– ECE 09.321: Final Project –

Fingerprint Sensor Security

Kyle Rettig and Eric Kryzanekas
Rowan University

December 22, 2018

1 Design Overview

The purpose of this design was to use more than one sensor to communicate with the MSP430F5529LP in order to implement a proof of concept for a Security System. This was done utilizing a Fingerprint Scanner, a Temperature sensor, the MSP430F5529LP, and an Arduino Uno. The goal was to create a security module that required the combination of an identified user in the fingerprint scanning system being scanned as well as a temperature read by the sensor to be in the threshold of human contact. This would mean that a security system would only allow access to someone who was presently holding a temperature probe as well as scanning their finger at the same time. This combination will be actuated by a toggling an LED, with a different color representing a different person. Due to being unsuccessful in getting the ESP8266 to work properly, this report will use Realterm to simulate the behavior of the ESP8266. Parts of the report will be written as if the ESP8266 was part of the design.

1.1 Design Features

These are the design features:

- UART Communication through multiple boards
- Temperature Sensing
- Fingerprint Identification
- Fingerprint Enrollment
- LED Actuation

1.2 Featured Applications

These are the featured applications:

- Dual Security System
- Fingerprint Scanning and Enrollment

1.3 Design Resources

All code for the project can be found at: https://github.com/RU09342-F18/intro-to-embedded-final-project-team-one
All parts were taken from the Rowan University Resource Center.

1.4 Block Diagram

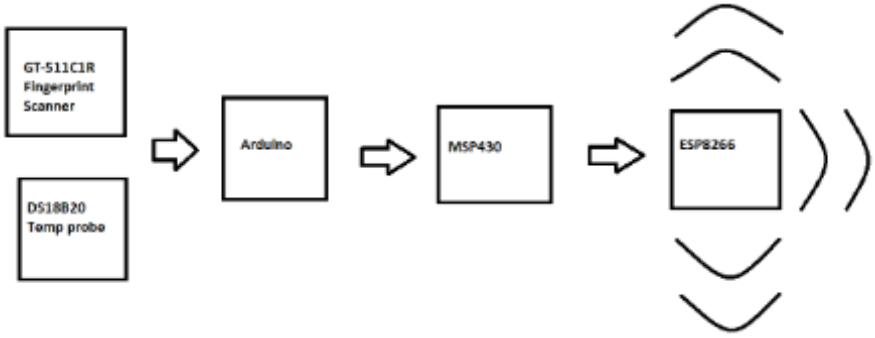

Figure 1: Block Diagram

1.5 Board Image

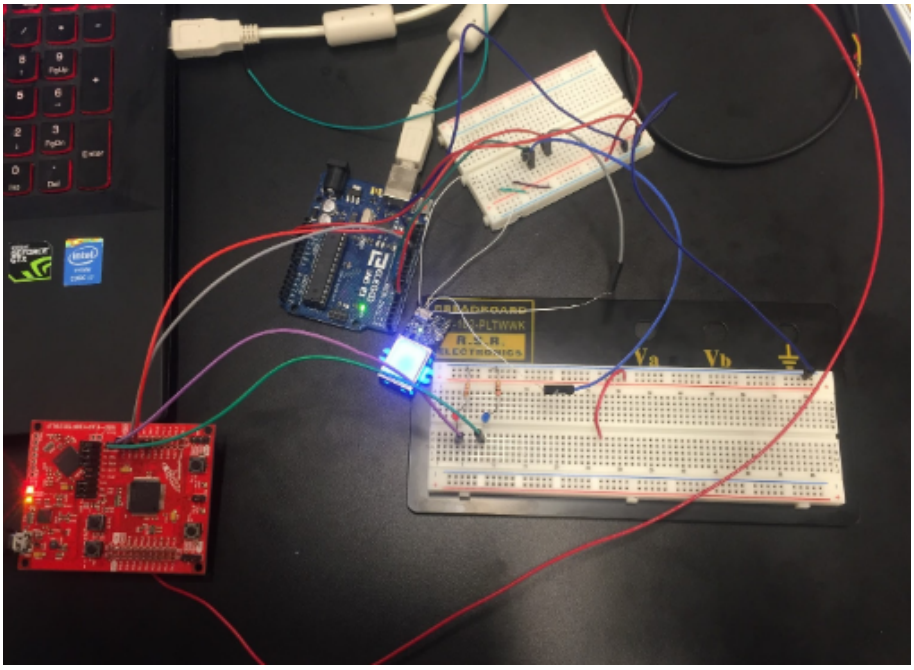

Figure 2: Block Diagram

2 Key System Specifications

PARAMETER	SPECIFICATIONS	DETAILS
Microcontroller 1	MSP430F5529	TI Microcontroller used
Microcontroller 2	Arduino Uno	Microcontroller used with GT-511C1R
Temperature Sensor	DS18B20	1-wire Communication
Fingerprint Scanner	GT-511C1R	Enrollment and Verification

3 System Description

The goal of this system was to act as a security measure/monitor so one could know who is scanning their fingerprint and when they are. Utilizing UART and Realterm allowed for realtime results of the data being transferred between the microcontrollers and the fingerprint scanner. For demonstration, two IDs are enrolled into the system,

and when their fingers are scanned, the system will toggle corresponding LEDs for each person. It will also notify the serial port monitor in Realterm who scanned their finger. The system is designed to only identify a successful fingerprint scan when the temperature probes are reading a value in the threshold of the heat of a human hand.

3.1 Detailed Block Diagram

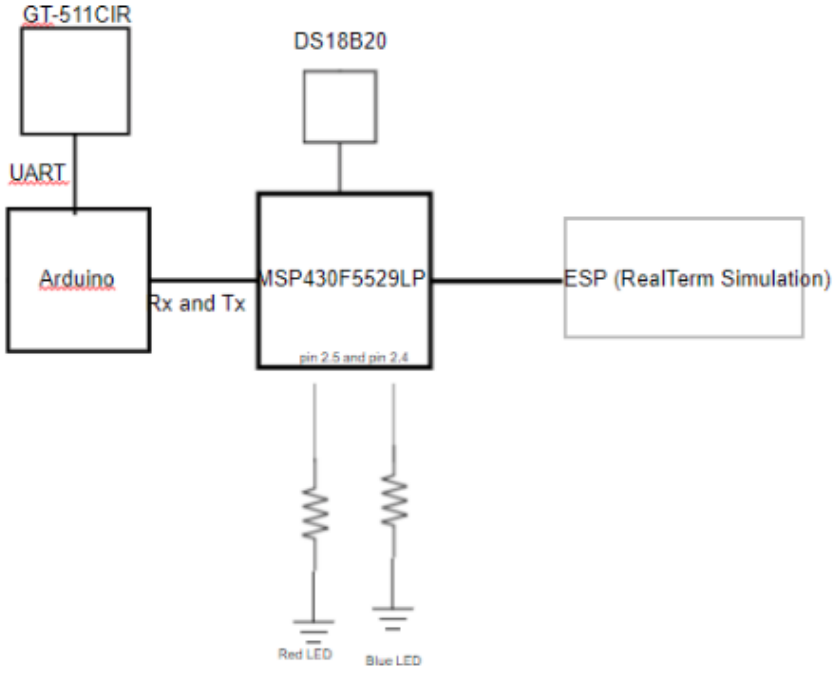

Figure 3: Detailed Block Diagram

3.2 Highlighted Devices

- MSP430F5529
- Arduino Uno
- GT-511C1R Fingerprint Scanner
- DS18B20 Temperature Sensor
- ESP8266 Wifi Module

3.3 MSP430F5529

This microcontroller was used for its several ports for UART communication as well as the GPIO pins. The pins were toggled to either low or high depending on if the fingerprint scanner identified a fingerprint. This was done by receiving the information through the Rx and Tx buffers from the Arduino Uno.

3.4 Arduino Uno

This was the second microcontroller used in this design. It was used for additional Rx and Tx ports to accommodate the parameters of the project. It was also used because the Arduino IDE is very user friendly when interfacing with the GT-511C1R as well as the ESP8266. This is because a lot of libraries are already built in that can help the user navigate how to complete the task at hand.

3.5 GT-511C1R Fingerprint Scanner

This scanner was used to enroll the users fingerprints into the system and then used to be able to identify those users if they pressed their fingerprint on the scanner. This scanner has four pins. Vcc, GND, Rx and Tx. It was easily connected and programmed with the Arduino to send a pulse to the MSP430 if a fingerprint matched one of the users.

3.6 DS18B20 Temperature Sensor

The temperature sensor proved to be a difficult task to integrate effectively into the rest of the design because the group was only capable of reading either a high or a low value from the sensor. Because of this, it did not reach a proof of concept demo.However, the goal and methods for implementing such a sensor are sound and make for an extra level of security in this system. The sensor was difficult to use because it sent and received data from only one wire. This was the groups first encounter with such a device and example code and resources online were unable to completely push the team into the right direction.

3.7 ESP8266 Wifi Module

The ESP8266 would allow the ability to display who had scanned their finger and when they did on a Wifi Server that is accessible to any who are subscribed to that given topic. That could be useful in several ways regarding security.

4 SYSTEM DESIGN THEORY

4.1 Design Requirement 1

One requirement for this design is to ensure that whoever enrolls their fingerprint beforehand must be sure that the scanner has gotten a good read of the finger. This can be tested by scanning your finger after you enroll it. If you get identified successfully several times in a row then you know that the scan was a good read.

4.2 Design Requirement 2

Another requirement is a laptop or other PC with at least 2 COM ports, since the Arduino and the MSP430F5529LP both need to be connected to two separate ports.

4.3 Design Requirement 3

Lastly, if using the ESP8266, one must be connected to Rowan_IOT wifi on their computer.

5 Getting Started/How to use the device

Use of the device is quite simple. The enrolled user will just put their finger on the fingerprint scanner and hold the temperature probe, and if their verified, their verification will be seen in the MQTT server.

6 Getting Started Software/Firmware

For this project to work, the fingerprint scanner must be correctly connected to the Arduino Uno. It must then be flashed with the Finger_Enroll.ino and a fingerprint(or two) can be enrolled. Then the Finger_ID.ino is flashed the the fingerprint scanner and the fingerprint scanner is ready to used. You can then connect the Arduino Uno along with the temp sensor to the pins specified in the main code of the program. Then the ESP8266 Wifi Module is connected to the MSP430F5529 along with the other parts, the esp8266setup.ino is flashed to the ESP8266 and the main.c is flashed to the MSP430F5529. The code is then ready to use. You may also connect LEDs to the output pins to show a successful fingerprint verification.

6.1 Communicating with the Device

The device, once flashed, is able to be interfaced completely through a smartphone that supports the MQTT app. The fingerprints will have to be adjusted using the arduino software and the serial terminal in the arduino software. The subscriptions

and topics can all be viewed from the app, and the fingerprint scanner is the only human interaction the device has.

7 Test Setup

First, one must connect the corresponding pins from the GT-511C1R into the Arduino. After this, one must connect the Arduino to the MSP430F5529 using UART pins that are enabled on the MSP. After this, the MSP must have its Rx and Tx pins connected to the ESP8266. The ESP8266 must be set into low power mode before compiling the example code by bootloading it. After this, the Arduino must be connected to one COM port of the users laptop and the MSP must be connected to another one. Next, one must enroll their finger print by compiling the example code for Enrollment. After that, the user must compile the Identification code on Arduino and the code for the MSP on code composer. Next, open Realterm and set it to the COM port that the MSP is connected to. This will track the data being sent from the fingerprint scanner. On Realterm, it will display arbitrary characters, and then display a different set of characters when an identified user is verified.

The most efficient way to test if it is working properly is to open the serial monitor on Arduino and see if the scanner is recognizing the fingerprint. The example code from Arduino on this device takes care of showing this on the serial monitor. It is also crucial to make sure that the baud rate for every device is 9600. This allows for the system to run because then every device is able to communicate with each other.

8 Results and Conclusion

Overall, there were a number of problems that were encountered that impeded the team's ability to create a system that covered the design overview. The final system consisted of everything except for the temperature sensor and the ESP8266 WiFi module. With what worked, the accomplishments were that if one enrolled their fingerprint, they could be identified and based on who was identified, a different color LED lit up on the breadboard. This was done by sending a pulse from the Arduino to a specific port on the MSP430F5529LP when a certain person was identified. This port was then connected to an LED. As a result, when User 1 is identified, a red LED lights up, and when User 2 is identified, a blue LED lights up. Because the ESP8266 module was not put in the final design, Realterm was used to read the data that the MSP was receiving.

The temperature sensor one wire communication proved to cause problems. It sends 8 bytes of data and the best that the group was able to interface with it was having the MSP read all 0s if the sensor was not reading anything, or all Fs if the sensor was reading something. After analyzation, it was determined that the reason that it was not working was because the Device Configuration Overlay (DCO) was not working properly. The temperature sensor is commonly used with microcontrollers that were

not accessible during this project from the resource center. Additionally, the ESP8266 was not used either. This is because problems were encountered after bootloading the chip and it was getting stuck and not connecting to the MQTT server. Since the code for the chip was example code supplied to us, it was out of the our scope to be able to debug it. Additionally, when setting up the device, the device commonly gets significantly hot to the touch. We deduce that a component in the chip could have been damaged causing it to not work properly.