VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

B.Tech – Electronics & Computer Engineering

ECE4003 – EMBEDDED SYSTEM DESIGN

LAB RECORD

(L64+L65+L66)

Submitted By

18BLC1017 – Dipan Polley

Submitted To

Prof. CHANTHINI BHASKAR

DATE: 25/09/2020

LAB – 09 Ultrasonic Sensor

LABTASK-1:

AIM:

Write a program to read distance value from HC-SR04 ultrasonic sensor module in cm and print it on the serial monitor. Simulate and verify this logic on Arduino Uno using Tinkercad circuits simulator.

API REQUIRED:

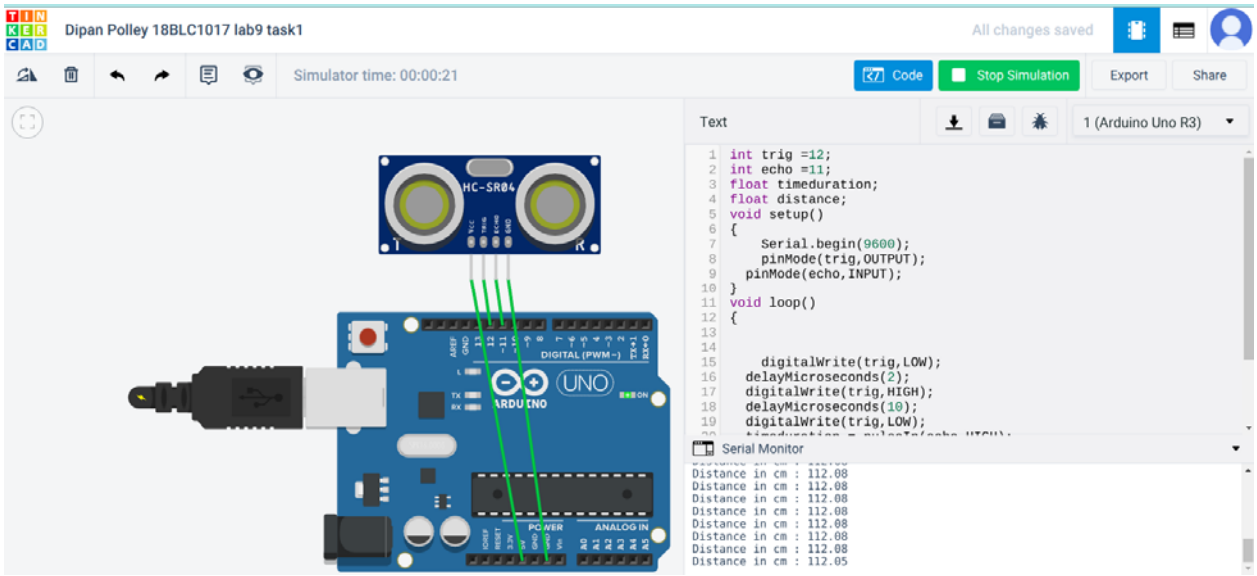
- **pinMode(pin, mode)** - Configures the specified pin to either input or output.
  - pin: the Arduino pin number to set the mode of.
  - **mode**: INPUT, OUTPUT, or INPUT_PULLUP.
- **digitalWrite(pin, value)** - Write a HIGH or a LOW value to a digital pin.
  - pin: the Arduino pin number.
  - value: HIGH or LOW.
- **delay(ms)** - Pauses the program for the amount of time (in milliseconds)
  - ms: the number of milliseconds to pause.
- **Serial.begin(speed) -** Configures a serial port with specified baud rate.
  - **Serial:** To access serial port in Arduino Uno.
  - **begin**: To configuring serial port.
  - **speed:** To specify the baud rate for serial communication.
- **Serial.println(val)** - Prints data on serial port as human-readable ASCII text.
  - **println:** To perform printing text followed by a newline character.
  - **val:** Specify value or text to be printed on serial monitor.
- **delayMicroseconds(us)** - Pauses the program for the amount of time (in microseconds)
  - us: the number of microseconds to pause.
- **pulseIn(pin, value)** - Reads a pulse (either HIGH or LOW) on a pin.

PROGRAM:

```
int trig =12;
int echo =11;
float timeduration;
float distance;
```

```
void setup()
{
        Serial.begin(9600);
        pinMode(trig,OUTPUT);
  pinMode(echo,INPUT);
}
void loop()
{



        digitalWrite(trig,LOW);
  delayMicroseconds(2);
  digitalWrite(trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig,LOW);
  timeduration = pulseIn(echo,HIGH);
  distance = 0.034 * timeduration/2;
  Serial.print("Distance in cm : ");
  Serial.println(distance);
  delay(500);
}
```

OUTPUT:



TINKERCAD LINK:

https://www.tinkercad.com/things/iAyEl0d8zwW-dipan-polley-18blc1017-lab9-task1/editel?sharecode=lvmpdPa-yox4pWVUkbeyVkdmSkSbL5wjwahRNWQ-HGc

INFERENCE:

We learned the basics of ultrasonic sensor and its working. We learned about the HC-SR04 module and its interfacing with Arduino, and how to take the distance as input.

LABTASK-2:

AIM: Write a program to design a reverse parking sensor module. This module consist of HC-SR04 ultrasonic sensor, LCD and buzzer interfaced with Arduino. The ultrasonic sensor continuously measure the distance (in cm) between the car and obstacle, then display it on LCD. Whenever the measured distance is lesser than 30cm generate warning signal to driver using buzzer also display a message "Obstacle !!!" on the first row of the LCD display. Simulate and verify this logic on Arduino Uno using Tinkercad circuits simulator.

API REQUIRED:

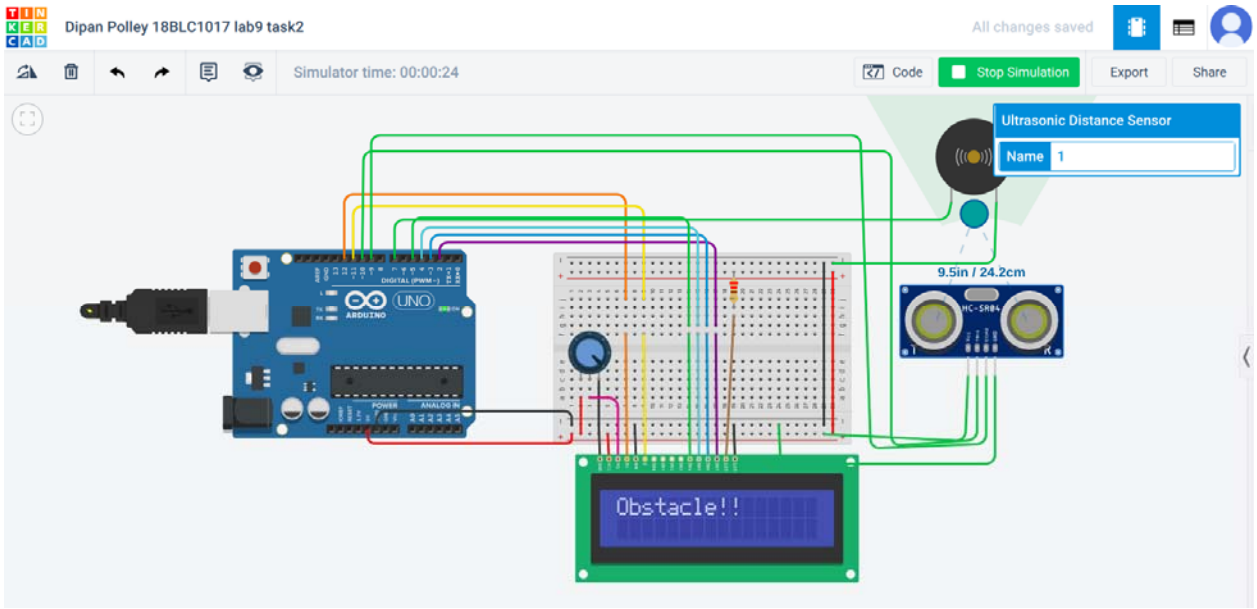
- **pinMode(pin, mode)** - Configures the specified pin to either input or output.
  - pin: the Arduino pin number to set the mode of.
  - **mode**: INPUT, OUTPUT, or INPUT_PULLUP.
- **digitalWrite(pin, value)** - Write a HIGH or a LOW value to a digital pin.
  - pin: the Arduino pin number.
  - value: HIGH or LOW.
- **delay(ms)** - Pauses the program for the amount of time (in milliseconds)
  - ms: the number of milliseconds to pause.
- **Serial.begin(speed) -** Configures a serial port with specified baud rate.
  - **Serial:** To access serial port in Arduino Uno.
  - **begin**: To configuring serial port.
  - **speed:** To specify the baud rate for serial communication.
- **Serial.println(val)** - Prints data on serial port as human-readable ASCII text.
  - **println:** To perform printing text followed by a newline character.
  - **val:** Specify value or text to be printed on serial monitor.
- **delayMicroseconds(us)** - Pauses the program for the amount of time (in microseconds)
  - us: the number of microseconds to pause.
- **pulseIn(pin, value)** - Reads a pulse (either HIGH or LOW) on a pin.
- **lcd.begin(cols, rows)-**Initializes the interface to LCD, and specifies dimensions
- **lcd.print(data)/lcd.write(data)** - the character to write to the display
- **lcd.clear()-**Clears the LCD screen and positions the cursor on upper-left corner

PROGRAM:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int trig=10;
int echo=9;
int bz=7;
float timeduration;
float distance;

void setup()
{
  lcd.begin(16, 2);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(bz, OUTPUT);
}

void loop()
{
  lcd.clear();
  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);
  timeduration = pulseIn(echo, HIGH);
  distance = 0.034 * timeduration/2;
  if(distance<30)
  {
    lcd.clear();
    lcd.setCursor(0, 0);
          lcd.print("Obstacle!!");
    digitalWrite(bz, HIGH);
  }
  else
```

```
{
  lcd.print("Distance: ");
 lcd.print(distance);

 }

 delay(500);
 digitalWrite(bz, LOW);
}
```

OUTPUT:



TINKERCAD LINK:

https://www.tinkercad.com/things/hgbjA0w6Zzh-dipan-polley-18blc1017-lab9-task2/editel?sharecode=Lm66zleBDEaXgrNylqvoBhlbX1vcl-l41bDGnv3mzM0

INFERENCE:

We used to the ultrasonic sensor to measure distance in real time, and give output in lcd on this basis.

LABTASK-3:

AIM:

Write a program to design smart parking system using HC-SR04 ultrasonic sensor, servo motor, buzzer, LCD and Arduino Uno.

- The ultrasonic sensor module place near the gate entrance continuously check for the incoming vehicles. The LCD display “Smart Parking” on the first row and “Avail. slot: XY” in second row of the display.
- When a vehicle comes closer to the ultrasonic sensor detection area and parking slot is available then the system open a gate barrier to 90° (close after 10 seconds) to allow the vehicle to the parking slot and decrement parking slot by 1.
- If no parking slot available then display a message “No Parking slot” on LCD (2nd line) and switch on the buzzer (for 5 Seconds). Have a similar system on the exit and increment the free slot by 1 for every vehicle leaves the parking slot.

Simulate and verify this logic on Arduino Uno using Tinkercad circuits simulator.

Note: XY is number of available slot and initially assume total available parking slot is 15

API REQUIRED:

- **pinMode(pin, mode)** - Configures the specified pin to either input or output.
  - pin: the Arduino pin number to set the mode of.
  - **mode**: INPUT, OUTPUT, or INPUT_PULLUP.
- **digitalWrite(pin, value)** - Write a HIGH or a LOW value to a digital pin.
  - pin: the Arduino pin number.
  - value: HIGH or LOW.
- **delay(ms)** - Pauses the program for the amount of time (in milliseconds)
  - ms: the number of milliseconds to pause.
- **Serial.begin(speed) -** Configures a serial port with specified baud rate.
  - **Serial:** To access serial port in Arduino Uno.
  - **begin**: To configuring serial port.
  - **speed:** To specify the baud rate for serial communication.
- **Serial.println(val)** - Prints data on serial port as human-readable ASCII text.
  - **println:** To perform printing text followed by a newline character.
  - **val:** Specify value or text to be printed on serial monitor.
- **delayMicroseconds(us)** - Pauses the program for the amount of time (in microseconds)
  - us: the number of microseconds to pause.

- **pulseIn(pin, value)** - Reads a pulse (either HIGH or LOW) on a pin.
- **lcd.begin(cols, rows)-**Initializes the interface to LCD, and specifies dimensions
- **lcd.print(data)/lcd.write(data)** - the character to write to the display
- **lcd.clear()-**Clears the LCD screen and positions the cursor on upper-left corner
- **servo.attach(pin)** or **servo.attach(pin, min, max)** - Attach the Servo variable to a pin.
    - **servo:** a variable of type Servo
    - **pin**: the number of the pin that the servo is attached to
    - **min** (optional): the pulse width, in microseconds, corresponding to the minimum (0-degree) angle on the servo
    - **max** (optional): the pulse width, in microseconds, corresponding to the maximum (180-degree) angle on the servo
- **servo.write(angle)** - Writes a value to the servo, controlling the shaft accordingly.
    - **angle**: the value to write to the servo, from 0 to 180

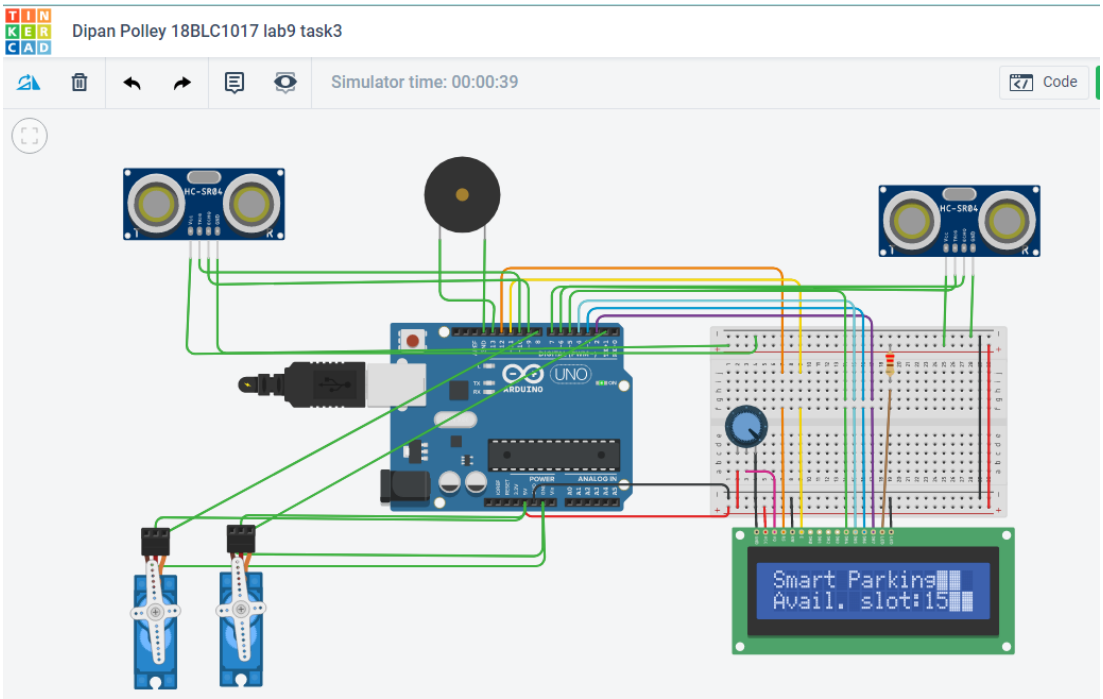
PROGRAM:

```
#include <LiquidCrystal.h>
#include <Servo.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
Servo s1;
Servo s2;
int t1=10;
int t2=7;
int e1=9;
int e2=6;
int b=13;
int c=15;
long duration, distance, TopSensor,BackSensor;

void setup() {
  s1.attach(8);
  s2.attach(1);
  s1.write(0);
  s2.write(0);
  lcd.begin(16, 2);
```

```
  lcd.setCursor(0,0);
  lcd.println("Smart Parking");
  pinMode(t1,OUTPUT);
  pinMode(t2,OUTPUT);
  pinMode(b,OUTPUT);
  pinMode(e1,INPUT);
  pinMode(e2,INPUT);
}

void loop()
{
  SonarSensor(t1, e1);
  TopSensor = distance;
  SonarSensor(t2, e2);
  BackSensor = distance;
  delay(2000);

  lcd.setCursor(0,1);
  if(c!=0){
    if(TopSensor <=70)
          {
          c=c-1;
          s1.write(90);
          delay(10000);
          s1.write(0);
          }
          else if(BackSensor >70 && c<15)
          {
          delay(2000);
          c=c+1;
          s2.write(90);
          delay(10000);
          s2.write(0);
          }
    lcd.print("Avail. slot:");
```

```
    lcd.println(c);
  }
  else if(c==0)
  {
    digitalWrite(b,1);
    lcd.println("No Parking Slot");
    delay(1000);
    digitalWrite(b,0);
  }
}
void SonarSensor(int trigPin,int echoPin)
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance= duration*0.034/2;
}
```

OUTPUT:



TINKERCAD LINK:

https://www.tinkercad.com/things/k9HtAlftucZ-dipan-polley-18blc1017-lab9-task3/editel?sharecode=fqptclQFLDeKooqd_fmZtpCHOOpbql4aN1GTYdNlETo

INFERENCE:

The ultrasonic sensor is used to detect incoming and outgoing vehicles. An internal count of the number of free slots available is maintained, and the gate opens based on the count; the gate being controlled by the servo motor.

CHALLENGING TASK 1:

AIM:

Write a program to design an Adaptive cruise control (ACC) system to help the vehicle to maintain a safe following distance and stay within the speed limit. This system adjusts a car's speed automatically so drivers don't have to and the detailed function of the system is as follows:

1. This system consists of one slide switch (to select normal or cruise mode), 4 DIP switches (to select different cruise speed), LCD (to display mode and speed of the vehicle), DC motor (to indicate the vehicle speed in rpm), a potentiometer (to control the vehicle speed in normal mode) and a buzzer (to indicate vehicle is very close to others)

2. Control switch(slide) decide whether the vehicle to be operated in Normal mode (LOW) or cruise mode (HIGH)

3. In normal mode, vehicle speed is controlled by adjusting the potentiometer knob. Also, display a message “Normal mode” in first row and “Speed:XYkmph” on the second row of the LCD. XY is the present vehicle speed.

4. In cruise mode, DIP Switch-1 to 4 select vehicle speed as 40, 60, 80 and 90 kmph respectively. Display the message “Cruising:ABkmph” on the first row and “Speed: XY kmph” on the second row of the LCD. AB is either 40/60/80/90 and XY is the present vehicle speed.

5. Using ultrasonic sensor, map the selected cruise speed with the distance. If other vehicle is not near to the coverage area of the ultrasonic sensor operate the vehicle at its maximum selected cruise speed with high rpm on DC motor.

6. If other vehicle is detected by ultrasonic sensor vicinity, depends on its distance adjust the speed (DC motor) of the vehicle accordingly.

7. Whenever another vehicle reaches very near to our vehicle (<1m) activate the buzzer to warn the driver. Also display the message “Very close alert” on first row of the LCD.

Simulate and verify this logic on Arduino Uno using Tinkercad circuits simulator.

API REQUIRED:

- **pinMode(pin, mode)** - Configures the specified pin to either input or output.
  - pin: the Arduino pin number to set the mode of.
  - **mode**: INPUT, OUTPUT, or INPUT_PULLUP.
- **digitalWrite(pin, value)** - Write a HIGH or a LOW value to a digital pin.
  - pin: the Arduino pin number.
  - value: HIGH or LOW.
- **delay(ms)** - Pauses the program for the amount of time (in milliseconds)
  - ms: the number of milliseconds to pause.
- **Serial.begin(speed) -** Configures a serial port with specified baud rate.
  - **Serial:** To access serial port in Arduino Uno.
  - **begin**: To configuring serial port.
  - **speed:** To specify the baud rate for serial communication.
- **Serial.println(val)** - Prints data on serial port as human-readable ASCII text.
  - **println:** To perform printing text followed by a newline character.
  - **val:** Specify value or text to be printed on serial monitor.
- **delayMicroseconds(us)** - Pauses the program for the amount of time (in microseconds)
  - us: the number of microseconds to pause.
- **pulseIn(pin, value)** - Reads a pulse (either HIGH or LOW) on a pin.
- **lcd.begin(cols, rows)-**Initializes the interface to LCD, and specifies dimensions
- **lcd.print(data)/lcd.write(data)** - the character to write to the display
- **lcd.clear()-**Clears the LCD screen and positions the cursor on upper-left corner
- **servo.attach(pin)** or **servo.attach(pin, min, max)** - Attach the Servo variable to a pin.
  - **servo:** a variable of type Servo
  - **pin**: the number of the pin that the servo is attached to
  - **min** (optional): the pulse width, in microseconds, corresponding to the minimum (0-degree) angle on the servo
  - **max** (optional): the pulse width, in microseconds, corresponding to the maximum (180-degree) angle on the servo
- **servo.write(angle)** - Writes a value to the servo, controlling the shaft accordingly.
  - **angle**: the value to write to the servo, from 0 to 180

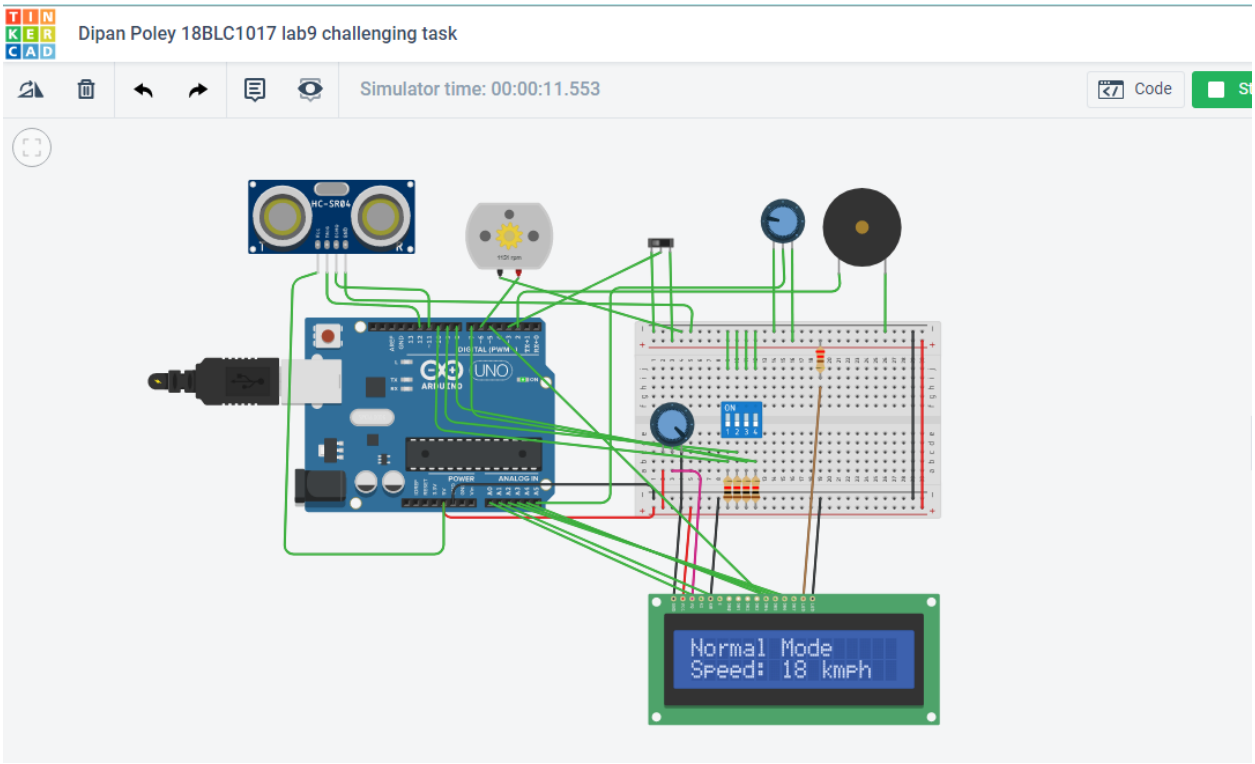
PROGRAM:

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(A0, A1, 5, A4, A3, A2);
int buz = 2;
int pot = A5;
int slide = 3;
int trig = 12;
int echo = 11;
int motor = 6;
int s1 = 10, s2 = 9, s3 = 8, s4 = 7;

float t, dist;
int mode, speed, cspeed;
float potinput, sout;

void setup()
{
 Serial.begin(9600);
 lcd.begin(16, 2);
 lcd.clear();
 pinMode(trig, OUTPUT);
 pinMode(echo, INPUT);
 pinMode(buz, OUTPUT);
 pinMode(slide, INPUT);
 pinMode(motor, OUTPUT);

 pinMode(s1, INPUT);
 pinMode(s2, INPUT);
 pinMode(s3, INPUT);
 pinMode(s4, INPUT);
}

void loop()
{
```

```
mode = digitalRead(slide);
digitalWrite(buz, 0);

digitalWrite(trig, 0);
delayMicroseconds(2);
digitalWrite(trig, 1);
delayMicroseconds(10);
digitalWrite(trig, 0);
t = pulseIn(echo, HIGH);
dist = (0.034 * t)/2;
lcd.setCursor(0,1);

if(mode==0)
{
  //normal mode
  potinput = analogRead(pot);
  lcd.setCursor(0,0);
  lcd.write("Normal Mode     ");
  lcd.setCursor(0,1);
  lcd.write("Speed: ");
        sout = (potinput / 1023) * 255;
  analogWrite(motor, sout);
  speed = (potinput/1023) * 90;
  lcd.setCursor(7,1);
  lcd.print(speed);
  lcd.write(" kmph  ");
}

if(mode==1)
{
  if(digitalRead(s1)==HIGH)
  {
    cspeed = 0;
  }
  else if(digitalRead(s2)==HIGH)
```

```
    {
      cspeed = 40;
    }
    else if(digitalRead(s3)==HIGH)
    {
      cspeed = 60;
    }
    else if(digitalRead(s4)==HIGH)
    {
      cspeed = 80;
    }
    else
      cspeed = 90;

    lcd.setCursor(0, 0);
    lcd.write("Cruising: ");
    lcd.setCursor(10, 0);
    lcd.print(cspeed);
    lcd.write(" kmph");

    digitalWrite(trig, 0);
    delayMicroseconds(2);
    digitalWrite(trig, 1);
    delayMicroseconds(10);
    digitalWrite(trig, 0);
    t = pulseIn(echo, HIGH);
    dist = (0.034 * t)/2;
    speed = (dist/336)*cspeed;  //mapping dist to speed.
  }

}
```

OUTPUT:



TINKERCAD LINK:

https://www.tinkercad.com/things/kYwnctq0lEe-dipan-poley-18blc1017-lab9-challenging-task/editel?sharecode=vjupD4dj8aomiS3D7-y8YvR3YR-mdlspQd6g6pcbxqY

INFERENCE:

The ultrasonic sensor is used to continually monitor the distance to the vehicle in front. The vehicle speed is adjusted accordingly by mapping max. distance to the maximum allowed speed set by the cruise control dip switches. In normal operation, the speed is controlled by the potentiometer (mapping pot range to range of voltage output to DC motor).

RESULTS:

Thus, we learned the working of ultrasonic sensor, and its interfacing with Arduino. We used it to carry out many applications.