Hardware, 3D Files & OS Setup Guide

Background & Motivation

Motivation and Learning Journey

My journey into building a homelab started with a desire to learn more about Docker Swarm and Raspberry Pi. Initially, my motivation was simply to experiment with these technologies, but as I delved deeper into the project, I realized that I was gaining much more than just technical knowledge about the hardware. I discovered the incredible potential of self-hosting a variety of services and the empowerment that comes with it.

Through this project, I had the opportunity to explore and understand various IT concepts and tools that have significantly enhanced my technical skills:

- **Docker Networks and Docker Swarm:** I learned how to set up and manage containers, orchestrate them across multiple nodes, and effectively use Docker Swarm for scaling and managing these containers.
- **Traefik Reverse Proxy:** I explored how to use Traefik as a reverse proxy, understanding its role in managing and routing incoming traffic to the appropriate services while ensuring SSL certification for secure connections.
- **VPNs and NAT:** I gained hands-on experience with both WireGuard VPN and OpenVPN, learning how to configure secure remote access to my network. I also deepened my understanding of Network Address Translation (NAT) and its significance in networking.
- **Portainer:** I learned how to use Portainer as a powerful and user-friendly tool for managing Docker environments, making it easier to monitor, deploy, and control my Docker containers.
- **Home Assistant and Jellyfin:** I explored home automation with Home Assistant and media server management with Jellyfin, realizing the convenience and control that come with hosting these services locally.
- **Vaultwarden:** I discovered the benefits of using Vaultwarden for secure password management, integrating it into my daily life and appreciating the privacy it offers by hosting it myself.

One of the most empowering aspects of this journey has been the realization that, with Docker and Linux as my foundation, I can host virtually any service I desire, provided it’s available for the ARMv8 architecture. This has opened up a world of possibilities for me, enabling me to customize my digital environment exactly to my needs.

A Project for Everyone with Passion

This homelab project is an excellent opportunity for anyone who wants to dive deeper into IT and self-hosting. It offers a hands-on way to learn about a wide range of technologies, from containerization and networking to automation and media servers. However, it's important to note that this project requires a genuine passion for learning and the willingness to research and troubleshoot on your own. While I provide full guides and resources, the ability to independently solve problems and continuously learn is essential for success.

If you’re excited about technology and ready to take on the challenge, building a homelab is a fantastic way to gain practical skills and a deeper understanding of the IT world.

Hardware Requirements

Partlist & Costs

About This Project

Welcome to my Guide on the Parts and Components I used to build my Raspberry Pi 5 Cluster. My name is aeoneros, I'm a 24-year-old IT enthusiast from Switzerland, and I’m relatively new to the IT world. This project was a great learning experience for me, and I’m excited to share the journey with you. Below, you'll find a comprehensive list of all the parts I used, complete with links for purchasing the components within Switzerland.

> To help you understand the financial aspect of the project, I've also included the prices I paid for each item in Swiss Francs (CHF) along with the equivalent in USD for your reference.
> **Exchange Rate:** 1 CHF = 1.18009 USD

Whether you're a fellow hobbyist or a professional looking to replicate this setup, I hope you find this resource helpful!

Parts List & Costs

1. **Raspberry Pi 5 (8GB)**
   - **Quantity:** 3
   - **Price per Unit:** CHF 86.70 / USD 102.36
   - **Total Price:** CHF 260.10 / USD 307.08
   - **Link:** Raspberry Pi 5 (8GB)

2. **Samsung 970 EVO Plus (500 GB, M.2 2280)**
   - **Quantity:** 3
   - **Price per Unit:** CHF 65.70 / USD 77.52
   - **Total Price:** CHF 197.10 / USD 232.56
   - **Link:** Samsung 970 EVO Plus SSD

3. **Geekworm X1001 PCIe to M.2 NVMe KEY-M SSD Shield for Raspberry Pi 5**
   - **Quantity:** 3
   - **Price per Unit:** CHF 12.26 / USD 14.47

- **Total Price:** CHF 36.78 / USD 43.41
- **Link:** Geekworm X1001 Shield

4. **Delock M.2 Heatsink, 70x22x5.2mm**
   - **Quantity:** 3
   - **Price per Unit:** CHF 9.75 / USD 11.51
   - **Total Price:** CHF 29.25 / USD 34.53
   - **Link:** Delock M.2 Heatsink

5. **Raspberry Pi Active Cooler Fan for Raspberry Pi 5**
   - **Quantity:** 3
   - **Price per Unit:** CHF 8.90 / USD 10.51
   - **Total Price:** CHF 26.70 / USD 31.53
   - **Link:** Raspberry Pi Active Cooler Fan

6. **Official Raspberry Pi 5 USB-C Power Supply 27 W, White**
   - **Quantity:** 3
   - **Price per Unit:** CHF 13.53 / USD 15.97
   - **Total Price:** CHF 40.59 / USD 47.91
   - **Link:** Official Raspberry Pi 5 USB-C Power Supply

7. **SanDisk Ultra Android microSDXC UHS-I Memory Card 128 GB + Adapter**
   - **Quantity:** 1
   - **Price:** CHF 11.16 / USD 13.18
   - **Link:** SanDisk Ultra 128 GB microSDXC

8. **Beikell SD Card Reader, Dual Plug USB 3.0/USB C Card Reader**
   - **Quantity:** 1
   - **Price:** CHF 9.46 / USD 11.17
   - **Link:** Beikell SD Card Reader

9. **Micro HDMI to HDMI Multifunctional Adapter, Compatible with Raspberry Pi 5 / 4B**
   - **Quantity:** 3
   - **Price per Unit:** CHF 8.90 / USD 10.51
   - **Total Price:** CHF 26.70 / USD 31.53
   - **Link:** Micro HDMI to HDMI Adapter

10. **Amazon Basics 0.3m RJ45 Cat7 Gigabit Ethernet High-Speed Patch Cable (Pack of 5)**
    - **Quantity:** 1
    - **Price:** CHF 9.67 / USD 11.41

- **Link:** Amazon Basics Ethernet Cable

11. **Geekworm Raspberry Pi Installation Tool (100 Pcs Hex Brass Spacer/Standoff + Nuts + Screws)**
    - **Quantity:** 1
    - **Price:** CHF 10.31 / USD 12.17
    - **Link:** Geekworm Installation Tool Kit

12. **EFB Elektronik 19" 04HE Wall Mount Case (450 mm Depth, 1-Piece, Flat Pack, RAL7035)**
    - **Quantity:** 1
    - **Price:** CHF 55.76 / USD 65.81
    - **Link:** EFB Elektronik Wall Mount Case

13. **GeeekPi Raspberry Pi 4 Cluster Case (4 Layers Acrylic Housing)**
    - **Quantity:** 1
    - **Price:** CHF 18.92 / USD 22.31
    - **Link:** GeeekPi Raspberry Pi 4 Cluster Case

    (Note: This case was used in my first version of the cluster, but I later switched to a custom-designed rackmount that I 3D printed on my Prusa MK4S due to better maintenance and accessibility.)

14. **5 Port Switch**
    - **Quantity:** 1
    - **Price:** Free
    - I already had one laying around :)

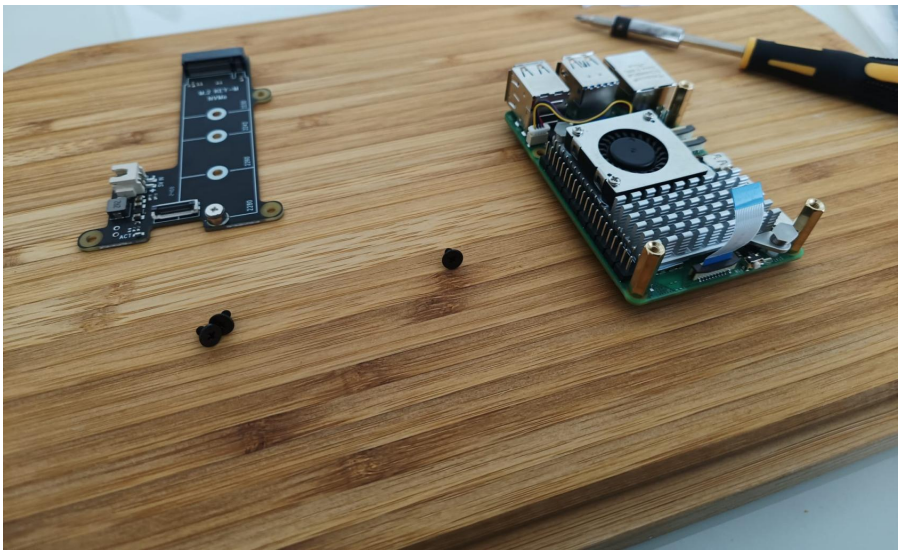
Total Costs:

- **CHF:** CHF 722.90
- **USD:** USD 853.00

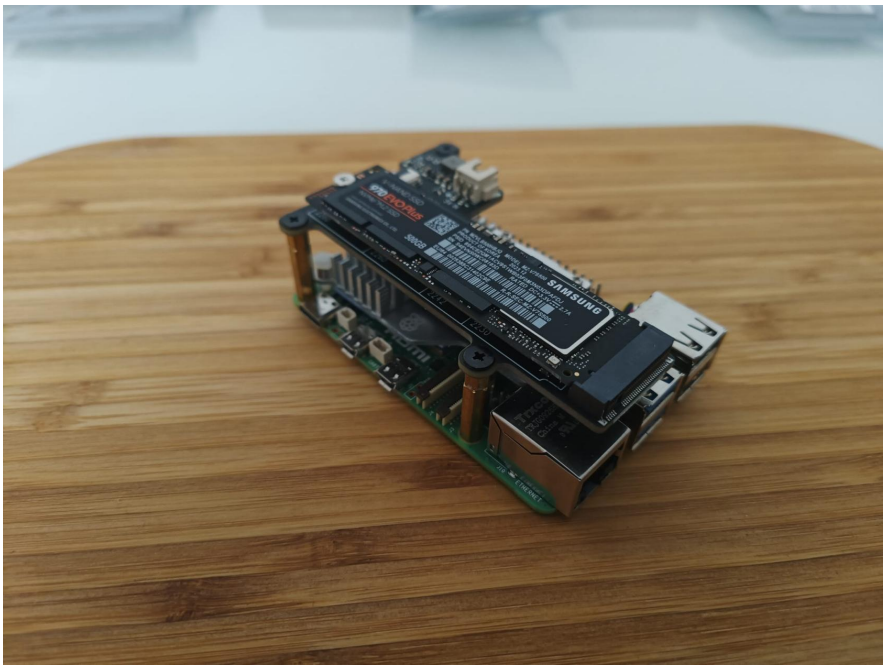
Building Process & OS Setup Guide

Install NVMe (Hardware)

How to Setup NVMe Guide

1.  First install the standoffs and screw them on the bottom of the Pi.

2. Then Mount the NVMe Base on the standoffs and click in the SSD,



3. Then you need to install the Thermal Pad on top of the NVMe.

> You DONT need to remove the Informationsticker from the SSD.
> In case you did, you will remove your Warranty.

4. On the last Step you need to remove the Blue Sticker and install the cooler on Top of the Thermalpad.

Flash OS onto NVMe

Important Notes

This Wikipage hase been integrated by aeoneros from the Original Source: ***Geekworm***

> Check Out the NVMe SSD Incompatibility List if you have Troubble at Booting your Device!

NVMe SSD Incompatibility List:

We recommend avoiding the following NVMe SSD drives which is equipped with a **Phison controller** due to their proven incompatibility:

- WD Blue SN550 series (Solved! Refer to New rpi-eeprom-update 2024-01-24 WD Blue SN550 nvme works now.)
- WD Green SN580 series (Solved! Refer to NVMe_SSD_boot_with_the_Raspberry_Pi_5#comment-4708)
- WD Green SN350 series (Solved! Refer to NVMe_SSD_boot_with_the_Raspberry_Pi_5#comment-4602)
- WD Black SN850 series
- WD Black SN770
- WD SN740
- Inland tn446 nvme drive
- Corsair MP600 SSD
- Micron 2450 SSD (Can be recognised but not support boot from NVME)
- Other NVMe SSD drivers equipped with the same **Phison controller**

These specific models have demonstrated compatibility issues, and it is advisable to avoid them when considering NVMe SSD options for the X10xx series NVMe shield. You can run "**lspci** " command to check the controller brand of the SSD.

We **confirm** that the following SSDs are **incompatible**:

- Micron 2200 256GB M.2 NVMe Gen3 x4, MODEL: MTFDHBA256TCH, The SSD is recognized but I cannot boot from it.

Also note:

- Compatible with M.2 **NVMe** SSDs only, **Not** compatible with M.2 SATA SSDs, M.2 PCIe AHCI SSDs, or other M.2 non-NVMe devices
- Older NVMe drives with less efficient flash media may not perform as well as newer drives
- New NVMe SSDs are not partitioned and will need to be both partitioned and formatted when first connected to the Raspberry Pi before they will be accessed in the Explorer.
- We get feedback from customers that **Polaris Controller** will also have compatibility problems. Please replace the other SSD test if it not work, whether it is compatible with the Raspberry Pi 5 does not depend on the X100X series boards
- NVMEs using the **MAP1202** controller may not support PCIe Gen 2, and must be forced to enable PCIe Gen 3 in order to be recognised. This is due to the fact that the controller is not backward compatible with PCIe Gen 2, and NVMEs using this controller will have compatibility issues, and are not recommended for use. Can refer to https://zhuanlan.zhihu.com/p/644984347

PS: There is also feedback from buyers that even NVME SSDs with *Phison controller* are supported after updating the latest firmware. Please refer to go to: X1001#comment-4638

Enable PCIe

> By default the PCIe connector is not enabled.

To enable it you should add the following option into /boot/firmware/config.txt and reboot:

```
sudo nano /boot/firmware/config.txt
```

Then add the following comment;

```
# Enable the PCIe External connector.
dtparam=pciex1

# This line is an alias for above (you can use either/or to enable the port).
dtparam=nvme
```

Press **Ctrl-O**, then enter, to write the change to the file.
Press **Ctrl-X** to exit nano (the editor).

And the connection is certified for Gen 2.0 speed (5 GT/sec), but you can force it to Gen 3.0 (10 GT/sec) if you add the following line after:

```
dtparam=pciex1_gen=3
```

> WARNING:
> The Raspberry Pi 5 is not certified for Gen 3.0 speeds, and connections to PCIe devices at these speeds may be unstable.

Then **DON'T** forgot to reboot :)

```
sudo reboot
```

You also can refer to official documentatio: Enabling PCIe
After reboot,use the **lspci** command to display your PCIe devices

```
sudo lspci
```

The output is as follows, note that the content of the third line depends on the NVME control you are using.

```
root@swarmpi1:~#  sudo lspci
0000:00:00.0 PCI bridge: Broadcom Inc. and subsidiaries BCM2712 PCIe Bridge (rev 21)
0000:01:00.0 Non-Volatile memory controller: Samsung Electronics Co Ltd NVMe SSD Controller SM981/PM981/PM983
0001:00:00.0 PCI bridge: Broadcom Inc. and subsidiaries BCM2712 PCIe Bridge (rev 21)
0001:01:00.0 Ethernet controller: Raspberry Pi Ltd RP1 PCIe 2.0 South Bridge
```

- Make sure your NVME SSD is plugged into the PCIe PIP, not just connecting PCIe PIP to the Pi 5, otherwise the **lspci** will not be able to display any PCIe device.

- If your NVMe SSD is not recognized, updating the bootloader firmware is essential!

  Refer to How to update eeprom firmware to update firmware or #FAQ Q1 to know more details.

Flash OS onto NVME SSD

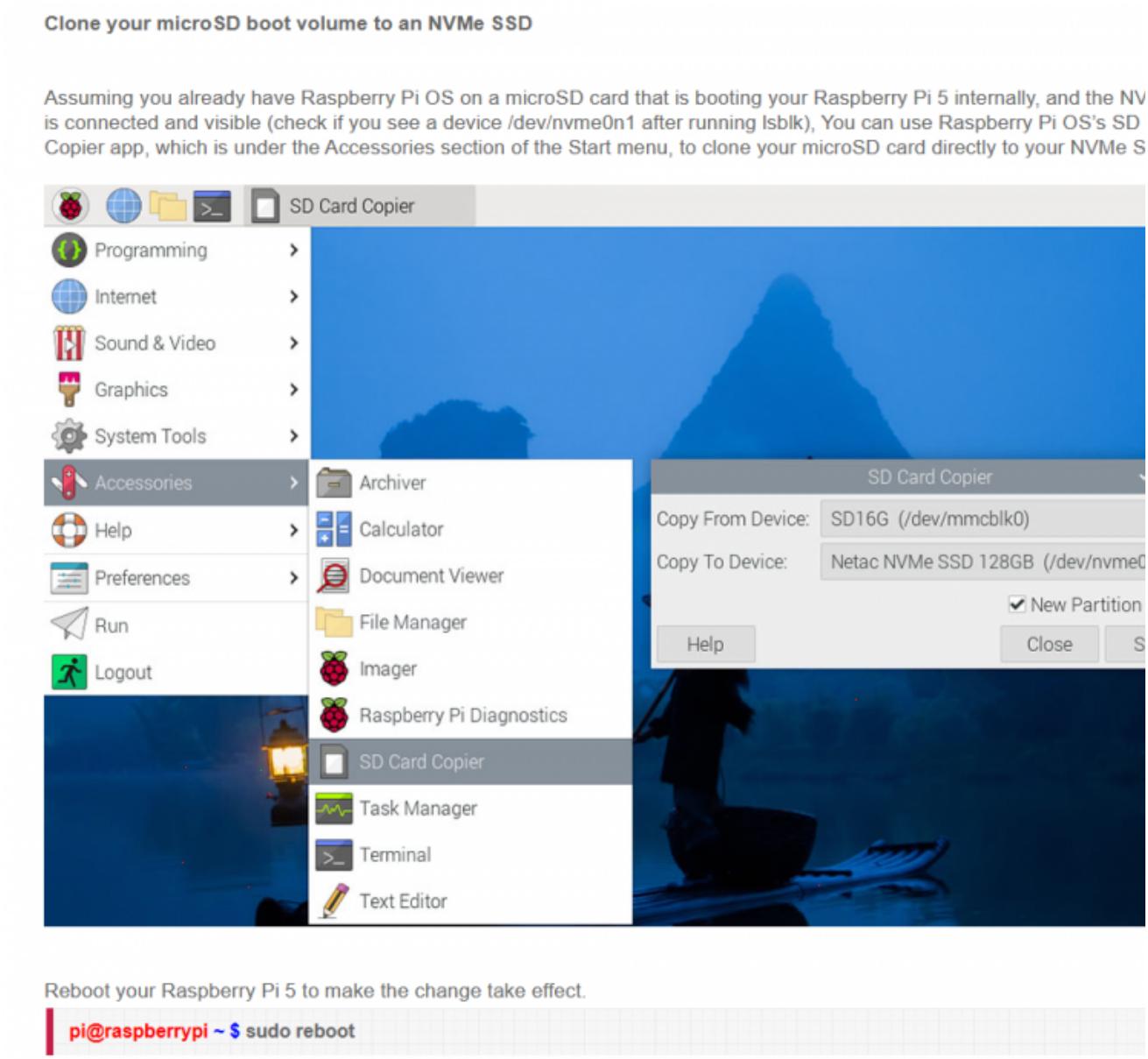
> To get the NVMe SSD to boot your Pi, it needs to have an OS, so the Raspberry Pi OS needs to be flashed onto NVME SSDs, this is very important!
> **Only** support Raspberry Pi OS (**Bookworm**) version, Raspberry Pi OS bullseye or Ubuntu or Home Assistant OS is NOT supported, refer to
>
> https://www.raspberrypi.com/software/operating-systems/

Here are a few ways to flash the OS to an NVME SSD:

> **This is a visual operation and i highly recommend it!**

1. **Use SD Card Copier tool to flash OS onto the NVME SSD On Raspberry Pi OS (Recommended)**
   1. Download Raspberry Pi Imager Tool on you Windows or Mac Machine and flash the OS (Raspberry Pi OS - 64bit) onto your SD-Card
   2. After flashing the Pi OS onto your SD-Card, insert the SD-Card into your Raspberry Pi 5 and Power the Device.
      1. **Sidenote**: If you Install the OS on multiple Devices i would recommend to flash the OS again onto the SD card after copying it to 1 Machine. That way you can avoid having Troubles later on.
   3. After powering your Pi & successful first boot,
      Cick **Applications** =>**Accessories** =>**SD Card Copier** on the main screen, run the **SD Card Copier** program, and copy the OS to the NVME ssd as shown in the figure below.

4. Click **Start** to run. Then shut down, unplug the SD card, and restart the device.



2. **Flash the SSD directly with Raspberry Pi Imager**
    1. You can also directly use the Raspberry Pi Imager tool on a MAC computer or windows computer to flash a fresh Pi OS to NVME SSD, but you will need an additional USB to nvme adapter.
        1. Install Pi Imager and open it
        2. Plug your NVMe SSD into your computer using a USB to NVMe adapter
        3. Choose an OS to install
        4. Choose the drive (connected through your adapter) to flash
        5. Click write (and set any options you'd like)
        6. Once you have finished flashing the OS, DON'T remove the nvme SSD. You must #Enable PCIe.

Then pull the NVMe drive, attach it to your Pi 5, and it should boot off it (with or without a microSD card inserted)—assuming you have the bootloader up to date and set the **BOOT_ORDER** appropriately!

> If you are flashing a fresh Pi OS to NVME ssd, you must #Enable PCIe; but if you are COPY or CLONE an old Pi OS from SD card to NVME ssd, and you have already enabled pcie in the old Pi OS in advance, then you don't need to do enable pcie again!

Set NVMe early in the boot order

The PCIe connection should work after a reboot, but your Pi won't try booting off an NVMe SSD yet. For that, you need to change the **BOOT_ORDER** in the Raspberry Pi's bootloader configuration:

- Use tool **raspi-config** to set boot order

```
sudo raspi-config
```

- Then select **6 Advanced Opitions** => **A4 Boot Order** => **B2 NVMe/USB Boot** answer **Yes**, then `sudo reboot`
  - Run the following command to set boot order.

```
sudo rpi-eeprom-config --edit
```

Then change the BOOT_ORDER line to the following:

```
BOOT_ORDER=0xf416
```

- Press **Ctrl-O**, then enter, to write the change to the file.
Press **Ctrl-X** to exit nano (the editor).

Read Raspberry Pi's documentation on BOOT_ORDER for all the details. For now, the pertinent bit is the **6** at the end: that is what tells the Pi to attempt NVMe boot first!

> **Reboot** your Raspberry Pi 5 to make the change take effect.

Set a static IP address with nmtui on Raspberry Pi OS 12 'Bookworm'

This Wikipage hase been integrated by aeoneros from the Original Source: jeffgeerling

> Old advice for setting a Raspberry Pi IP address to a static IP on the Pi itself said to edit the `/etc/dhcpcd.conf` file, and add it there.
> But on Raspberry Pi OS 12 and later, `dhcpcd` is no longer used, everything goes through Network Manager, which is configured via `nmcli` or `nmtui`. If you're booting into the Pi OS desktop environment, editing the IP settings there is pretty easy.

> You can also configure a static IP entirely via `nmcli` without using the UI; see this article on nmcli` from Cyberciti.biz.

But setting a static IP via the command line is a little different.

Install `nmtui`

The `nmtui` Command is Part of the Network-Manager Package, to install use this Commands:

```
# For Debian-based distributions
sudo apt-get install network-manager

# For RPM-based distributions
sudo yum install NetworkManager-tui
```

Check for Device Status

First, get the interface information—you can get a list of all interfaces with `nmcli device status`:

```
DEVICE          TYPE      STATE                 CONNECTION
eth0            ethernet  connected             Wired connection 1
docker_gwbridge  bridge    connected (externally)  docker_gwbridge
lo              loopback  connected (externally)  lo
docker0         bridge    connected (externally)  docker0
wlan0           wifi      disconnected          --
```

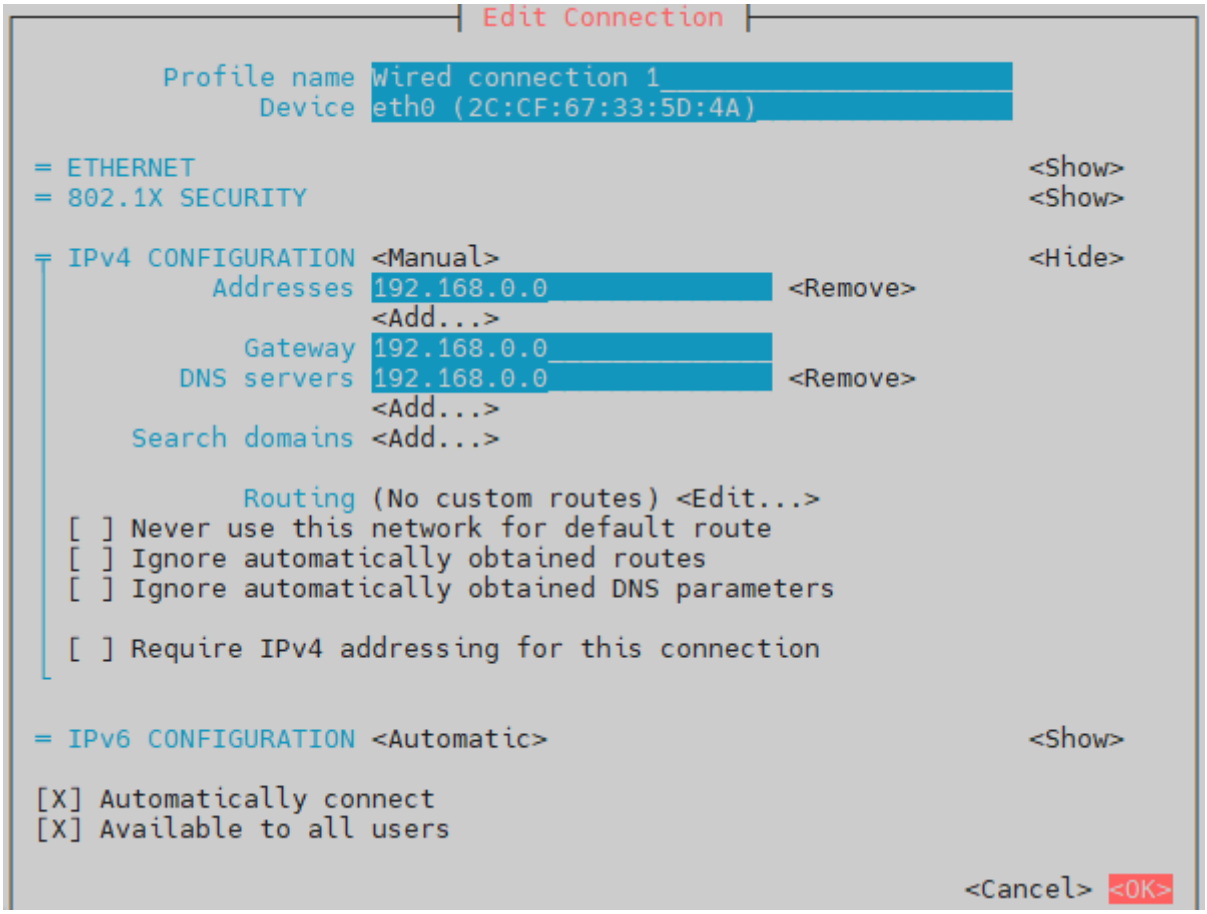
In my case, I want to set an IP on `eth0`, the built-in Ethernet.

Set static IPv4 Address

I can get all the current information about that port with `nmcli device show eth0`, and I can edit the connection using the terminal UI (`nmtui`):

```
$ sudo nmtui edit "Wired connection 1"
```

Go through each setting adding in at least an IPv4 address, Gateway, and DNS Server, for example:



The IP's are just for Show. Change them to fit your Setup.
Then go down to the bottom and select 'OK'.

This saves the static IP configuration, but doesn't *apply* it immediately. To apply the changes, you need to restart NetworkManager:

```
$ sudo systemctl restart NetworkManager
```

Then if you run `nmcli device show eth0`, you should see the new IP address (the old one might still be attached to the interface at the same time until you reboot):

```
root@swarmpi1:~# nmcli device show eth0
GENERAL.DEVICE:                         eth0
GENERAL.TYPE:                           ethernet
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     Wired connection 1
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnection/2
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         192.168.0.0/24
IP4.GATEWAY:                            192.168.0.0
```

> You successfully changed the IPv4 Address from your PI :)

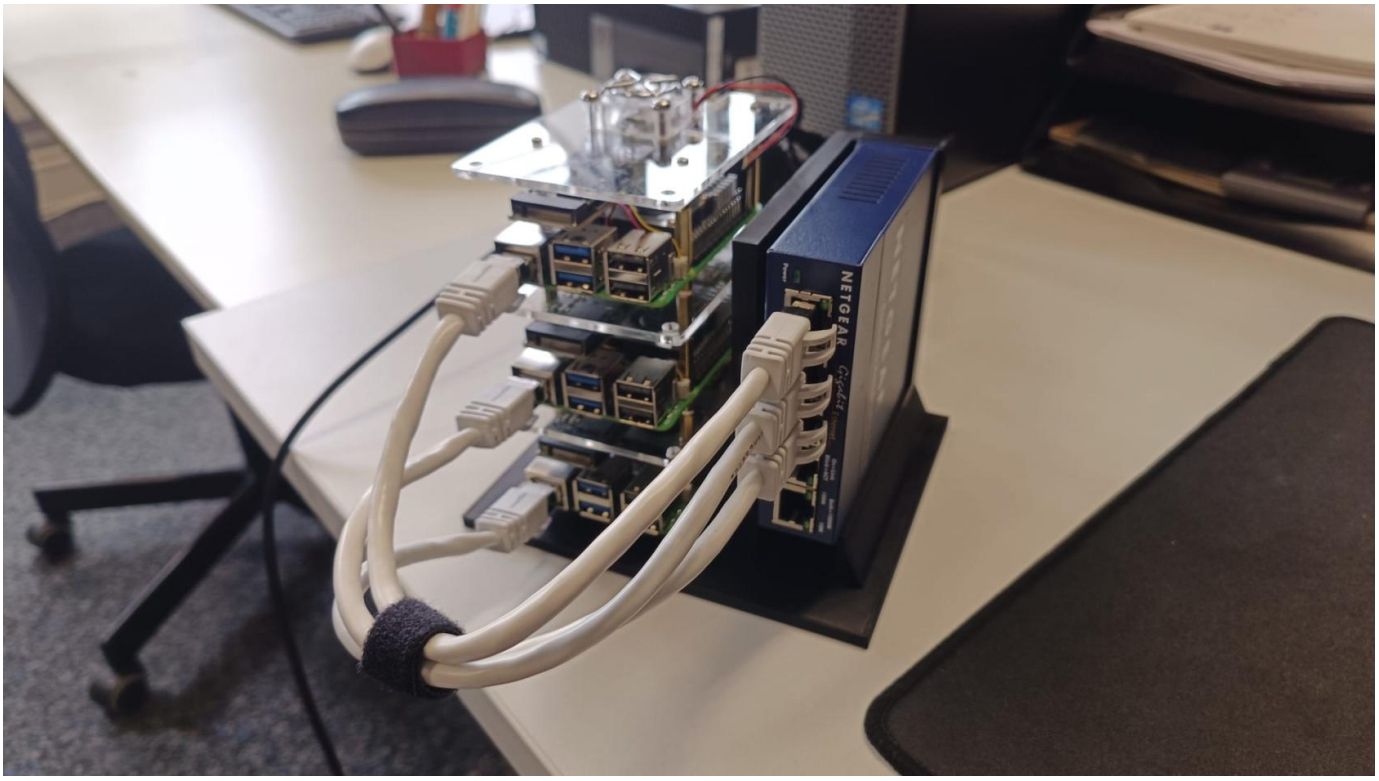
3D Mounts

All Designs have been created by aeoneros with Fusion360 Student License

3D Mounts

Raspberry Pi Cluster Stack (Freestanding) + 3D Files

> All 3D Files in this Post are made by aeoneros! :) If you like to Support me you can Buy me a Coffee :)



Background Information

I initially experimented with Proxmox and VMs at my company, trying out Docker Swarm. Eventually, I decided I really wanted to build a Raspberry Pi cluster. Being a person who likes to get things done quickly, I found the parts for a freestanding Pi cluster and ordered them right away. I immediately set to work designing a baseplate for it and also added a mount for a 5-port switch from Netgear. This is the final version of my freestanding Pi cluster.

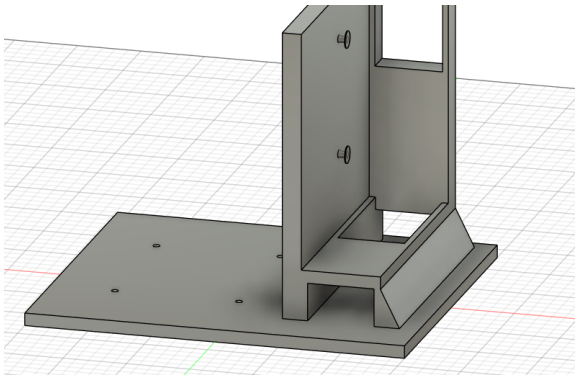
Parts List

- **GeeekPi Raspberry Pi 4 Cluster Case (4 Layers Acrylic Housing)**
- Amazon-Link
- Price**:** CHF 18.92 / USD 22.31

- **Geekworm Raspberry Pi Installation Tool (100 Pcs Hex Brass Spacer/Standoff + Nuts + Screws)**
- Amazon-Link
- Price: CHF 10.31 / USD 12.17

- **Netgear GS105GE (5Ports)**
- Amazon-Link
- Digitec-Link
- Price: CHF 29.- / USD 34.25

3D Files (For Free)



I designed this Mountstand in Autodesk Fusion 360 Student Version and give it away publicly avialable for free.

You can Access the Autodesk-Fusion-360 Project when clicking on the Link.

The Password for the Project is "**pistand+aeoneros**".

You can also download the 3D Files directly here on my Website here:

Floor_Plate.3mf

Mount_For_NetgearSwitch.3mf

PI_Stand_FinalVersion.f3d

19" Server Rack Mount (for 3 Pi's) + 3D Files

Coming Soon....