SMART WALKING STICK FOR THE VISUALLY IMPAIRED REPORT

CENG 3006, INTRODUCTION TO EMBEDDED SYSTEMS

Kıymet Deren Toy
Computer Engineering
Muğla Sıtkı Koçman University
Muğla, Turkey
kiymetderentoy@posta.mu.edu.tr

May 4, 2021

Abstract

The aim of this project is to improve the living standards of visually impaired individuals, albeit a bit. This project will make the walking stick used by every visually impaired individual smart by using the arduino UNO board and the necessary software. This report shows the simulation prepared before the project was implemented. The simulation was prepared in Proteus simulation software. As a result of the simulation, it was observed that there was no problem in the implementation of the project.

1 Introduction

In this project, it is addressed to minimize the problems that visually impaired individuals face in life with developing technology, to ensure that they can go out alone in their daily lives and to prevent accidents that may occur due to the noise and chaos of the streets. In order to avoid these problems, a smart walking stick will be offered for the use of visually impaired individuals. On this walking stick, there will be devices that measure the distance to the objects around the walking stick and give warnings by making sounds in risky situations. This report describes the testing/simulation of the project in a virtual environment. First, a distance meter was simulated using devices such as arduino UNO board, ultrasonic distance sensor, sounder in Proteus simulator. Thanks to the distance in this distance meter, a warning is given by sounding the sounder. Finally, the Bluetooth connection of the system with the devices was tested and the simulation was completed.

2 Project Details

Android phone, walking stick and headphones will be used in the project. Our system will be placed on the walking stick. Arduino UNO board, HC-05 bluetooth module, HC-SR04 ultrasonic distance sensor and buzzer are used in our system. The following devices were used in the simulation process.

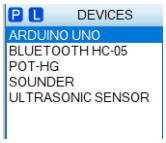

Figure 1: Devices

Here, distance information transfer via bluetooth and text to speech application, which cannot be simulated, will be done in the real state of the project. The completion of the system consists of three steps: implementation, simulation and real life implementation.

3 Implementation

I started the implementation of the system by making the necessary connections. I plugged the "Trigger" output of HC-SR04 ultrasonic distance sensor to port 2 of the Arduino UNO board, and the "Echo" output to port 3. While I combined the "5V" output with power, I connected the "Gnd" output to the ground. Using the sounder instead of the real life buzzer, I connected one end of it to the Arduino UNO's 9 output and one end to the ground. After these connection processes, I adjusted the pins that I connected the outputs of the Arduino according to the input or output in my code.

```
const int trig = 2;
const int echo = 3;
const int buzzer = 9;

void setup() {
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(buzzer, OUTPUT);
```

Figure 2: Setting up Pins

In order to test the distance in the simulation, I added the POT-HG [1] resistance, whose value I can change with physical interventions from outside. I connected the pot-hg's one end to the power and the other to the ground. And then, I calculated the elapsed time in microseconds using the pulseIn()[2] command with the changing resistance. I converted this time to centimeters with the necessary math. Based on this centimeter value, I created an if else block to decide whether the sounder should have high or low voltage. You can see my code below.

```
void loop() {
  digitalWrite(trig, HIGH);
  delayMicroseconds(1000);
  digitalWrite(trig, LOW);
  duration = pulseIn(echo, HIGH);
  cm = (duration / 2) / 29.1;

  if (cm <= 10) {
    digitalWrite(buzzer, HIGH);
    delay(100);
    digitalWrite(buzzer, LOW);
    delay(90);
  }
  else if (cm <= 20) {
    digitalWrite(buzzer, HIGH);
    delay(200);
    digitalWrite(buzzer, LOW);
    delay(150);
  }
  else if (cm <= 30) {
    digitalWrite(buzzer, HIGH);
    delay(350);
    digitalWrite(buzzer, LOW);
    delay(250);
  }
  else if (cm <= 40) {
    digitalWrite(buzzer, HIGH);
    delay(500);
    digitalWrite(buzzer, LOW);
    delay(450);
  }
  else if (cm <= 50) {
    digitalWrite(buzzer, HIGH);
    delay(750);
    digitalWrite(buzzer, LOW);
  }
  else {

  }
}
```

Figure 3: Calculating Cm and Providing Audio Output

I added Virtual Terminal to the simulation to be able to view the serial port in the simulation. I connected the RXD end of the Virtual Terminal to the TX end of the arduino, and the TXD end of the terminal to the RX end of the arduino. I started serial communication with "Serial.begin (9600)" in the setup() part of the code. In the loop() part of the code, I printed the distance instantaneously using "Serial.print()".

```
Serial.print("Distance:");
 Serial.print(cm);
 Serial.print("cm");
 delay(100);
 Serial.println();
```

Figure 4: Virtual Terminal Code

Finally, I made the connections of the HC-05 bluetooth module. Since the RX and TX ports of the Arduino that are exchanging are full, I imported the "SoftwareSerial" library to my code and used the 10th and 11th pins of the Arduino as RX and TX. Thus, I provided data transfer via arduino and bluetooth module. I started the object I defined using the SoftwareSerial library as "mySerial.begin (38400)" in the setup () part of the code and made the necessary settings such as name, password and uart.

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX,TX
String nameUno = "DEREN UNO";
int password = 1412;
String uart = "9600,0,0";
```

```
mySerial.begin(38400);
delay(5000);
mySerial.print("AT+NAME=");
mySerial.println(nameUno);
Serial.print("The name has been set: ");
Serial.println(nameUno);
delay(1000);
mySerial.print("AT+PSWD=");
mySerial.println(password);
Serial.print("The password has been set: ");
Serial.println(password);
delay(1000);
mySerial.print("AT+UART=");
mySerial.println(uart);
Serial.print("The baud rate has been set: ");
Serial.println(uart);
delay(2000);
Serial.println("Process completed.");
```

Figure 5: Bluetooth Settings in Code

After all these steps, the necessary code and necessary connections for simulation were established.

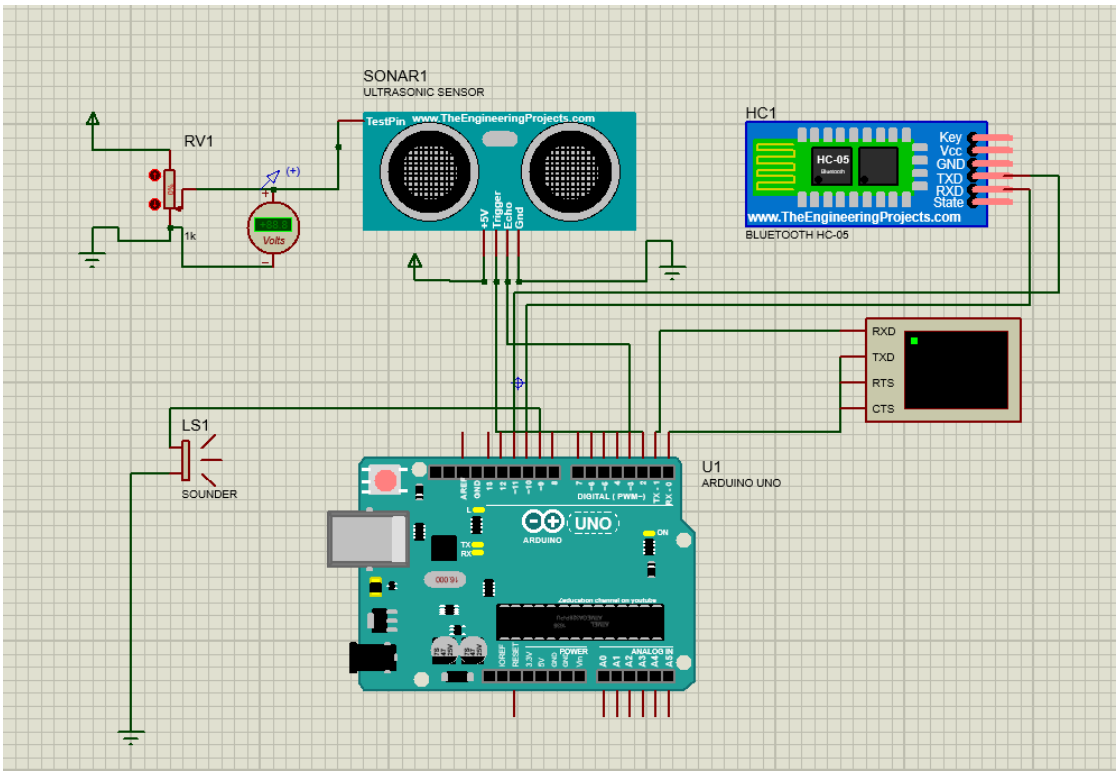

Figure 6: System Diagram

4 Simulation

I used the Proteus software program for the simulation of my project. First of all, I downloaded the necessary library files to use devices such as Arduino UNO board, Bluetooth module, buzzer, and saved them to the section where Proteus libraries are located. In this way, I was able to use the device I wanted. After the necessary connections I explained in the Implementation section were established and my code was finished, I compiled my code and extracted the compiled binary. Thus, I was able to upload my ".hex" file to my Arduino UNO board as follows.

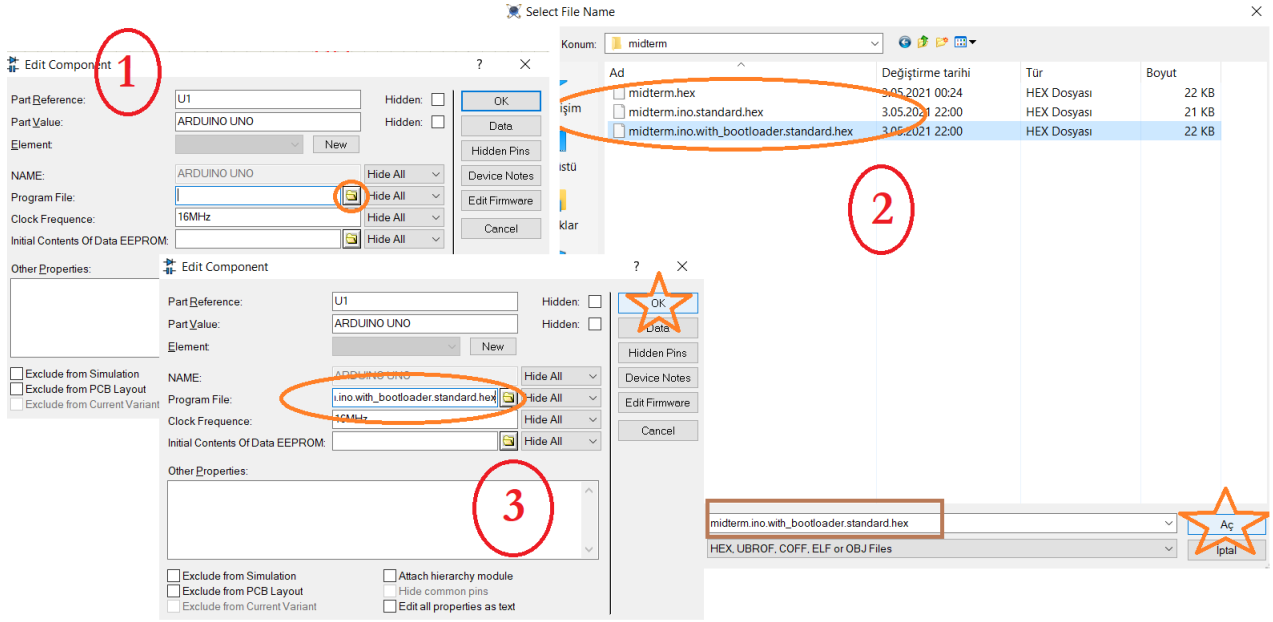

Figure 7: Arduino UNO Settings in Simulation

I downloaded the library necessary for the ultrasonic distance sensor to work and defined the path to the library in the edit component section as follows.

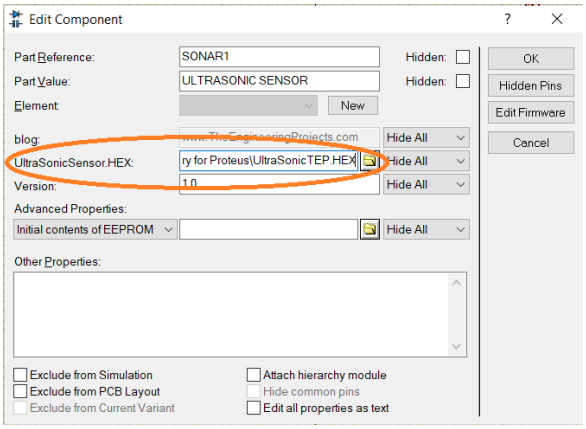

Figure 8: Ultrasonic Sensor Settings in Simulation

Finally, I made the necessary settings to control the bluetooth connection. For this, I first entered the bluetooth settings of my computer and added the COM3 serial connection and opened the edit component section of the HC-05 module in Proteus and defined the physical port as COM3. I set the virtual baud rate to 38400 to be the same as the code in Figure 5.

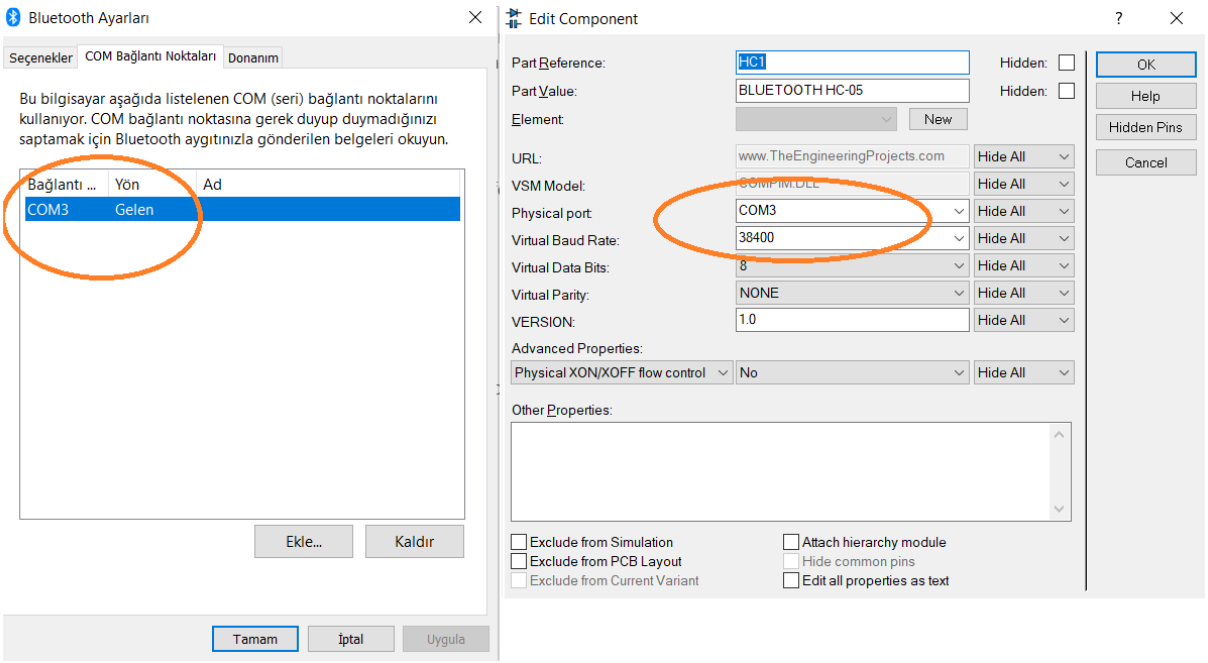

Figure 9: Bluetooth Settings in Simulation

After all these adjustments, my simulation ran smoothly. View of the simulation while running:

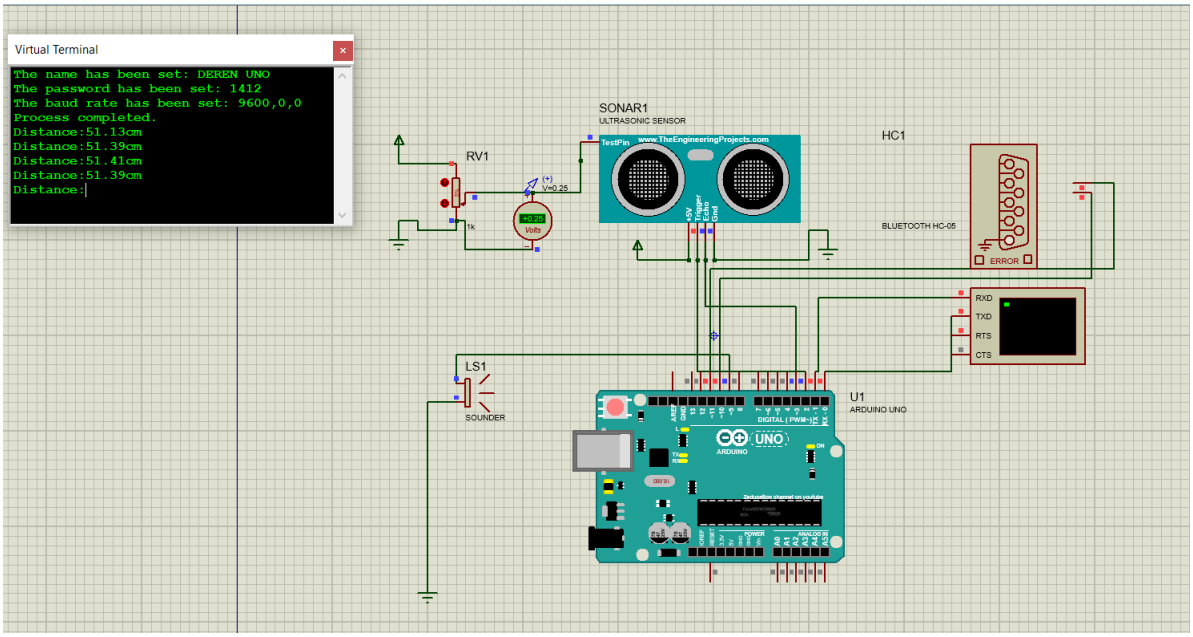

Figure 10: Simulation

You can access the simulation video by clicking here.

5 References

[1]https://www.elektrikrehberiniz.com/elektronik/potansiyometre-nedir-10639/: :text=Potansiyometre%20N
[2]https://fma.arslanalp.com/arduino-dersleri-pulsein-komutu/: :text=Alp%200%20Comments-,pulseIn()%20Komutu,darbenin%20mikrosaniye%20cinsinden%20uzunlu%C4%9Funu%20verir.