Smart Home Entry System

Team:
Robert Kaufman (rkauf@umich.edu),
Kobe Howcroft (korb@umich.edu),
Grant Hincher (ghincher@umich.edu),
Brian Good (brigood@umich.edu)

1 Customer

People concerned with home security. Particularly those who prefer to not carry around a house key, as this device would allow the user to unlock their door from their phone. People who use delivery services for packages or food.

2 Value

Provides a vast quality of life improvement by allowing for long distance locking and unlocking of one's home. This would make the anxiety of wondering whether or not you remembered to lock your home a thing of the past, and allow friends and relatives who may need to enter your home while you're away to do so. Furthermore, our system could help remedy the increasingly common issue of package theft. Companies have tried creating boxes that package or food delivery drivers can lock and unlock, but these efforts failed because workers didn't use them. However, if the box can lock/unlock automatically, delivery drivers would be more willing to simply put them in the box.

3 Approach

The main hub is inside the home and is accessible via wifi. This system contacts subsystems via XBEE; namely, a door lock and package lock box. The door lock uses a servo motor to control the deadbolt to allow it to extend and retract when signaled from the main server. The door also has a keypad to unlock when at the door as well. The package box uses an infrared sensor to detect when a package or food has been inserted into the box, the box will then close automatically and will lock. The lockbox could also be controlled remotely or through a keypad in the same way as the door lock. The system also has a camera to capture the entryway and see who is entering.

Major functions of the project:

Wireless connection between user and subsystems

- Use XBEE and Wifi to establish communication between main system and subsystems
- Apply built in XBEE encryption for security

Touch screen

- This allows the user to monitor and change the status of subsystems from inside the home.

LCD screen to display to display status

- Display numbers pressed on the keypad
- Display whether the password entered was correct or not

Servo motor to open and close the lock

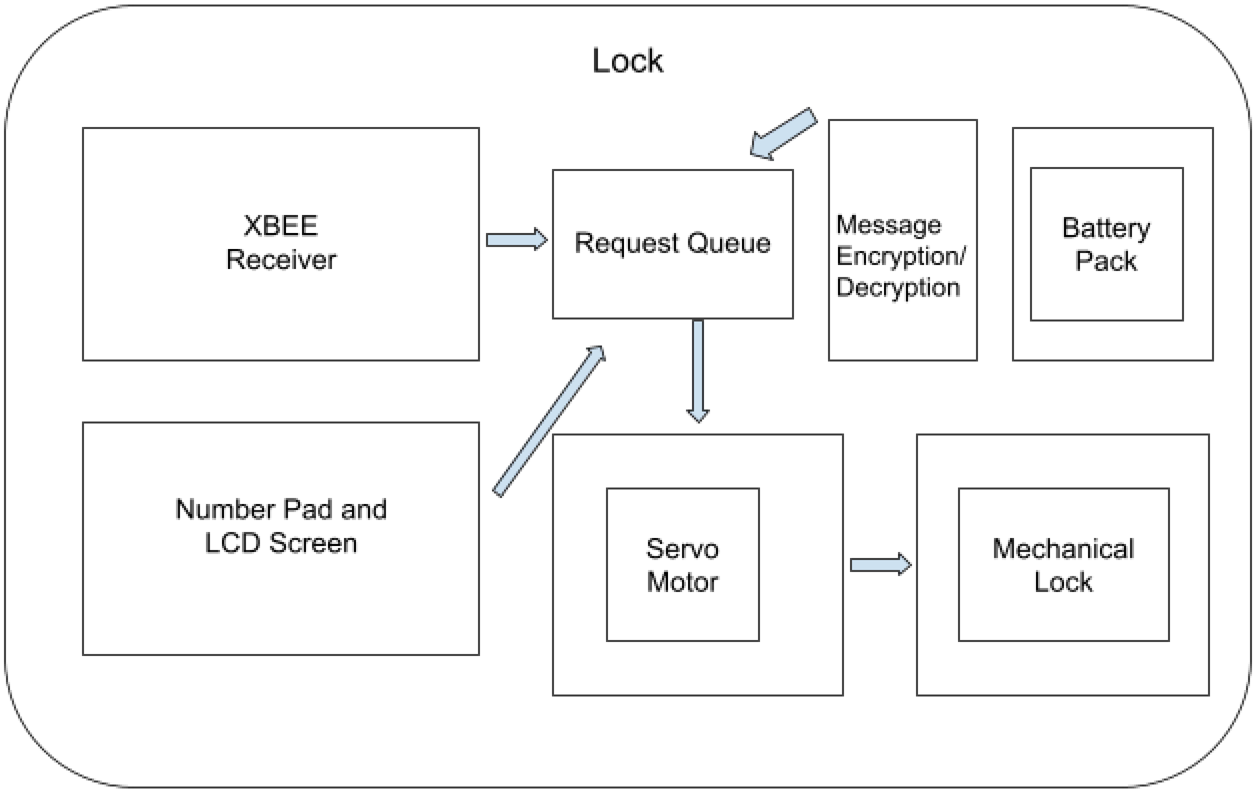
- Interface with shaft connected to manual lock/unlock from inside

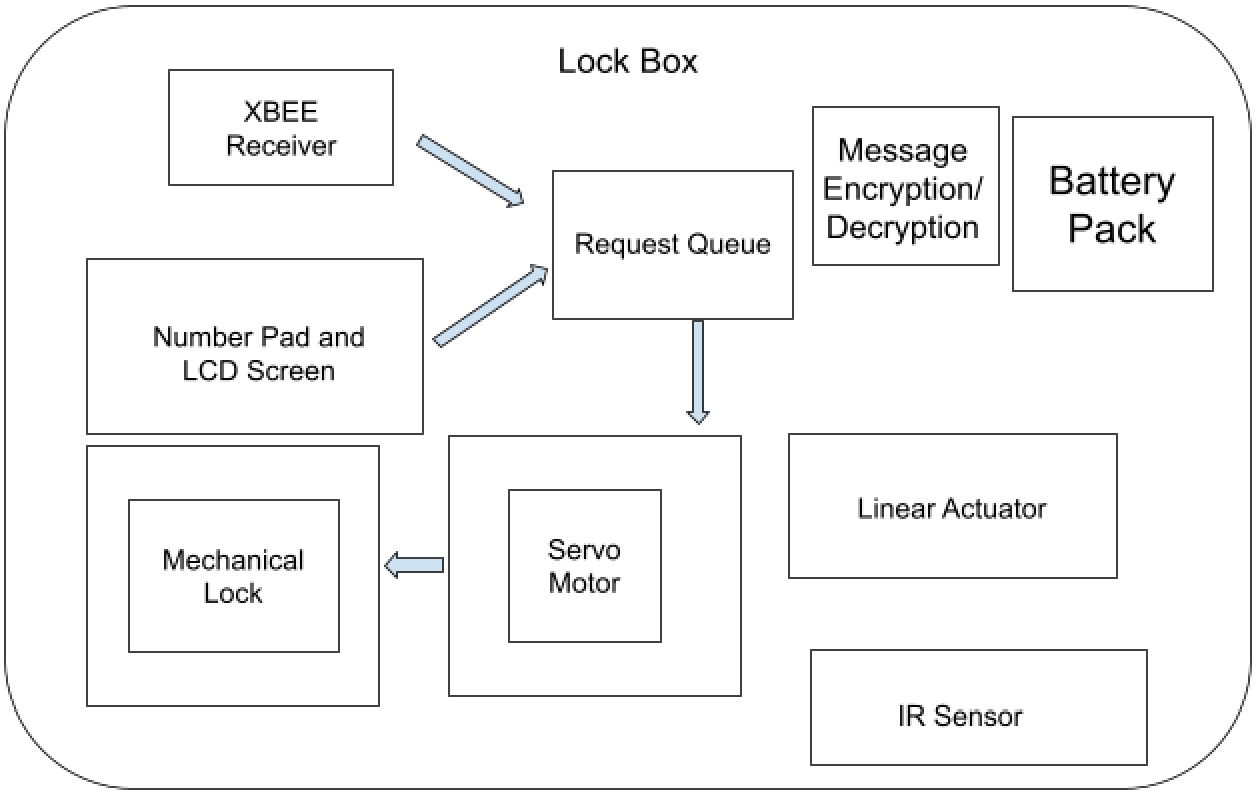
Lock box with IR detection

- Interface with shaft connected to manual lock/unlock from inside

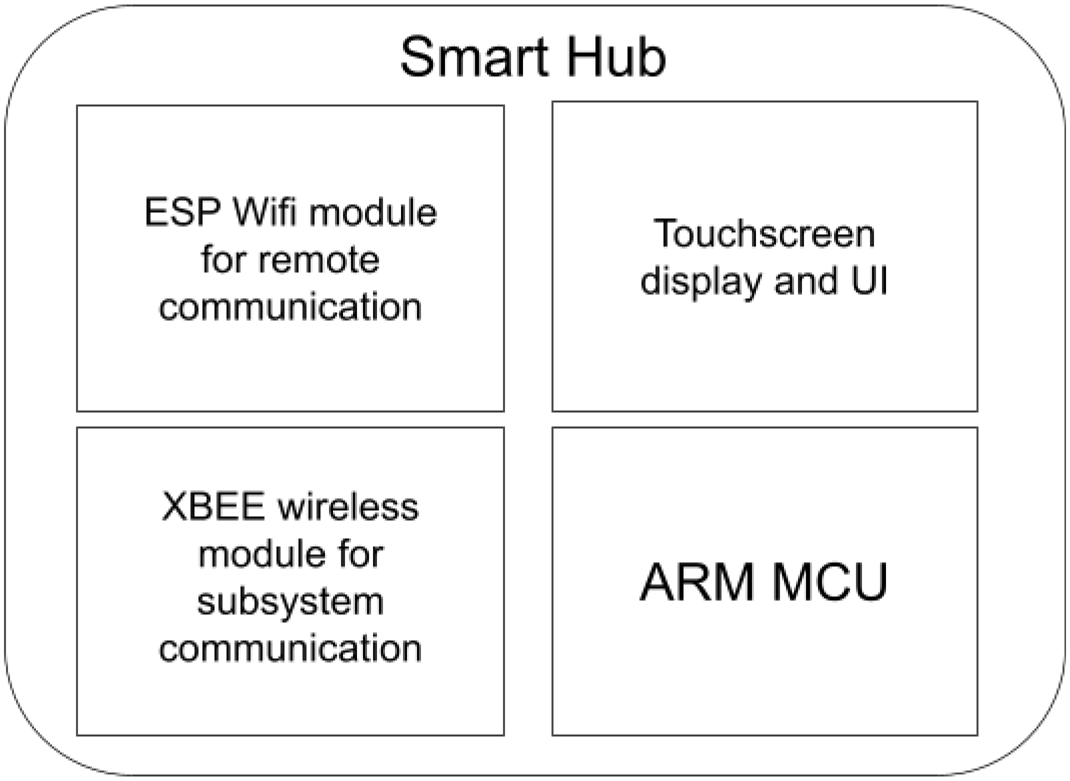
4 Final System Components

1) Main Hub
    a) 3.2" ILI9341 SPI TFT LCD Display 320x240 Touch Screen Shield Display Module
    b) ESP8266-01 wifi module
    c) XBEE transmitter/receiver
    d) Arm microprocessor: Nucleo L4R5ZI-P board
2) Door Lock
    a) Hitec HS-422 Servo motor to open and close lock
    b) Switch Keypad 16 Button 4x4 with 8-Pin Solder Pad 24VDC 20mA Keypad complexity points
    c) Serial enabled LCD screen to show currently typed keys
    d) XBEE receiver/transmitter
    e) Arducam Mini 2MP JPEG Camera
    f) Arm microprocessor: Nucleo L4R5ZI-P board
3) Package lock box
    a) Medium duty linear actuator to open and close box (also gives position) (12 inch travel)
    b) Hitec HS-422 Servo motor to open and close lock
    c) Switch Keypad 16 Button 4x4 with 8-Pin Solder Pad 24VDC 20mA Keypad
    d) XBEE receiver/transmitter
    e) IR sensor (Sharp infrared 2Y0A21)
    f) Arm microprocessor: Nucleo L4R5ZI-P board





*Note: The encryption/decryption is built into the XBEE modules we used


Smart Hub
ESP Wifi module for remote communication
Touchscreen display and UI
XBEE wireless module for subsystem communication
ARM MCU

5 Changes to Plan

For the most part, we did not change our overall plan. The main thing that didn’t quite pan out was the JPEG camera. We had originally planned on having this be operated by an interrupt from the Nucleo strobed by a lock interaction and then communicate this image to the hub over XBEE. The precise issues are discussed further in technical challenges, but this proved impossible within our given time frame. As a compromise, we interfaced with the camera using an Arduino and generated a live feed on a computer to at least have some sort of imaging capability in our system.

Additionally, we had planned a presence detecting component using another IR sensor that would trigger a light over the door. We ultimately abandoned this component since it was more of the same and didn’t add complexity, but would’ve made for enough work to bog down other, more important systems. Aside from these, everything else went more or less according to plan.

6 Complexity Points

Part	Points
Touchscreen	⅔-1
Key Pad	⅔
JPEG Camera	~1/4
WiFi	~1/2
XBEE	1/2
IR Sensor	1/4
Servo	1/4
Linear Actuator	2/3
LCD Display	1/2
Total	4.25-4.583

Note: For the JPEG Camera and the WiFi, we talked to Matt, and he gave us an estimate for the complexity point values, so those are what we added to the table. Additionally, with the JPEG Camera, we weren't able to hook it up to the Nucleo board after a significant amount of effort, but did get it to work on the Arduino, which we used in our final demonstration. For this reason, we decreased the complexity points earned in the above table from what Matt originally quoted us.

7 Technical Challenges and Solution:

Initially, we had an issue with the keypad, likely due to bad solder, that led us to believe that it was active high instead of active low. This caused us a significant amount of time trying to figure out what was wrong until we eventually figured out that the keypad was active low and we simply had a bad solder.

Another issue we had was a power concern. When we turned on the servo to lock the door, it would draw too much power from the Nucleo board and would cause our LCD screen to go dark. We fixed this problem by directly hooking the servo to an external battery, and powering it that way. That way, when we wanted to turn on the servo, it wouldn't drain the power and the LCD would stay bright.

Another issue we had was with the touchscreen and trying to get that to work. We initially started off with the blue Adafruit touchscreen, but after a while of trying to get that to work, we found a github repo for the red touchscreen, and we switched to that.

During testing, our Linear Actuator had a gear that fell off of its shaft, causing it to suddenly stop working. After a while of taking it apart and trying to find what was wrong, we finally found the gear not in the right place and fixed it. After that, however, we still decided to go with a shorter one due to the fact that the one we had was too long, and did not have a good angle to be able to open the door to our lockbox properly and would get bound up in the process.

When trying to get our JPEG camera to work, we found a library that would get it to send pictures to an Arduino board that we had. However, we were not able to get it to communicate with the Nucleo board, as we were dropping bytes over the UART communication. Unfortunately we never did figure out how to get this to work. Our solution was to demo the camera feed directly on a laptop for demo day, to show that while we couldn't get data onto the Nucleo successfully, we were in fact able to get the raw JPEG data off of the camera.

We spent a lot of time trying to talk to and configure the ESP8266 wifi module directly over UART. There is AT-Command documentation which shows that sending the ESP8266 specific messages over UART would allow you to configure the ESP8266 to connect to a network, and service TCP requests. We played with this for about a week and could not get consistent results. In the end, our solution was to directly flash a program to the ESP's memory, which would allow it to be pre-configured on startup to connect to the correct network and service requests. Now we could talk to the ESP using our own custom messages over UART consistently.

8 Final Team Member Contributions

Robert (25%)- Keypad, LCD, Servo Operation, Touch Screen
Grant (25%)- Keypad, Linear Actuator, H Bridge, Physical Construction
Kobe (25%)- LCD, Keypad, IR sensor, Materials, Touch Screen
Brian (25%)- Xbee, Keypad, Encryption, Camera, WiFi

9 Design material and code

All code available on GitHub: https://github.com/briangood35/eecs373-project