Final Project: Keypad Security System

Group #25

William Presley and Jay Hayes

M10334151 and M10261044

Fall 2019

University of Cincinnati

Electrical Engineering and Computer Science

Abstract:

The purpose of this project is to build a keypad security system using the Microchip Curiosity Board along with any external hardware such as an ultrasonic sensor, piezoelectric speaker, LCD display, and more. We will write this program in C using all our knowledge that we have learned throughout the semester to accomplish this task. The project will be an extension on the functionalities that we had implemented in Lab 8 during the semester. Throughout this report we will discuss the materials needed for the project and what they cost, a user manual, timeline to plan out the project, and our takeaways after completing the project.

Introduction:

For our embedded design circuit, we wanted to use the keypad membrane switch and ultrasonic sensor together to detect movement and then afterwards sound off an alarm until the user enters the correct 4-digit passcode using the keypad membrane switch. We had come up with the idea before we did Lab 8 in this course, which helped have a solid grasp of using the LCD display and keypad membrane switch. The functionality will operate like an invisible tripwire that sounds off an alarm when it is crossed, except that the user can disable the alarm if they know the password and enter it. If the user enters the correct passcode to disable the alarm, we will reset the program and allow the user to set the alarm whenever they want. We will also offer the user the option to change the password as well at this point if they can provide the old passcode, initially we will set this to be 0000.

Timeline:

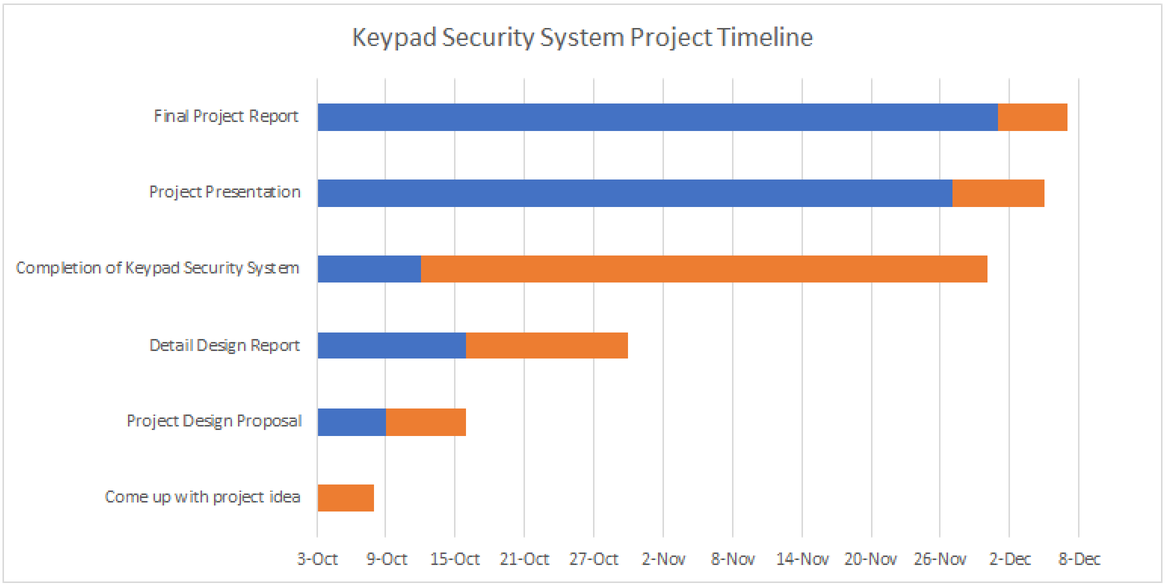

Figure 1: Timeline for Keypad Security System

Parts List and cost:

Material	Cost
LCD display	\$8.99
Keypad membrane switch (2-pack)	\$6.45
Ultrasonic sensor	\$3.95
Piezoelectric buzzer	\$5.80
Microchip Curiosity development board	\$20.00
Arduino Uno	\$18.50
10 kΩ resistors (4)	\$0.28
Total Cost	\$63.97

Figure 2: Materials and cost for Keypad Security System

Detailed design:

The LCD screen will be used as a UI element, displaying information for the user. The keypad membrane switch will be used for user input. The ultrasonic sensor will be used to sense motion to set off the alarm. The alarm will be the buzzer and displayed by the screen.

Figure 8 is a block diagram of the hardware and displays how it will connect to the curiosity board.

Figure 7 is a flowchart to display the sequence the software will follow to satisfy a proper security system.

Circuit diagrams:

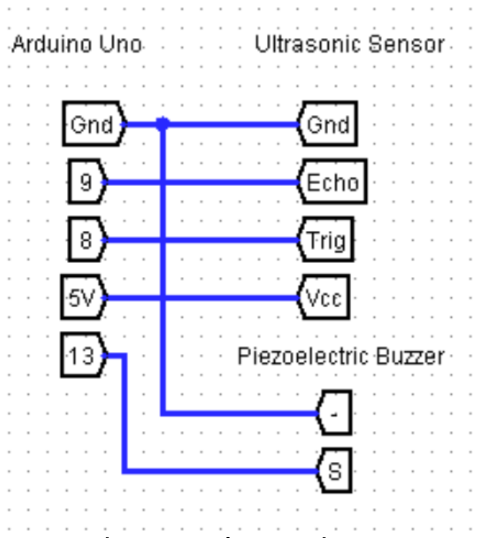

Figure 3: Arduino Uno connections to ultrasonic sensor and piezoelectric buzzer

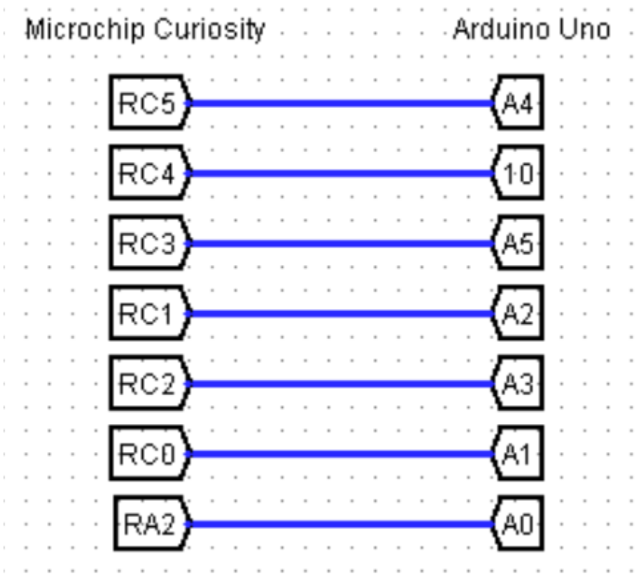

Figure 4: Microchip Curiosity connections to the Arduino Uno

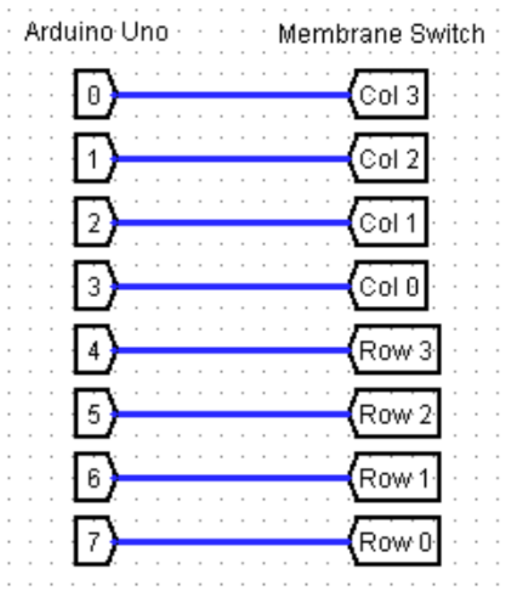

Figure 5: Arduino Uno connections to membrane switch

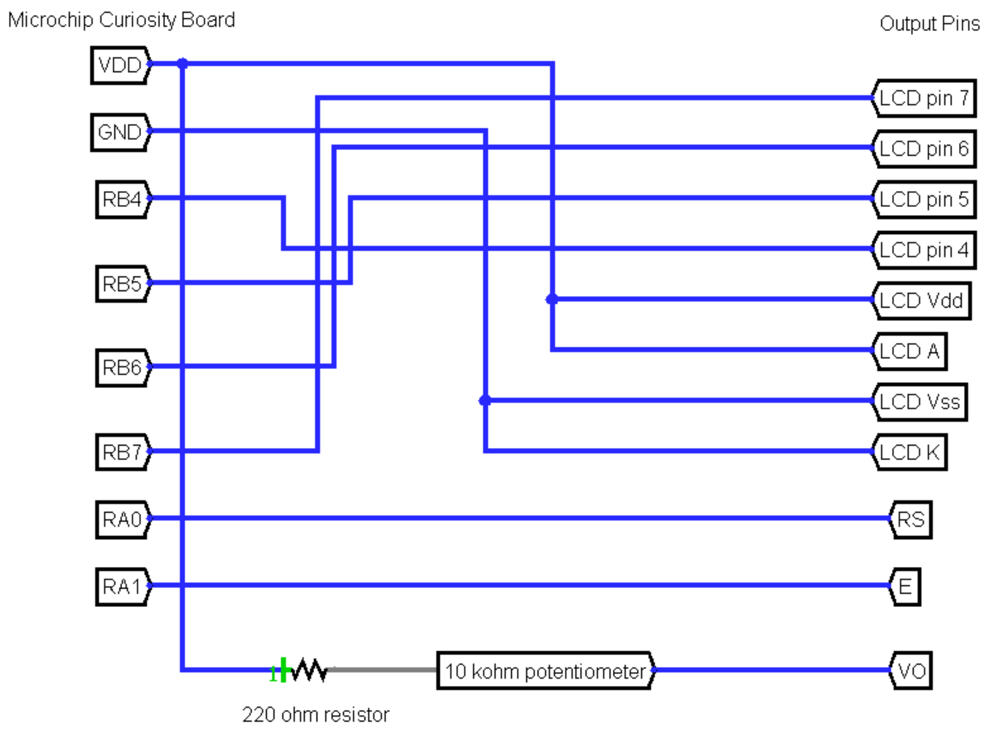

Figure 6: Connections from Microchip Curiosity to LCD display

Flow chart:

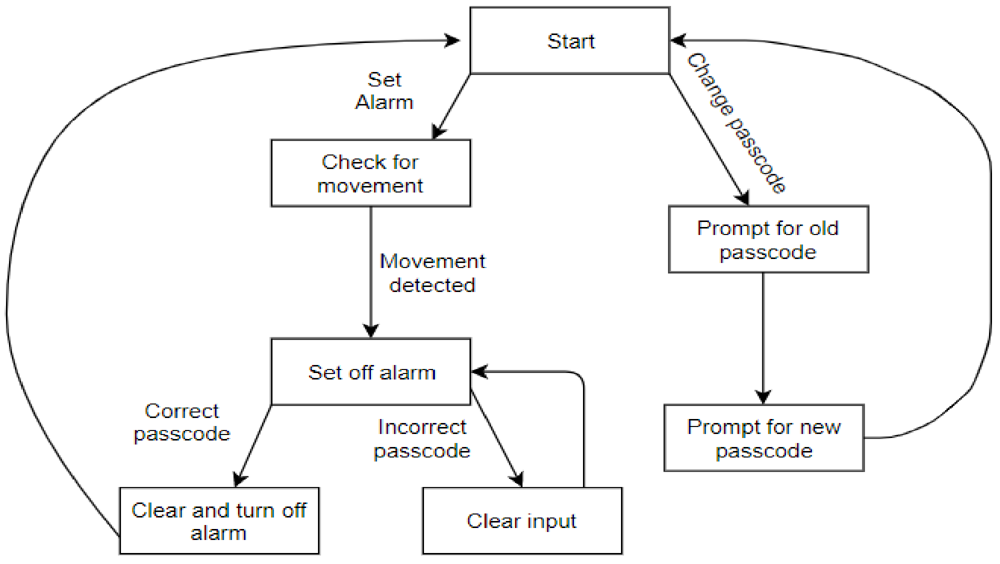

Figure 7: Flow chart for the Keypad Security System

Block diagrams:

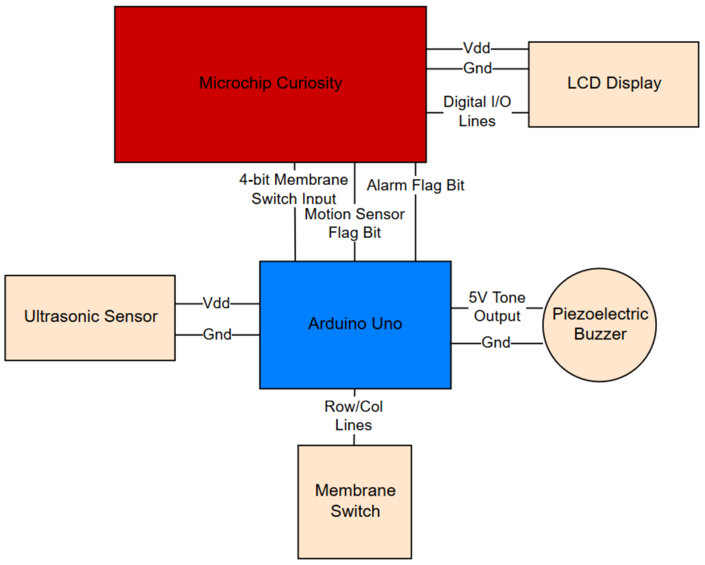

Figure 8: Block diagram of our final circuit

Final circuit:

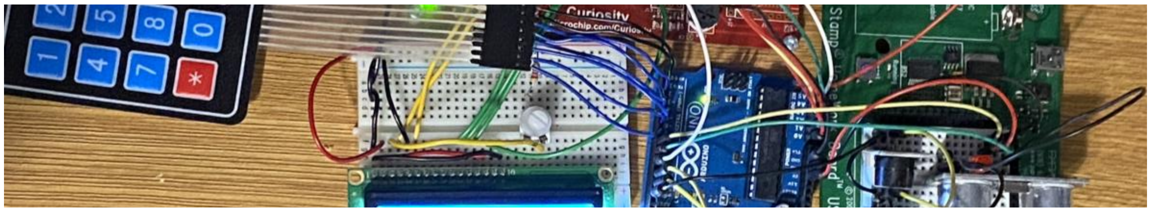
Figure 9: Final circuit setup for the Keypad Security System

User's manual:

Setup

1. Ensure that any wiring has not come loose.
    a. Check that the buzzers, ultrasonic sensor, and membrane switch are connected to the Arduino Uno.
    b. Check that the LCD is connected to the Microchip Curiosity.
    c. Check that wires connecting the Arduino Uno to the Microchip Curiosity are not loose.
2. Connect 9V battery to the Arduino Uno.
3. Power on the Microchip Curiosity with battery or through a USB connection.

Operating Instructions

To interface with the Keypad Security System, the membrane switch acts as the keypad input from the user and the LCD acts as the output interface to the user.

Once powered on, the system boots to the main screen which displays options for the user to set the alarm or change the password.

Pressing A will start a five-second countdown before the alarm is set and then any motion detected after it is set will trigger the alarm. If the disarm key (displayed on the LCD) is pressed, the user gets one opportunity to enter the correct passcode or else the alarm will sound. If the password is entered correctly, then the system is disarmed. Otherwise if motion is detected, then the alarm will sound and prompt the user for the passcode until it's entered correctly.

Pressing B on the main screen will start the process to change the password for the system. Initially, the password is set to '0000'. To complete a password change, the user is prompted for the current password, the new password, and then a confirmation of the new password. If the current password is entered incorrectly, the user is re-prompted for the current password. If the new passwords do not match, the user is asked to restart the password change process.

Design Decisions/Alternative designs:

The Keypad Security System utilizes a 16x2 LCD screen, keypad membrane switch, ultrasonic sensor, piezoelectric buzzer, Microchip Curiosity board, and Arduino Uno board. Since Arduino has existing libraries, it was more efficient to connect peripheral equipment to the Uno.
As seen in Figure 8, the membrane switch, ultrasonic sensor and piezoelectric buzzer are connected to the Uno. The Curiosity is connected to the Uno and the 16x2 LCD display.

By connecting the membrane switch to the Uno, we can save hardware resources and pins on the Microchip Curiosity. When the membrane switch is connected to the Uno, it does not require the same pull-up resistors that it does when connected directly to the Curiosity. Additionally, it reduces the pins used on the Curiosity from 8 to 4 by sending a 4-bit binary value from the Uno to the Curiosity corresponding to the character pressed on the membrane switch.
By connecting the ultrasonic sensor to the Uno, one pin was saved on the Curiosity, but more importantly, the Curiosity board did not have to dedicate any processing time to set and monitor the data pins, and instead was mainly tasked with handling UI and maintaining responsiveness.
By connecting the buzzer to the Uno, less code was necessary since Arduino has libraries for tone/frequency generation. The tone function was the only necessary line of code for the noise generation, where if connected to the Curiosity, code would have been needed to handle changing and timing of the output pin to the buzzer to create the frequency manually.

To keep as much UI with the Curiosity as possible, the 16x2 LCD is connected to the Curiosity. Using 4-bit mode to preserve pins, the LCD uses the library that was used in Lab 8.
The Curiosity was the main board that handled UI and all the I/O from the Uno and peripherals. The code that looped and handled the functionality of the system was on the Curiosity.

Most of the alternate designs that were tried were too inefficient on pins or required more pins than were available on the Curiosity. Since Arduino has libraries for all the peripherals, it was better to connect the sensors to the Uno and then send data from the Uno to the Curiosity to save pins. The only downside to this is the use of the additional board.

Conclusion:

For our final project we made a keypad security system using the Microchip Curiosity Board. The project would prompt the user initially to either “create a new passcode” or “set the alarm.” If the user sets the alarm, then the alarm would set itself after 5 seconds so that the user can get away from it. Once 5 seconds has passed, we use the ultrasonic sensor to detect if anyone comes near the sensor. If the sensor detects movement it will sound off an alarm coming from the piezoelectric buzzer until the user is able to successfully enter the correct passcode. We did not run into any issues with regards to the software aspect of this project but did face some with the hardware. We did not have enough pins available on the Microchip Curiosity Board for all our components which is why we needed to have the Arduino Uno kit so that we could extend the number of inputs that we could use. When we first started implementing the code, we did not get any readings on our LCD initially even when we were manually writing straight to it. We eventually found that the problem was our potentiometer was not working and thus we could not adjust the contrast on the display, which is why we could not see anything on the display. If we had more time to work on the project one thing that we would have wanted to try is use our system along with a mechanical mechanism, like opening a garage door or the front door to your house. We would have wanted to take what we created for our project and attach it to the front door of someone’s house and instead of activating the sensor when the user crosses the motion sensor, we set it off if they try to open the door with the wrong password. After completing this project, we feel comfortable completing any projects moving forward written in C and using any of the mid-range family microcontrollers.