ArdAlarm

Tyler Wellington
Computer Science
SUNY Binghamton
Binghamton, NY
twellin1@binghamton.edu

Abstract

The ArdAlarm is a smart home security system that is controlled by an Android device. The feeling of safety and security is one of great value, and this product seeks to provide that with the use of an Arduino and an Android device. The ArdAlarm system itself is composed of an Arduino and several modules to detect motion, retrieve data over Bluetooth, and alert the user. The application to control the system was developed in Android Studio, and allows the user to activate, deactivate, turn off the alarm after it has been triggered, and set the desired password to turn the alarm off. This implementation provides a straightforward, and effective system for all potential users.

Keywords

ArdAlarm, Arduino, Motion Detection, Android Application, Android Studio, Bluetooth

1 Introduction

Over the past decade, there has been a surge in the development and use of IoT technology on a domestic level. It is becoming increasingly common for people to have a number of "smart devices" throughout their homes, such as smart speakers or TV's, that are integrated with the user's phone for added convenience and functionality. The ArdAlarm is a smart security system designed to be added to the growing list of household smart devices. Using an Arduino equipped with several modules, and any Android device, this product provides a security system that can be controlled entirely through an Android application via Bluetooth.

2 Design and Implementation

This section will go into detail about the hardware elements of this project, including a breakdown of all necessary Arduino modules and their purposes, as well as the software elements, including a brief discussion of the code that was written and the libraries used.

2.1 Hardware

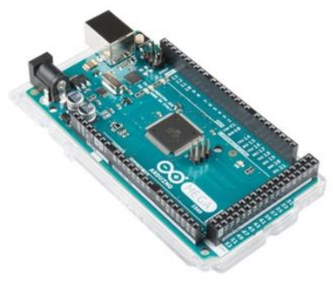

Figure 1: Arduino Mega 2560

2.1.1 Arduino Mega 2560. The primary component of the alarm system itself is an Arduino Mega 2560 (shown in Figure 1 above). This is a microcontroller board equipped with several input and output pins that allow for expansion of functionality through various modules. This board stores and runs all of the necessary code in order to function as the alarm. Once the code is written and deployed to the Arduino, any time the Arduino is powered on, it will automatically run the deployed code. Thus, there is no further setup on the user's end. Since the Arduino is preprogrammed, all it needs is power, and it will function as an alarm system. However, while the Arduino stores and runs the alarm program, it cannot serve as an alarm system on its own. It requires the following modules to work as intended (shown in Figure 2 below):

(1) Piezo Buzzer

(2) Bluetooth Module

(3) Ultrasonic Sensor

(4) I2C Display

Figure 2: Arduino Modules

1. Piezo Buzzer
2. HC-05 Bluetooth Module
3. HC-SR04 Ultrasonic Sensor
4. I2C LCD Display

The HC-05 Bluetooth module is one of the most crucial modules for this system. It allows for communication between the Arduino and an external device via a Bluetooth connection. This module is responsible for the establishing the connection between the Arduino and the Android application in order for it to receive information from the user.

The ultrasonic sensor is what acts as the trigger for the alarm. It uses sonar in order to detect distance by sending out an inaudible ultrasonic sound and converting the time it takes for that sound to bounce back into a unit of distance. Thus, by being able to calculate distance, and can detect a change in distance. This change in distance is treated as motion and will trigger the alarm.

The I2C module is an 16x2 LCD display that allows the Arduino to display custom messages. This module is used to provide information to the user in the form of various messages displayed based the status of the alarm. For example, when the system has been activated, the Arduino will display a message saying “Alarm Activated” to inform the user.

Lastly, the Piezo buzzer module serves as the actual alarm for the system. When the ultrasonic sensor detects motion the Arduino will trigger the buzzer to emit a loud sound so that anybody who passes through the sensor knows that the alarm has been set off.

2.1.2 Motorola Moto G6. This Android device used for development testing of the ArdAlarm was a Motorola Moto G6. This was used to test and run the application that works in conjunction with the Arduino itself. While this was the device used for development, any Bluetooth capable Android device with the ArdAlarm app will work with the alarm system.

2.2 Software

2.2.1 Arduino Sketch. The program deployed onto the Arduino was written and tested entirely in the Arduino IDE. It uses the following libraries that correspond to the various modules:

1. SoftwareSerial.h [2]
2. NewPing.h [3]
3. LiquidCrystal_I2C.h [4]
4. Wire.h [5]

These libraries allow the Arduino to take full advantage of the HC-05, ultrasonic sensor, and I2C display modules, respectively. They are necessary in order to implement the features of the system.

As for how the Arduino code functions, it first initializes all global variables and sets all modules to their default statuses, as they pertain to the ArdAlarm. The Arduino then begins to run its *loop()* function. This function is constantly looped through as long as the Arduino is powered on and contains all of the methods calls and other code needed the operate the security system. It works by continuously scrubbing the Bluetooth serial monitor for incoming data, interpreting said data, and responding appropriately through the program. Depending on the data that was sent, the Arduino can enter one of three states: armed, disarmed, or triggered.

In the disarmed state the Arduino simply displays a message the alarm is not activated and waits for a signal to change that. This is also the default state of the system. Once in the armed stated the ultrasonic sensor first calibrates itself by finding the current distance to the nearest object it is facing (within its maximum range of 400cm). It then continues to monitor the distance and upon seeing a significant change, will trigger the alarm. When in the triggered state, the Arduino sounds the buzzer, displays a message to the user, and starts a 30 second timer (the time frame for the user to send the password). If the user sends the correct password within the given time frame, it will put the system back into a disarmed state. Otherwise, it will alert the user through the app that they have failed to disable the alarm, and it will return the armed state.

2.2.2 Android Studio Application. The partnering ArdAlarm Android app was written and tested in the

Android Studio IDE. The app takes advantage of various tools such as Buttons to trigger certain functions, EditText objects to allow the user to input text for sending or setting a password, Bluetooth sockets, and others [1]. Upon opening the app, the device will automatically pair with the HC-05 module (assuming the device has been previously paired to the HC-05 in through the devices Bluetooth settings). It will then display the application menu as shown in Figure 3 below:

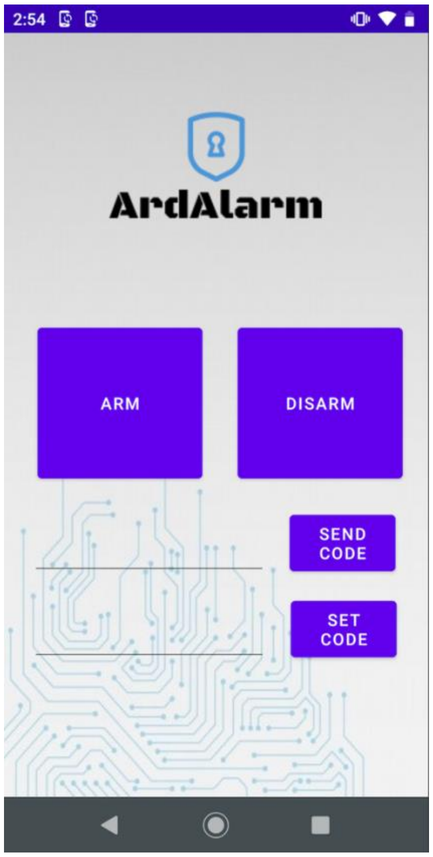

Figure 3: ArdAlarm Application

The *Arm* and *Disarm* buttons simply send a message to the Arduino signaling it to respectively activate or deactivate the alarm. The *Send Code* takes whatever text is in the input field next to it, puts it in a Bluetooth message, and sends it to the Arduino as an attempt to disable the alarm after it has been triggered. Lastly, the *Set Code* button takes the text in the input field next to it and sends it to the Arduino in order to change the current password on the Arduino to this new one. For both the *Send Code* and *Set Code* buttons, if their corresponding input field is empty, the app sends no message and displays a pop-up informing the user to input something. The interface was made to be simplistic, in order to provide the user with an easy experience while also providing them with the information and feedback that they need to work the app.

3 Evaluation

In its current state, this product is a simple to use security system using motion detection and Bluetooth as its backbone. It has the necessary functionality to serve as a working alarm system and demonstrates the potential of Bluetooth security devices. Over time, this system could improve to provide more features to the user such as managing multiple alarm systems or changing certain default settings on the Arduino such as the buzzer frequency or display messages. Features like these and others could provide the user with a more positive experience.

4 Conclusion

Developing this system gave me with a broader view of the IoT world. There are thousands of applications for it, and security is only one of them. The ArdAlarm is meant to act as an entry into the world of smart home security using IoT. The straightforward user experience of the app, along with the plug and play nature of the Arduino allows the user an easy way to get a glimpse into smart security by providing a user friendly, fully functioning system. This is exactly what I set out to accomplish with this project, and thus I am satisfied with the final product.

References

[1] https://developer.android.com/docs
[2] https://www.arduino.cc/en/Reference/softwareSerial
[3] https://playground.arduino.cc/Code/NewPing/
[4] https://www.arduino.cc/en/Reference/LiquidCrystal
[5] https://www.arduino.cc/en/reference/wire