor**lithium polymer battery**, or more correctly**lithium-ion polymerbattery** (abbreviated as**LiPo**,**LIP**,**Li-poly**,**lithium-poly** and others) is arechargeablebatteryoflithium-ion technology using apolymerelectrolyte instead of a liquid electrolyte. High conductivity semisolid (gel) polymers form this electrolyte. This battery has been used here because of its ability to recharge.

PROJECT METHODOLOGY

In this project we have used an Arduino board, two ultrasonic sensors, an LCD display, a breadboard and jumper wires. To effectively count the number of people in a room, the whole setup is attached to the entry/exit door of the room. The two ultrasonic sensors are placed on the top frame of the door, one on the outside and the other on the inside. For the rest of this chapter, these two sensors shall be denoted as outside sensor and inside sensor respectively.

It works on the method of identifying the direction of motion through the door.When a person enters into the room through door, the outside sensor returns a distance less the maximum distance (75% of the height of the door) and hence detects the presence. This output is digitized to 1, i.e. change detected. Had there been no change, the output shall have been 0.Now at this point, the inside sensor cannot detect any presence and its output is digitized to 0. Now, after just entering the room, the inside sensor gives output 1 and outside sensor gives output 0 (As the person moved from outside to inside). Thus, for entrance of a person, data sets we get as output (in format [outside_sensor,inside_sensor]) from the sensors are[1,0] and then [0,1]. From these sets of data, we determine that a complete procedure of entrance is performed with the help of a variable ***state***. Here state is set as follows:

STATE	*CONDITION*
0	No movement
31	Entry
13	Exit

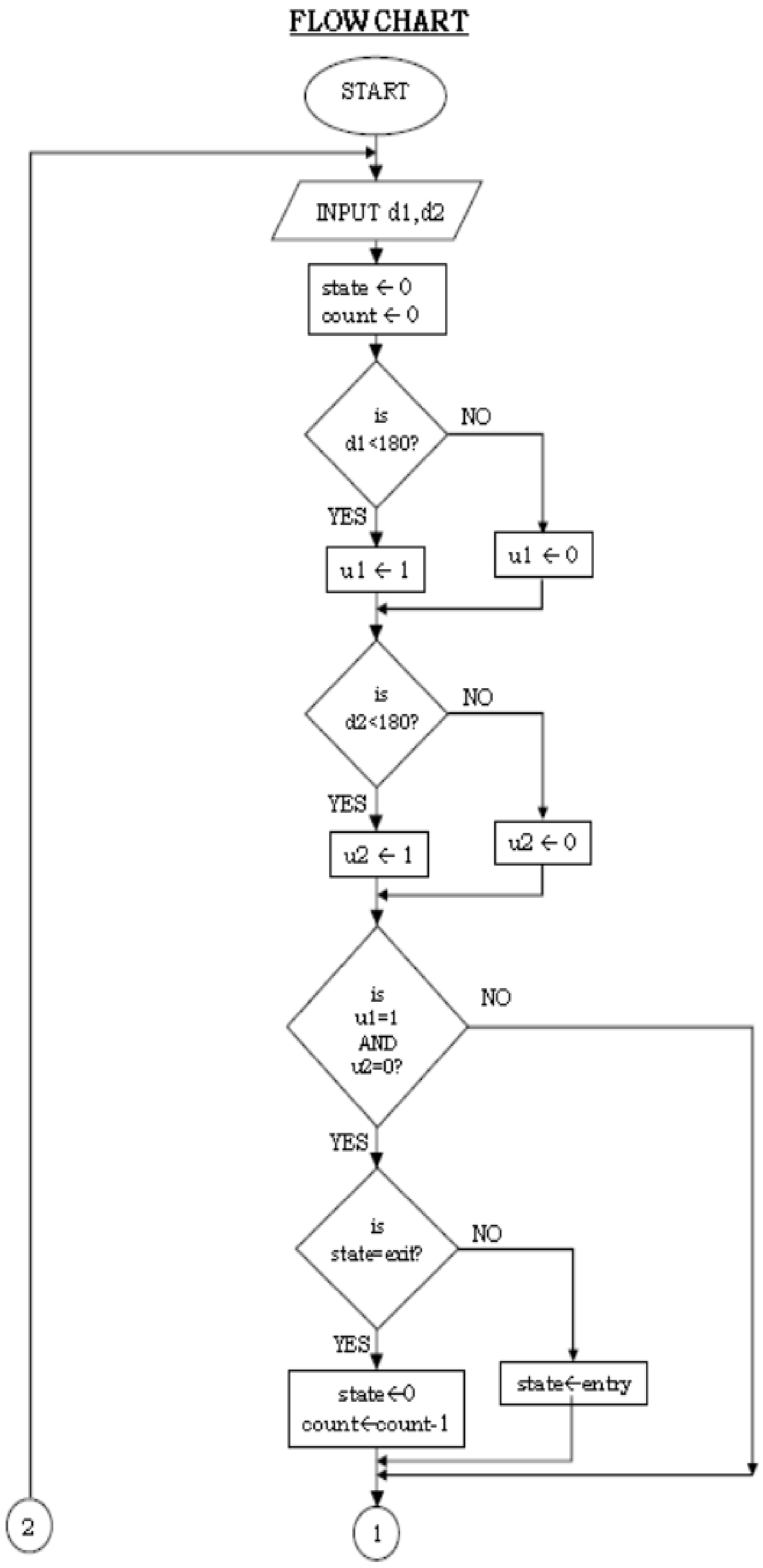
The state changes from 0 to 31 indicating an entrance and as soon as the person has completed his entrance, the state is changed back to 0. At that same time, the count of people in the room in incremented by one and the number of persons present inside the room is displayed through the LCD module.

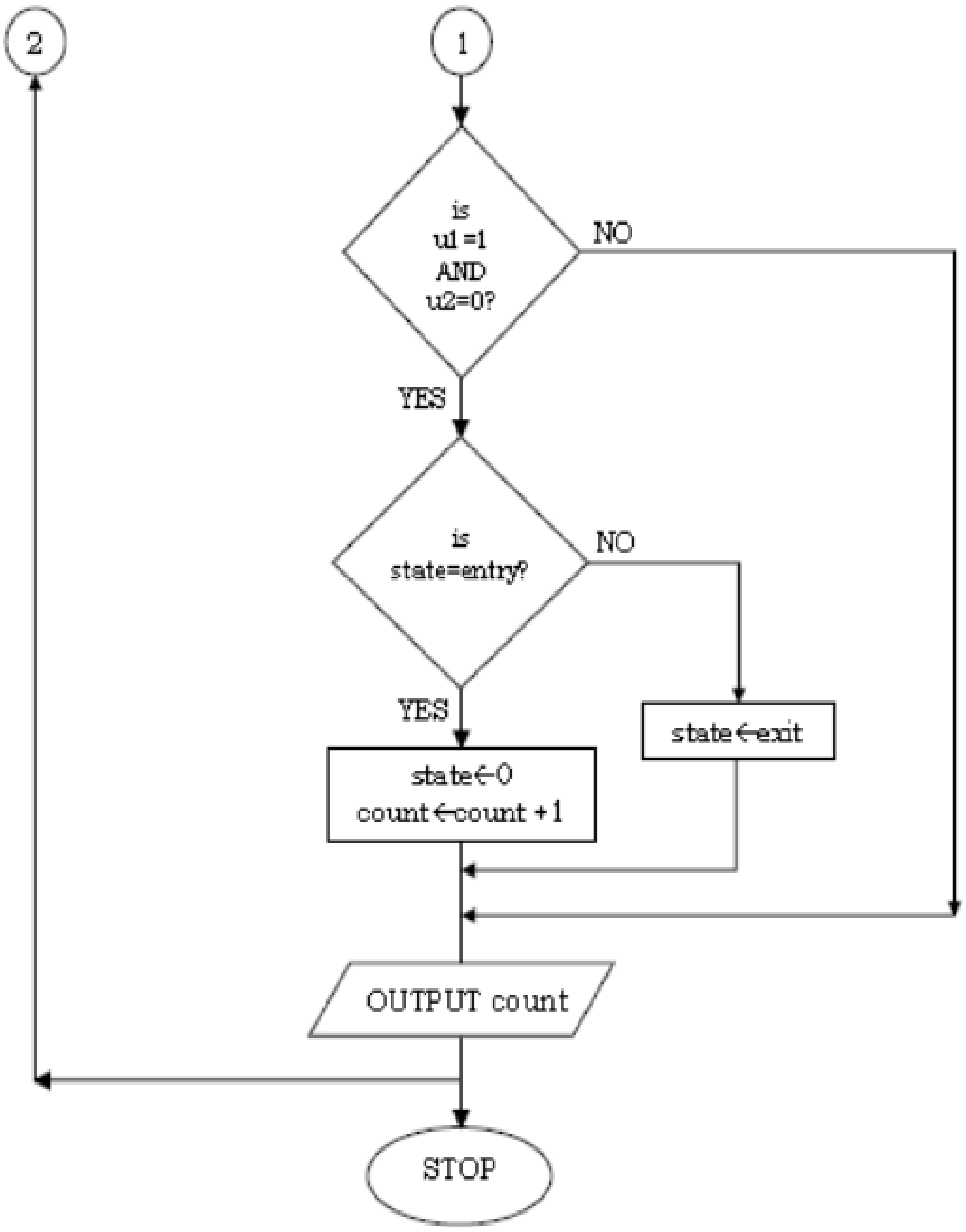
Similar to the above methodology, for departure of a person, we will get output data sets [0,1] and then [1,0], and the state shall be changed to 13.In similar logic of the entry, at the exit of a person, the counter will be decremented by 1 and the respective result is displayed on the LCD.

In case, the total number of people in the room exceeds the critical limit, the LCD is made to show, “MAXIMUM CAPACITY”.

For conventional simplicity of our project, we have assumed, that [0,0] and [1,1] output from the sensors makes no change to the count or the state.

A flowchart to demonstrate the program logic used in this project is given below :



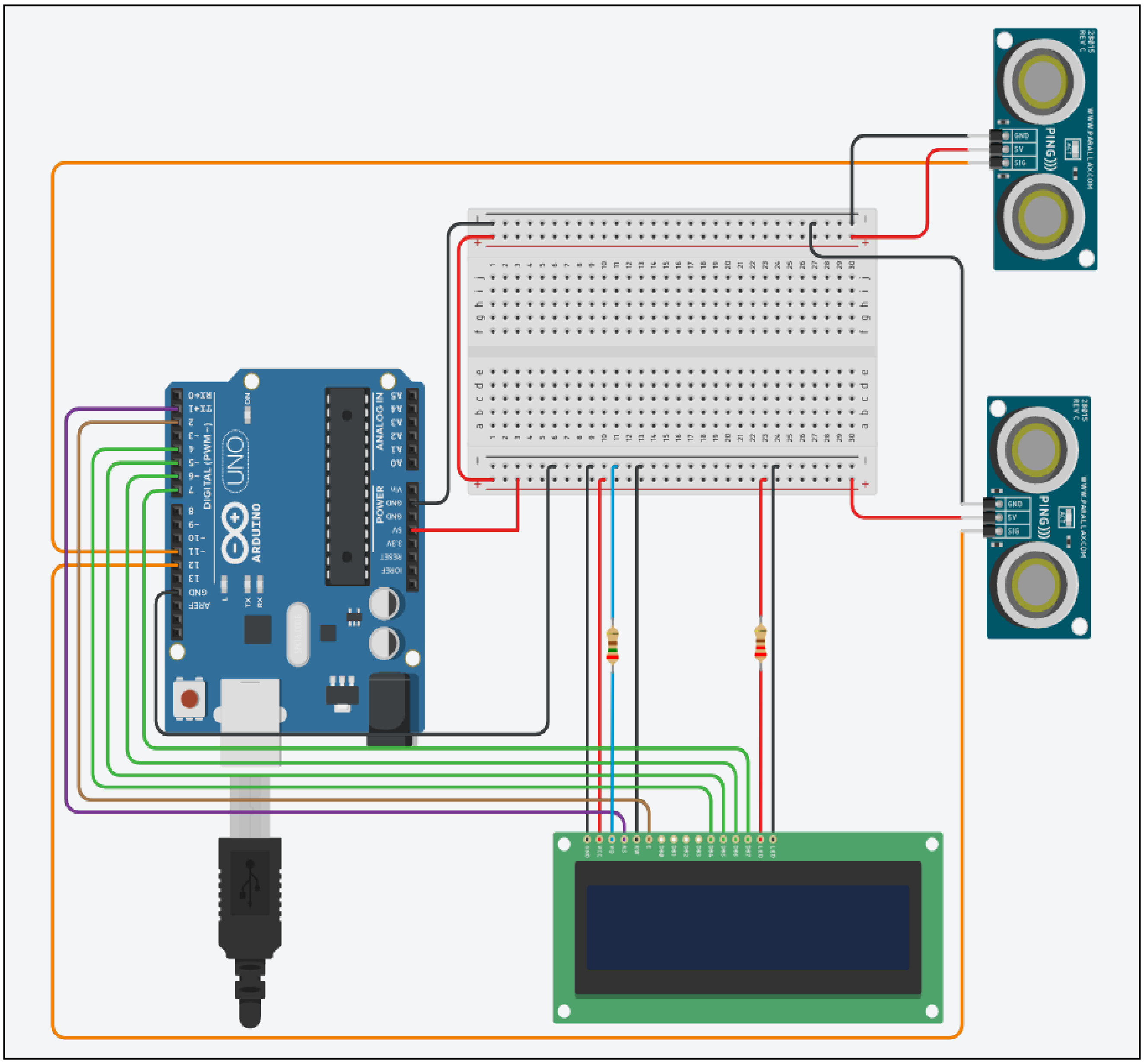

2
1
is u1 =1 AND u2=0?
NO
YES
is state=entry?
NO
YES
state←0
count←count +1
state←exit
OUTPUT count
STOP

SOFTWARE SIMULATION

This project has been simulated in Tinkercad, a free online computer-aided design (CAD) program website. The website facilitated us to make a simulated circuit along with the Arduino code so that simulation can be carried out before implementing this project with actual components and circuit.

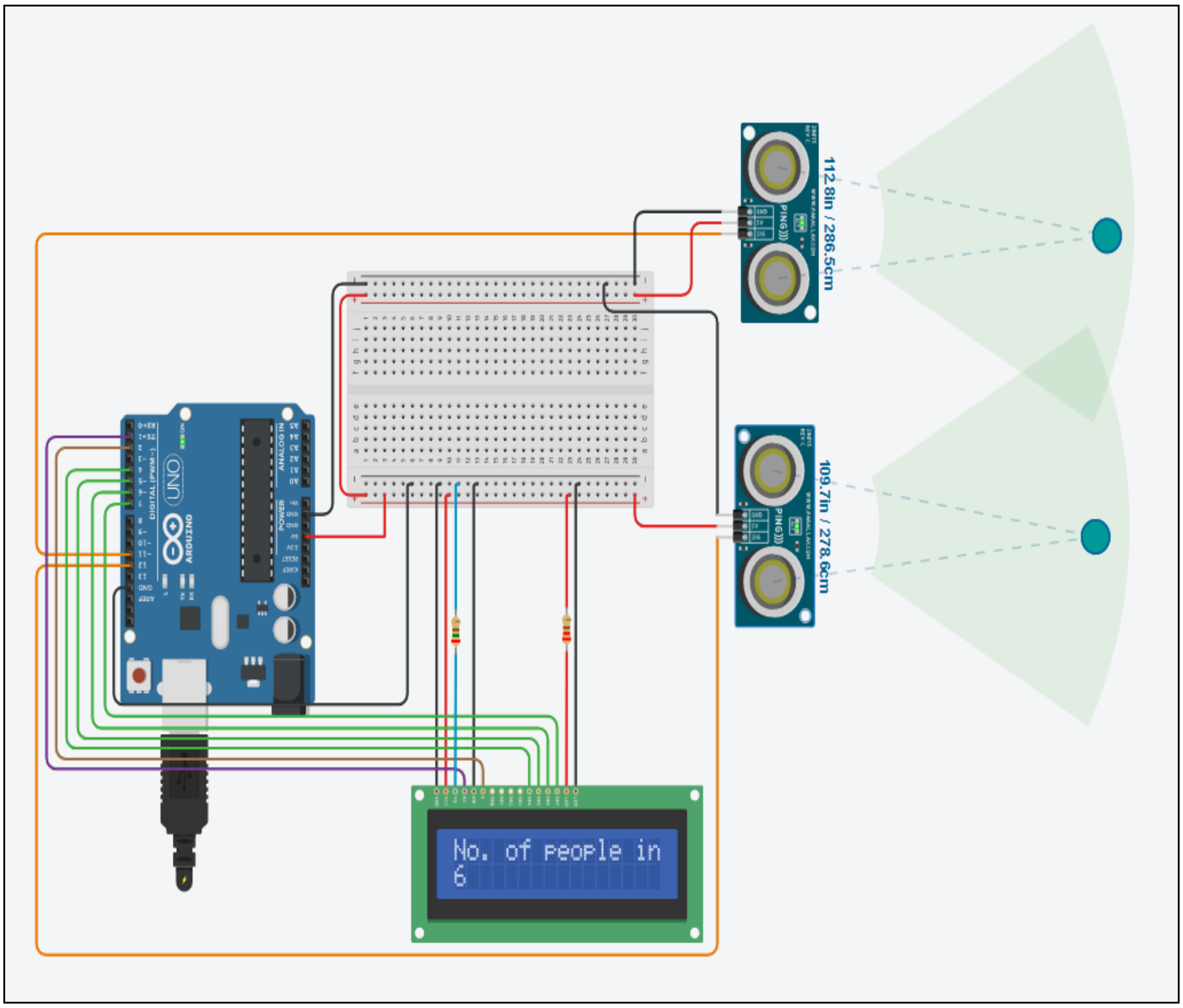
The code used here in this simulation was made and compiled in the Arduino IDE and then that code is used in Tinkercad.

The simulated circuit diagram is given below :



Here, the upper sensor is the sensor outside the door and the lower sensor is the sensor inside the door. The batteries are not shown here in this simulation.

The simulation provides perfect results here as shown below :



The code implemented in this project is provided in Appendix for reference.

LIMITATIONS & FURTHER SCOPE

According to our project logic, we detect any being with height below 2 feet, considering the door height to be 8 feet. As a result, we can assume the system won't count any smaller animal's presence.

In this project we have considered many assumptions as per our convenience. Also, we could only perform software simulation. So, ideal conditions have been considered throughout the mechanism of the project. Hence, the project is limited in ways described below :

1. Here we have assumed a normalised walking speed of a person controlled by the delay function.
2. We have also assumed that the sensors are separated by enough distance so that their spectrum of detection won't interfere with each other.
3. According to our assumption the program execution takes place at conditions where the speed of sound is 340 m/s. But,the speed of sound gets affected by change in temperature(in ^{o}C) and humidity according to the given formula,

   *Speed of sound (m/s) = 331.4 + (0.606*temperature) + 0.0124*humidity*

   To accommodate this change in speed, temperature and humidity sensors may be implemented to give accurate results.
4. In this project we have assumed, that the dead zonecan be prevented by the door frame.A dead zone refers to the area directly in front of the transducer face where the sensor can't reliably make measurement. This is due to a phenomenon called ringing. Ringing is a continued vibration of transducer after the excitation pulse. The energy must dissipate before the transducer can listen for a return echo.

REFERENCES

[1] IoT and it's Applications by Prof.Satish Jain, Shashi Singh.

[2] https://www.arduino.cc/en/Reference/PulseIn

[3] https://www.tinkercad.com/

[4] https://www.arduino.cc/en/software

[5] Sensors and Transducers by Ian R. Sinclair.

[6] Programming Arduino Getting Started with Sketches by Simon Monk.

APPENDIX

The code used in the project is given below :

```
#include<LiquidCrystal.h>
const int echo1 = 11; // Echo Pin of Ultrasonic Sensor 1
const int echo2 = 12; // Echo Pin of Ultrasonic Sensor 2

long t1, t2, cm1, cm2, u1=0, u2=0;
long state=0, count=0, h=240, lim=50;

LiquidCrystallcd(1,2,4,5,6,7);

long readDuration(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  /* Reads the echo pin, and returns the sound wave travel time
  in microseconds*/
  return pulseIn(echoPin, HIGH);
}

long dist(long microS)
{
  return microS*0.017;
}

void display()
{
  lcd.begin(16,2);
  lcd.clear();
  if (count<lim)
{
    lcd.print("No. of people in the room");
    lcd.setCursor(0,1);
    lcd.print(count);
  }
  else
  {
    lcd.print("MAXIMUM CAPACITY");
  }
}
```

```
void State()
{
  if(u1==1 && u2==0)
  {
       if(state!=13)  //13 is exit motion, 31 is entry motion
       state=31;
       if(state==13)
       {
       state=0;
       count--;
           delay(1000);
       }
  }
  if(u1==0 && u2==1)
  {
  if(state!=31)
       state=13;
  if(state==31)
  {
       state=0;
       count++;
           delay(1000);
  }
  }
}

void setup()
{
  //Serial.begin(9600);
}

void loop()
{
  t1 = readDuration(echo1, echo1);
  t2 = readDuration(echo2, echo2);
  cm1 = dist(t1);
  cm2 = dist(t2);
  if(cm1<(0.75*h))
    u1=1;
  else
    u1=0;
  if(cm2<(0.75*h))
    u2=1;
  else
    u2=0;
  State();
  display();
  delay(500);
}
```