CSE321 Project 3
Security System Documentation

Miguel Bautista, Senior
Fall 2021 CSE321
University at Buffalo
Department of Computer Science and Engineering
28 November 2021

TABLE OF CONTENTS

1 PROJECT OVERVIEW

This system is a traditional security alarm system which will be used to contribute to the public interest of safety by providing consumers with the peace of mind of a safer area. This is achieved by using an ultrasonic sensor to keep track of nearby objects and a microphone to detect loud noises. If these are detected, the system will activate and notify the user that something has been detected. The user can interact with the system by using a keypad to arm/disarm the system through the LCD interface.

The Nucleo-L4R5ZI will be the microcontroller used to control this system and achieve the specified design constraints and specifications.

2 PROJECT REQUIREMENTS

The system must be capable of the following functionality:

- Define an initial security code when first powered on
- Arm the system or disarm the system by pressing "A" (After security code is entered)
- Display the system status when idling (No button has been pressed in 10 seconds)
- Trigger the armed security system when an active sensor has been tripped
- Notify system owner which sensor has been tripped upon security system trigger
- At all times, show which sensors are activated
- Must run "forever"

3 PROJECT SPECIFICATIONS

The system is subject to the following constraints:

- The active alarm sensors shall be a microphone sensor and ultra-sonic range sensor
- The active sensor status must be shown on individual LEDs
- The ultrasonic sensor shall not be activated by stationary distant objects
- The microphone sensor shall not be triggered by ambient sound
- When triggered, an active buzzer shall activate, and LEDs must flash
- The system shall take input from a 4x4 keypad matrix

4 WATCHDOG

The watchdog is an important hardware specific feature which acts as a system failsafe in the event something goes wrong in the software.

In this system, the watchdog timer is set to reset the system if the timer is not reset every five seconds. The thread which handles the powering of the keypad rows oversees resetting the timer after switching each row. Since the row power should be cycling every few milliseconds, it can be concluded that if the watchdog is not reset within the allotted time that the system is not operating correctly and should be reset. This means that if the system encounters some unknown or unintended operational state, there is still a way for the system to recover.

5 SYNCHRONIZATION TECHNIQUE

The system utilizes threads to maintain separate parts of the system. To maintain synchronization between these treads and ensure data is not accessed by different parts of the system concurrently, mutexes are used.

There is one mutex which protects key system data. Before any threads access system data – such as the current system mode or the key press status – the thread must wait to lock the mutex and then unlock it once its task is completed. This ensures that unintended data access does not corrupt the program if it were changed while a thread is still utilizing that resource.

6 BITWISE DRIVER CONFIGURATION

While MBed OS provides classes which abstracts away complex driver configurations for interfaces such as interrupts, the four pins associated with the rows of the 4x4 matrix keypad are controlled using manual bitwise configuration.

Located within the sub-module, stm_methods, bitwise driver configuration functions are manually defined to enable the RCC, set the mode of a pin to input or output, and to write a digital state to a pin.

7 THREADS

The system uses two main threads to control the keypad and then to control all other system functions. This was implemented this way for the system to always be cycling through the keypad row power and allow for continuous input. More specifically, the keypad will not be interrupted during longer system operations such as waiting for the ultrasonic sensor, blinking the LEDs, or longer LCD statements.

8 INTERRUPTS

The system utilizes interrupts to detect when either the ultrasonic sensor or the microphone has been tripped and when a button has been pressed on the 4x4 matrix keypad. This is done to eliminate the need to constantly poll the status of those pins to see if they have been activated.

The interrupt gives the system the ability to carry on its tasks and then jump to the interrupt handler if the hardware has detected a high input. More specifically, each ISR sets the appropriate flag to indicate a button or sensor has been activated and is then handled based on the current system mode.

9 SOLUTION DEVELOPMENT

After deciding to create a system that contributes to the consumer safety domain, research was done to figure out what would be the most feasible product that could help many people at a cost-effective price.

After deciding to work on a security system, many different sensors were considered for the system. Aside from the ultrasonic sensor and microphone which was present in the final implementation, motion sensors and cameras were also considered. The reason the former sensors were chosen for the final product was due to their simplicity to use, cost efficiency, and versatility if additional functionality was added in the future. For example, when designing the solution, additional features, such as microphone passthrough and distance tracking, were also considered.

Once the scope of the project was finalized, many different methods were tested using the MBed OS based on their versatility and performance. After testing out using, queues, timeouts, interrupts, threads, and tickers, the most optimal solution seemed to be a combination of all these design languages.

More specifically, the threads handle the powering of the rows and all other system functions, interrupts are used to react immediately to sensors being triggered, queues are used to run non-blocking code from the ISR, timeouts are used to determine system idling and the ultrasonic sensor distance checking, and tickers are used to trigger the ultrasonic sensor measurement.

These design choices were used – after many test runs of other designs – because they met the efficiency requirements necessary which meet the timing constraints set forth by many sections of the system design such as the ultrasonic sensor and the keypad power.

10 BLOCK DIAGRAM

The following block diagram is a simplified visualization of the security system which shows the 4x4 matrix keypad input, ultrasonic sensor, microphone, the 1602 LCD output, the LED outputs, and the microcontroller used to control the inputs and outputs.

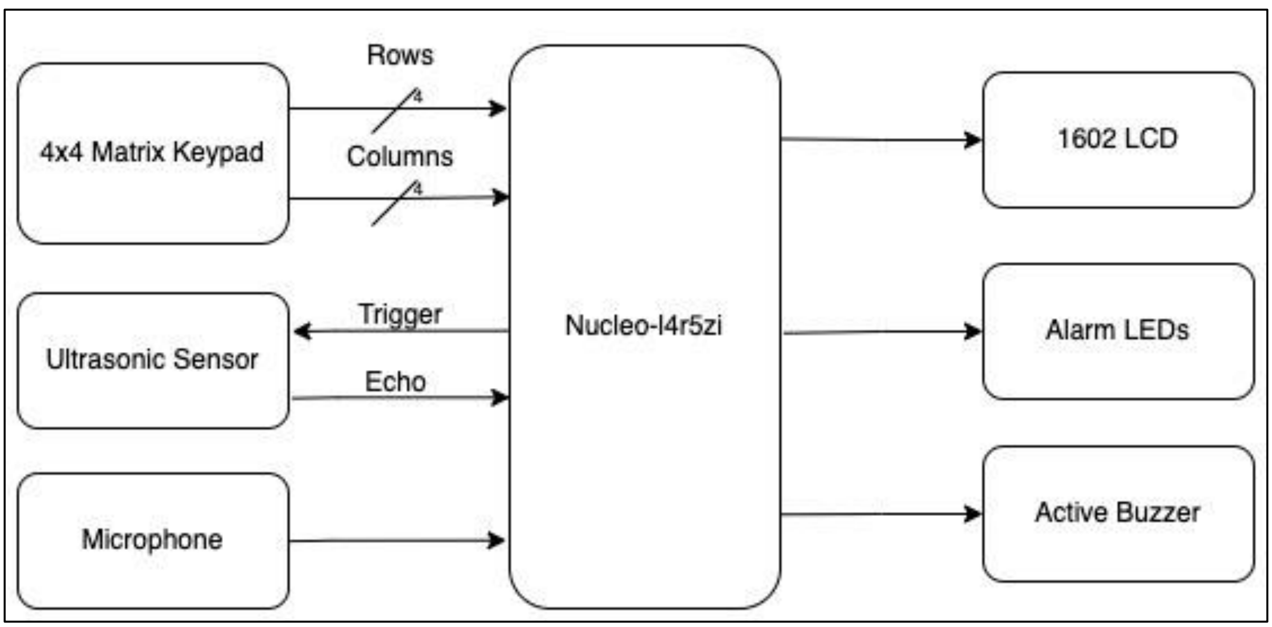

Figure 1.0: *Security System Block Diagram*

11 FUNCTIONALITY DIAGRAM

The following flowchart depicts the operation of the system:

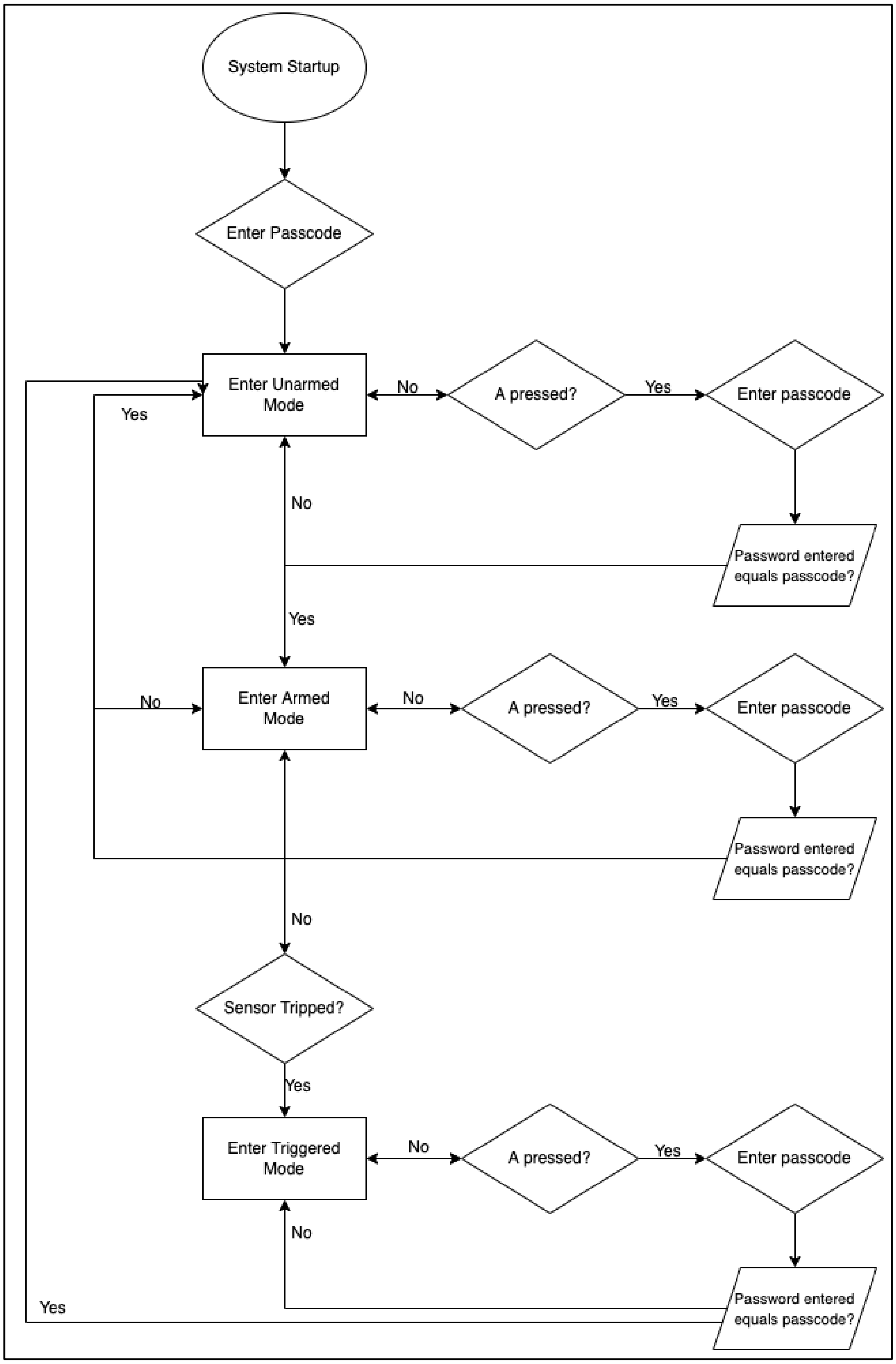

Figure 2.0: *Security System Flowchart*

12 BILL OF MATERIALS

The following materials are needed for this system:

- 1602 LCD with I2C chip: Used to display the UI
- Nucleo-L4R5ZI: This microcontroller will control the system logic
- 4x4 Matrix Keypad: System input device the user will control
- Solderless Breadboard: Provides an easy way to connect the parts together
- Jumper wires: Provides connections between the pieces
- LEDs: Will be used as outputs to indicate various events such as alarms or inputs
- Microphone sensor: Used to detect nearby sound
- HC-SR04 Ultrasonic Sensor: Used to detect nearby motion
- Active Buzzer: Used as the alarm sound of the triggered system

13 USER INSTRUCTIONS

The schematic in conjunction with the build and use instructions serve to provide an in-depth tutorial on how to successfully assemble and operate this system.

I. SCHEMATIC

The following is a schematic which provides a more in-depth view of the electrical setup of the system:

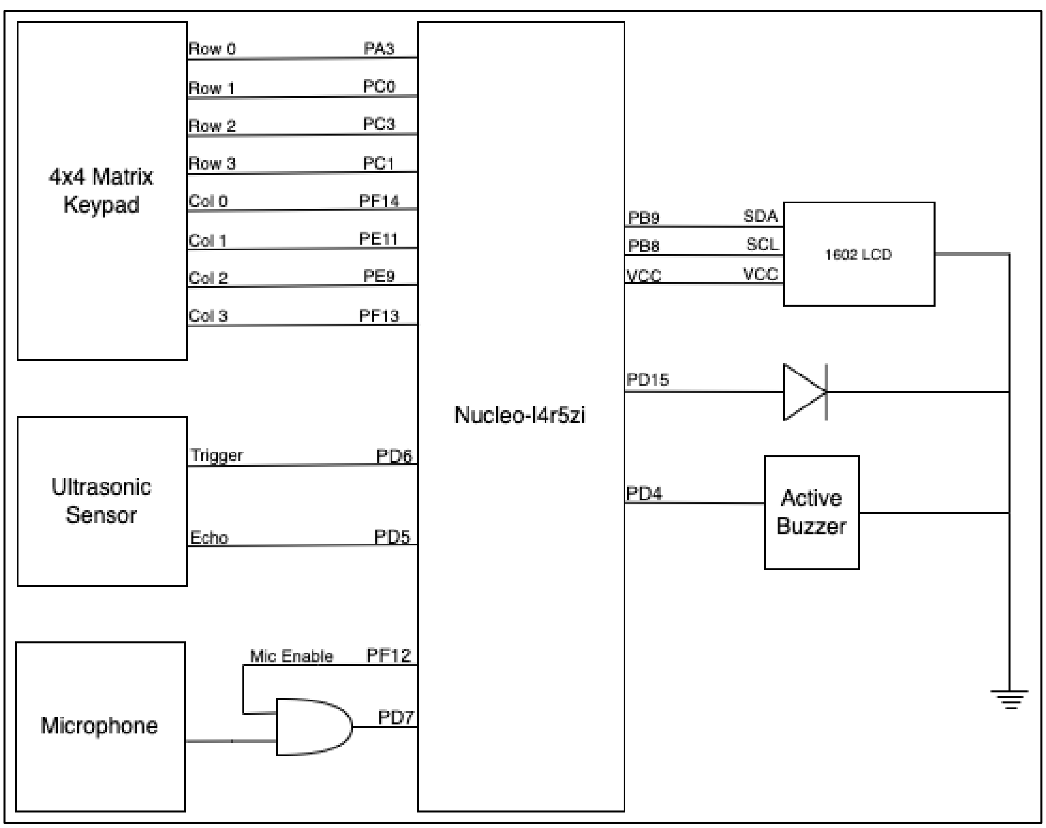

Figure 3.0: *Security System Schematic*

II. INSTRUCTIONS TO BUILD SYSTEM

To build the system, following these instructions:

1. Connect the 4x4 matrix keypad to the Nucleo
    a. With the keypad facing upwards, connect the keypad starting from left to right
    b. Going from left to right, the pins should be connected to PA3, PC0, PC3, PC1, PF14, PE11, PE9, PF13
        i. Visually, there should be 4 wires on the left of the Nucleo – corresponding to the keypad rows – and then 4 wires on the right of the Nucleo – corresponding to the keypad columns.
2. Connect VCC and Ground to the red and blue lines of the breadboard respectively
3. Place the ultrasonic sensor on the breadboard
    a. Connect to VCC and ground
    b. Connect the Trigger pin to PD6 of the Nucleo
    c. Connect the Echo pin to PD5 of the Nucleo
4. Place an and gate on the breadboard
    a. Connect to VCC and ground
5. Place the microphone on the breadboard
    a. Connect to VCC and ground
    b. Connect the Digital Output pin to the first And gate input
    c. Connect PF12 on the Nucleo to the second And gate input
    d. Connect the And gate output to PD7
6. Adjust the microphone sensitivity by turning the potentiometer clockwise or anti-clockwise so that the first LED only blinks for loud noises and doesn't activate for ambient noise.
7. Place the 1602 LCD on the breadboard
    a. Connect to VCC and ground
    b. Connect the SDA pin to PB9
    c. Connect the SCL pin to PB8
8. Place LEDs on the board in parallel
    a. Connect to ground
    b. Connect the positive side of the LEDs to PD15 of the Nucleo
9. Place the active buzzer on the breadboard
    a. Connect to ground
    b. Connect the positive side to PD4

III. INSTRUCTIONS TO USE SYSTEM

Once initializing the Nucelo, define the system passcode by typing in four digits with the 4x4 matrix keypad. After setting the passcode, the system can be switched between unarmed and armed mode by pressing A on the matrix keypad and then entering in the previously defined passcode.

If the alarm is activated after placing the system in armed mode, it can be unarmed by pressing A on the matrix keypad and typing in the previously defined passcode.

To ensure the sensors have been setup correctly, there are LEDs that blink every time the ultrasonic sensor activates, and the microphone has an LED constantly on when listening. This should be verified immediately after setting up the system.

17 TEST PLAN

To test the functionality of the system, the functionality will be checked to ensure every point from the specification section is satisfied.

After programming the Nucleo, it was verified that the system is initialized correctly and runs forever. This was done by setting the initial password, changing between armed and unarmed modes, triggering each sensor, and letting the system run for a long time to ensure all edge cases were covered and that functionality does not break during system usage.

18 OUTCOME

After completing the previous steps, the system functionality has been successfully verified and implemented as per the described system specification subjected to the specified features and constrains listed.

19 FUTURE CONSIDERATIONS

There are several features that can be added in the future to improve the usability of the system. Namely, adding in switches to each active sensor to allow for manually user enabling and disabling will increase privacy in conjunction with the LEDs used to display whether the sensors are active. The importance of such a feature is steadily growing in demand as more consumers are becoming increasingly aware of companies tracking and collecting user data.

Another useful addition to the system would be to create an audio passthrough feature utilizing the on-board microphone. By doing this, consumers can listen to the audio that potentially tripped the system or just to hear what is occurring in the system vicinity. This feature is very similar to what is currently available in Ring Doorbells and similar technologies and would greatly increase the versatility and protection the system offers.

Lastly, a ROM should be added to the system to record the passcode that the system is initialized to. This means that in the event of a power outage, the system cannot just be taken or disabled. In other words, you must know the code to make changes, even if it is disconnected from a power source. This is a preventative measure in the event of worst-case system scenarios – this inclusion would undoubtable provide extra safety and peace of mind for the end user even if it is unlikely to happen.