Building My Own Linux-based Wireless Access Point

NOTE: This article is not complete.

Basically, I got everything working, but the signal strength was really low. I had this box and an Asus Wi-Fi router in the same spot, and the signal strength on this box was much lower (1-2 bars) versis the Asus router next to it (full bars).

I gave up because I needed the box for something else.

I'll finish this – maybe I'll give it a go again at some point in the future.

Contents

Context

First, My Router – A Linux-based PC

Before we get into my project, we'll talk about a journey I traveled before: building my own Linux-based router.

I Do This Because…

I like tinkering with networking. I find the ability and mechanics of running a small network from a residence, and having it securely interconnect with the world through the Internet, to be extremely interesting.

"Small Plastic Routers": For The Sane

"Small plastic routers" are common household devices now. (This is in contrast to "huge metal rackmount enterprise routers".)

Normal people tend to use equipment provided by their ISP that are "combo" boxes – those combine the functions of modem, router, wired Ethernet switch and wireless Ethernet switch into one box. Or they may buy their own equipment – modem, router, switch, or any combination of those.

Normal people just want a convenient, out-of-the-box solution to getting devices in their home online, and ISP-provided or store-bought devices are fine. These devices are obviously geared toward typical consumer Internet use and typical consumer technical knowledge, and that is quite lacking for the technically interested, curious network tinkerer.

However, there is a bigger problem with these small plastic routers … keep reading.

Aftermarket Firmware

Is there middle ground? Sure – in the late 2000's and 2010's most small plastic routers (most already running Linux) could be flashed with aftermarket firmware – DD-WRT likely being the most famous, but others such as Tomato and OpenWRT also out there. You're leaving "normal person" territory by doing this.

Limited Lifespan

Most small plastic routers have a limited lifespan, because source code for the networking hardware drivers are often not released-only binary modules. This is particularly a problem for devices with Broadcom hardware. This means the Linux kernel on these devices is not upgradeable because binary modules only work for a specific kernel version. With the source code you can rebuild for any kernel version that's compatible.

- You always want to be able to upgrade the kernel to take care of security vulnerabilities.
- Open source drivers often get updates for longer than manufacturers who don't release the code and abandon drivers for nonprofitable or old product lines.

I Decided To Not Be Sane

Hence I permanently left the small plastic router world about two years ago, and replaced my very-long-in-the-tooth Linksys E2100 with a Debian Linux-based PC. It's not the first time I used a PC as a router--I've been making a go at doing that in some fashion or another since I got my first broadband connection all the way back around 2005.

Benefits versus Drawbacks Of Linux-PC-As-A-Router

The benefits of using a Linux PC as a router are:

- No need to worry about out-of-date firmware.
    - Debian—the operating system I chose--always gets updates.
- What is running is completely under my control
    - No cloud or remote-management services running that I don't want or can't turn off.
- Able to put gobs of RAM and storage in the system to run advanced stuff right on the router.
    - If I WANT cloud services running on the router, I can add the RAM and storage they need to thrive.
- Able to put as many NICs and other hardware in the system as I want.
    - No need for VLANs when physically separate LANs can be used!

There are drawbacks:

- You need to know a lot about Linux, or be willing to learn.
    - It took me a long time to get this knowledge, and is something I enjoy. Not for everyone.
- You are 100% responsible for security.
    - This is a big deal.
- You are 100% responsible for everything, really. No more calling your ISP because your combo box stopped working.
    - This means you need to know your stuff with networking – I think I do a little bit.
- Power consumption is higher than a small plastic router.
- A PC takes up more space and is heavier than a small plastic router.

For my home network's router, I judged the benefits to outweigh the drawbacks for me – especially since learning and using Linux and its networking features are a source of fun to me.

What Does This Have To Do With A Wireless Access Point?

Any modern home network needs Wi-Fi. We're fully in the post-PC age now and wired devices are going to be in the minority. My lovely self-built router above is wired only.

Linux and Wi-Fi

Your basic non-wireless router needs 2 network interfaces. So if you want to take a PC that has 1 onboard NIC, and add a second NIC to make a wired-only router, it's easy – just buy the cheapest wired PCIe NIC you can find on Amazon and throw it in.

Can you do the same with Wi-Fi? Yep. Just like there are PCIe cards that add wired network jacks to a PC, there are PCIe cards that add Wi-Fi radios. But things are more complicated with Wi-Fi.

Wi-Fi interfaces have roles (modes)

Wi-Fi, apart from working using radio waves, has other important differences from wired networking. One of those is the fact that there are roles for Wi-Fi interfaces and the interface can only be in one of them: the important ones here are STA ("station" - system wanting Wi-Fi) or AP ("access point" – system providing Wi-Fi).

AP mode is needed

Not all Wi-Fi cards support AP mode – or more specifically, their *drivers* don't support AP mode or the manufacturer doesn't make them available. I can't blame them because most people buy Wi-Fi cards for laptops or other devices that connect to an AP. I'm the weird one here trying to build my own AP.

Furthermore, that's not Linux's fault - everything on the Linux end supports the AP mode fine and has for quite some time – because… most small plastic routers actually run Linux.

Hardware that supports AP mode is hard to find

It would be lovely if you could just open up a small plastic router, take out it's mini-PCIe card, and throw it in a PC. However, the Wi-Fi hardware in these plastic boxes is often specific to the box and part of the motherboard, and not something you can just buy in a PC Wi-Fi card. (You actually could pull the mini-PCI (not PCIe!) card out of the first hardware revision of the famous Linksys WRT54G).

- Because of this, I currently actually *do* have a small plastic router connected to my PC-as-a-router – but it's in AP mode and doesn't do any routing. More on that later.

It's actually a challenge to find PCIe Wi-Fi cards that both support modern Wi-Fi speeds (802.11ac) and AP mode, but they are out there – and I bought a couple.

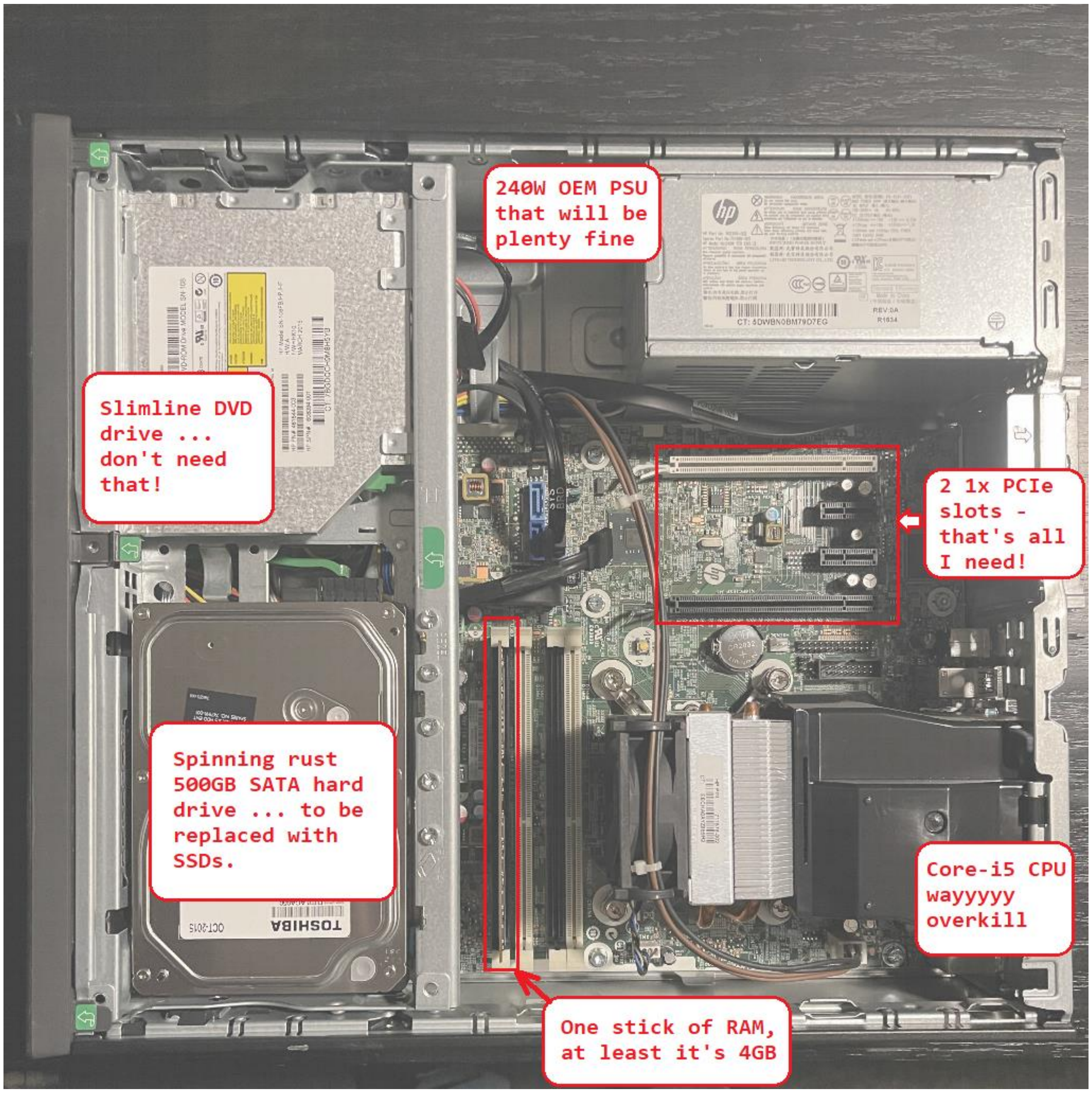
Gathering The Hardware

The PC

Well, if we're gonna build a PC-based wireless access point, we need a … PC!

My workplace graciously let me have a decommissioned HP EliteDesk 800 G1 SFF. Decent desktop office computer from circa 2015.

It's big, but not too big.   Let's have a look at the innards.


240W OEM PSU that will be plenty fine
Slimline DVD drive ... don't need that!
2 1x PCIe slots - that's all I need!
Spinning rust 500GB SATA hard drive ... to be replaced with SSDs.
Core-i5 CPU wayyyyy overkill
One stick of RAM, at least it's 4GB

So we have what's probably a dual-core Intel Core i5, single stick of 4GBytes of RAM, enough PCIe slots for our Wi-Fi cards, and enough SATA ports to install 2 SSDs in a RAID-1 configuration.

Importantly it features an Ethernet port on the back, so we can bridge the Wi-Fi into a wired network or router. I haven't checked yet but I'm certain it's gigabit.

Firmware Settings

When entering the firmware screens, I was expecting HP's newer graphical UEFI menu, and was pleasantly disappointed. Random opinion no one asked for: I like the no-nonsense Compaq menu quite a lot.

Anyway, looking around reveals that this box supports virtualization (both VMX and VT-d), Secure Boot (disabled), and Intel AMT (also disabled).

We don't need any of that, so we'll disable all of those functions.

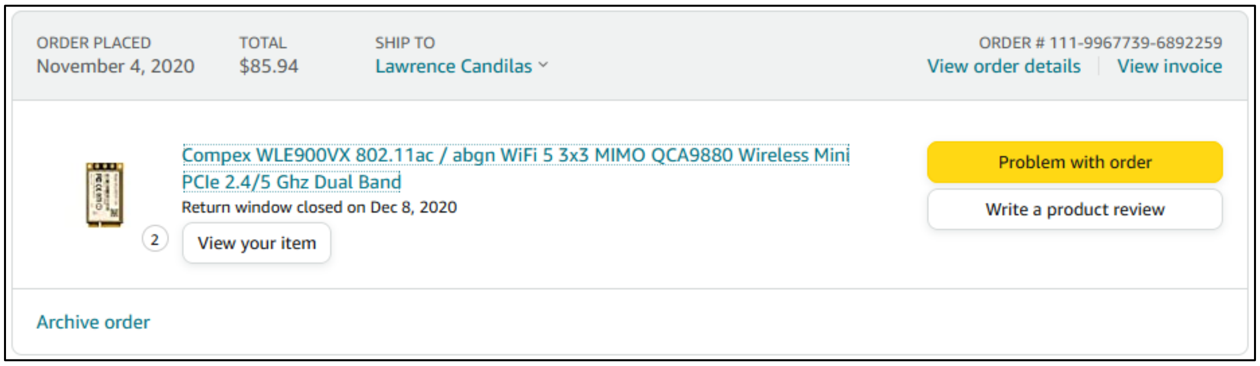
Verdict

This PC will be far more than adequate for this purpose, from a base hardware standpoint.

The Wi-Fi Cards

Wi-Fi Cards

So here is my Amazon order for my selected Wi-Fi card – the WLE900VX, supporting 802.11ac.



ORDER PLACED
November 4, 2020

TOTAL
$85.94

SHIP TO
Lawrence Candilas

ORDER # 111-9967739-6892259
View order details | View invoice

Compex WLE900VX 802.11ac / abgn WiFi 5 3x3 MIMO QCA9880 Wireless Mini PCIe 2.4/5 Ghz Dual Band
Return window closed on Dec 8, 2020

This was selected after some research about the QCA9880, and finding that the QCA9880 had good Linux support, AP mode support, and everything needed for an access point.

- You'll notice the date – November 4, 2020.
- Yep, a few years ago.
- Confession: I actually tried to get wireless working right on my main PC-based router, but it did not work out, and I'll detail that later.
- So this isn't really my first rodeo with this, but maybe it'll work this time!

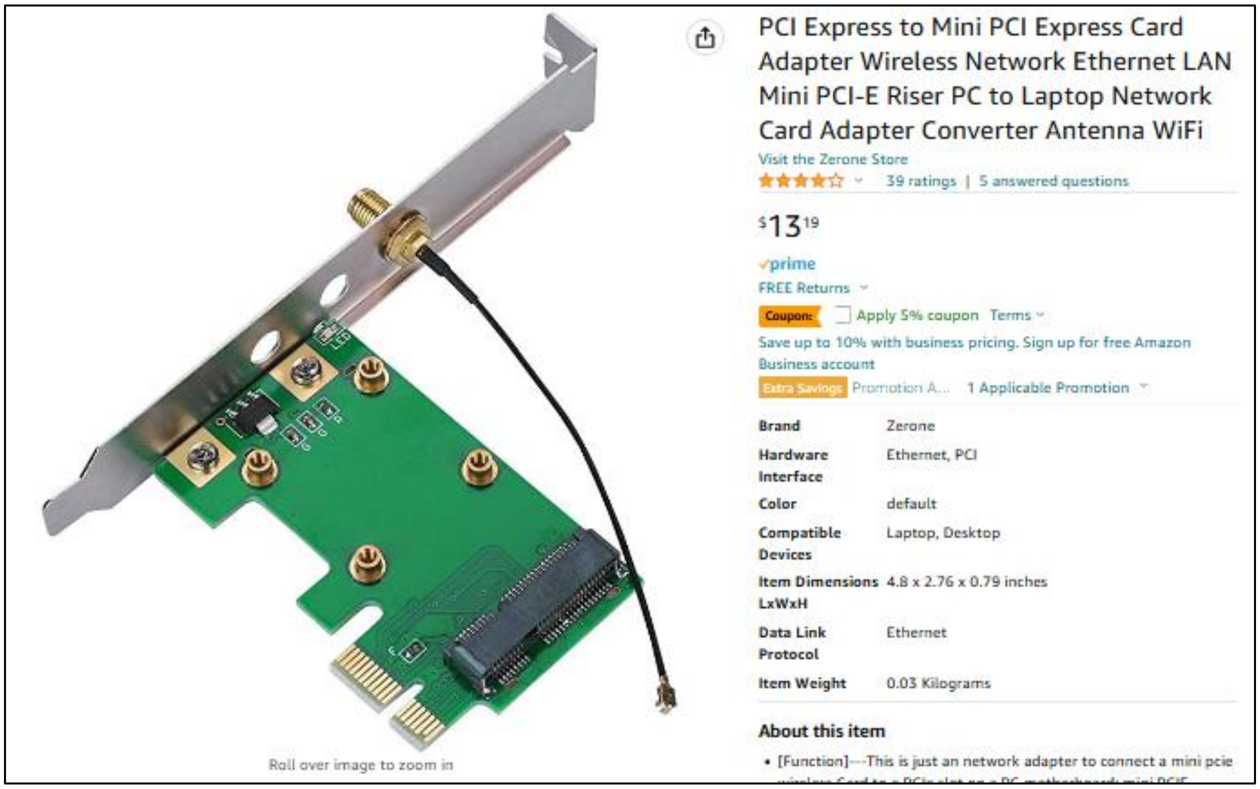
You'll notice I ordered two. That's because this card's radio works on *either* the 2.4GHz or 5Ghz bands, but not both. So two radios are needed.

- You're seeing a snapshot of my Amazon order because--it's not available anymore! Guess I'm lucky I got it when I did.

Turning The Cards Into Something That Can Be Connected To A PC

So the motherboard of the PC I'm using does not have onboard mini-PCIe slots. Not surprising – I think having one for Wi-Fi on desktop motherboards is becoming a thing but it definitely wasn't a thing on business computers in 2015.

No big deal, we'll just get two of these:



PCI Express to Mini PCI Express Card Adapter Wireless Network Ethernet LAN Mini PCI-E Riser PC to Laptop Network Card Adapter Converter Antenna WiFi

Visit the Zerone Store

★★★★☆ 39 ratings | 5 answered questions

$13.19

✓prime

FREE Returns

Coupon: Apply 5% coupon Terms

Save up to 10% with business pricing. Sign up for free Amazon Business account

Extra Savings Promotion A... 1 Applicable Promotion

Brand	Zerone
Hardware Interface	Ethernet, PCI
Color	default
Compatible Devices	Laptop, Desktop
Item Dimensions LxWxH	4.8 x 2.76 x 0.79 inches
Data Link Protocol	Ethernet
Item Weight	0.03 Kilograms

About this item

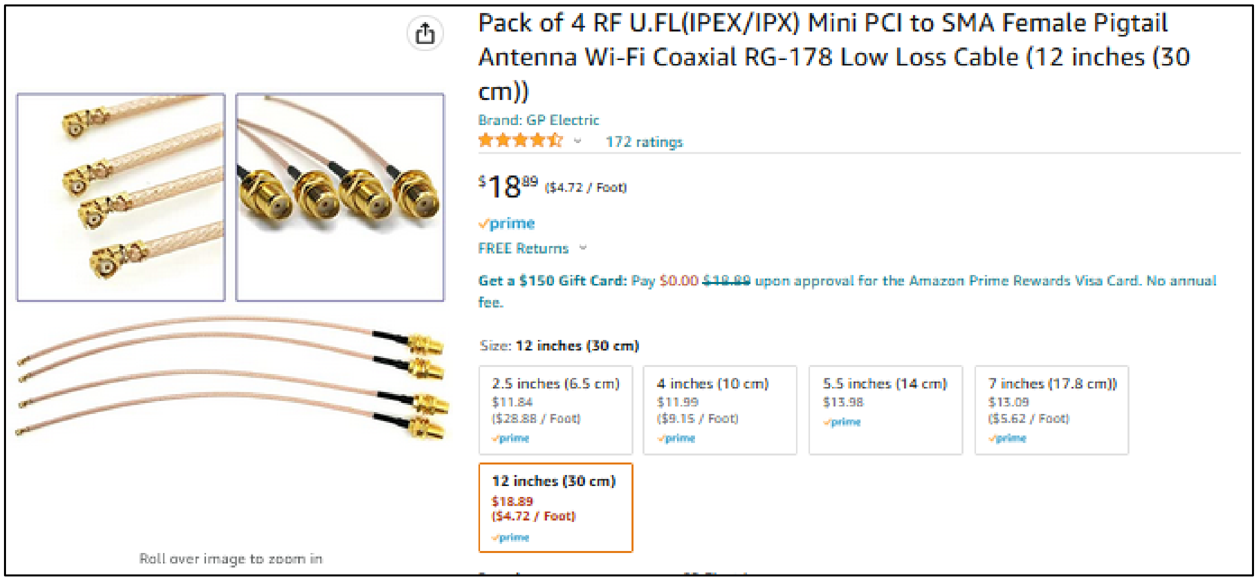
- [Function]---This is just an network adapter to connect a mini pcie

Roll over image to zoom in

The mini-PCIe card will connect right to that, and it has convenient holes for the antenna pigtails.

More Pigtails

The WLE900VX has 3xMIMO meaning … it uses 3 antennas. Glad the adapter I chose had 3 holes for the pigtails, but it only came with 1 pigtail. No biggie, just had to get 2 sets of thesez:



Pack of 4 RF U.FL(IPEX/IPX) Mini PCI to SMA Female Pigtail Antenna Wi-Fi Coaxial RG-178 Low Loss Cable (12 inches (30 cm))

Brand: GP Electric

★★★★☆ 172 ratings

$18.89 ($4.72 / Foot)

✓prime

FREE Returns

Get a $150 Gift Card: Pay $0.00 ~~$18.89~~ upon approval for the Amazon Prime Rewards Visa Card. No annual fee.

Size: 12 inches (30 cm)

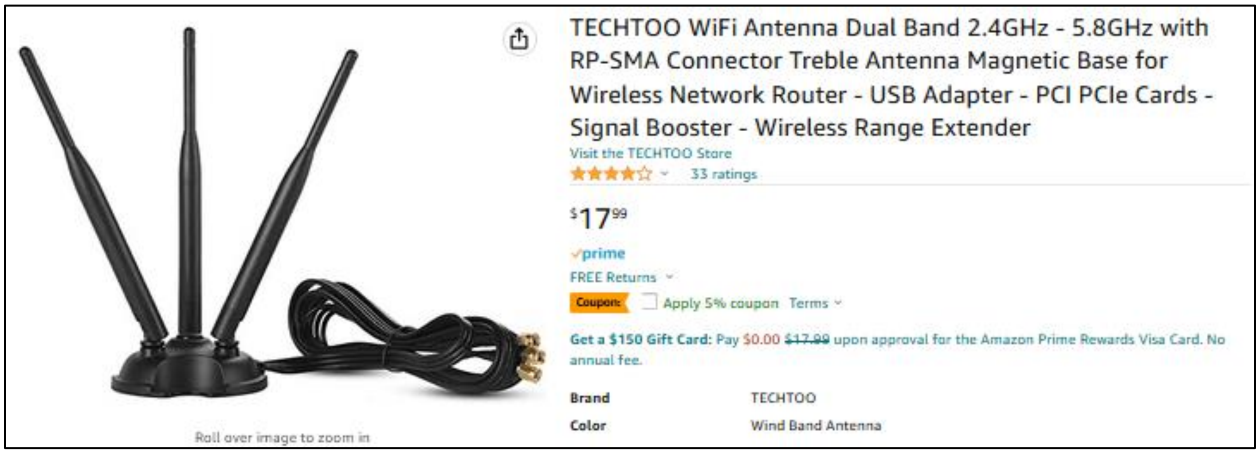
2.5 inches (6.5 cm) $11.84 ($28.88 / Foot) ✓prime	4 inches (10 cm) $11.99 ($9.15 / Foot) ✓prime	5.5 inches (14 cm) $13.98 ✓prime	7 inches (17.8 cm)) $13.09 ($5.62 / Foot) ✓prime
12 inches (30 cm) $18.89 ($4.72 / Foot) ✓prime			

Roll over image to zoom in

Incidentally, I learned the name of the connector that the antennas screw into is called SMA.

Antennas

I had the feeling I would need antennas, so I got two of those:



TECHTOO WiFi Antenna Dual Band 2.4GHz - 5.8GHz with RP-SMA Connector Treble Antenna Magnetic Base for Wireless Network Router - USB Adapter - PCI PCIe Cards - Signal Booster - Wireless Range Extender

Visit the TECHTOO Store

33 ratings

$17.99

prime

FREE Returns

Coupon: Apply 5% coupon Terms

Get a $150 Gift Card: Pay $0.00 ~~$17.99~~ upon approval for the Amazon Prime Rewards Visa Card. No annual fee.

Brand	TECHTOO
Color	Wind Band Antenna

Roll over image to zoom in

SMA compatible—and also the screw connectors look like they'll fit.

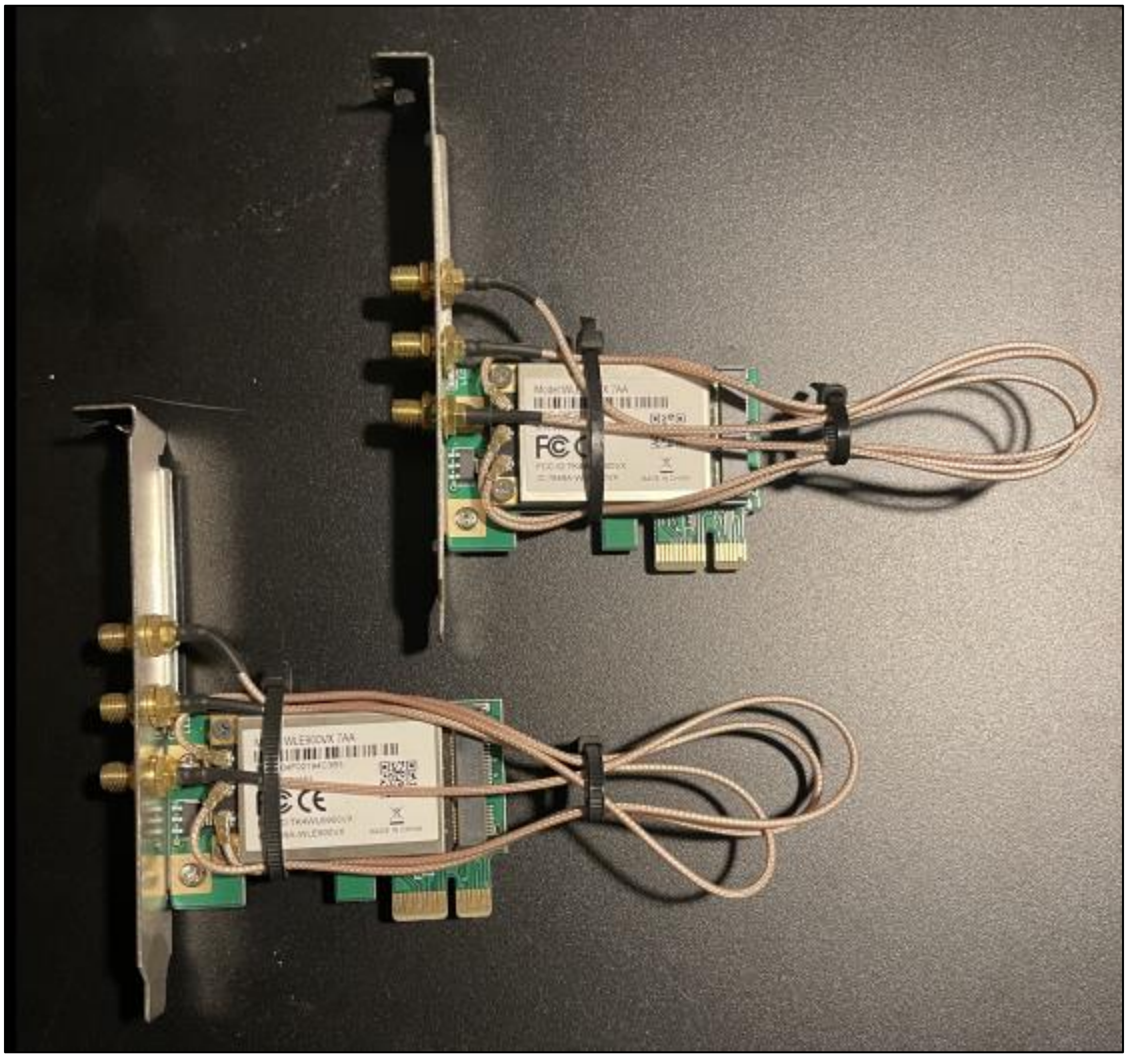
So those will plug right into the back of the cards from the PC.

Another cool thing about these not mentioned in the product description is that the base is magnetic. It conveniently attaches to metal.

Assembling The Cards

Some assembly required: namely, putting the mini-PCIe cards in the adapters, and connecting the pigtails from the cards to the back of the adapter bracket.

To keep the pigtails out of the way, I used 2 zip ties. Here's the result:



Ugly, sure—and that will be a recurring theme here.

I'm not sure if this is good for a radio interference perspective. I'm wondering if shorter ones would be better, but it's functional. Maybe if I put aluminum foil in between the wires to protect the signal…

Just kidding, I didn't do that.

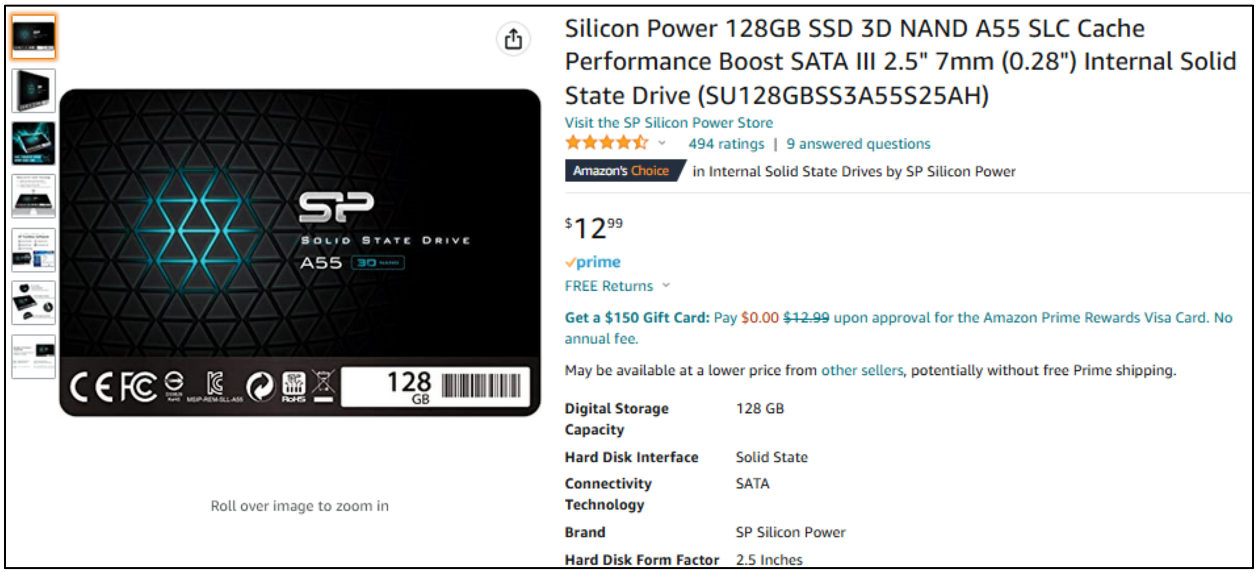
SSDs for the operating system

So we need to provide a home for Linux on this box.

If you were paying attention to the photos of the PC above, this system came with a 500GB mechanical hard drive. There is no reason in 2023 to be using a mechanical hard drive for anything other than bulk storage, especially in an "infrastructure" device that's going to sit out of sight, potentially for years, and not even have a monitor connected.

128GB SATA SSDs are dirt cheap at the moment. So I got 2 of them – a non-major brand that I've used before.

We will install Linux on a RAID-1 for reliability. For $12.99, why not.



Silicon Power 128GB SSD 3D NAND A55 SLC Cache Performance Boost SATA III 2.5" 7mm (0.28") Internal Solid State Drive (SU128GBSS3A55S25AH)

Visit the SP Silicon Power Store

494 ratings | 9 answered questions

Amazon's Choice in Internal Solid State Drives by SP Silicon Power

$12.99

prime

FREE Returns

Get a $150 Gift Card: Pay $0.00 ~~$12.99~~ upon approval for the Amazon Prime Rewards Visa Card. No annual fee.

May be available at a lower price from other sellers, potentially without free Prime shipping.

Digital Storage Capacity	128 GB
Hard Disk Interface	Solid State
Connectivity Technology	SATA
Brand	SP Silicon Power
Hard Disk Form Factor	2.5 Inches

Pros and cons of other options here

Of course… other options were possible:

- Get 2 NVMe to PCI adapters and use NVMe,
  - BUT that's more expensive.
  - Also I don't know if this motherboard is weird about devices other than graphics cards in the x16 slot. This is business OEM hardware here that sometimes is weird like that.
- Could have installed and ran Linux off of USB storage device,
  - BUT they're not too reliable, slow, and might be accidentally removed.
  - RAID over multiple USB storage devices is possible but probably a big pain.
  - Regarding slowness, while I don't need super high storage performance I would like it to boot quickly

  Run it from the optical drive that's in the system,
  - BUT
    - Probably would take ages to boot.
    - The optical drive might not even work.
    - I don't even know if I have any blank CDs or DVDs anywhere.
    - It's 2023.

- Setup PXE boot off of my main router or home server,
  - BUT I want this to be a standalone device and not dependent on my router.
    - However with this I could have avoided putting any storage in the box at all.
    - Maybe another time.
- Get 2 SD-card-to-SATA adapters and use SD cards.
  - BUT that's actually more expensive than 2 128GB SSDs.
  - It'll also probably be slow unless I get expensive SD cards.
  - SD cards are sometimes unreliable.

Is 128GB enough?

If you think it isn't, you are looking at this from a Windows or desktop Linux perspective. We'll be installing the minimum we need to get this working.

- The base Debian Linux operating system is very lean and can comfortably run in 8GB if you don't need GUI tools.
- If we were to do things like run servers, web interfaces, etc. we would need more space.

So 128GB is more than enough. I would have selected a lower capacity SSD drive if it was cheaper than the 128GB; it seems 128GB is the sweet spot right now, at least on Amazon.

Assembly

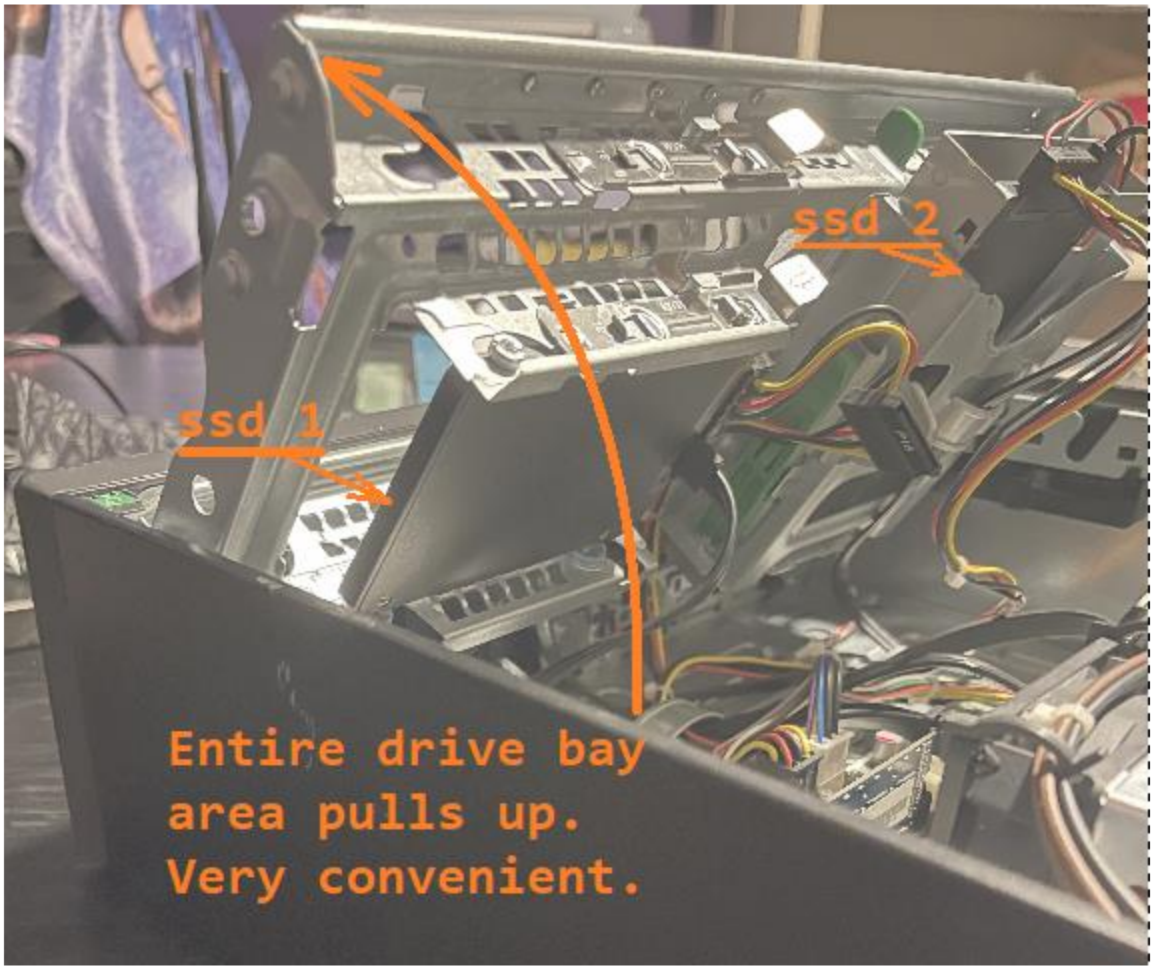
Installing the SSDs

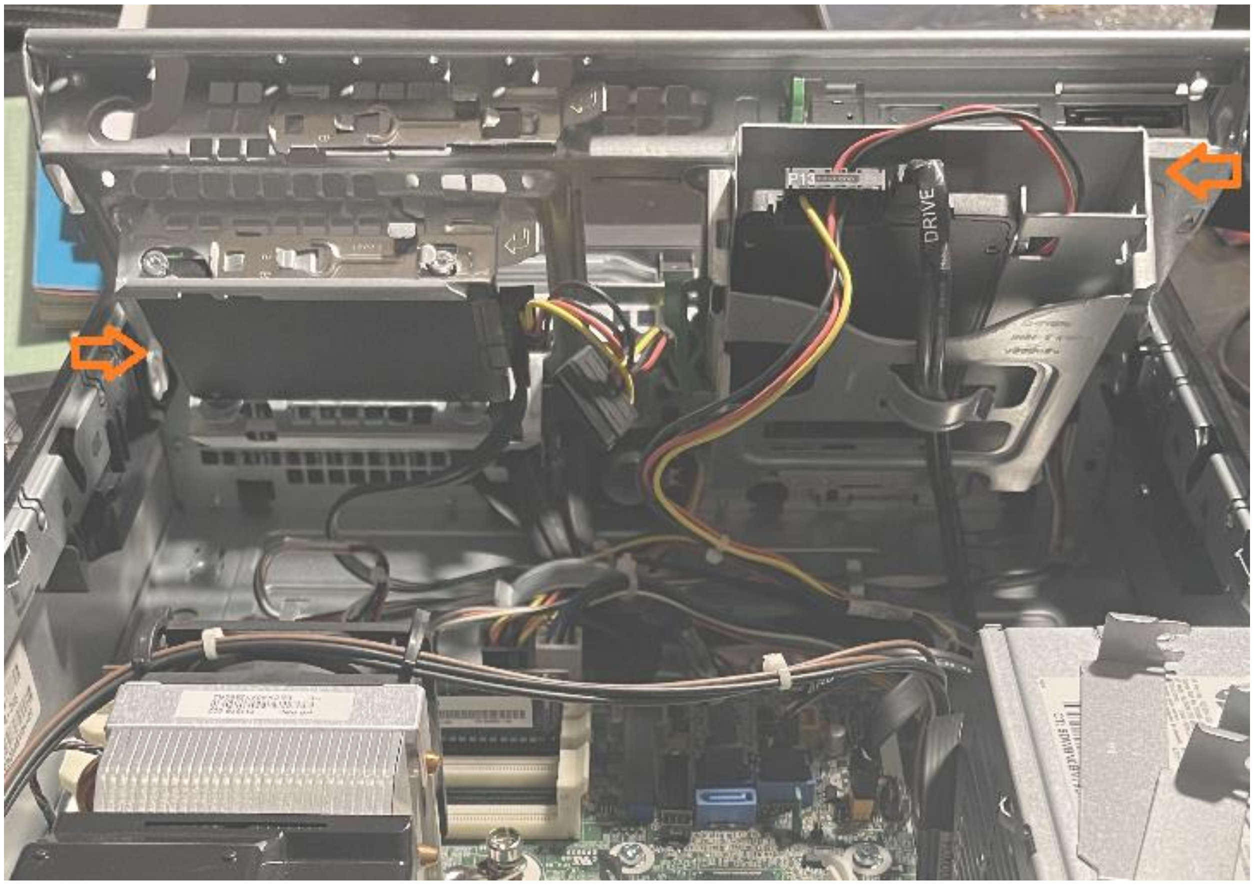
This wasn’t too bad.

Looks like this PC has room enough for 2 3-inch hard drives and 1 2.5-inch hard drive.

One SSD will go in the 2.5-inch space and I have an adapter to put the other in the 3-inch space. Below you see the SSDs installed and the old mechanical hard drive removed.

I know these pictures are terrible, but it’s OK. This whole thing is terrible. I should not be doing this.





We will plug our SSDs into SATA0 and SATA1 on the motherboard.

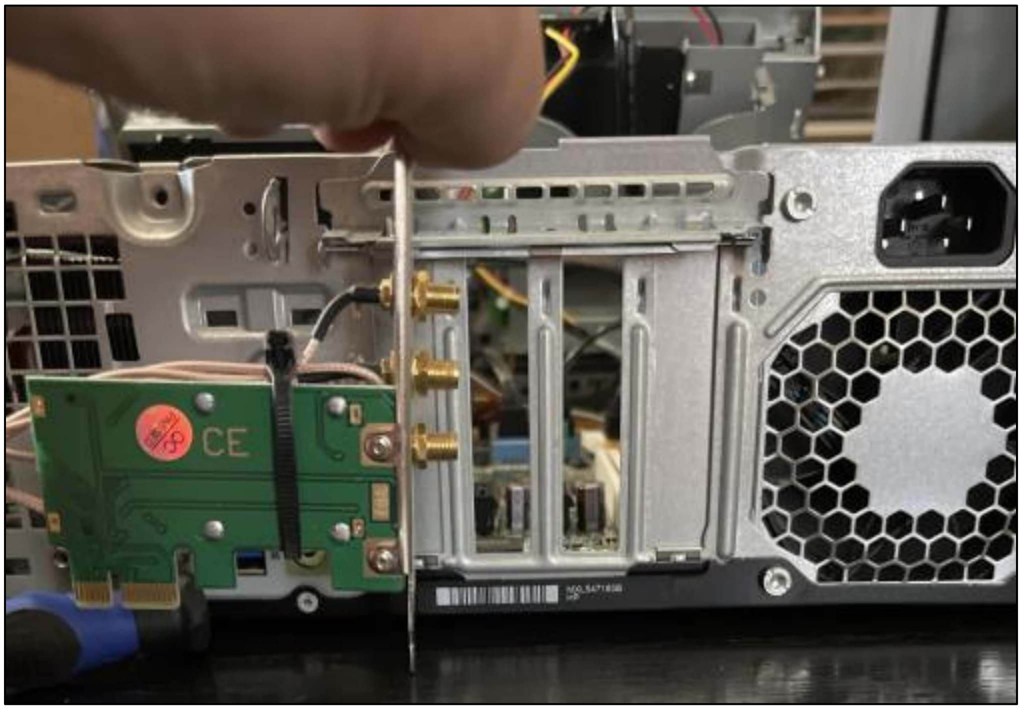
I’m leaving the optical drive disconnected but in the chassis – don’t really have a use for that and don’t have anything to replace it at the moment.

(If you’re looking at the motherboard picture, you’ll see SATA3 and SATA4 and I think a SATA5—there’s solder pads but no port there.)

Installing the Wi-Fi cards

Well, looks like I made a mistake.

If you were particularly astute you may have noticed it and were chuckling up to this point.



I didn't order LOW-PROFILE adapters – but to be fair, the *original* computer I put these into (my precious PC-based Linux router I talked about earlier) was not low-profile.

I could, of course, just stick the cards in there without a bracket and let the pigtails drop outside of the chassis. I'm afraid I will accidentally pull on something and damage it, though.

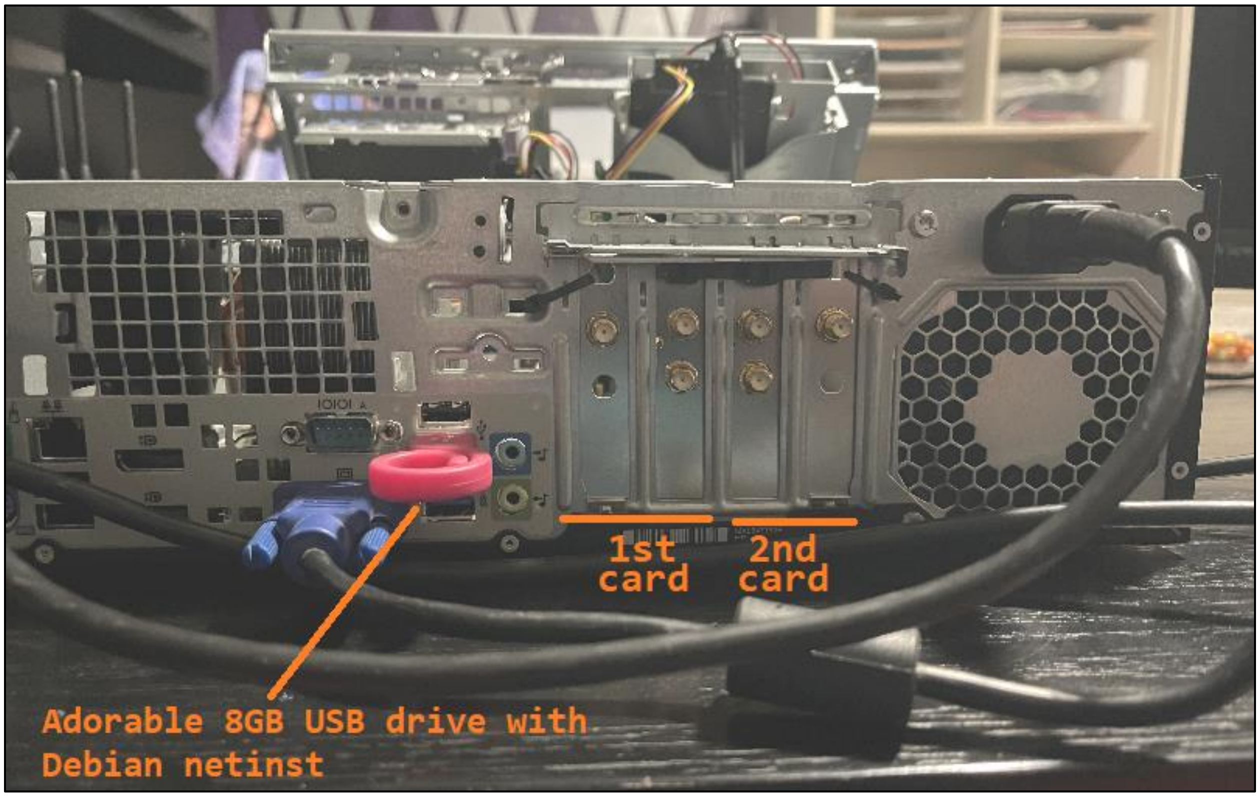
Bolt Cutters And Zip Ties

What I can do is just cut the existing brackets down to size.

That presents a second problem, the top antenna hole is covered by the clasp that keeps the cards secured. I could remove that clasp (somehow), or … for some reason I have some additional normal-profile brackets with 4 antenna holes—I really don't know why I have these.

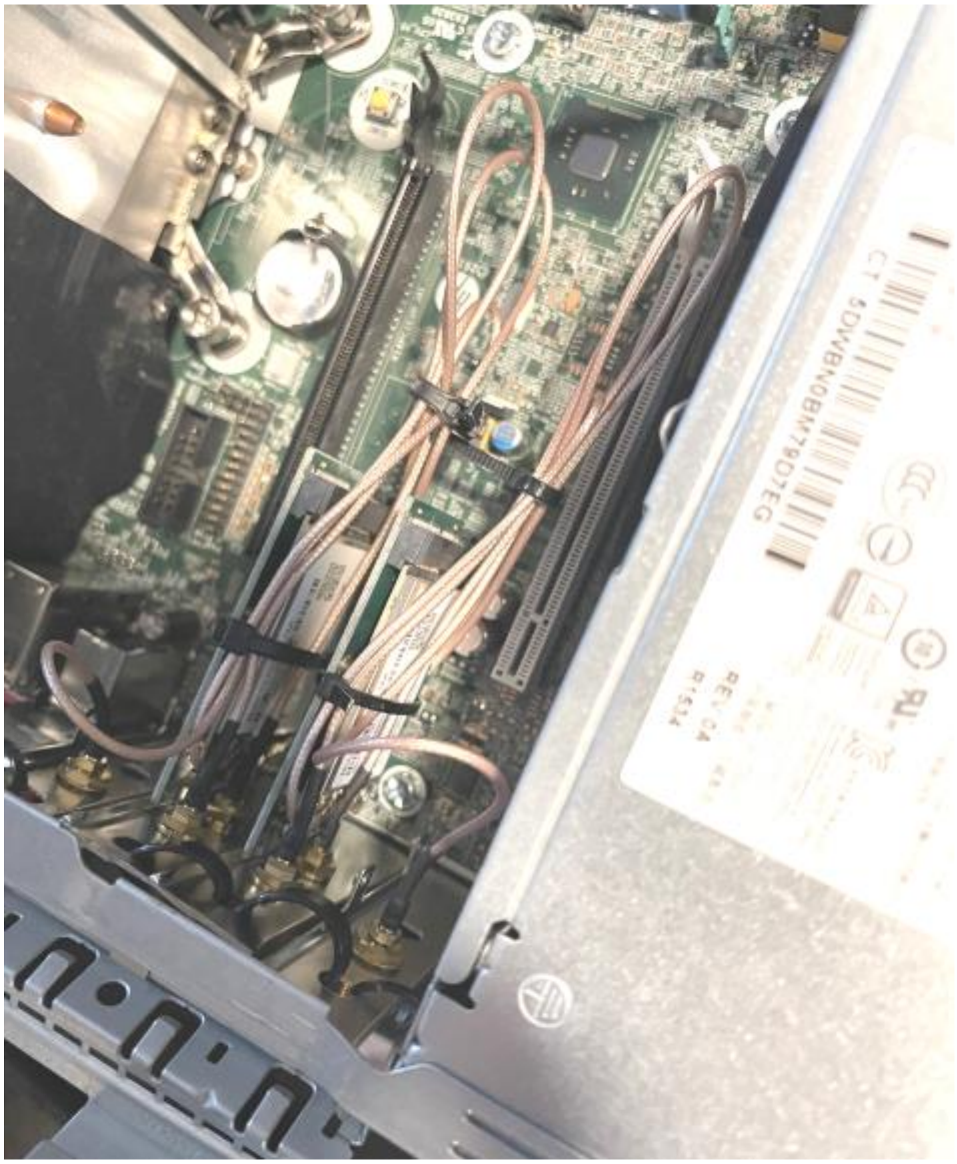
I can thread 2 antennas through the original brackets and 1 antenna through the other brackets.

And that’s what I did. Here’s the beautiful result.



Things are reasonably steady there.

Here’s a top-down view.



Installing Linux

Appliance Philosophy – Not Desktop

My Linux-based access point is basically an appliance—one that's going to be directly providing connectivity. and I want to approach the operating system installation and configuration like an appliance. Meaning:

- The less installed, the better. Less software installed means less to update, less to break, and less to worry about. (Told you that 128GB SSD was overkill)
- We don't need desktop GUI stuff on a box that will basically be out of the way and invisible. We can SSH into this and do whatever we need.

I'm wanting to start with as barebones of a Linux installation as possible, but with the option to easily install anything that's needed.

Why I Chose Debian

Debian is famous for its dedication to FOSS ideals, dedication to stability over new features, dedication to keeping up with security updates, and has been around since 1993. It's not going anywhere. Plus I've been using it in one form or another since 2004.

Plus…

The Debian `netinst` image

Debian's `netinst` install fits our minimalist "appliance philosophy" perfectly.

Nothing will be installed at first except the bare minimum to get a useable system, but there are the literally tens of thousands of Debian packages to choose from and install later if needed (and of course external `.deb`'s can be installed as needed).

The intent of `netinst` is to allow installation over the Internet without having to download all of the CDs or DVDs that make the entire distribution. You are able to select collections of packages, then it downloads and installs straight from the Internet.

Of course you do not have to select any collections for a true, minimal, barebones system – you simply select nothing. Some of the collections are useful, such as "SSH Server" and "Basic system utilities."

Actual Installation Process

Legacy Mode

This hardware is UEFI based with support for "legacy mode." I attempted to use UEFI mode and ran into problems—documentation forthcoming. So a step was to put the firmware into legacy mode.

If interested in the nitty-gritty details..

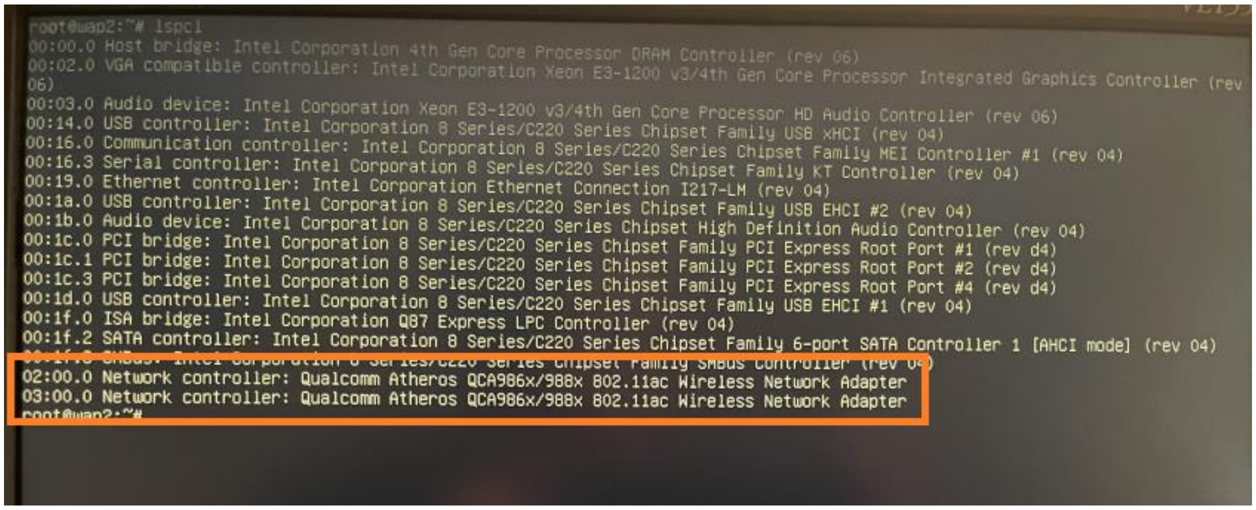
For all the details about installing Debian, which could be applicable to more than just this guide, please see my separate document "Installing Debian Linux netinst Image On A Two-128GB-SATA RAID-1" for the steps of the installation process, if interested.

Finding The Firmware For Our Wi-Fi Cards

All right, we have Linux installed and booting.

Now to make sure both Wi-Fi cards that are installed are recognized by Linux. Fairly certain they are just because of the firmware prompt during the installation, but we'll check again for the heck of it.

We'll use the **`lspci`** command - it gives us a list of all PCIe devices it sees in the system.



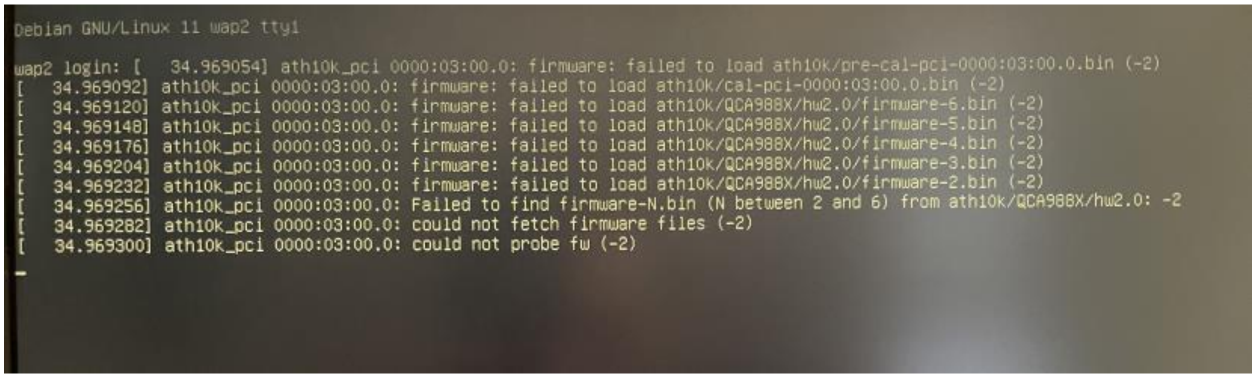
```
root@wap2:~# lspci
00:00.0 Host bridge: Intel Corporation 4th Gen Core Processor DRAM Controller (rev 06)
00:02.0 VGA compatible controller: Intel Corporation Xeon E3-1200 v3/4th Gen Core Processor Integrated Graphics Controller (rev
06)
00:03.0 Audio device: Intel Corporation Xeon E3-1200 v3/4th Gen Core Processor HD Audio Controller (rev 06)
00:14.0 USB controller: Intel Corporation 8 Series/C220 Series Chipset Family USB xHCI (rev 04)
00:16.0 Communication controller: Intel Corporation 8 Series/C220 Series Chipset Family MEI Controller #1 (rev 04)
00:16.3 Serial controller: Intel Corporation 8 Series/C220 Series Chipset Family KT Controller (rev 04)
00:19.0 Ethernet controller: Intel Corporation Ethernet Connection I217-LM (rev 04)
00:1a.0 USB controller: Intel Corporation 8 Series/C220 Series Chipset Family USB EHCI #2 (rev 04)
00:1b.0 Audio device: Intel Corporation 8 Series/C220 Series Chipset High Definition Audio Controller (rev 04)
00:1c.0 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #1 (rev d4)
00:1c.1 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #2 (rev d4)
00:1c.3 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #4 (rev d4)
00:1d.0 USB controller: Intel Corporation 8 Series/C220 Series Chipset Family USB EHCI #1 (rev 04)
00:1f.0 ISA bridge: Intel Corporation Q87 Express LPC Controller (rev 04)
00:1f.2 SATA controller: Intel Corporation 8 Series/C220 Series Chipset Family 6-port SATA Controller 1 [AHCI mode] (rev 04)
00:1f.3 SMBus: Intel Corporation 8 Series/C220 Series Chipset Family SMBus Controller (rev 04)
02:00.0 Network controller: Qualcomm Atheros QCA986x/988x 802.11ac Wireless Network Adapter
03:00.0 Network controller: Qualcomm Atheros QCA986x/988x 802.11ac Wireless Network Adapter
root@wap2:~#
```

We can see both of our "**Qualcomm Atheros QCA986x/988x 802.11ac Wireless Network Adapter**" devices.

But we don't have working Wi-Fi interfaces visible yet…

Device firmware errors – expected because Debian

So, upon my first boot after the OS install, after the login prompt came up, this spit forth:



```
Debian GNU/Linux 11 wap2 tty1

wap2 login: [   34.969054] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/pre-cal-pci-0000:03:00.0.bin (-2)
[   34.969092] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/cal-pci-0000:03:00.0.bin (-2)
[   34.969120] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-6.bin (-2)
[   34.969148] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-5.bin (-2)
[   34.969176] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-4.bin (-2)
[   34.969204] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-3.bin (-2)
[   34.969232] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-2.bin (-2)
[   34.969256] ath10k_pci 0000:03:00.0: Failed to find firmware-N.bin (N between 2 and 6) from ath10k/QCA988X/hw2.0: -2
[   34.969282] ath10k_pci 0000:03:00.0: could not fetch firmware files (-2)
[   34.969300] ath10k_pci 0000:03:00.0: could not probe fw (-2)
_
```

These are "firmware: failed to load" messages on boot related to our lovely Wi-Fi cards.

Many Wi-Fi cards will not do anything (even if recognized) on startup or connection until Linux sends them the device firmware. Linux normally does this when setting up the device, but it can't if the device firmware files are missing.

This is totally expected, and the Debian installer warned us in the beginning of the install process.

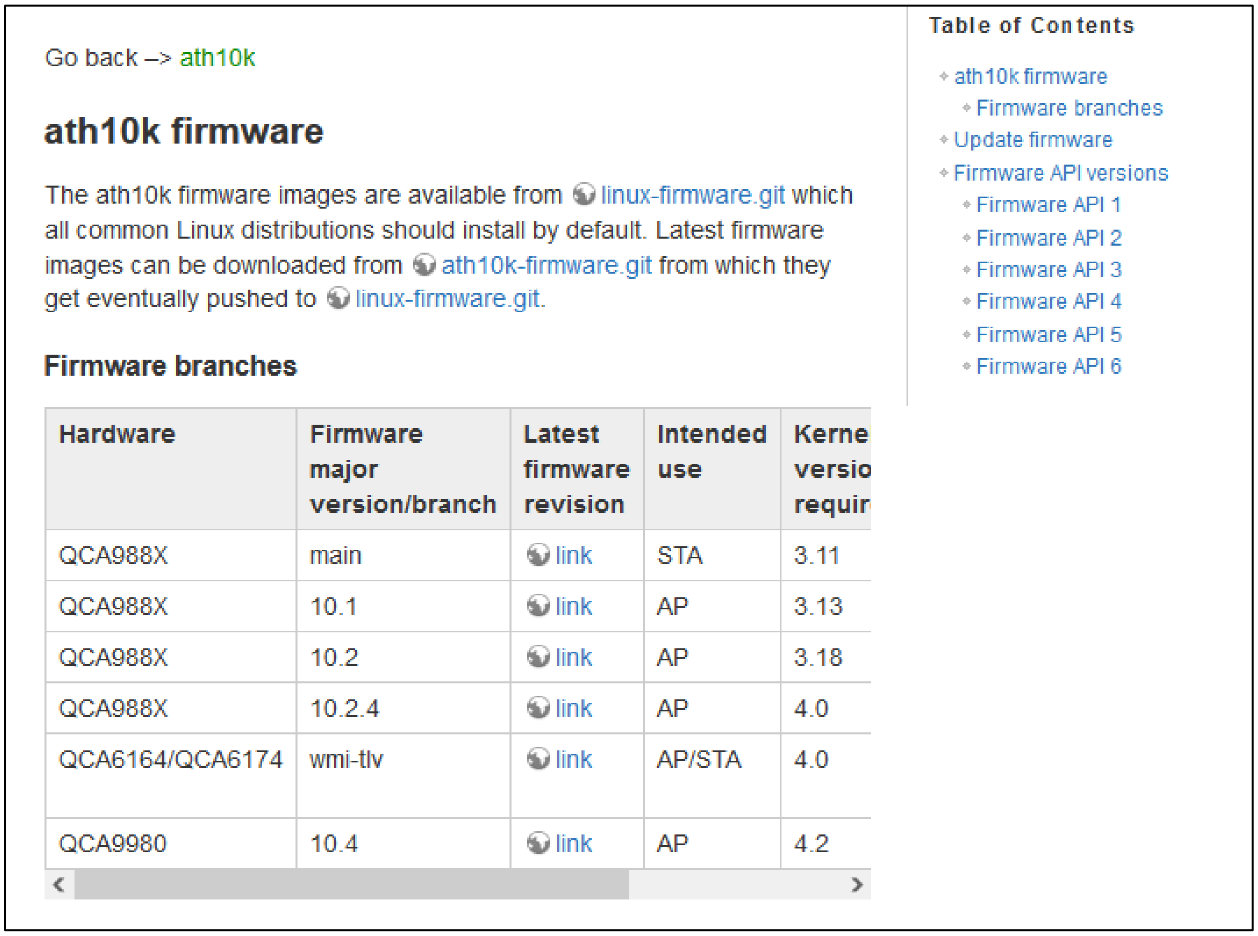
- Why do some peripherals require firmware files?
  - Peripherals are like small computers, they have a small CPU and amount of RAM on them and run code that makes the device behave a certain way.
  - Wi-Fi cards tend to not have the code built-in to the device, but want the host PC to give it the code it needs when it sets the device up on boot or connection.
- Why doesn't Debian include this firmware?
  - Debian doesn't include this code, or firmware, because it's not released under a FOSS compatible license.

Finding the QCA9880 device firmware

Not to worry. Debian doesn't include the firmware so we have to get them ourselves.

The Amazon product description said these had a QCA9880 chipset, so we'll look for device firmware for that model.

Some searching led me to this page:
`https://wireless.wiki.kernel.org/en/users/Drivers/ath10k/firmware`



Go back –> ath10k

ath10k firmware

The ath10k firmware images are available from linux-firmware.git which all common Linux distributions should install by default. Latest firmware images can be downloaded from ath10k-firmware.git from which they get eventually pushed to linux-firmware.git.

Firmware branches

Hardware	Firmware major version/branch	Latest firmware revision	Intended use	Kerne versio requir
QCA988X	main	link	STA	3.11
QCA988X	10.1	link	AP	3.13
QCA988X	10.2	link	AP	3.18
QCA988X	10.2.4	link	AP	4.0
QCA6164/QCA6174	wmi-tlv	link	AP/STA	4.0
QCA9980	10.4	link	AP	4.2

Table of Contents

After a bit more looking and experimenting, what I ended up doing that worked was this:

- I actually downloaded the latest 10.2.4.70 **`firmware-5.bin`** file from here (it was the newest):

[`https://github.com/kvalo/ath10k-firmware/blob/master/QCA988X/hw2.0/10.2.4.70/firmware-5.bin_10.2.4.70.70`](https://github.com/kvalo/ath10k-firmware/blob/master/QCA988X/hw2.0/10.2.4.70/firmware-5.bin_10.2.4.70.70)

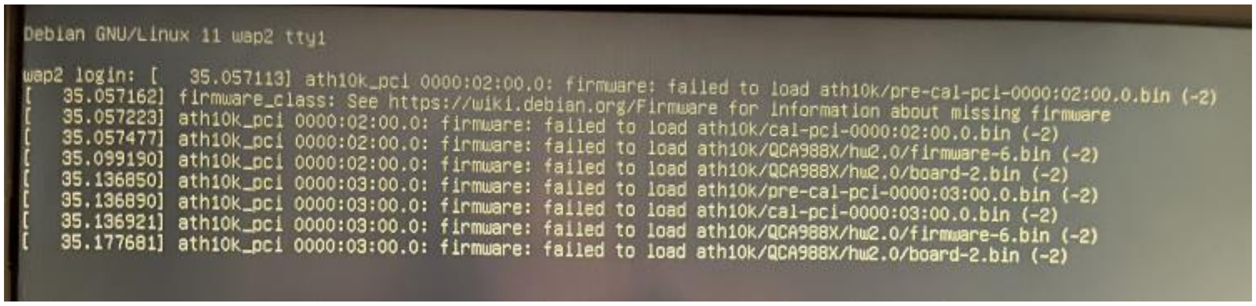
- And looks like the board.bin file from here is also needed:

  [`https://github.com/kvalo/ath10k-firmware/blob/master/QCA988X/hw2.0/board.bin`](https://github.com/kvalo/ath10k-firmware/blob/master/QCA988X/hw2.0/board.bin)

I downloaded both files, created the directory **`/lib/firmware/ath10k/QCA988X/hw2.0`**, and placed both **`firmware-5.bin`** and **`board.bin`** in there.

Then a reboot.

I still get some errors …



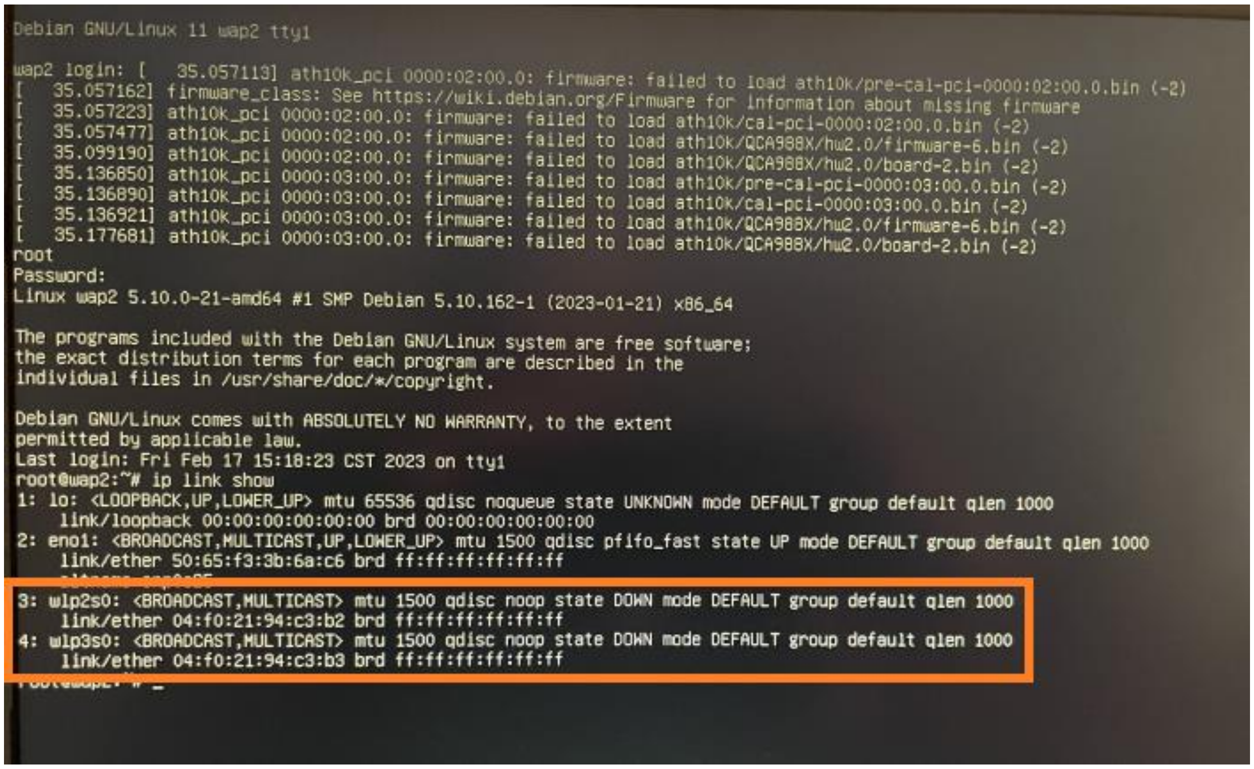
```
Debian GNU/Linux 11 wap2 tty1

wap2 login: [   35.057113] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/pre-cal-pci-0000:02:00.0.bin (-2)
[   35.057162] firmware_class: See https://wiki.debian.org/Firmware for information about missing firmware
[   35.057223] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/cal-pci-0000:02:00.0.bin (-2)
[   35.057477] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-6.bin (-2)
[   35.099190] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/board-2.bin (-2)
[   35.136850] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/pre-cal-pci-0000:03:00.0.bin (-2)
[   35.136890] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/cal-pci-0000:03:00.0.bin (-2)
[   35.136921] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-6.bin (-2)
[   35.177681] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/board-2.bin (-2)
```

but it's simply the driver probing for every possible file, including ones it won't really use – probably because the driver supports many cards other than this exact one, but other cards would need different firmware files.

In any event: it's not complaining any more above that it can't find **`firmware-5.bin`** or **`board.bin`**.

Actual confirmation things are OK: the Wi-Fi interfaces are now showing up in **`ip link show`**.



```
Debian GNU/Linux 11 wap2 tty1

wap2 login: [   35.057113] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/pre-cal-pci-0000:02:00.0.bin (-2)
[   35.057162] firmware_class: See https://wiki.debian.org/Firmware for information about missing firmware
[   35.057223] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/cal-pci-0000:02:00.0.bin (-2)
[   35.057477] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-6.bin (-2)
[   35.099190] ath10k_pci 0000:02:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/board-2.bin (-2)
[   35.136850] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/pre-cal-pci-0000:03:00.0.bin (-2)
[   35.136890] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/cal-pci-0000:03:00.0.bin (-2)
[   35.136921] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/firmware-6.bin (-2)
[   35.177681] ath10k_pci 0000:03:00.0: firmware: failed to load ath10k/QCA988X/hw2.0/board-2.bin (-2)
root
Password:
Linux wap2 5.10.0-21-amd64 #1 SMP Debian 5.10.162-1 (2023-01-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Feb 17 15:18:23 CST 2023 on tty1
root@wap2:~# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 50:65:f3:3b:6a:c6 brd ff:ff:ff:ff:ff:ff
    altname enp0s25
3: wlp2s0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 04:f0:21:94:c3:b2 brd ff:ff:ff:ff:ff:ff
4: wlp3s0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 04:f0:21:94:c3:b3 brd ff:ff:ff:ff:ff:ff
root@wap2:~# _
```

If we see interfaces here, we have things that will send/receive Ethernet packets and that we can assign IP addresses to and bridge and all that good stuff.

SSH setup

You're probably tired of seeing screenshots from my phone. Getting SSH setup will enable me to remote into the box using PuTTY and then I can give you nice screenshots instead.

/etc/network/interfaces

Giving an interface a static IP

We used the wired Ethernet interface during install—DHCP, so the Debian installer carried that configuration over into the installation.

I'd like to change that--give the wired Ethernet interface a static IP so I can SSH into it without having to guess the IP address or check my router's DHCP tables. We'll also need to specify a netmask and default gateway.

Setting the wired Ethernet's IP address and netmask

On Linux, you can statically set the IP address and other attributes of network interfaces directly with the **`ip addr`** command.

```
ip addr add dev eno1 192.168.3.229/24
```

- Yes, that's an 'add' - Linux will let you stack as many IP addresses as you want on an interface.
- There are situations where you don't want an interface to have an IP address (and we'll be running into that situation later).
- If you want to change an interface's IP address, you need to delete the existing one (ip addr del) and add a new one.

Adminstratively enabling the wired Ethernet interface

Interfaces are administratively down by default, to enable them, use **`ip link`**:

```
ip link set dev eno1 up
```

Those were live changes – we want persistent changes

But the above commands don't change any persistent configuration. What we did above won't survive a reboot.

ifupdown

Fortunately, part of the "Basic system utilities" we installed earlier includes the **`ifupdown`** package.

`ifupdown` provides a mechanism to define network configuration persistently, automatically apply it on boot, and a few other things we need.

More specifically, the following is provided:

- **`/etc/network/interfaces`** – a text file where we can define each interface and all of it settings, such as IP address, whether to statically assign IP addresses or get them via DHCP, default gateway, and much more.
- Two commands – **`ifup`** and **`ifdown`** – that will turn up and turn down interfaces and use information the file **`/etc/network/interfaces`** to do it.

    - Note: Not all interfaces have to appear, or will appear automatically, in **`/etc/network/interfaces`**.
        - The Debian installer will add ones it finds during install, but after that, you must add them manually.
        - If you try to **`ifup`** or **`ifdown`** an interface that's not listed in **`/etc/network/interfaces`**, or is listed incorrectly, you'll get an error.
- **`ifup`** and **`ifdown`** provide hooking for scripts
    - First method: In **`/etc/network/interfaces`** it's possible to define commands/scripts that…
        - run right *before* an interface is brought up (pre-up)
        - run right *after* an interface is brought up (post-up)
        - run right *before* an interface is brought down (pre-down)
        - run right *after* an interface is brought down (post-down)

      and these are just lines in the **`/etc/network/interfaces`** file.
    - Second method: Scripts/executables in specific directories.
      The "hook" directories are:
        - **`/etc/network/if-pre-up.d`** (pre-up)
        - **`/etc/network/if-up.d`** (post-up)
        - **`/etc/network/if-down.d`** (pre-down)
        - **`/etc/network/if-post-down.d`** (post-down)
        - All executable scripts/files in these directories will be run (as root) when **`ifup`** or **`ifdown`** is run, at the appropriate time (before or after the interface state change).
            - If an executable/script needs to appear in multiple directories, symlinks can be used.
        - This is a *hook* mechanism that other networking commands, utilities, and applications use to make sure things happen when interfaces are brought down or up.
        - These commands are *not* executed if you issue an **`ip link set dev`** command.
- **`systemd`** unit files
    - Unit files will run **`ifup`** on any interface listed as auto in **`/etc/network/interfaces`**, on startup (specifically when enabling the "network" target)

So, basically any static network configuration, or configuration we need to have "saved" and automatically applied on boot, it needs to be defined in /etc/network/interfaces.

Installing a few things we need

Right now we have pretty much a blank Linux system with only basic system utilities installed. Basic networking commands like **`ip`** is part of that. We'll need some additional commands:

- **`iw`**
    - This command gives us lots of detail on Wi-Fi PHYs.
        - **PHY** is a term for the part of a network card (Wi-Fi, wired, optical, whatever) that actually generates the signals on the medium (radio/air, electricity/wired, optical/fiber).

            - Aside: Other related terms are **MAC** (Media Access Control – faces the CPU) and **MII** (Media Independent Interface – connects CPU and PHY). Not iportant at the moment.
        - Basically: mn All the things our Wi-Fi card is capable of can be found out by looking at the PHY.
- **`hostapd`**
    - This makes AP associations work and hooks them into the network once authenticated.

To be completed …