Final Report

Solar-Powered Wi-Fi Repeater

Zephan Enciso, Chloe Frentzel, Matthew Hawkins, Lisa Huang, Jake Leporte

13 May 2021
EE Senior Design

Contents

Overview

GroundLink is a **solar powered** Wi-Fi range extender. It is a stand alone device that can be placed outside to amplify the Wi-Fi signal from a nearby building. The heart of the device is the **ESP32**-WROVER-IE microcontroller that acts both as an access point and station, providing **NAT routing** for Wi-Fi signals on the **2.4 GHz Wi-Fi** range. After the device has been configured using the device's **web interface** and placed in its desired location, the device will connect to an existing wireless network and provide up to four devices with a reliable WiFi connection.

Problem Statement and Proposed Solution

We address limited **Wi-Fi coverage** in **outdoor college campus spaces**. With the ongoing global pandemic, there has been an increased need for strong Internet connection in outdoor spaces for various tasks such as Zoom for class sessions and other meetings. In addition to ease of use and access to Zoom in outdoor spaces, our device will also encourage students and faculty to spend more time outside.

Our device presents a **802.11n Wi-Fi AP** with **WPA2 security** to protect users' privacy and limit access to the network. The device operates on the **2.4 GHz** Wi-Fi range, which features longer range at the cost of raw throughput. Upon connecting, the user can configure the device via a **web interface** (at IP address 192.168.4.1). This allows the user to connect to an existing wireless network or change the default password and SSID. To cover a larger area, the devices can be **chained together** (albeit with a latency penalty).

The device is powered by a system consisting of solar cells, batteries, power regulators, and a charge controller. When external conditions do not allow the solar cells to output enough to power the device, the batteries allow the device to continue to run, as well as provide a steady source of power. Our **6 V, 6 W solar panel** occupies a footprint of approximately **220 mm by 175 mm**, which sets the length and width dimensions of the device.. The charge controller balances solar charging across all the battery cells, ensuring even wear. In the event the user wishes to charge the device in the absence of sunlight, the device features a micro USB-B connection for **DC charging**.

At the heart of the device is the **ESP32-WROVER-IE microcontroller**, which both services web requests and provides routing for Wi-Fi signals. The ESP32 comes with extensive Wi-Fi functionality onboard, and the connectivity is enhanced with an **external antenna**.

To package the device, we created a **3-D printed prototype enclosure**. The enclosure provides minimal protection from dust and water while housing the entire device in a self-contained unit. With the enclosure, the device is able to be transported easily and is an isolated unit.

System Requirements

- Power system consisting of a solar input and DC input powered by a USB power adapter alongside a regulatory circuit
- Circuitry for interfacing the different hardware components of the project with each other, mainly the power system, the microcontroller, and the antenna.
- An external antenna to enhance the internal antenna trace of the ESP32.
- UART interface connected a microUSB-B port on the device. This allows the ESP32 to show up as a virtual COM port on a PC and be programmed in a manner similar to Arduino microcontrollers.
- TCP/IP stack with routing capabilities.
- A web server running on the microcontroller, which will host the configuration site. The configuration site is designed using web programming APIs provided with the ESP32 microcontroller and written with HTML and CSS. The design attempts to balance ease-of-use with configurability.
- Enclosure designed in CAD and 3-D printed to house all elements while also being slightly water and dust resistant (protect the device)

Subsystems

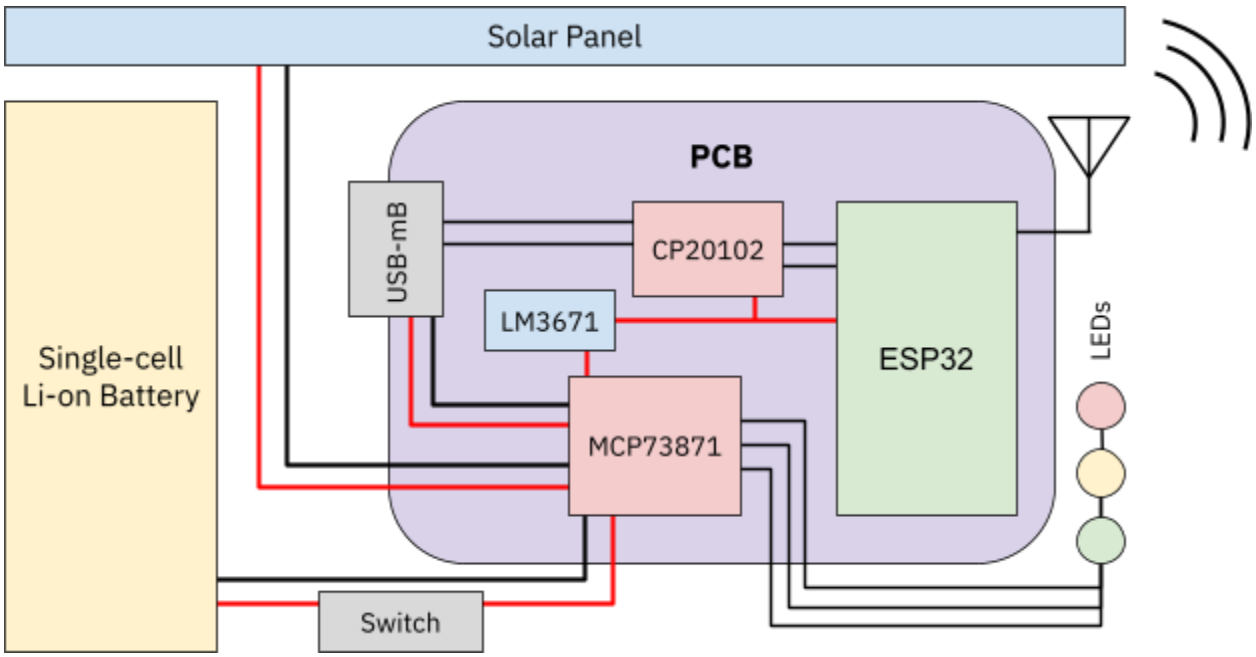


Figure 1. Block Diagram

Power System

The power system of the device is powered by a battery that is charged either by the solar panel or from a **USB mini-B** input using a **MCP73871** charge controller. The charge controller sends the input power from the solar panel or the USB to charge the battery and shuts off the device if the battery power gets too low or if there is not enough power coming from the input sources. LEDs on the side of the device indicate the status of the charge controller. The Red LED indicates there is sufficient input power to charge the battery. The yellow LED indicates that charge is actively being stored, and the green LED indicates that the charging is complete. A switch is also included to easily disconnect the battery from the device to prevent it from discharging while the device is not being used.

Board

The PCB contains circuitry for the power system and the microcontroller. A **USB mini-B connector** allows the system to be plugged into a laptop for programming or charging. A **DC barrel plug** attaches the solar panel to the board. These act as inputs to the MCP73871 charge controller which is used to charge the 3.7 V battery that powers the board. The **LM3671** DC-DC converter steps this voltage down to 3.3 V which is used to power the ESP32-WROVER-IE microcontroller and the **CP2102**. The CP2102 is used to program the ESP32 using a **UART interface**. An external 2.4 GHz Omni-directional WiFi antenna connects to the internal antenna trace of the ESP32 to further extend the signal.

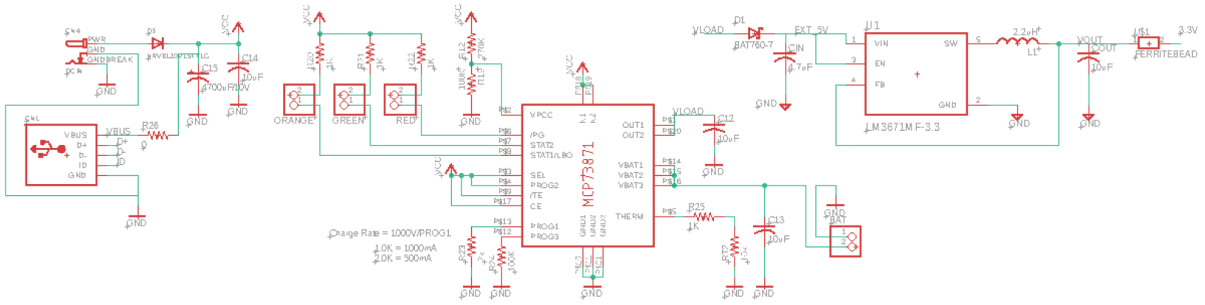


Figure 2. Power Circuitry

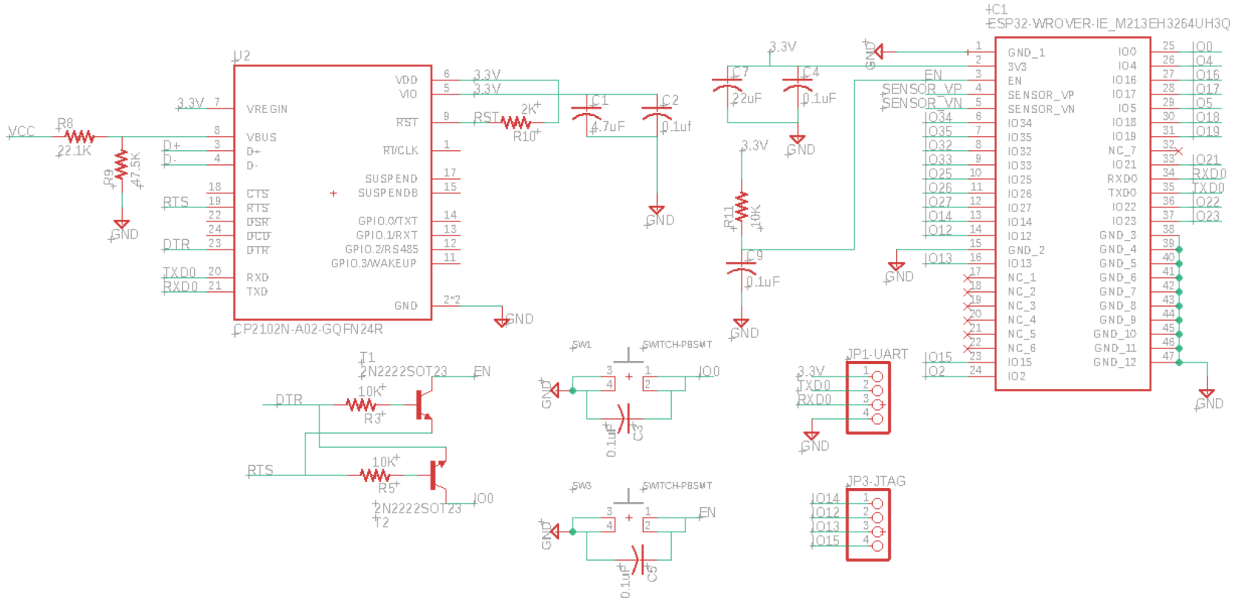


Figure 3. Programming Circuitry

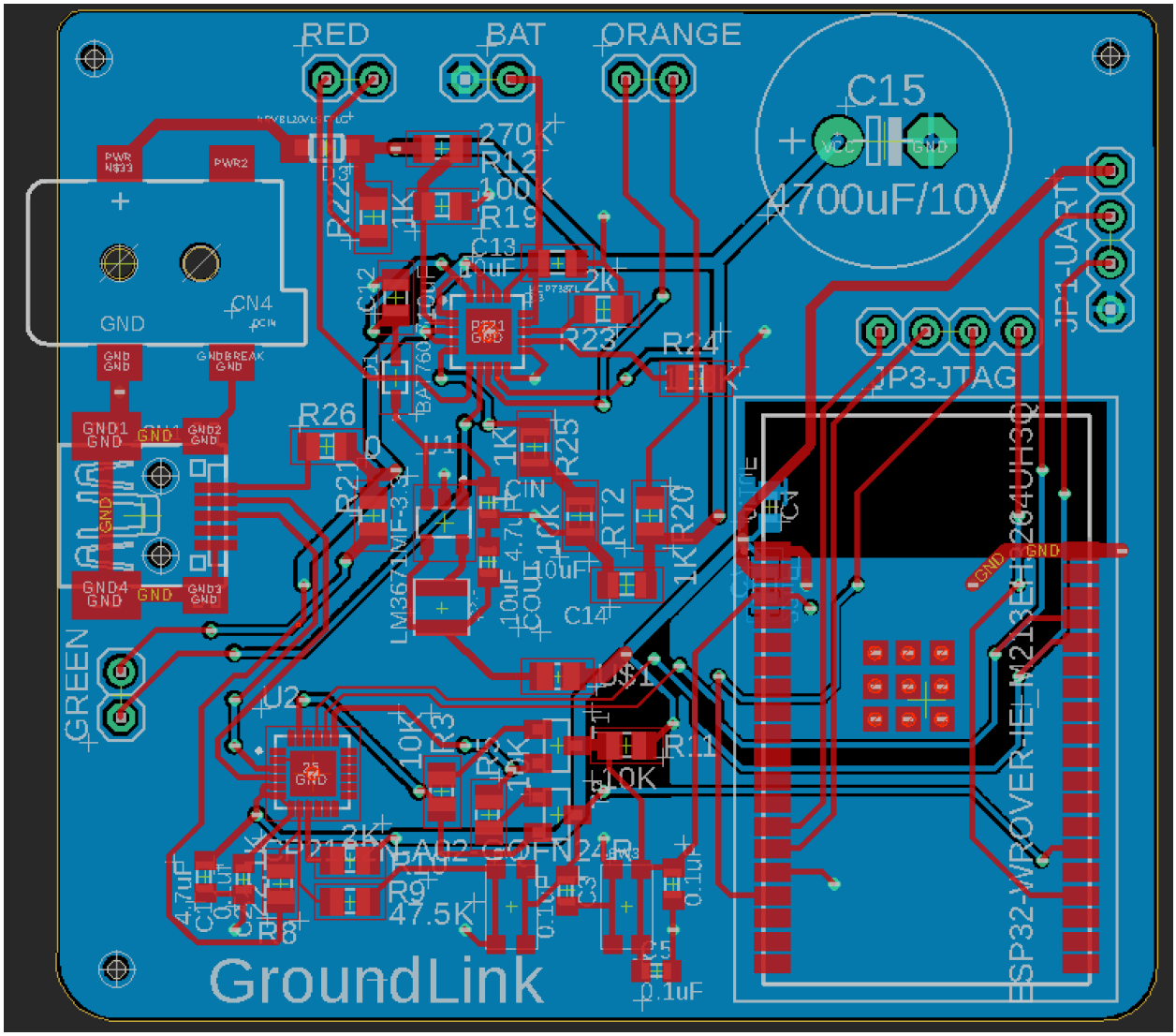
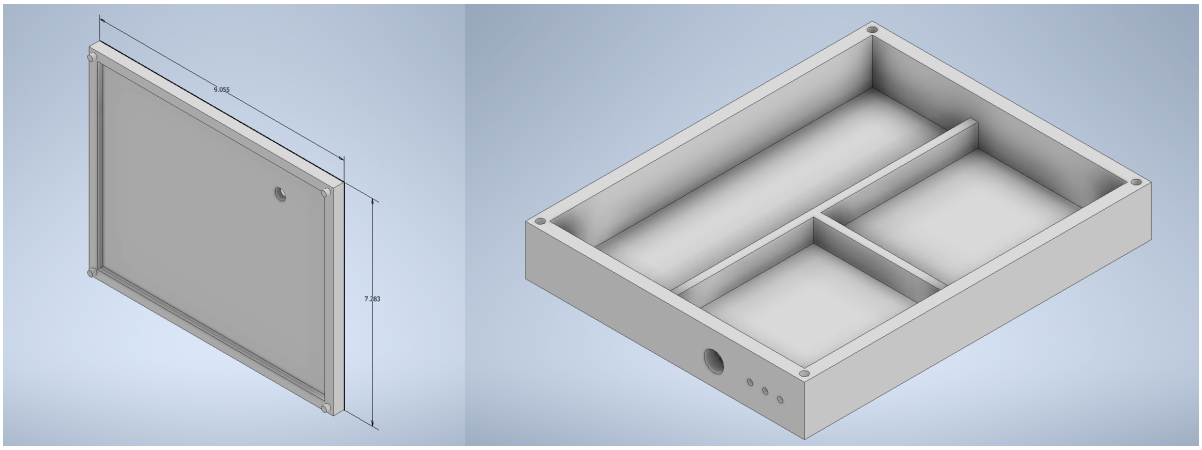


Figure 4. EAGLE Board Layout

Enclosure

The enclosure houses the board, the battery, antenna, and operation LEDs. It also serves as a stand for the solar panel. With the enclosure, the device is a self-contained unit ready for transport. The Navari Family Center for Digital Scholarship (CDS) was used in order to print the enclosure after its design in Autodesk Inventor.



Software

The software uses an **Network Address Translation** (NAT)-capable IP stack called "lwip", which Espressif includes in its SDK for the ESP32. Our application code is forked from free and open source code provided by Martin Ger (ESP32 Nat Router) which uses the underlying ESP IP stack to perform NAT routing between the **STA interface** and the **AP interface** on the chip. We modified the code to provide **additional configuration** options,

with the hope of using the underlying software stack to implement WPA Enterprise security (specifically PEAP). Unfortunately, due to inconsistencies between the networking stack included in the Espressif ESP32 SDK and other ESP PEAP implementations, coupled with the closed-source nature of the lowest levels of the stack, meant that implementation of this feature proved infeasible given our time constraints. We save this for future work.

Bill of Materials

The total cost of all the components in our project totaled to $277.09 and included the peripheral parts of our device and the components on our PCB as outlined in Figure 1 below.

Part Description	Source/Supplier	Part Number	Quantity	Cost/piece	Total Cost	Manufacturer	Manufacturer part number
Omni-Directional WiFi Antenna	Amazon		1	$8.59	$8.59	Highfine	B01KBU61S8
3mm Assorted LEDs	Amazon		1	$7.99	$7.99	DiCUNO	US 3MM-10T-200
USB / DC / Solar Lithium Ion/Polymer charger	Adafruit	390	1	$17.50	$17.50		
Huge 6V 6W Solar panel - 6.0 Watt	Adafruit	1525	1	$69.00	$70.50		
Lithium Ion Polymer Battery - 3.7v 2500mAh	Adafruit	328	1	$14.95	$14.95		
ESP32-WROVER-IE	DigiKey	1965-ESP32-WROVER-	3	$2.80	$8.40	Espressif Systems	ESP32-WROVER-IE (4MB)
Diode	digikey	BAT760-7	3	$0.56	$1.68	Diodes Incorporated	BAT760-7
Diode	digikey	NRVB120VLSFT1G	3	$0.43	$1.29	ON Semiconductor	NRVB120VLSFT1G
DC barrel connector	digikey	ADC-028-1-T/R-PA10T	3	$0.58	$1.74	Adam Tech	ADC-028-1-T/R-PA10T
Inductor	digikey	LQH3NPZ2R2MMEL	3	$0.36	$1.08	Murata	LQH3NPZ2R2MMEL
Capacitor	digikey	ESK478M010AL4AA	3	$0.89	$2.67	Kemet	ESK478M010AL4AA
Charge Controller IC	digikey	MCP73871-2CCI/ML	3	$1.88	$5.64	Microchip Technology	MCP73871-2CCI/ML
DC/DC Converter	digikey	LM3671MF-3.3/NOPB	3	$1.48	$4.44	TI	LM3671MF-3.3/NOPB
USB-UART bridge	digikey	CP2102-GMR	3	$3.54	$10.62	Silicon Labs	CP2102-GMR
PCB			2	$60.00	$120.00		

Figure 2. Bill of Materials

Conclusions

Summary

Our device successfully extends the range of a Wi-Fi network, pairing to a home network and the user's desired device. The team successfully met the requirements set in the Fall of 2020, with key subsystems all interacting together to accomplish the stated objective of Wi-Fi connection. Furthermore, our device was under budget and on schedule, which points to the feasibility of manufacturing at scale, especially with further modifications and improvements in product efficiency.

Recommendations

Fearing the complexity of the Espressif IDF, we wasted a lot of development time using the Arduino IDE, which appears more user-friendly on the surface. Unfortunately, using such an abstracted development framework means losing out on a lot of low-level functionality that our project would eventually depend on. For example, there is no way to implement NAT forwarding using the Arduino IDE because not enough of the network functions are exposed to the programmer. As it turns out, the Espressif IDF is not only more powerful, but it is also more pleasant to program in. Installation of the IDF is a breeze, and all of the tools run from a single function:

```
idf.py build           # Build project from source
idf.py flash           # Flash compiled project to device
idf.py monitor         # Open serial monitor
idf.py erase-all       # Erase memory
idf.py make-menuconfig # Pull up a GUI for setting configuration
```

Programming in the Espressif IDF is also straightforward, thanks to the well-written documentation and abundance of example projects included in the IDF. Finally, and perhaps most importantly, the make flow can be completely customized should users want to use different libraries. Overall, we would strongly recommend groups interested in the ESP32 to use the Espressif IDF from the get-go.

With respect to 3D printing and board delivery, we'd recommend other groups be prepared for longer than expected lead times. Such lead times could get in the way of progress, but our group managed to keep making progress on other aspects of the device such as software. Furthermore, 3D printing at the CDS may not always come out clean on the first iteration, so groups should expect to wait or make modifications to any 3D prints.

Future Work

In the future, there will be improvements to the fit and finish of the enclosure. Instead of 3D printing, one could use more advanced machining methods and materials. This will ensure weatherproofing and upgraded IP65 dust and water resistance for outdoor use. As an additional consideration for the future, shielding for EMI from RF sources and additional access points may be included. To the overall device, a USB A power out from the GroundLink device will also be added. The Wi-Fi repeater will be more versatile, providing battery charge to other devices as well. In addition, there will be increases to the

security of the web protocols. This includes transmission of configuration data over HTTPS and storing encrypted configuration information. Finally, with more time and communication with other members of the ESP community, WPA Enterprise support can be added to the device.

References

[Charge Controller Demo Board]

[ESP32 Dev-Kit]

[ESP-IDF SDK]

[Martin Ger's ESP32 NAT Router Repository]