– ECE 09.342: Final Project –

SAIF: Security at its Finest

David Sheppard
Rowan University

December 18, 2018

1 Design Overview

One of the foremost concerns associated with owning a home is security. While traditional security systems have been connected to phone lines to alert first responders in the event of an intrusion, recent home security systems have begun to implement principles of smart devices. The SAIF system is an affordable motion sensing device that can be implemented as a smart security system. The SAIF system is designed to detect motion and alert the homeowner of an intrusion. Thanks to a series of buttons on the device, the system can stopped by inputting the correct combination if it is accidentally triggered.

The system is a very passive one except when motion is detected. When motion is detected, the user has 10 seconds to enter the correct preset combination on the three buttons. If the combination is incorrect or the 10 second timer overflows, then a buzzer is sounded and an alert is sent over UART to a receiving device. That device can be a PC or a WiFi module which could publish warnings via MQTT. The WiFi implementation is not discussed in detail here, as that was not fully tested and implemented.

1.1 Design Features

The major features are as follows:

- Easy to use
- Adaptable
- Affordable
- Compact
- Versatile

1.2 Featured Applications

- Home security
- Occupancy sensing
- Room vacancy sensing

1.3 Design Resources

The entire code and design file repository can be found at: `https://github.com/RU09342-F18/intro-to-embedded-final-project-saif`

1.4 Block Diagram

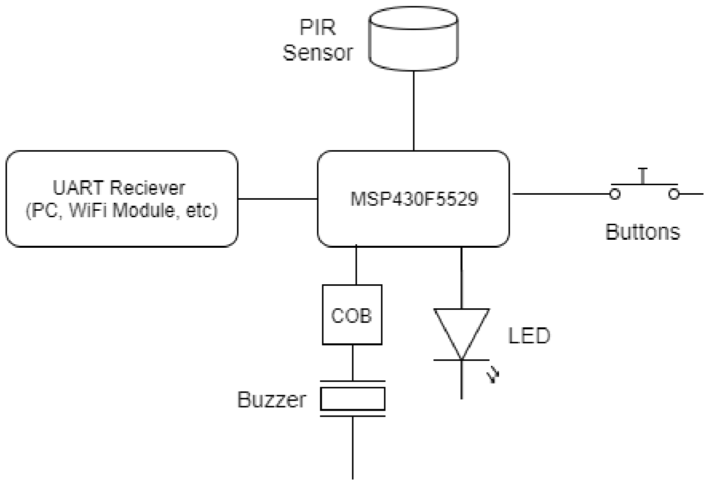
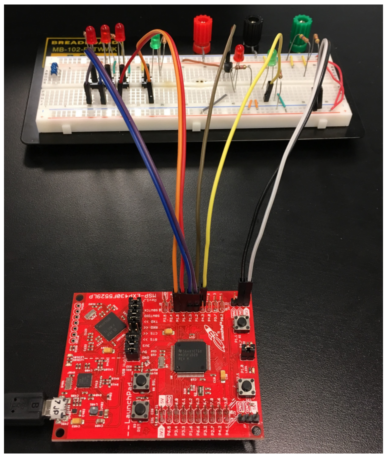


Figure 1: Basic System Block Diagram

1.5 Board Images


(a) Complete setup

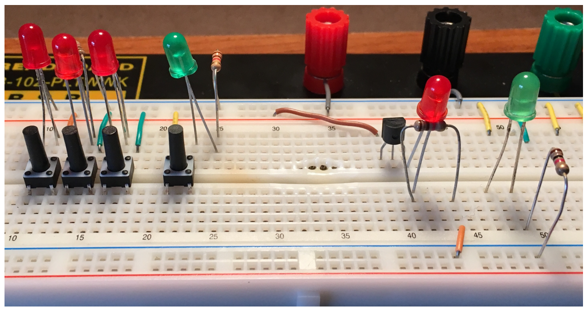
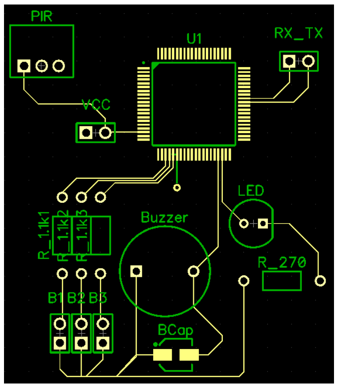
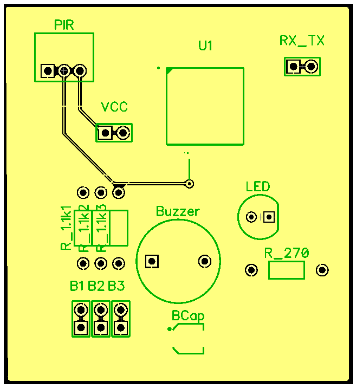

(b) Detailed View of Breadboard

Figure 2: Images of the SAIF Setup

Due to time limitations, a PCB could not be printed. As a result, the demo was implemented on a breadbaord as seen in Fig. 2. A PCB was designed to implement the SAIF system with an isolated MSP430F5529 chip (no launchpad needed). Such an implementation would allow for a much more compact design, although time did not allow for the board to be printed for the demo. Images of the PCB with and without the ground plane visible are shown in Fig. 3.



(a) PCB Design with Ground Plane Hidden



(b) PCB with Ground Plane Visible

Figure 3: PCB Design with the Isolated MSP430F5529 Chip

2 Key System Specifications

PARAMETER	SPECIFICATIONS	DETAILS
Source Voltage	3.3 VDC	Microprocessor and other elements all run at 3.3 V
Baud Rate	115200	Used for serial communication
Max current per pin	6 mA	
Max total current output	48 mA	

3 System Description

The SAIF system offers a simple solution to those needing an affordable motion detector. The SAIF system is primarily designed for use in security systems, but can be adapted to occupancy and strict motion sensor applications as well. The SAIF system is affordable with a cost in the range of $20-30 for everything necessary for a functional system. It is also very energy efficient, operating with very little current drawn when in standby mode.

3.1 Detailed Block Diagram

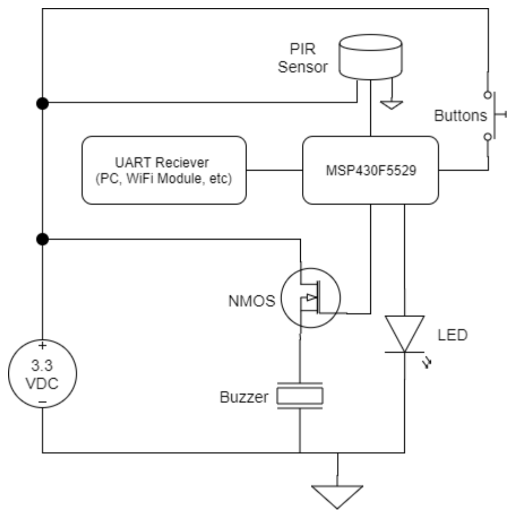


Figure 4: Detailed System Block Diagram

3.2 Highlighted Devices

Overall, the system is rather simple. The MSP430F5529 launchpad acts as the main control and the PIR sensor acts as the primary input. The buzzer is used to sound an alarm when necessary. The resistors limit current and the capacitors help to limit noise. Lastly, the NMOS allows the microprocessor to power the buzzer. The list of components is given below.

- MSP430F5529LP
- PIR Sensor (ZRE200GE)
- Buzzer (PS1240P02BT)
- NMOS (2N7000)
- 1.1 kΩ Resistor (x4)
- 270 Ω Resistor
- 1 nF bulk Capacitor

3.3 PIR Sensor

The PIR sensor is the primary input device for the SAIF system. The PIR contains two small infrared sensors that can detect infrared waves nearby. As a person walks by, the infrared radiation is detected by one sensor first and then the other. The infrared readings change quickly when this happens. This allows the sensor to detect motion easily. The sensor has three terminals: power, ground, and output. The output is registered as high when motion is detected and is low otherwise. The output pin of the PIR sensor is connected as an input to the microprocessor. The PIR sensor is capable of receiving power from the microprocessor since it draws very little current.

3.4 Buzzer with COB

The buzzer acts as the auditory output of the device. When motion is detected and a correct password is not provided, the buzzer sounds for 10 seconds as a warning while the UART data is being sent. In order to drive enough current for the buzzer, a convert-o-box (COB) was used to control the buzzer. The COB consisted of a simple NMOS device connected to power, the buzzer, and the microprocessor. When the microprocessor's buzzer control pin goes high, the buzzer is set off until the pins goes low again.

4 SYSTEM DESIGN THEORY

Overall, the SAIF system is simultaneously simple and complex. The user's operation is quite simple and the setup is easy to understand. Nonetheless, the device has a complex design to help reduce interference from noise and the make the system durable.

4.1 Hardware/Software Trade-off: Button Debouncing

One of the most prominent challenges that arises when using button-based user input is the need to debounce the buttons. When a button is pressed, it creates some vibration that can causes its output to toggle between high and low for a short period of time before settling on the correct value. This can cause a problem for the microprocessor, as the oscillating values can cause multiple interrupts to trigger, making the software believe that the button has been pressed and released more than once. While debounced buttons can be purchased, this is an unnecessary cost that can be easily eliminated with proper design techniques.

The common hardware method for debouncing buttons involves placing a capacitor across the button to eliminate the quick oscillations experienced when a button is pressed. While this is the method that was first attempted with the device, the results were consistent enough. For a security system that heavily depends on proper user input, the buttons must be debounced very accurately. After having some unsatisfactory results with the hardware debouncing, software debouncing was attempted. This involves placing a delay in the code such that the microprocessor will not accept any input for a certain period of time. It was ultimately decided that a delay of about $\frac{1}{3}$ of a second would be used, because this gave the best results during trial and error testing. When this method was implemented, every button press registered correctly with the code. Additionally, the use of software to debounce the buttons helped to decrease cost, as capacitors were no longer needed for this task.

4.2 Power Supply Constraints and COB Implementation

One of the main restraints that I decided to implement was the idea of limiting the SAIF system to only one power source. This creates simplicity and well as lowering the overall cost of the device. During the design process, a PIR sensor and a buzzer were found that were capable of running off of the same 3.3 V as the microprocessor. Nonetheless, the buzzer should be operated at over 6 mA, which is the current limit for the microprocessor. As a result, a COB had to be implemented to drive the buzzer. The COB simply consists of an NMOS device with the source connected to the anode of the LED, the drain connected to the 3.3 V supply, and the gate connected to the microprocessor's output pin designated for controlling the buzzer. With this implementation, the buzzer sounds when the pin is driven high and turns off when the pin is driven low.

5 Getting Started/How to use the device

5.1 Triggering and Stopping the Alarm

When the device is powered on, it will begin to look for motion with the PIR sensor. Once motion is detected, the internal timer begins to count to 10 seconds. During this timer period, the user can enter a 5-digit combination on the 3-button keypad to stop the alarm. If the combination is correct, the green status LED will light up and the device will go back into standby mode to resume normal operations. If the combination is incorrect or the 10 second countdown overflows, then the buzzer will be activated for 10 seconds and an alert will be send over the UART serial communication channel. The device will then return to standby mode after the buzzer has turned off.

5.2 Serial Communication

The SAIF system uses the UART serial communication protocol. This can be immediately used with a PC running a terminal such as Realterm to read the data, or could be implemented with a WiFI module to send and receive alerts via MQTT or another communication method. When the device is triggered, it sends the ASCII values for the message ”Intruder” over the serial communication line. This can be read in on Realterm provided that the correct settings are chosen. This process is further explained in section 6.2.

6 Getting Started Software/Firmware

The entire code for the SAIF system is written in C using Code Composer Studio version 8.1. Code Composer studio is needed for loading the software to the device or for making changes to the code. The code is dependent upon the msp430f5529.h header file in order to use the libraries built by Texas Instruments. A hierarchy view of the entire system and its procedures can be seen in Fig. 5.

6.1 Hierarchy Chart

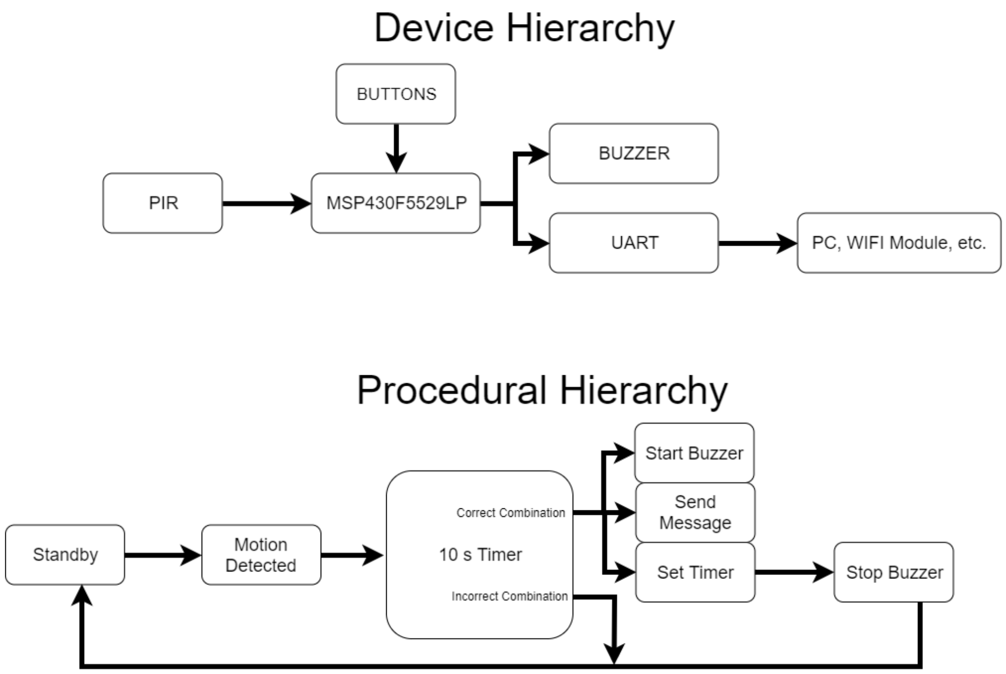


Figure 5: Hierarchy Chart of Inputs

6.2 Communicating with the Device

The SAIF system uses UART serial communication to provide status updates when an intrusion is detected. To read this information on a PC, Realterm (or a comparable terminal program) must be installed. The terminal must be set to receive information from the correct COM port. To determine the correct COM port, open the computer's device manager and look for the category ”Ports (COM & LPT).” Click the dropdown menu to view all ports and find the one that says ”MSP Application UART1 (COMX),” where ”X” is the COM port number. This is the number that must be entered in Realterm under the Port tab. The baud rate must also be set to 115200 before clicking ”Change” and ”Open” on the terminal. Additionally, the display setting under the ”Display” tab should be set to ”Display as ASCII” in order to properly read in the information. Once this is set up, the terminal is ready to receive messages from the SAIF system.

6.3 Device Specific Information

This specific device is designed to run at a 115200 baud rate for serial communication. It is also important to note that the microprocessor should never be used to output more that 6 mA per pin or more that 48 mA total. Not heeding these requirements could potentially cause damage to the microprocessor as well as system malfunctions.

7 Test Setup

To set up the device for testing, a number of connections need to be made. The microprocessor connections are shown in Fig. 8.2. In addition to these connections, the device needs a 3.3 VDC power connection.

Table 1: 3

Pin Number	Circuit Element
1.2	Button 1
1.3	Button 2
1.4	Button 3
1.5	Buzzer COB gate
3.0	PIR output
3.1	Status LED

The device was tested by first performing expected tasks. The PIR sensor was triggered and then an incorrect or correct combination was entered. Additionally, unexpected actions were performed such as entering combinations when the device has not been triggered as well as trying to trigger the device while the buzzer was sounding. This helped to test the usefulness of the device as well as its response to unexpected conditions.

7.1 Test Results

During testing, the device performed well. After the software debouncing was used to replace the hardware debouncing, the button presses registered correctly with the device. Correct and incorrect combinations were tested thoroughly and the results confirmed the accuracy of the code. Each time an incorrect combination was entered, the buzzer sounded for 10 seconds and no inputs were accepted until the buzzer stopped. Additionally, it was ensured that pressing the combination buttons before the PIR triggered did not in fact cause any unintended consequences.

8 Design Files

8.1 Schematics

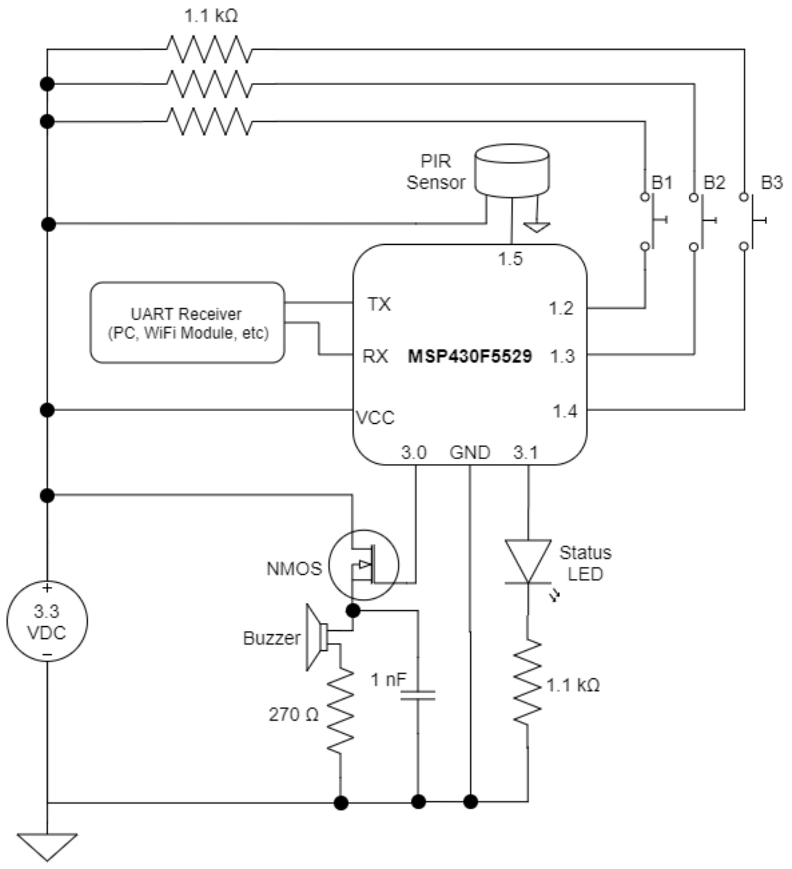


Figure 6: System Schematic

8.2 Bill of Materials

Table 2: 3

Item	Cost
MSP430F5529LP	\$12.99
PIR Sensor (ZRE200GE)	\$2.31
Buzzer (PS1240P02BT)	\$0.67
1.1 kΩ Resistor (x4)	\$0.04
270 Ω Resistor	\$0.01
1 nF bulk Capacitors	\$0.40
Breadboard	\$5.00
Total	**\$21.42**

8.3 Diptrace Project

The DipTrace PCB files can be found at `https://github.com/RU09342-F18/intro-to-embedded-final-project-saif/tree/master/PCB_Files`and the images of the design can be see in Fig. 3.