CS225-226 Project Report

Smart Dustbin

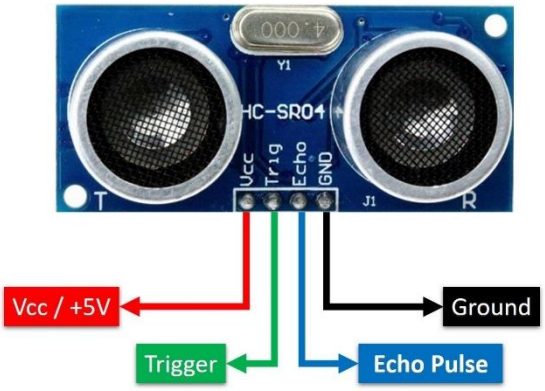
Adarsh Kumar

2001CS02

In this project, I have designed a simple system called **Smart Dustbin** using Arduino, Ultrasonic Sensor and Servo Motor, where the lid of the dustbin will automatically open itself upon detection of human hand or any other object near it.

Main Components Used

Ultrasonic sensor



It is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity.

It has four pins:

1. VCC pin: connected to 5V VCC, powers the sensor.
2. TRIG pin: has to be kept high for 10 microseconds to initialize measurement by sending ultrasonic wave.
3. ECHO pin: goes high for a period of time which will be equal to the time taken for the ultrasonic wave to return back to the sensor.
4. GND pin: connected to 0V (GND).

It works in five major steps:

1. TRIG pin receives a 10 microseconds pulse from the Arduino.
2. With the falling edge of the pulse, the sensor gets activated and sends out 8 cycles sonic burst of 40 kHz pulses.
3. As the sonic bursts are sent, the ECHO pin sets itself to be HIGH and the timing starts.
4. When the sonic bursts return, the ECHO pin goes down to LOW and the timing stops.
5. ECHO pin sends the duration of time for which it was HIGH to Arduino, which is used to calculate distance.

Micro Servo Motor

It is a small device that has an output shaft. This shaft can be positioned to specific angular positions by sending the servo a coded signal. As long as the coded signal exists on the input line, the servo will maintain the angular position of the shaft.

It has three pins:

1. VCC pin: connected to 5V VCC.
2. Signal pin: receives the Pulse-width Modulation (PWM) signal from Arduino.
3. GND pin: connected to 0V (GND).

Its working:

The width of the PWM signal decides the angle of rotation. The minimum width and the maximum width of the PWM signal are fixed in Arduino Servo Library.

If the width of our PWM signal is equal to the minimum width, then the servo rotates to 0º, if it is equal to the maximum width, then the servo rotates to 180º and if the width is in between, then the servo rotates according to the proportion.

PWM signal can be generated by using Arduino, but for ease, we use the Arduino Servo Library to control the servo by just giving the angle that we need to rotate.

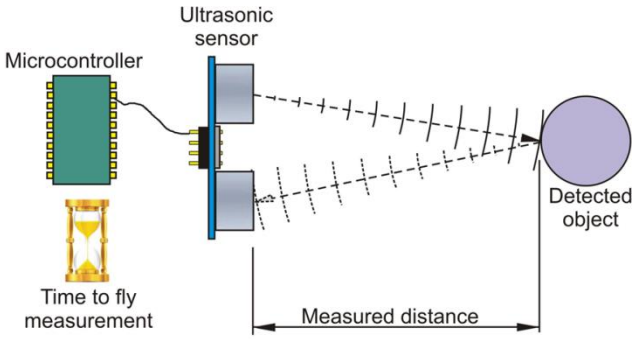
Working Principle

It is based on the same principle as a radar system. The microcontroller sends a trigger signal to the ultrasonic sensor. The duty cycle of this trigger signal is 10 microseconds for the HC-SR04 ultrasonic sensor. When triggered, the ultrasonic sensor generates eight acoustic (ultrasonic) wave bursts and initiates a time counter. As soon as the reflected (echo) signal is received, the timer stops.

Let T be the time the pulse took to travel back to the transducer, then we can say that it took T/2 time to reach the object after being emitted by the transducer.

Therefore, distance of the object from the transducer = speed of sound x T/2

If this distance is less than 50 cm, we trigger the servo motor and it opens the lid of the dustbin. After some delay, we again trigger the servo and it closes the lid.



Code Overview

```
#include <Servo.h> // this library allows an Arduino board to control RC (hobby) servo motors

Servo servo;

int trigPin = 5, echoPin = 6, servoPin = 7; // initialising all three pins
int led = 10;
long duration, dist, average;
long arr[3];
```

At the top, we have included Servo library. It allows Arduino boards to control a variety of servo motors. Below that we have defined variables that will be used for calculating the distance of the object from the dustbin.

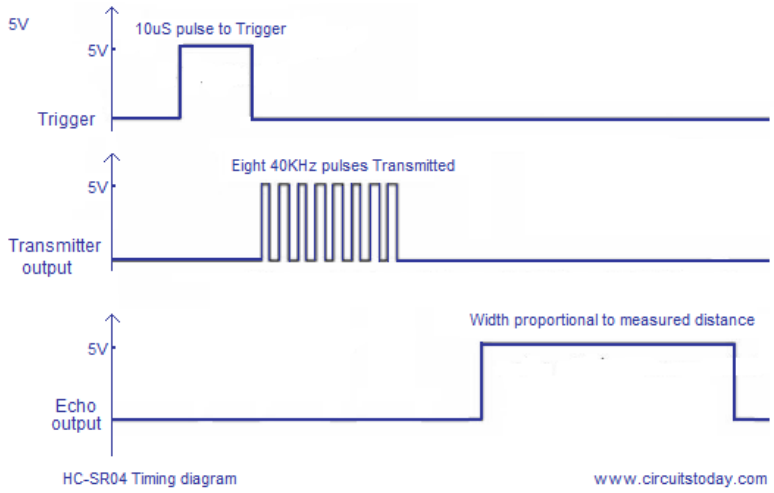
```
void setup() {
  Serial.begin(9600); // establishes serial communication between Arduino board and another device
  servo.attach(servoPin); // attaches servo motor to the Arduino board
  pinMode(trigPin, OUTPUT); // configuring trigPin to behave as an output
  pinMode(echoPin, INPUT); // configuring echoPin to behave as an input
  servo.write(0); // sets the angle of the shaft (in degrees)
  delay(100); // delaying by 100 milli seconds
  servo.detach(); // detaches the servo variable from its pin
}
```

The setup function is executed only once at the beginning. It has been used to configure various pins to behave as an output/input. It has also been used to set the angle of servo motor to 0º in the beginning.

```
void measure() {
  digitalWrite(10, HIGH); // sets the digital pin 10 on
  digitalWrite(trigPin, LOW); // sets the trigPin off
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH); // sets the trigPin on
  delayMicroseconds(15);
  digitalWrite(trigPin, LOW); // sets the trigPin off
  pinMode(echoPin, INPUT); // configuring echoPin to behave as an input
  duration = pulseIn(echoPin, HIGH); // returns the length of the pulse in microseconds
  dist = (duration / 2) / 29.1;  // obtaining distance
}
```

Here, the Pulse-width Modulation (PWM) pin is set to high. Then first we set TRIG pin LOW, then HIGH, then again LOW. This sends a pulse to the ultrasonic sensor after which it gets activated.

Once it gets activated, it sends out 8 cycles sonic burst of 40kHz pulses, which after bouncing off from an object returns to the sensor. Also, as soon as it sends this burst, ECHO pin is set to HIGH.



As soon as ECHO pin becomes HIGH, **pulseIn()** function starts timer. When the burst is received by the sensor, ECHO pin becomes LOW and this function stops the timer. Now it returns the duration for which ECHO pin was HIGH.

```
void loop() {
  for (int i = 0; i < 3; i++) {
    measure();
    arr[i] = dist;
    delay(10);
  }
  dist = (arr[0] + arr[1] + arr[2]) / 3; // above code takes three continuous
  measurements with 0.01 seconds delay and then take their average
  if ( dist < 50 ) {
    servo.attach(servoPin); // attaches servo motor to the Arduino board
    delay(1);
    servo.write(0); // sets the angle of the shaft (in degrees)
    delay(3000);
    servo.write(150); // sets the angle of the shaft (in degrees)
    delay(1000);
    servo.detach(); // detaches the servo variable from its pin
  }
  Serial.print(dist); // prints data to the serial port as human-readable ASCII text
}
```

The loop function runs repeatedly till the Arduino is powered on.

We take three continuous measurements with 0.01 seconds delay and then take their average to reduce errors. If this average value is less than 50 cm, we activate the servo and use the Servo library functions to move the servo motor to our desirable angle.

Here, we hold the servo at an angle of 0 degrees for 5 seconds and then move the servo to 150 degrees. After a one second delay, we detach the servo variable from its pin.

Complete Functioning Diagram