Molayo Faseru

IoT-based Key Finder

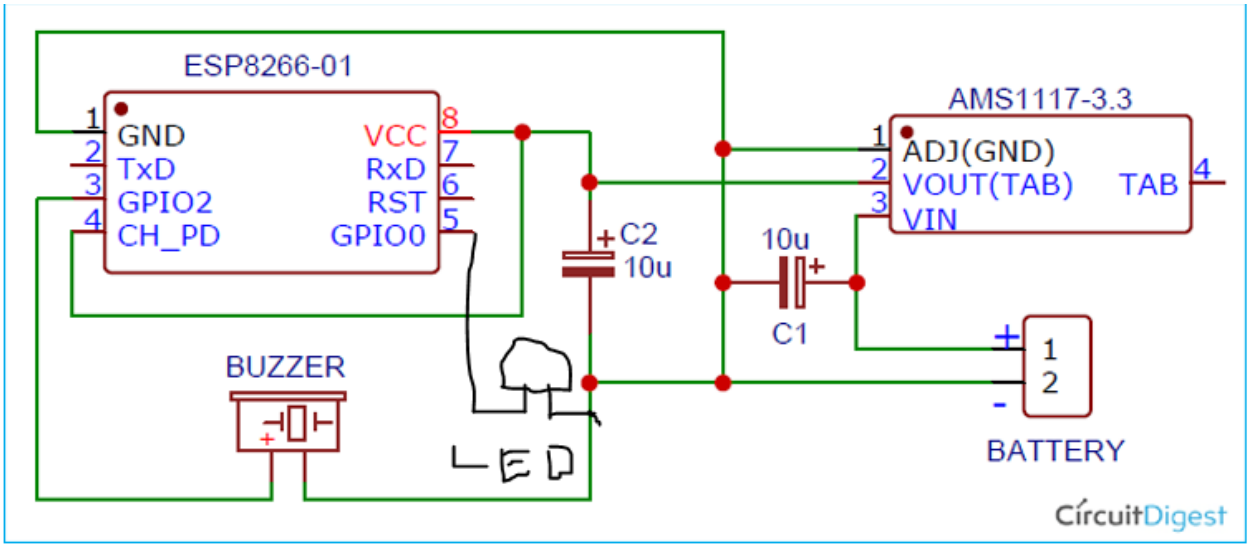
Design Project #3 – Final Project

Design Overview

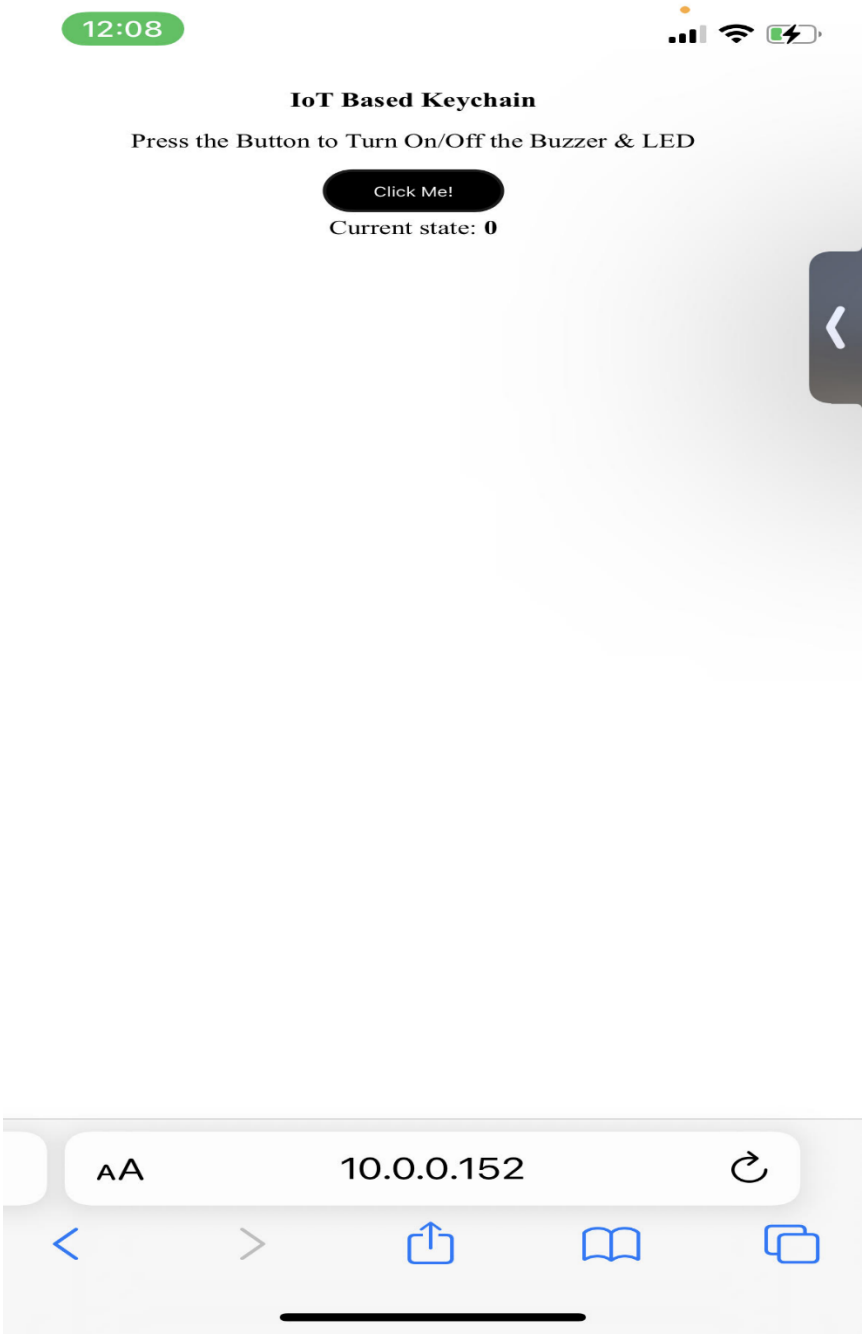
Most of us have experienced the feeling of spending hours searching for keys that you misplaced and after a long search we end up finding them. That search time can be significantly shortened with the aid of an IoT-based Key Finder. In this project, I have designed a simple IoT-based Key Finder using ESP8266-01, Buzzer, LED, and Battery.

Operation: In case you can't find your keys and you have the Key Finder attached to it, you can take out your phone and open the key finder webpage. There you will click an icon and begin to hear a beep sound and see flashing lights coming from it. With this, you should be able to track your keys if they are within the vicinity. This design builds on a preexisting design I found on a Smart Key Chain. The source is referenced in the sources section [1]. I modified this design by adding an LED bulb to the GPIO0 pin so that the Key Finder can be located in the dark as well as it being more accessible to people hard of hearing. A hole can be made on the PCB to provide a means of attaching to a keychain.

Schematic: This below is the schematic of the original design that I am referencing with the proposed modification highlighted.



For the code, I made use of the ESP8266WiFi library to connect ESP8266-ESP01 to a Wi-Fi network for communication between the user's phone and the ESP8266 chip. I also created a html webpage that can be opened in a phone browser using the ESP8266-ESP01 IP address. Here is a screenshot of the page.


12:08
IoT Based Keychain
Press the Button to Turn On/Off the Buzzer & LED
Click Me!
Current state: 0
AA
10.0.0.152

Project Justification

My motivation for this project is to get into the Internet of Things in a practical setting. I enjoyed learning about that sector of technology in my Computer Networks and Embedded Systems classes, and I would like to explore more interesting practical projects and hopefully career in that field. I feel this project is appropriately scoped because there a various stages in the development process. From the PCB Design to the Circuit Verification, and ordering of parts, writing the software, testing, troubleshooting, and enclosure/finalization and lastly the report/documentation.

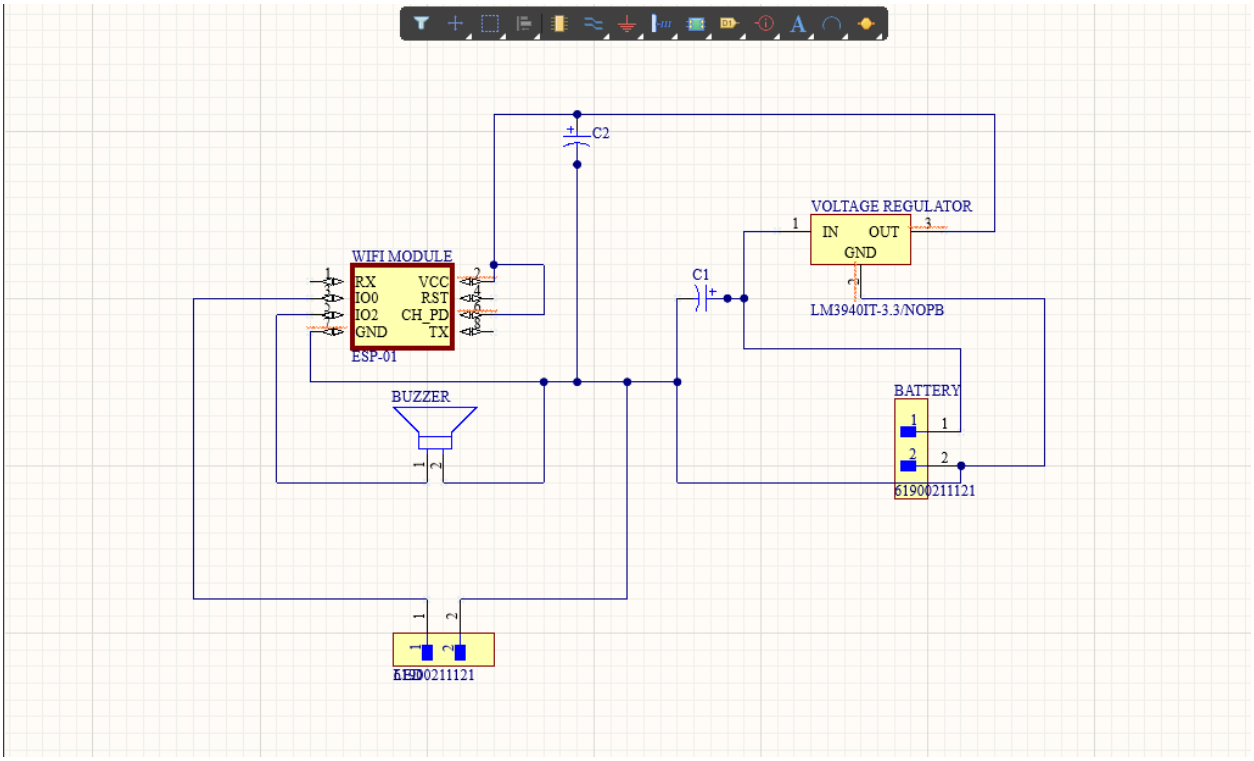
Preliminary Design Verification

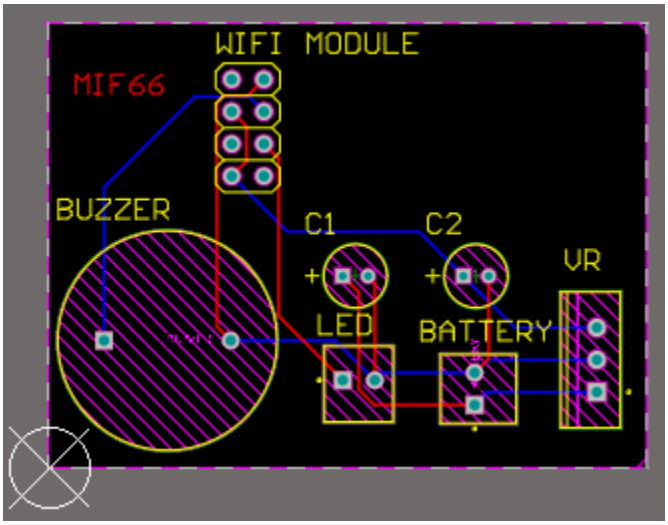
For my design verification, I made the schematic in Altium and verified all valid connections. I then breadboarded the circuit after programming the ESP01 module and confirmed operation. Here is a recording of my breadboard verification. As I hit the "Click Me!" button on the webpage, the buzzer and the LED start to flash at the same period. Attached below is a recording of my fully functioning breadboard circuit. https://youtube.com/shorts/1g-szEXvZwQ?feature=share.

Design Implementation

My overall design consists of a built PCB circuit with a 9v battery connected to it. I used a 3.3v regulator because the max voltage for esp01 module is 3.7v and the board will be damaged if given any higher. I connected two 10microFarad capacitors to between VCC and ground of the ESP01 module and the VCC and ground of the 3.3V regulator respectively to reduce power supply noise and voltage spikes on the supply lines. The Buzzer and LED are the output for this design, When the button is clicked from the webpage, it flashes both the Buzzer and LED at the same period and the User can now locate the device with hearing and sight.

Below are screenshots of my PCB schematic and layout.





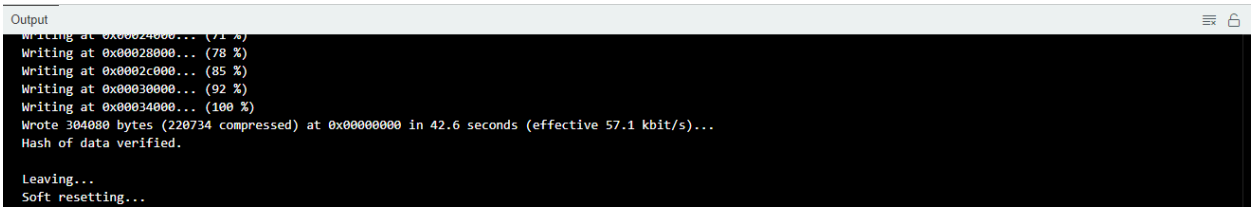
Design Rule Verification Report

Date:	11/18/2022		
Time:	2:26:25 PM		
Elapsed Time:	00:00:01	**Warnings:**	0
Filename:	C:\Users\Public\Documents\Altium\Projects\Final_Project\Final_Project.PcbDoc	**Rule Violations:**	0

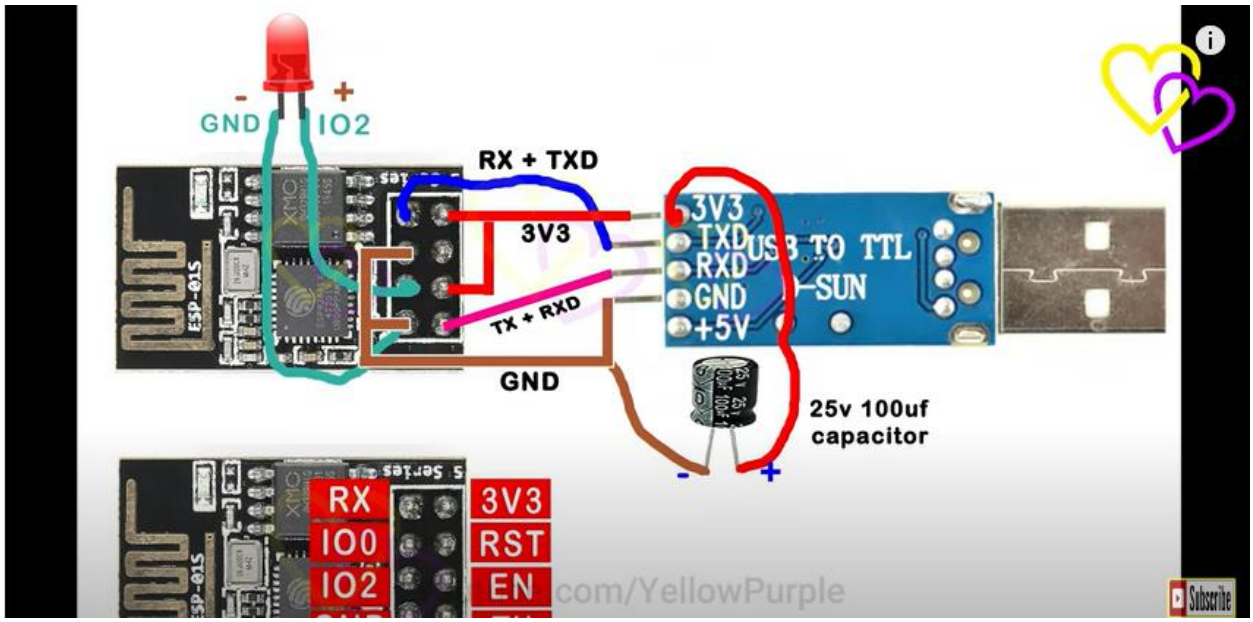
I faced a huge challenge with programming the board. I had no previous experience in programming ESP8266 boards and ran into many issues/errors while trying to flash the device. I got errors such as “esptool.FatalError: Failed to connect to ESP8266: Invalid head of packet (0x00)” and Timed out waiting

for packet header, and had to do a ton of research in order to troubleshoot this issue and it really slowed down my design process.

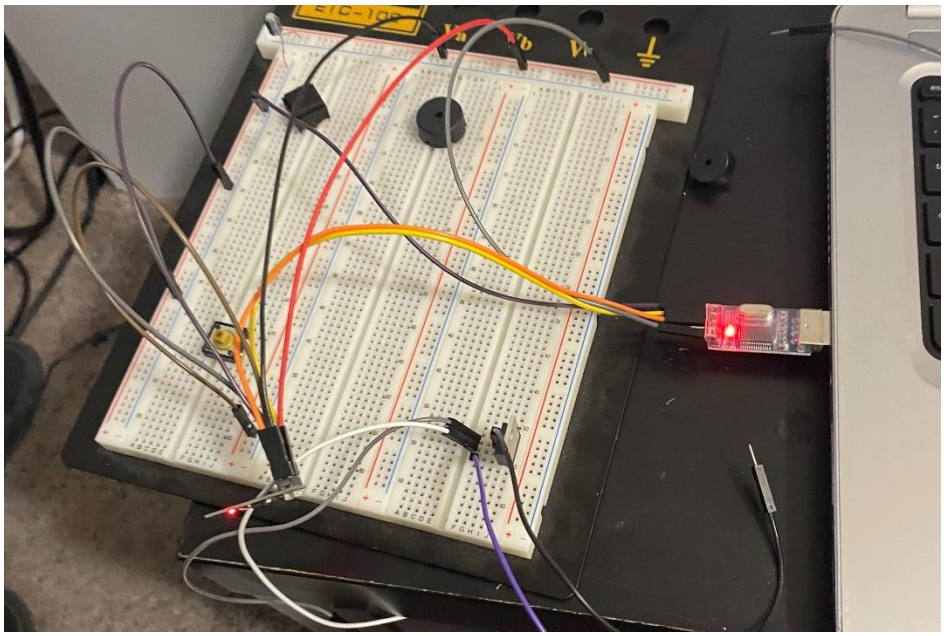
After tons of research and debugging, I found a reliable solution and was able to successfully flash the ESP01 module with my Key Finder software.



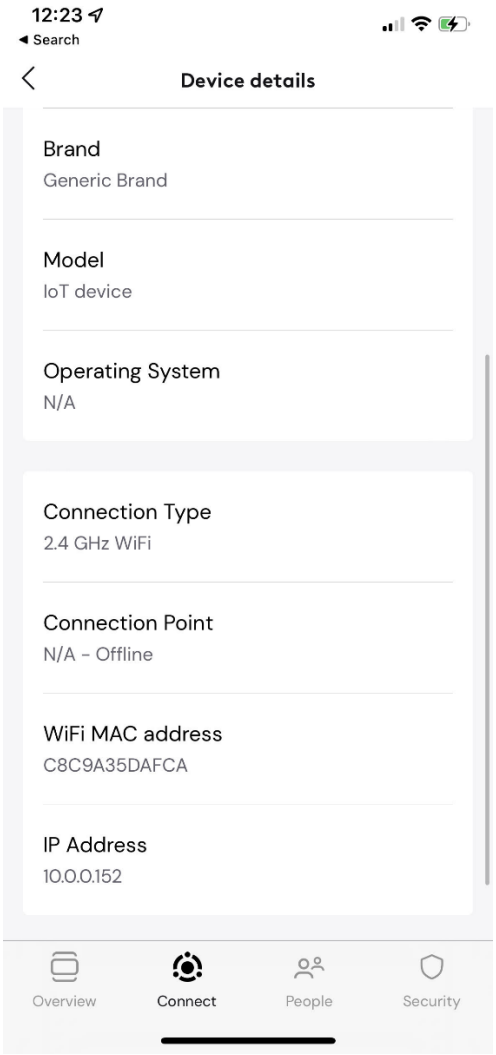
Below is the schematic I used to flash the ESP01 module [2].



Here is my programming setup



Another issue I faced was connecting the esp01 module to a Wi-Fi network. I read to use my phone's personal hotspot as a Wi-Fi network to connect the esp01, but it wasn't connecting to the hotspot at all. I then used my home Wi-Fi network to connect to the esp01 module and I had to log in to my provider account and see if it was connected and it was!! I then was able to fetch the Ip address from my wireless provider dashboard



Now the program has been flashed to the esp01 module and can begin the testing process.

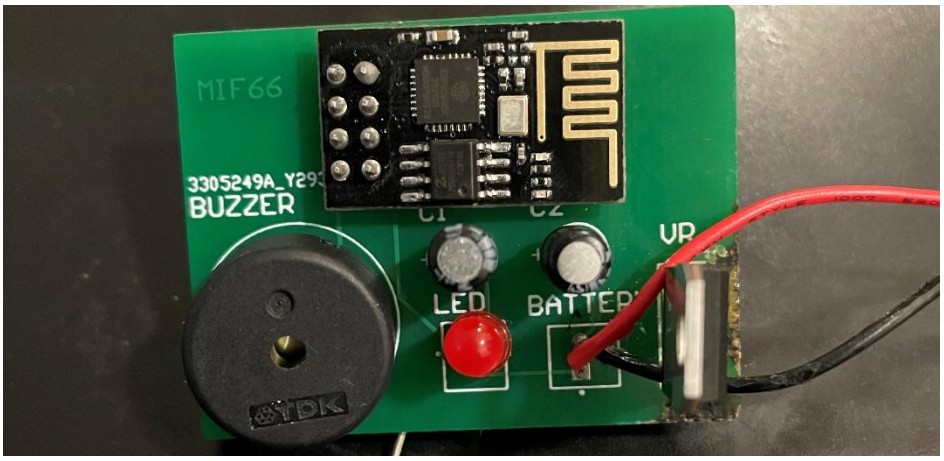
Design Testing

My testing procedure was simply to breadboard the circuit and verify the operation. With the programmed esp01 module, I built the circuit on a breadboard and connected to the esp01 by typing the Ip address in my phone browser and opened up the Key finder Webpage. I toggled the Click Me button and the Buzzer and LED started to flash/blink. This means my Design has 100% passed. For this kind of design, the programming of the esp01 took the most debugging, but once the module is programmed, the circuit is not complicated, hence why the testing procedure was simple/smooth.

Below is a recording of my tested design showing that it works as supposed. If you turn the volume up, you can hear the buzzer clicking.

https://youtube.com/shorts/1g-szEXvZwQ?feature=share

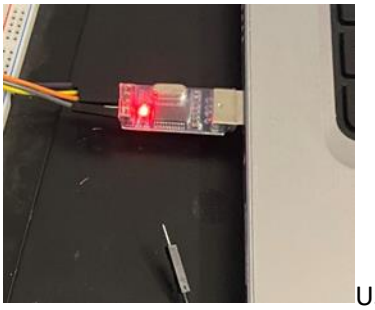
After the breadboard testing, as I was soldering the components to the PCB, I ran into minor issues. The LED was soldered with the wrong poles, so I had to take it out and resolder it to the PCB. Below is an image of the fully soldered circuit.



Here is a video of my functioning design. https://youtu.be/dYm1bjgFuvg

Summary, Conclusions and Future Work

This design was a mix of software programing and hardware electrical work(soldering) and after tons of research and debugging I was able to program the device. My major issue with programming was the USB to TTL serial converter that I used. Usually, one will use an Arduino UNO board to easily program this device, but I took a more cost-effective method with the USB to TTL converter and It came out alright at the end. In another iteration I will still make use of this USB to TTL since I now know the proper ways to use this device, but for new users working with the ESP8266, I recommend using an Arduino Uno board if available. With this device attached to your keychain, you should never worry about misplacing your keys again 😉.


USB to TTL Serial converter

Final Presentation

I created a PowerPoint voiceover slideshow showing the operation of my design. Please find attached in my canvas submission with all my design files.

Sources

[1] https://circuitdigest.com/microcontroller-projects/diy-iot-based-key-chain-finder-usingesp8266

[2] https://www.youtube.com/watch?v=GQIzDSK4tMw&t=78s