

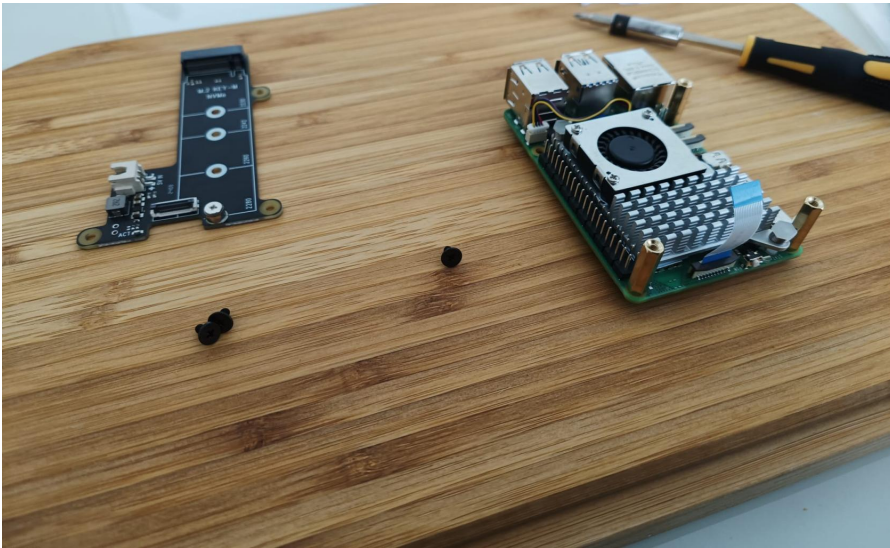
Building Process & OS Setup Guide

- Install NVMe (Hardware)
- Flash OS onto NVMe
- Set a static IP address with nmtui on Raspberry Pi OS 12 'Bookworm'

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 has been integrated by aeoneros from the Original Source: [Geekworm](#)

Check Out the NVMe SSD Incompatibility List if you have Trouble 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-eeeprom-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!

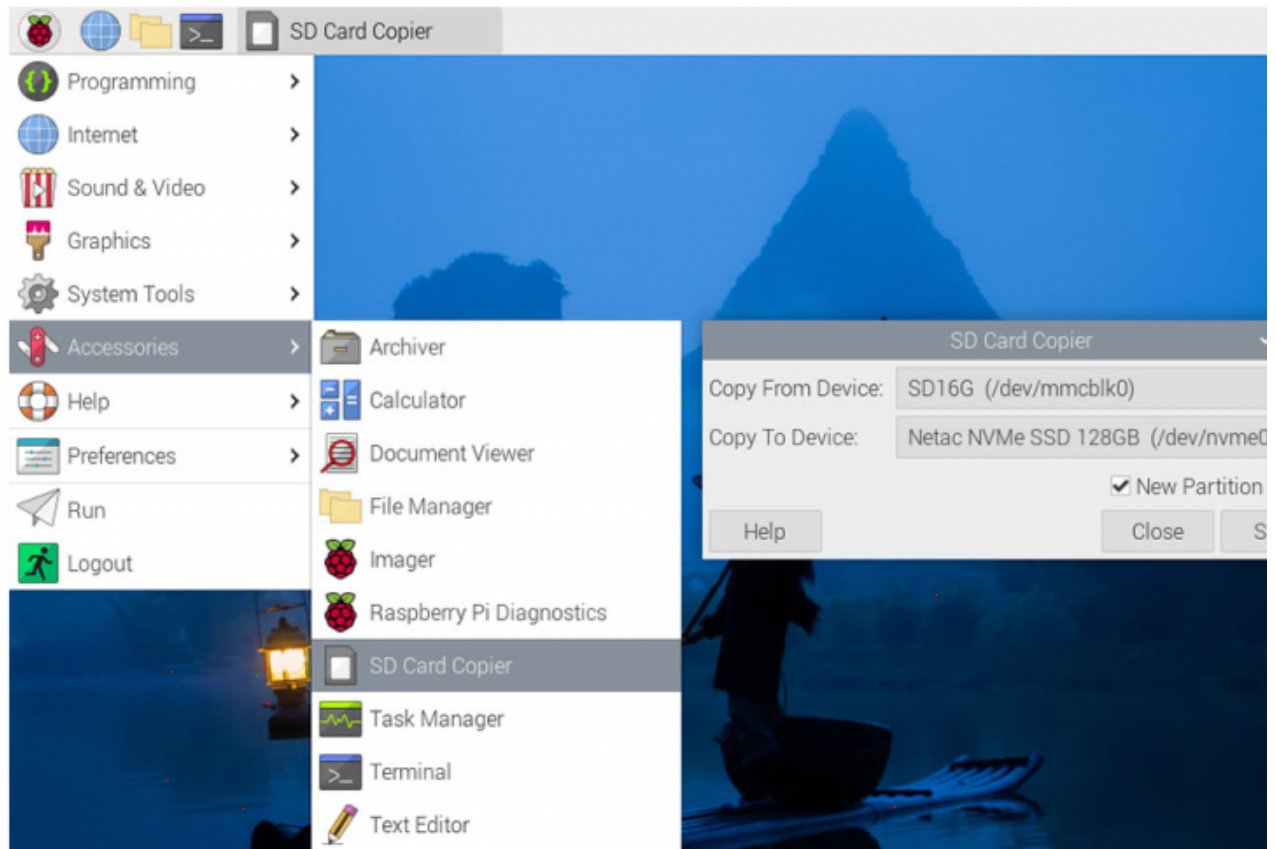
1. Use SD Card Copier tool to flash OS onto the NVME SSD On Raspberry Pi OS (Recommended)

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.

Clone your microSD boot volume to an NVMe SSD

Assuming you already have Raspberry Pi OS on a microSD card that is booting your Raspberry Pi 5 internally, and the NV is connected and visible (check if you see a device `/dev/nvme0n1` after running `lsblk`), You can use Raspberry Pi OS's SD Copier app, which is under the Accessories section of the Start menu, to clone your microSD card directly to your NVMe S



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

```
pi@raspberrypi ~ $ sudo reboot
```

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 Options => 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 has 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/dhcpd.conf` file, and add it there.

But on Raspberry Pi OS 12 and later, `dhcpd` 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:

Edit Connection

Profile name: Wired connection 1
Device: eth0 (2C:CF:67:33:5D:4A)

= ETHERNET <Show>
= 802.1X SECURITY <Show>

= IPv4 CONFIGURATION <Manual> <Hide>
Addresses: 192.168.0.0 <Remove> <Add...>
Gateway: 192.168.0.0
DNS servers: 192.168.0.0 <Remove> <Add...>
Search domains: <Add...>

Routing (No custom routes) <Edit...>
[] Never use this network for default route
[] Ignore automatically obtained routes
[] Ignore automatically obtained DNS parameters
[] Require IPv4 addressing for this connection

= IPv6 CONFIGURATION <Automatic> <Show>
[X] Automatically connect
[X] Available to all users

<Cancel> <OK>

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 :)