# Design Application Report - Microcontroller

ES2A8 – Engineering Design

Harjeet Brar - U1521184

School of Engineering

13/01/2016

## Principle of the Operation

The principle operation of this task was to firstly develop a circuit which included the following components: an ultrasonic module (HC-SR04), an LCD (16×2), a 10 kΩ potentiometer onto of which the ultrasonic module will be mounted, a 220 Ω resistor, breadboard, wires, 3 paper cups (obstacles) and the Arduino UNO microcontroller. Using these components, the objective was to produce such a circuit, as seen in figure 1 and 2, through which I would be able to detect obstacles (paper cups) and the angles at which they were placed at relative to an origin position. The layout is displayed clearly in figure 13.

**Step one** was designing the circuit, which required the use of previous tutorials to link the LCD screen with the Arduino, as well as a few external links to understand how the potentiometer is correctly connected with the ultrasonic module and then the Arduino. **Step two** involved developing the code for the prototype, which was conducted in a similar manner. Finally, **Step three** was a matter of testing the design and code for any faults, resolving them and then carrying out the experiment.

## STEP 1: Designing the Circuit

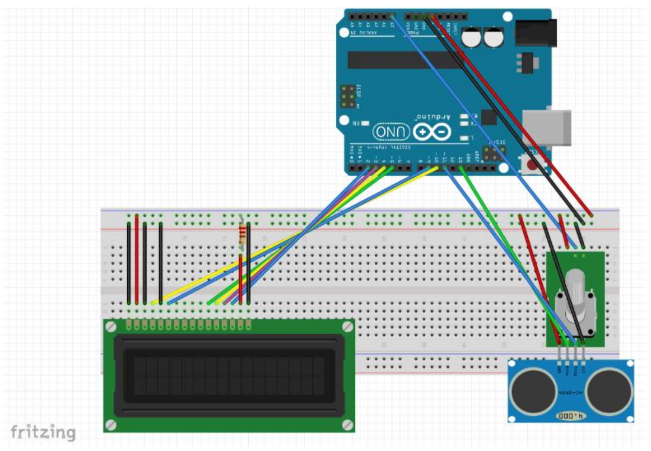


*Figure 2: Schematic of circuit connected to the Arduino device*

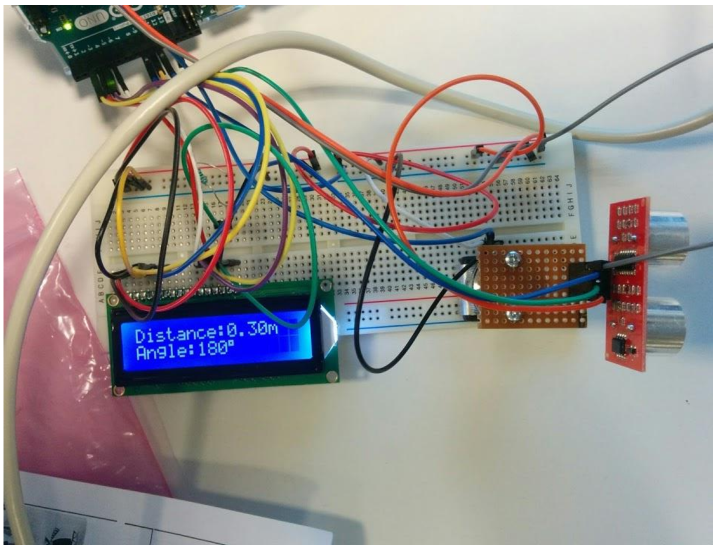


*Figure 1: Schematic of the breadboard*

I was familiar with connecting many of the components involved in this experiment, such as the connection between the LCD and the Arduino was something we had done in previous labs. However, I came across a few difficulties when connecting the potentiometer and the ultrasonic module to the Arduino, as well as displaying both, the distance and the angle of the cups located near the ultrasonic sensor.

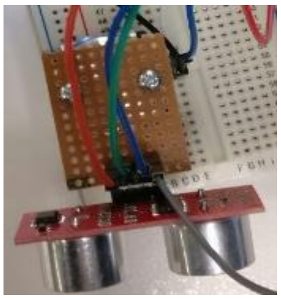


*Figure 3: Ultrasonic module mounted onto the potentiometer*

This required mounting the ultrasonic module onto the potentiometer, as shown in figure 3. Having resolved the issue of not being able to measure the angles of the cups from an origin position I was able to utilize both rows of the LCD, which can be seen in figure 4. All the components were powered by the Arduino, which was being power by the

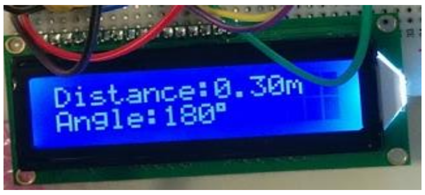


*Figure 4: LCD*

computer through a USB connection, which leads me on to the code which as being supplied from the computer.

<u>STEP 2: The Code</u>

**Setting up the LCD**

The following code in figure 7 shows where the pins are being input into the Arduino, in order to produce an output on the LCD.

```
else {
 LCD.print("Out of range");
}
```

*Figure 5: Code used for when an object is "out of range"*

```
#include <LiquidCrystal.h>
const int pLCD_RS = 10;
const int pLCD_EN = 9;
const int pLCD_DB4 = 5;
const int pLCD_DB5 = 4;
const int pLCD_DB6 = 3;
const int pLCD_DB7 = 2;
const int sensorPin = A0;
int angle;
LiquidCrystal LCD(pLCD_RS, pLCD_EN, pLCD_DB4, pLCD_DB5, pLCD_DB6, pLCD_DB7);
LCD.begin(16,2); //Tell arduino to start our 16 column and 2 row LCD
LCD.setCursor(0,0); // Set cursor to top left corner
if (targetDistance <= 0.4 && targetDistance >=0.05){
```

*Figure 6: Code for the connection between the LCD and the Arduino*

```
  LCD.print("Distance:");
  LCD.print(targetDistance);
  LCD.print("m");
  LCD.setCursor(0,1);
  LCD.print("Angle:");
  LCD.print(angle);
  LCD.print(char(223));
}
```

*Figure 7: Code used to print distance and angles*

The “targetDistance” command is used to set the following range as shown in figure 7: 5 cm ≤ d ≤ 40 cm. Any objects outside of this will not show readings on the LCD, but will instead display what is shown in figure 5 (“out of range”). The function used to position the cursor to the top left corner of the LCD, is shown through “LCD.setCursor” which leads on to the used of “LCD.print” function, to display the distance and the angle the cups were positioned from the ultrasonic module. This is shown in figure 7 (on the left).

**Setting up the Ultrasonic Module**

```
int trigPin = 13;
int echoPin = 11;
int myCounter = 0;
int servoControlPin = 6;
float pingTime; //Time for pin to travel from sensor to the target and rebound back
float targetDistance; //Distance to target in meters
float speedOfSound = 343; //Speed of sound in meters per second
void setup() {

  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(sensorPin, INPUT);
```

*Figure 9: code for setting up the ultrasonic module*

The schematic of the ultrasonic module HC-SR04 is shown in figure 9 below, which displays the various pins on the module itself, for example the trigPin and the echoPin. The numbers in the code above (figure 6), represent what pins in the Arduino the module in connected up to, for example the first line implies that the trigPin is linked to the number 13 pin on the Arduino. The code on the left in figure 7 shows how the ultrasonic module is setup, with which pins being used as INPUTs and which pins being used as OUTPUTs. The “float” function also shows how the ultrasonic sensor is being programmed to feedback the results it captures in meters, through the use of speed of sound in m/s. After sending a ping, it bounces of an obstacle and

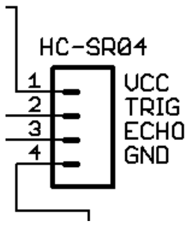
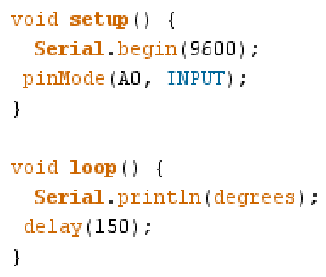


*Figure 8: Schematic of the ultrasonic module*

returns back to the ultrasonic sensor, which detects this and records the time taken. From this, the ultrasonic sensor finds the distance using the equation, distance = speed x time.

## Setting up the Potentiometer

In order to set up my potentiometer, I used the following code shown in figure 11. This was necessary in finding my reference point ($0^o$ point), from which I could measure my angles relative to this.

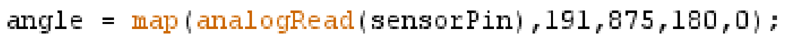


```
void setup() {
  Serial.begin(9600);
 pinMode(A0, INPUT);
}

void loop() {
  Serial.println(degrees);
 delay(150);
}
```

*Figure 10: Code used to set up my potentiometer*

I began at $0^o$ and rotated the potentiometer anticlockwise. As the potentiometer rotates, there is an effect on the voltage being detected by the Arduinos A0 input pin. When this is printed out to the serial monitor, through the "serial.println" function, values are acquired which represent the angles from 0 to 1024. This is because the microcontroller uses a 10 bit analog (voltage) to (numerical) conversion, which gives a possible $2^{10}$ possible numbers. From this we know a voltage of 0V would correspond to 0 and a voltage of 5V would give a value of 1024. Therefore, the values 191 and 875 represent where both $0^o$ and $180^o$ lie respectively. It is noteworthy that these values will differ for every individual, due to the unique factor of the equipment.



```
angle = map(analogRead(sensorPin),191,875,180,0);
```

*Figure 11: Code used to set up potentiometers bearings to $180^o$ and $0^o$*

## STEP 3: Carrying out the experiment

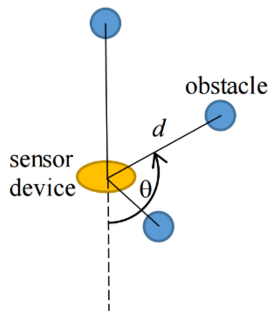


*Figure 13: Layout of obstacles in reference to the ultrasonic module*

The figure 8 suggests how the experiment should be carried out, with "d" representing the distance of the cups from the ultrasonic module, and "θ" representing the angle created in an anticlockwise direction. Note that the angle the picture was taken from, effects the perception of the angle sizes.

Making use of this arrangement I moved the ultrasonic sensor in an anti-clockwise direction, positioning the cups as shown on the right in figure 13. Each time the sensor was rotated, I used to a protractor to measure the angle and check the degree of accuracy. I also used a ruler to confirm the distance being recorded was accurate as well, as shown in figure 14.

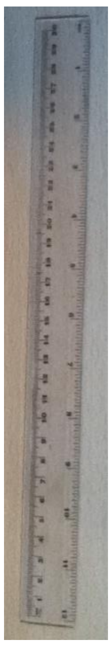


*Figure 14: Ruler used to measure distances*

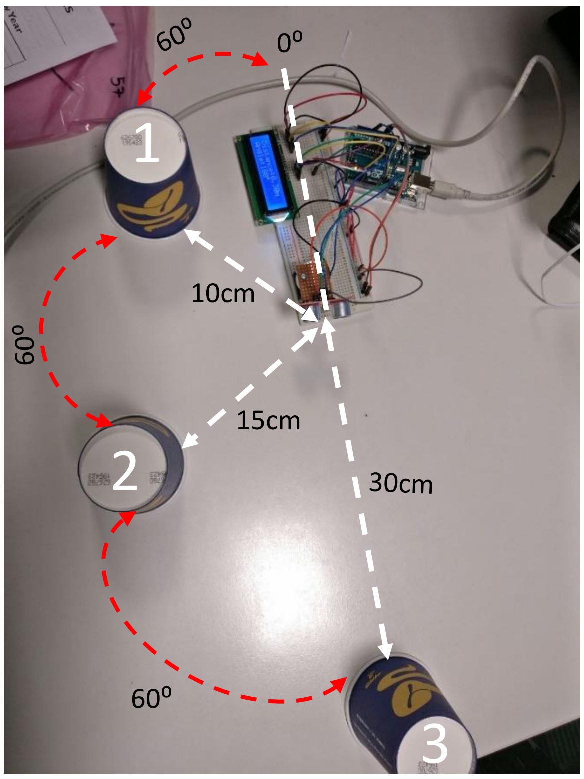


*Figure 12: Layout of the paper cups when conducting the experiment*

## Technical Limitations with the Prototype

- The ultrasonic module is very sensitive to any object within is range, therefore there could be errors involved if obstacles are placed within small distances of each other objects. As the sensor may detect other obstacles rather than the obstacle you are testing.
- The ultrasonic sensor only measures to a 1cm degree of accuracy, which could leave a high level of uncertainty, when measuring obstacles only a short distance away from the ultrasonic sensor.
- The movement of the ultrasonic sensor itself is very fragile and this make it difficult to hold in position on occasions, and can be easily knocked out of position by small vibrations within the environment.
- Lastly, the large number of wires on the breadboard makes it difficult for the ultrasonic sensor to move around freely. This could be resolved through the use of two breadboards.

## Bibliography

- Toptechboy.com. (2017). *LESSON 20: Arduino LCD Project for Measuring Distance with Ultrasonic Sensor | Technology Tutorials*. [online] Available at: http://www.toptechboy.com/arduino/lesson-20-arduino-lcd-project-for-measuring-distance-with-ultrasonic-sensor/ [Accessed 8 Jan. 2017].
- Arduino Laboratory 2 Interfacing Arduino with an analogue sensor and a Liquid Crystal Display. (2017). 1st ed. [ebook] University of Warwick - School of Engineering. Available at: http://www2.warwick.ac.uk/fac/sci/eng/eso/modules/year2/es2a8/resources/es2a8_microcontroller_workshops/arduino_lab2_v4.pdf [Accessed 8 Jan. 2017].
- Playground.arduino.cc. (2017). *Arduino Playground - UltrasonicSensor*. [online] Available at: http://playground.arduino.cc/Main/UltrasonicSensor [Accessed 8 Jan. 2017].
- ES2A8: Engineering Design ES2A8A2- Design Application Report Topic 4 - Microcontroller. (2017). 1st ed. [ebook] University of Warwick - School of Engineering, pp.1-2. [Accessed 8 Jan. 2017].
- Ezdenki.com. (2017). *HC-SR04 Ultrasonic Sensor on an Atmel ATtiny13 at EZdenki.com*. [online] Available at: http://www.ezdenki.com/ultrasonic.php [Accessed 8 Jan. 2017].
- Arduino.cc. (2017). *Arduino - AnalogRead*. [online] Available at: https://www.arduino.cc/en/Reference/analogRead [Accessed 9 Jan. 2017].
- Potentiometer, A. (2017). *Arduino + HC SR04 Ultrasonic + Potentiometer*. [online] Stackoverflow.com. Available at: http://stackoverflow.com/questions/27429205/arduino-hc-sr04-ultrasonic-potentiometer [Accessed 9 Jan. 2017].
- "Arduino 5 Minute Tutorials: Lesson 3 - Potentiometer - Robotshop Blog". *RobotShop Blog*. N.p., 2017. Web. 12 Jan. 2017. Available at: http://www.robotshop.com/blog/en/arduino-5-minute-tutorials-lesson-3-potentiometer-3638