Tri-Mark Corporation UWB Digital Key

Final Report

sdmay23-09
Tri-Mark Corporation
Advised by Timothy Bigelow
David Bone, Embedded Systems Specialist
Shayla Lunn, Embedded Systems Specialist
Kaili Lawson, Embedded Systems Specialist
Lakin Jenkins, Application Development
Hanan Zahid, Application Development
Erica Hollander, Application Development

Email: sdmay23-09@iastate.edu
Website: https://sdmay23-09.sd.ece.iastate.edu

Table of Contents

Project Design

Our project contains three separate modules that need individual development. The first module to design was the mobile application, which was done on IOS. The decision to make the mobile application for iPhones initially stemmed from the fact that the Nearby Interactions API contains a sample application similar to what we would implement. Taking this sample application, we were able to expand upon it and add our own functionality.

The next module to design was the UWB module. This module was provided to us by a company called Mobile Knowledge, which provided the hardware as well as initial demo UWB applications. After spending time exploring the codebase provided, we were able to implement our own communication, adding in what was needed for our application.

The final module to design was the Tri*Mark* module. Hardware and initial software was provided to us by our client. This codebase was updated to communicate to the UWB module, implementing all of the functionality of our project.

Combining these three modules together, end-to-end communication was established, with the ability to communicate from the mobile application to the UWB module through bluetooth and UWB, as well as communicate between the UWB module and the Tri*Mark* module using soldered on wires connecting UART transmit and receive.

Project Plan

In order to speed up the development process, development of each module was done in parallel. Below is the plan followed for each module:

Mobile Application

For the Mobile App, our process started with deciphering where the best place to start is. We looked into different APIs for different mobile devices operating systems and tried out a cross-functional approach. From the research we did and examination of the technology we had to work off of, we decided that we would develop an Apple App using Swift and Nearby Interactions as our API to communicate via UWB. The project started with the UWB and Bluetooth Apple sample application. Off the bat, we could run a Nearby Interacts Session and read distancing from the Tri*Mark* boards. After that, the team worked to understand the various threads working while the app was running. From here, the team worked to implement Bluetooth communication to send lock and unlock messages. Concurrently while the team worked on that, we re-designed the user interface to accommodate the designs we had proposed and approved. Once that worked, the team focused on authenticating a user and implementing pairing mode.

UWB Module

On the UWB module side, there were several changes to be made to the initial codebase. Firstly, UART transmit and receive functionality needed to be added in order to connect and communicate with the Tri*Mark* module. After UART functionality was added, control flow needed

to be implemented in order to make sure that the communication was as reliable as possible. In addition to UART, the UWB module needed to be able to handle incoming bluetooth messages from the application and execute the proper functionality. The UWB module mainly serves as a midpoint between the application and the Tri*Mark* module, forwarding messages to the necessary endpoint.

Tri*Mark* Module

The Tri*Mark* module interacts with the lock actuator as well as EEPROM memory in order to execute the necessary functionality for our project. Firstly, UART transmit and receive as well as control flow needed to be implemented in order to communicate with the UWB module. After this was implemented, the module needed to be able to pulse the lock/unlock actuator based on distance input. Once this was implemented, the module needed the ability to parse EEPROM memory and look for the phone's unique UUID. This is the authorization process that allows each phone to have full access to unlock/lock functionality.

Design Evolution

The majority of our design stayed the same throughout development. End-to-end communication stayed consistent with how we initially outlined it. The mobile application connects and communicates to the UWB module through BLE and UWB, and the UWB and TriMark module communicate through UART lines.

Our application's design has evolved since the initial mockup to be more simple and intuitive. The application now only has one screen, which contains all functionality desired by our clients. This screen contains lock and unlock buttons, which can each be pressed as many times as the user would like. The screen also displays distance information from the UWB module, providing useful data to the user. A requirement that was added this semester was the ability to have a "vehicle nickname" stored in the TriMark module's memory and displayed when an application connects to the vehicle. This was implemented and displayed on the home screen as well.

Throughout the second semester, the embedded side has been added to and functionality increased. Initially, we had planned on only a few UART messages being passed such as manual lock/unlock and distance messages. After this semester, we have added several new messages such as pairing mode and UUID authentication data flow.

Requirements

Functional

- The application will authenticate an IOS device.
- The application shall be able to unlock and lock the vehicle autonomously.
- When the unlock/lock button is pressed and the Digital Key is in range of the vehicle, the vehicle shall unlock/lock.

- The application shall check whether the Digital Key corresponding to the vehicle is in range and unlock the door.

Nonfunctional Requirements

- Look & Feel:
    - The app should contain the TriMark Corporation logo and incorporate the style and feel of the company itself.
- Usability:
    - The app should be well organized and have a good user interface.
- Performance:
    - After pressing the unlock/lock the vehicle should respond within a reasonable amount of time.
- Operational:
    - The vehicle should unlock by using the application when the device is within 15 feet away from the vehicle.

Relevant Standards

- IEEE 802.15.4 HRP UWB Standard
    - This standard works to promote and standardize the Digital Key technology. It also gives recommended practices when working with UWB.

- IEEE 802.15.1: Bluetooth and BLE
    - This standard is a low-data-rate, low-power wireless networking standard aimed at replacing cables between lightweight devices.

- AES128 Encryption Standard
    - This standard gives open-source functions that provide secure encryption of digital data.

Engineering Constraints

- Car Connectivity Consortium Digital Key
    - This is a standardized ecosystem that allows for mobile devices to store, authenticate and share Digital Keys for smart vehicles. The goals is to create a universal standard, so that all vehicles and devices are able to utilize this technology

- Mobile Device Specific APIs

- Apple devices and other devices use different APIs to access their U1 chips. The constraint is that different operating systems require different APIs, which makes it difficult to architect cross-functional applications.

Security Concerns

- Physical Security:
    - Stealing a phone
        - Stealing a phone with the application. If the thief could open the application and find the vehicle, they could unlock/lock and eventually remote start the vehicle. This could make it easier to steal vehicles.

    - Pairing Mode (when not at the dealer)
        - The module could be put in pairing mode and an attacker could connect their device. This could allow for an unauthorized user to authenticate to vehicles that do not belong to them.

- Cyber Security:
    - Man in the Middle: Replay Attack
        - Recording messages and replaying them to unlock vehicles when

    - Denial of service attack
        - Attacker continues to attempt to connect via bluetooth, not allowing an authenticated user and owner of the vehicle to connect and lock/unlock vehicle

Implementation Details

Application

Our project was implemented for IOS devices which include iPhone 14 or later. This is because these devices have a U1 chip inside developed by Apple. The application was developed on Xcode using Swift. There were a few different API’s and libraries used in our application. To make an initial connection and begin sending messages, the Bluetooth Core library was used. Once connected, the nearby interaction API is used to get the range for distance. The distance measurements and lock/unlock messages get sent over bluetooth.

UWB Module

The UWB module contains code written in the C programming language, which runs on a QN9090 microcontroller. The initial codebase and hardware was given to us by our client, which was provided by a company that specializes in UWB technology, Mobile Knowledge. We updated this codebase by adding our own APIs and UART communication files. This module receives and transmits UART messages to the TriMark module, which are outlined in our Master Document shared with the client. This module also has the ability to establish an initial bluetooth connection and communicate with the mobile application through this.

Tri*Mark* Module

The Tri*Mark* module was provided to us by our client. To add our application's functionality, we created our own API's that executed all of the UWB functionality. This included adding UART lines on the physical board, as well as writing the code that sent and received messages on these lines. After the UART communication was established, we were able to work with the existing codebase to toggle the actuators and read/write to EEPROM memory. This module executed all physical functionality, as well as sent explicit acknowledgements to the UWB module.

Testing Process and Results

Embedded Testing

On the embedded side, we had two main parts to test both separately and together- the TriMark module and the UWB module. On both sides, we did stress testing to make sure that our UART transmit and receive functionality works. We did this by simulating a UART receive through PuTTY to test our interrupt handler, as well as reading the PuTTY logs to test our transmit functionality. We also implemented tests to ensure the module connects and communicates with the mobile application. We made sure to test several use cases such as if the phone is within unlock range, if the phone leaves lock range, and if a manual lock/unlock is initiated by the phone.

App Testing

We made sure all functions implemented in the application are working. This includes testing a button press and making sure the right function is being called. Functionality of the buttons include, unlocking/locking a vehicle, pairing a phone to the UWB module, and making a connection once a phone has been previously paired.

System Testing

We ensured the entire system was functioning properly by implementing end-to-end testing. We established different protocols such as having the key lock or unlock within a certain distance. We tested to ensure that if you are outside of the automatic unlock distance, but still in UWB range, you will be able to lock and unlock the vehicle via the application.

Acceptance Testing

For acceptance testing, we implemented multiple procedures to guarantee that all of our requirements would be met. We executed this by using end-user testing. In our end-user testing, we focused on the functions of our application most used by the user, such as the lock/unlock mechanism by walking up, the distance capability in the application, and the manual lock/unlock functionality from inside the lock/unlock radius.

Related Work

Currently, digital keys are implemented in some newer consumer vehicles, which includes BMW, Mercedes, and Audi. This excludes the market of bigger vehicles such as tractors, RV’s, and industrial vehicles. Unlocking an industrial vehicle via your mobile phone does not currently exist so this project will lead the future of automation in this market. This benefits these users because it allows them to ditch the key fob and enables them to unlock their vehicle using biometric protected or password protected devices. We did research initially by watching videos of BMW’s current digital key technology.

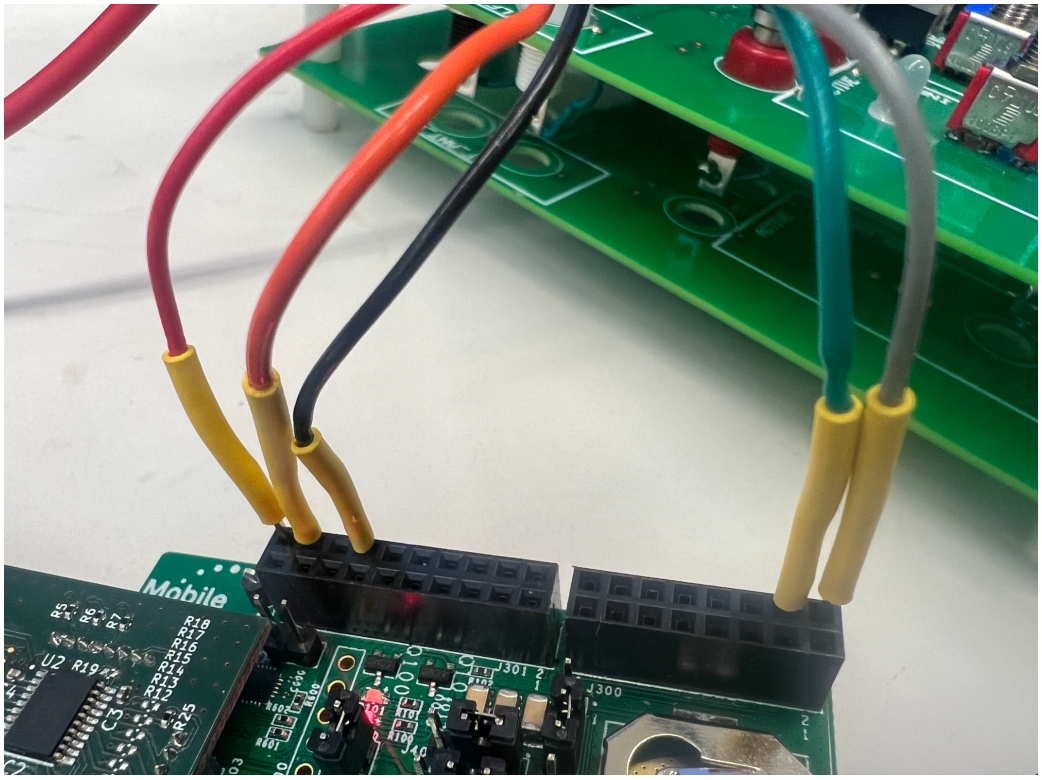
Appendices

Appendix I: (Operation Manual)

Testing and Installing Mobile Application to iPhone:

1. Environment Setup and Requirements:
    a. MacBook running macOS Monterey Version 12.6 or newer
    b. Xcode version 14.3 or newer is required to build the project
    c. iPhone 11 or later devices running IOS 16.3 or newer
2. Installing application to iPhone:
    a. Plug in your iPhone to the MacBook via USB-C cable
    b. On Xcode, select the output device to the plugged in iPhone
    c. On Xcode, add your developer account under project settings
    d. Build the project by clicking “Product” in the menu and then clicking “Run”
    e. Once the application finishes installing, go to the settings application on your iPhone and navigate to General → VPN & Device Management → Developer App, then click trust on your application development account
    f. The application should now run on your iPhone

Embedded Setup

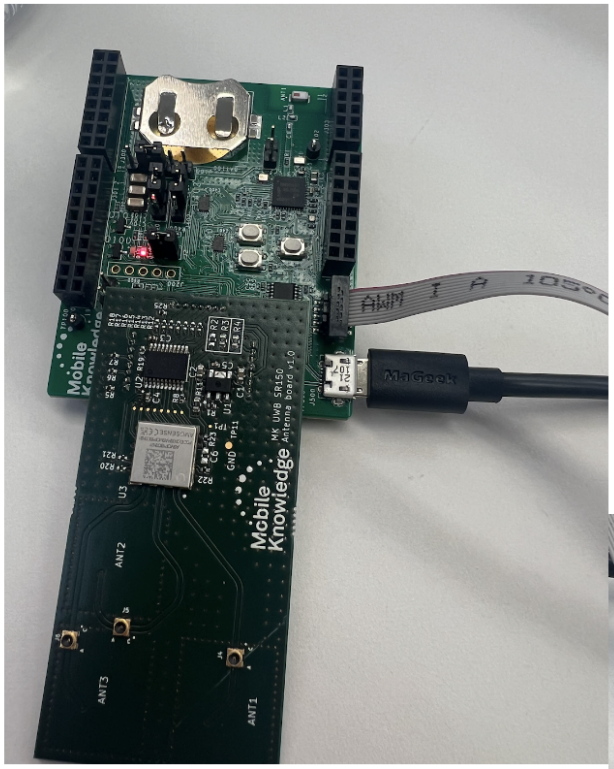


(Wire connection from TriMark module to UWB module)

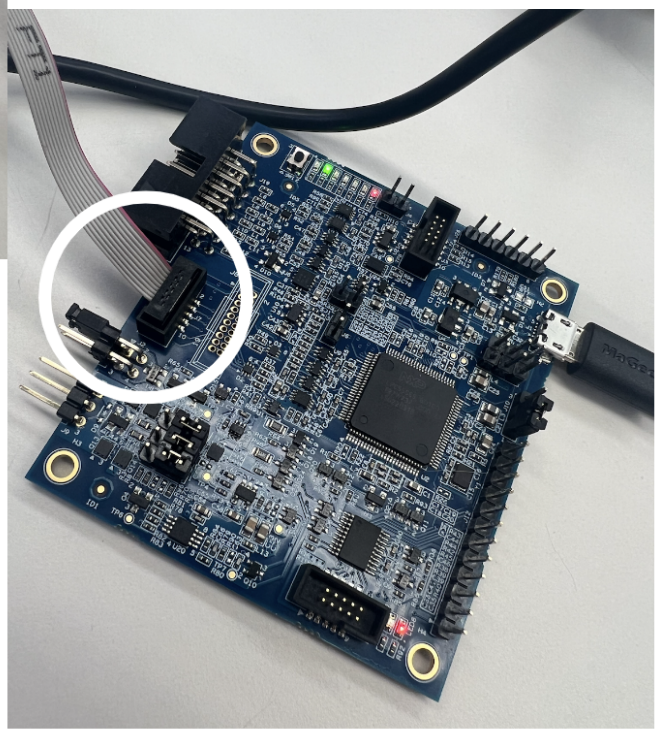
1. TriMark Module
    a. Download the MPLabX IDE
    b. Import the UWB.x project
    c. Attach 12V power supply to the TriMark module
    d. Build the project and download to the module

2. UWB Module

Anchor Setup:



NXP Debugger Setup:
(jumper 2 and debugging cable circled)



Flashing Instructions:
- MCUXpresso IDE is needed.
- Follow all instructions from video, but switch build configuration to UWB Shield 2 SR150
- For Mobile Edition, you will need to download the necessary SDK into MCUXpresso. This SDK can be found at this link: https://mcuxpresso.nxp.com/en/select.
    - select QN9090DK6 board
    - include FreeRTOS and Wireless BLE components

Interacting With The Vehicle

1. Accept bluetooth access permissions on application.
2. The application will automatically connect to the UWB module if nearby.
3. Once connected to the module, the distance will be shown and the lock/unlock buttons will be enabled for interaction.