Raspberry Pi 4 USB Boot Config Guide for SSD / Flash Drives

Published: June 29, 2019 / Last Modified: November 27, 2022 / Benchmarking, Hardware, Raspberry Pi, Storage / 1,022 Comments

Raspberry Pi 4 with Samsung 950 Pro NVME SSD

The new Raspberry Pi bootloader is out which makes these instructions only necessary if you want to continue to use the SD card as a bootloader. View the Raspberry Pi Bootloader Configuration Guide here!

The Raspberry Pi 4* is finally here and has a lot of exciting changes. One very major downside is that it doesn't support true USB booting yet out of the box (like the 3 series did). The Raspberry Pi foundation states that it is being worked on and will be added back with a future update. No timeline has been given yet for that to happen but they state it's one of their top priorities.

Most of my projects heavily depend on having good performing storage so sitting and waiting was not an acceptable solution. In this guide I'll show you a workaround to use USB devices as your rootfs device and use a Micro SD card as bootloader only which gives us full SSD performance after boot! To see exactly how much of a performance difference this makes (spoiler: it's gigantic) check out the Raspberry Pi Storage Benchmarks.

I highly recommend doing this on a completely new install. If you try to upgrade your old ones and something goes wrong there's a good chance you might lose data. We will be modifying the boot partition, resizing partitions, etc. so don't use a drive with any data on it unless you are positive you have all of the steps down!

Compatible USB Adapters

The Raspberry Pi 4 is proving to be picky about what SATA, M.2, etc. adapters will work in the USB 3.0 port. The USB 3.0 ports are the ones in the middle that are blue inside. The black ones are USB

2.0 and won’t give you the faster speeds the new Pi offers.

It’s very likely that some of these will be fixed via software and firmware updates and the Raspberry Pi Foundation has several open known issues related to USB 3. Until that happens though I will maintain a list here of known working ones and known problematic ones. It’s still very early in the release of the Pi 4 so we still have a lot to learn about which adapters work / don’t work. If you have working and nonworking adapters leave a comment and I’ll add it in this list.

If the adapters worked before on older Pis then one thing you can try is putting them in the black USB 2.0 ports. Obviously this is stupid because we all want the Pi 4 performance gains but if you end up needing to buy a new adapter this will give you a workaround until a replacement arrives!

Find USB adapter chipset

There are certain chipsets used in adapters that are known to be working/not working.

```
 pi@raspberrypi:~ $ lsusb
  Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
  Bus 002 Device 002: ID 174c:55aa ASMedia Technology Inc. Name:
 ASM1051E SATA 6Gb/s bridge, ASM1053E SATA 6Gb/s bridge, ASM1153 SATA
 3Gb/s bridge, ASM1153E SATA 6Gb/s bridge
  Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
  Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
  Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

This is a lsusb dump of all my connected USB devices. I have bolded the line with the USB bridge device. We can see that the chipset is ASM1153E. This is a really common one that works well with the Pi.

Known Working Adapters

This is a compiled list of known working adapters built by myself from adapters I’ve purchased and commenters from ones they have purchased in this article and my newer guide that utilizes the new Raspberry Pi 4’s native bootloader for USB booting.

StarTech 2.5″ SATA to USB 3.1 Adapter*	2.5″ SATA to USB 3.1	Verified working in comments (thanks Fredrick)
Inateck FE2004 2.5″ SATA to USB 3.0 Hard Drive Enclosure*	2.5″ SATA to USB 3.0	Mirco reports that this enclosure is working but trim is not supported
Samsung 2.5″ SATA to USB 850 EVO Kit /w Adapter* (Alternate amazon.de link*)	2.5″ SATA to USB 3.0 Kit	This is a kit that comes with a drive and adapter. Rene confirms the adapter works including with non-Samsung drives.
CSL 2.5″ SATA to USB 3.0 Adapter*	2.5″ SATA to USB 3.0	The CSL SL adapter is confirmed to be working by Krikitt in the comments. Available in Europe. Not available in US.

UGREEN 2.5″ SATA to USB-C 3.1 Enclosure Drive Caddy*	2.5″ SATA to USB-C 3.1	Confirmed to be working by CAProjects in the comments. Available in both Europe and US
UGREEN 2.5″ SATA to USB-C 3.1 “Protect What You Love” Case* (AliExpress Listing* – Make sure to select USB-C 3.1)	2.5″ SATA to USB-C 3.1	Reported working by Michal in the comments, thanks!
UGREEN 2.5″ to USB 3.0 “SATA USB Converter” Adapter* (AliExpress Listing*)	2.5″ SATA to USB 3.0	Also reported by Michal as working in the comments, thanks again!
UGREEN 2.5″ SATA to USB 3.0 Adapter Cable with UASP Converter*	2.5″ SATA to USB 3.0	This adapter is reported to be working by Mirco in the comments
SABRENT 2.5″ SATA to USB-C 3.1 Type A Adapter*	2.5″ SATA to USB-C 3.1 Type A	The new USB-C 3.1 Type A version of the Sabrent adapter is reported as working in the comments by UEF. DO NOT get the USB 3.0 version as that one is below on the naughty list and won’t work!
SABRENT 2.5″ SATA to USB 3.0 Dual Bay Docking Station*	2.5″ SATA to USB 3.0	One of very few Sabrent adapters/enclosures to work. Reported working well by William Grey in the comments. Has two bays!
AliExpress Generic 2.5″ SATA to USB 3.0 3 colors Hard Disk Case*	2.5″ SATA to USB 3.0	Reported as working with UASP support by pierro78 in the comments
Orico 2.5″ 2139C3-G2 2.5 inch USB 3.1 Gen 2 10Gbps Transparent Enclosure*	2.5″ SATA to USB 3.1 Gen 2 10Gbps	Make absolutely sure it’s the 10Gbps USB 3.1 Gen 2 version. There is another one that looks identical that is a USB 3.1 Gen 2 6Gbps that will not work. Confirmed working by RRT in the comments.
ASUS ROG STRIX Arion Aluminum Alloy M.2 NVMe SSD External Portable Enclosure Case Adapter* – (AliExpress Listing*)	M.2 NVMe (B+M Key) to USB/USB-C 3.2 Gen 2	It’s bold. It’s beautiful. It’s also confirmed working by TADRACKET and Steve B. However, be warned, it takes a *lot* of power! Steve B. reports that even with the oversized 3.5A CanaKit adapter* it does not work. If you have the standard 3.0 adapter you can be practically certain it won’t power this enclosure. Does work with a powered USB hub*.
ICY BOX M.2 NVMe (M Key) to USB-C 3.1 Gen 2 Enclosure* (Alternate amazon.de listing*)	M.2 NVMe (B+M Key) to USB-C 3.1 Gen 2	Returning legendary commentary Frank Meyer reports: Does not work with a 3.0A power adapter (also reported by TTE). It’s not enough power for this enclosure. Does work with a powered USB hub*.

TDBT M.2 NVMe (B+M Key) to USB-C 3.1 Gen 2 Enclosure*	M.2 NVMe (B+M Key) to USB-C 3.1 Gen 2	Confirmed to be working well by WorkHard in the comments
AliExpress Generic M.2 NVMe (B+M Key) to USB 3.1 “M2 SSD Case NVME Enclosure”*	M.2 NVMe (B+M Key) to USB 3.1 Type A	Confirmed working by Jens Haase, thanks Jen!
SSK Aluminum M.2 NVMe (M Key) to USB 3.1 Gen 2 SSD Enclosure*	M.2 NVMe (M Key) to USB 3.1 Gen 2	Brian L reports this is working well with beta firmware upgrades, but that it did not work at all without them!
ORICO M.2 NVMe SSD Enclosure, USB 3.1 Gen 2 (10 Gbps)* (AliExpress Listing*)	M.2 NVME to USB 3.1 Gen 2	M.Yusuf has given the first ever report of a working Orico adapter! Make sure it’s the USB 3.1 Gen 2 version that says “Support UASP for NVMe SSD”. This is the way.
DELOCK 42570 M.2 SATA (B Key) to USB Micro-B 3.1 Gen 2 SSD Enclosure*	M.2 SATA (B Key) to USB Micro-B 3.1 Gen 2	Andreas Franek reports that the enclosure works with a 3.0A power adapter (gets a little warm)
Shinestar M.2 NVMe (M Key) to USB 3.0 Adapter*	M.2 NVMe (M Key) to USB 3.0	This is the adapter I’m using in the picture at the top of the article. It is for NVMe M.2 drives only and is getting hard to find
UGREEN M.2 NVMe (B+M Key to USB-C 3.1 Gen 2 SSD Enclosure*	M.2 NVMe (B+M Key) to USB-C 3.1 Gen 2	Confirmed working in comments by Chad D
UGREEN M.2 SATA (B+M Key) to USB 3.1 Enclosure*	M.2 SATA (B+M Key) to USB 3.1 Enclosure	Reported as working well in the comments by John H. Reinhardt with a ASM1051E chipset
QNINE M.2 SATA (B Key) to USB 3.0 Enclosure*	M.2 SATA (B Key) to USB 3.0	I used this enclosure to benchmark M.2 SATA Lite-On and SanDisk drives — working great in 3.0 ports
Argon One M.2 SATA (B+M Key) Pi 4 Case*	M.2 SATA (B+M Key) Pi 4 Case	This case gives you a M.2 SATA port for your Raspberry Pi and is also a case! Confirmed working by Frank.
Tanbin mSATA to USB Adapter*	mSATA to USB 3.0	I used this mSATA to USB adapter for my Crucial M550 benchmark — working in 3.0 ports

Generic mSATA to USB 3.0 Adapter (fe2008)*	mSATA to USB 3.1	Confirmed working in comments by Nico
Canakit Raspberry Pi 4 Power Supply (USB-C)*	3.5A USB-C Power Supply	Canakit has been making very reliable power supplies for several Pi generations now. Using a 3.5A power supply will give enough extra power for your Pi to power the drive without causing instability
Simplecom SE502 M.2 SSD Adapter*	M.2 SATA (B Key) to USB 3.0	Quirks required, reported working by alan but only with quirks
Delock #61883 SATA to USB 3.0 Converter*	2.5″ SATA to USB 3.0	Reported working well by Joerg_H
Vantec SATA/IDE TO USB 3.0 Adapter*	2.5″ SATA to USB 3.0	Reported as working by JeffG but with an ugly messy appearance

Known Working Adapters

Known Problematic Adapters (Naughty List)

Here is a list of common USB adapters that are known to have problems with the Raspberry Pi 4. You can get some of these adapters working by using quirks mode (see the “Fix (some) USB Adapter Problems Using Quirks” section below).

FIDECO M207CPS USB3.2 to M2 NVME/SATA SSD Enclosure*	M.2 NVME to USB 3.2 Gen 2	Lee Myring reports that the FIDECO M207CPS has issues working with the Pi
UGREEN 30848 2.5″ SATA to USB 3.0 Hard Drive Enclosure*	2.5″ SATA to USB 3.0	Reporting as not working properly and disconnecting often by Mirco, thanks!
Sabrent USB 3.0 to 2.5″ SATA adapter*	2.5″ SATA to USB 3.0	Only works in the USB 2.0 ports. Will not boot in a USB 3.0 port. I have two of these and can confirm they don’t work. RIP to Sabrent, our previous king of the Pi 3 era of adapters.
Sabrent USB 3.0 to 2.5″ SATA Tool-Free External Hard Drive Enclosure*	2.5″ SATA to USB 3.0	Another nonworking Sabrent adapter reported by Alex, thanks Alex!
ELUTENG 2.5″ SATA to USB 3.0 Adapter*	2.5″ SATA to USB 3.0	Despite earlier reports as working Ryan and one other have reported this adapter does not work unless you enable quirks mode! Don’t make Ryan’s sacrifice in vain and avoid this one.
USB 3.0 to 2.5″ SATA III Hard Drive Adapter UASP	2.5″ SATA to	reported by dzm in the comments as having very poor I/O performance

Support-20cm, Black*	USB 3.0	
ORICO 2.5″ SATA to USB C 3.0 Enclosure (Transparent)*	2.5″ SATA to USB 3.0	Several commenters have stated the transparent ORICO is not working. Avoid!
ORICO 2.5″ SATA to USB 3.0 Enclosure (Black) 2588US3-BKT*	2.5″ SATA to USB 3.0	Commenters report that the USB-C variant of the transparent ORICO enclosure also does not work
ORICO 2.5″ SATA to USB 3.0 Enclosure (Black/White) 2520U3*	2.5″ SATA to USB 3.0	Reported as not working by by Richon in the comments
ORICO 2.5″ SATA to USB-C 3.1 Gen 1 Enclosure (Transparent)*	2.5″ SATA to USB-C 3.1 Gen 1	Confirmed as not working by Andrea De Lunardi in the comments (thanks!)
ORICO 2.5″ SATA to USB 3.1 Gen 1 Enclosure (Silver) M2PF-C3-BK-EP*	2.5″ SATA to USB-C 3.1 Gen 2	Looks really similar to the ICY BOX. Confirmed not working by auanasgheps in the comments.
Vantec 2.5″ SATA to USB 3.0 USB Adapter with Case*	2.5″ SATA to USB 3.0	Does not work after hours of testing and frustration by Moshe Katz in the comments!
AliExpress Generic 2.5″ SATA to USB 3.0 “New USB 3.0 To 2.5in SATA 7+15Pin Hard Drive Adapter”*	2.5″ SATA to USB 3.0	Extremely cheap adapter from AliExpress — MADATALIEXPRESS bought 5 of them and none worked, PPCM had one working, very unreliable and slow when it does work, not recommended even if you get lucky!
EWENT USB 3.0 to SATA EW7017	2.5″ SATA to USB 3.0	Does not work – reported by Wouter in the comments, thanks!
CableCreation USB 3.0 to SATA Adapter Compatible 2.5″ SATA III HDD Hard Disk Driver, 0.5FT, Black*	2.5″ SATA to USB 3.0	Morgon reports not working in the comments — I also recognize this adapter as the “fake” StarTech adapter that is sold on AliExpress, thanks Morgon!
JSAUX USB 3.0 to SATA Adapter, USB 3.0 to 2.5 Inch SATA III Hard Drives/SSD/HDD Adapter*	2.5″ SATA to USB 3.0	Reported as not working by Bennie in the comments, thanks!

EZCast M.2 NVME SSD Enclosure Adapter*	M.2 NVMe to USB 3.1 Gen2	Reported as problematic due to duplicate USB ids — best to avoid — thanks MikeC
Sabrent 2.5″ Aluminum Enclosure*	2.5″ SATA to USB 3.0	Reported as not working by JeffG

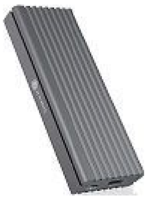
Known Problematic Adapters

Equipment Used

Samsung 980 Pro NVMe SSD

The Samsung 980 Pro (NVMe) is a professional grade SSD and one of the fastest in the world. The Samsung NVMe drives have been at the top of this category for a long time and are well trusted for both their performance and reliability / long life.

Links: AliExpress*, Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.jp*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*, Amazon.sg*


ICY BOX M.2 NVMe Enclosure

The ICY BOX is basically a giant heatsink that you mount a high performance M.2 NVMe drive inside of. This enclosure is really fast but requires a powered USB hub. Not even the 3.5A adapter can reliably power it! The enclosure works well and will physically feel warm to the touch as it is pulling the heat off your NVMe drive!

Links: Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*, Amazon.sg*

2.5″ SATA Option:

Kingston A400 2.5″ SATA SSD

The Kingston A400 has been a great drive to use with the Pi for years. It’s reliable, widely available around the world, has low power requirements and performs very well. It’s also very affordable. This drive has been benchmarked over 1000 times at pibenchmarks.com and is the #1 most popular SSD among the Pi community!

Links: AliExpress*, Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.jp*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*, Amazon.sg*

Compact Option:

SanDisk Extreme Pro SSD

The SanDisk Extreme Pro USB SSD is a true solid state drive. This is different than a typical “flash drive” which uses extremely cheap memory and has very low random I/O performance/throughput compared to a real solid state drive. I’ve used both the USB 3.1 and USB 3.2 variants with the Pi successfully and they benchmark very well!

Links: AliExpress*, Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.jp*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*, Amazon.sg*

You may use other types of drives with the Pi such as M.2 SATA to USB 3.0 and m-SATA to USB 3.0. Here's some adapters I've used for those types of drives:

VL716 mSATA Enclosure

The VL716 mSATA enclosure lets you connect micro SATA drives to the Pi. These drives are an older type of SSD (usually seen in laptops) predating the M.2 slot but are still widely available and perform extremely well!

Links: AliExpress*, Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.jp*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.se*, Amazon.sg*

UGREEN M+B Key M.2 Enclosure

The UGREEN M+B enclosure is a great enclosure for the Pi for M.2 SATA 2280 NGFF drives. It supports both B-key and M-key drives. Does not support newer NVMe drives. As with other types of enclosures it requires more power than other options!

Links: AliExpress*, Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.jp*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*, Amazon.sg*

Argon ONE Pi 4 M.2 Case

The Argon ONE M.2 is a M.2 SATA Pi 4 case / storage solution. With the case and M.2 SATA expansion board you can completely enclosure your Pi 4 and have a built in M.2 slot! The M.2 SATA board is sometimes sold separately from the case itself and can be used as well. Does not support NVMe, this is for SATA M.2 drives only!

Links: AliExpress*, Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.jp*, Amazon.co.uk*, Amazon.de*, Amazon.es*, Amazon.fr*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*, Amazon.sg*

Power Requirements

Power can be a serious problem with these drives. We are learning from the comments that you are especially likely to run into power issues with NVMe enclosures. A powered USB hub or a power adapter that puts out 3.5A comes not only just strongly recommended, it may actually be required that you choose one option or the other for your drive to function.

The specific requirements of how much power you'll need depend on the adapter/enclosure and the model of your drive itself. As a very rough guideline, older models of drives tend to use more power than newer models of drives. 3.5″ form factor drives also use more power than 2.5″ drives. The earliest SSD models like first and second generation models are also well understood to use significantly more power than newer models. This is due to changes and improvements in technology over the years and even using different more efficient memory like 3D NAND. Some super high end performance drives will consume more power as well.

Here's the current recommendations based on everyone's comments combined with stuff I've personally used with the Pi:

CanaKit 3.5A Power Adapter

The CanaKit 3.5A adapter has an extra half an amp (500 mA) of capacity to give some breathing room to your accessories. This is bigger than the official Pi power supply which provides 3.0A.

Links: Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.sg*

Sabrent Powered USB 3.0 Hub

The Sabrent powered USB hub delivers a whopping 2.5A of dedicated power for your USB attached devices. This is almost as much as the Pi adapter itself is rated for (3.0A). It will easily power the most thirsty of setups such as NVMe enclosures.

Links: Amazon.com*, Amazon.ca*, Amazon.com.au*, Amazon.co.uk*, Amazon.es*, Amazon.it*, Amazon.nl*, Amazon.pl*, Amazon.se*

Note: Make sure Amazon doesn't try to take you to the non-powered version and that it's the one with the AC adapter that plugs in to provide extra power

Fixing (some) USB Adapter Problems Using Quirks

Some of the very common adapters on the naughty list above (such as the Sabrent) can be made to work by using USB quirks to disable UAS mode on the drive. This lowers performance, but it's still much faster than a SD card and your adapter won't go to waste.

To find out the quirks we need to find the device ID string for your adapter and then add an entry to cmdline.txt telling the kernel to apply them on boot.

Find Your Adapter

To apply the quirks we first need to get the adapter id. We will use the sudo lsusb command:

```
$ sudo lsusb
 Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
 Bus 002 Device 002: ID 174c:55aa ASMedia Technology Inc. Name:
ASM1051E SATA 6Gb/s bridge, ASM1053E SATA 6Gb/s bridge, ASM1153 SATA
3Gb/s bridge, ASM1153E SATA 6Gb/s bridge
 Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
 Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
 Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

On line 2 we can see my ASM1051E SATA 6Gb/s bridge adapter (it's the known working StarTech.com 2.5″ SATA to USB adapter*). You will see something very similar to mine when you run the command and it shouldn't be too hard to figure out which device it is. If you need more information add a -v switch to make the command sudo lsusb -v. This can sometimes add some additional details to make it easier to figure out which one is your adapter.

If you're still not sure, we have another command that between the two that can narrow things down. Type / paste the following:

```
sudo dmesg | grep usb
```

```
 [    0.828535] usb usb3: New USB device found, idVendor=1d6b,
idProduct=0002, bcdDevice= 4.19
 [    0.828568] usb usb3: New USB device strings: Mfr=3, Product=2,
SerialNumber=1
 [    0.828597] usb usb3: Product: DWC OTG Controller
 [    0.828620] usb usb3: Manufacturer: Linux 4.19.75-v7l+ dwc_otg_hcd
 [    0.828644] usb usb3: SerialNumber: fe980000.usb
 [    0.830051] usbcore: registered new interface driver uas
 [    0.830182] usbcore: registered new interface driver usb-storage
 [    0.836488] usbcore: registered new interface driver usbhid
 [    0.836511] usbhid: USB HID core driver
 [    0.971598] usb 1-1: new high-speed USB device number 2 using
xhci_hcd
 [    1.154217] usb 1-1: New USB device found, idVendor=2109,
idProduct=3431, bcdDevice= 4.20
 [    1.154254] usb 1-1: New USB device strings: Mfr=0, Product=1,
SerialNumber=0
 [    1.154281] usb 1-1: Product: USB2.0 Hub
 [    1.301989] usb 2-1: new SuperSpeed Gen 1 USB device number 2 using
xhci_hcd
 [    1.332965] usb 2-1: New USB device found, idVendor=174c,
idProduct=55aa, bcdDevice= 1.00
 [    1.332999] usb 2-1: New USB device strings: Mfr=2, Product=3,
SerialNumber=1
 [    1.333026] usb 2-1: Product: ASM105x
 [    1.333048] usb 2-1: Manufacturer: ASMT
 [    1.333071] usb 2-1: SerialNumber: 123456789B79F
```

This is the dmesg log showing the hardware detection as hardware is activated on the Pi. If your log is really long you can generate fresh entries by just unplugging a device and plugging it back in and running the command again. Here we can clearly see that the ASM105x is what our StarTech adapter is being detected as.

Now we can go back to our first lsusb command and we want the 8 characters from the ID field that comes right after the Device:

```
Bus 002 Device 002: ID 174c:55aa ASMedia Technology Inc. Name: ASM1051E
SATA 6Gb/s bridge
```

Our adapter’s ID is: 174c:55aa

Applying Quirks

To apply the quirks to our USB adapter we are going to edit /boot/cmdline.txt. Type:

```
sudo nano /boot/cmdline.txt
```

We are going to add the following entry into the very front of cmdline.txt:

```
usb-storage.quirks=XXXX:XXXX:u
```

In place of the X's above you will put in your adapter's ID that we got before. With the example commands I gave above mine would look like this: usb-storage.quirks=174c:55aa:u. After this my cmdline.txt looks like this (everything should be one continuous line, no line breaks!):

```
usb-storage.quirks=174c:55aa:u console=serial0,115200 console=tty1
root=PARTUUID=d34db33f-02 rootfstype=ext4 elevator=deadline
fsck.repair=yes rootwait
```

Now reboot the Pi. If the Pi fails to boot you can plug the SD card into the computer and go to /boot/cmdline.txt and undo the change we did so you can boot back in with your SD card.

Verifying Quirks

Once you have rebooted after changing cmdline.txt we can verify the quirks have been applied by doing another dmesg | grep usb command:

```
sudo dmesg | grep usb
 [    1.332924] usb 2-1: New USB device found, idVendor=174c,
idProduct=55aa, bcdDevice= 1.00
 [    1.332957] usb 2-1: New USB device strings: Mfr=2, Product=3,
SerialNumber=1
 [    1.332983] usb 2-1: Product: ASM105x
 [    1.333006] usb 2-1: Manufacturer: ASMT
 [    1.333028] usb 2-1: SerialNumber: 123456789B79F
 [    1.335967] usb 2-1: UAS is blacklisted for this device, using usb-
storage instead
 [    1.336071] usb 2-1: UAS is blacklisted for this device, using usb-
storage instead
 [    1.336103] usb-storage 2-1:1.0: USB Mass Storage device detected
 [    1.336479] usb-storage 2-1:1.0: Quirks match for vid 174c pid
55aa: c00000
 [    1.336611] scsi host0: usb-storage 2-1:1.0
```

This time we can see in dmesg that UAS was blacklisted for the device and it has loaded with the usb-storage driver instead. This driver tends to be more compatible with the "problematic adapters" but the performance is usually significantly lower. It's definitely worth a try though as some adapters do better with the quirks performance-wise. The only way to know for sure is to run a benchmark (see storage benchmark section near the end).

USB Boot Instructions

There are a lot of steps to follow to set everything up properly. If you make a mistake the first time don't spend too much time trying to correct it or figure out what you did wrong. It's usually faster to burn the images again and reconfigure again rather than try to figure out which step you might have made a typo on. It's much easier the second time!

Prepare SD Card

Download the latest Raspberry Pi OS release from the Official Raspberry Pi download page. Both Lite or Desktop versions will work. Win32DiskImager (Windows) or balenaEtcher (Linux, Mac OS X, Windows) are highly recommended to burn the images.

Note: Don't attempt to use raw dd commands to write the images. Too many silly things can go wrong that are checked for/fixed by the recommended programs. Do yourself a big favor and use one of the recommends to avoid spending a ton of time troubleshooting basic imaging problems!

Preparing SSD

We are going to burn a second identical copy of Raspbian to the SSD. This ensures everything the Pi needs to boot is there so we can use the SD card as a bootloader but our actual system will be on our nice fast SSD drive.

Note: Make sure you create the empty file named "ssh" on the boot partition of both drives if you are headless or don't have a mouse/keyboard attached so you can ssh in on the first boot.

First Boot (SD card only, SSD unplugged)

Insert your freshly imaged SD card into the Pi and connect the power. Sign into the Pi for the first time.

Don't do an apt-get upgrade/dist-upgrade or any additional configuration yet. Just stick with the instructions until we have finished configuration (especially since if something goes wrong you may have to start over and any other setup you did will be lost).

Once the Pi has finished booting and you have signed in for the first time plug in your SSD to your Pi's USB 3.0 ports. The USB 3.0 ports are the ones that have the blue plastic inside instead of the black plastic (the black ones are USB 2.0 ports).

Change PARTUUID

We need to change the PARTUUID of our SSD's partitions so the Pi doesn't get confused about what device to boot from. Right now the partitions on both the SD card and the SSD are an exact match and we need them to be different so we can tell the Pi to boot specifically from our SSD's partition.

We are going to use fdisk to change the SSD's PARTUUID to the hexadecimal d34db33f to make our SSD easy to identify. Use the following:

```
$ sudo fdisk /dev/sda

 Welcome to fdisk (util-linux 2.33.1).
```

```
  Changes will remain in memory only, until you decide to write them.
  Be careful before using the write command.
  Command (m for help): p
  Disk /dev/sda: 238.5 GiB, 256060514304 bytes, 500118192 sectors
  Disk model: ASM105x
  Units: sectors of 1 * 512 = 512 bytes
  Sector size (logical/physical): 512 bytes / 512 bytes
  I/O size (minimum/optimal): 512 bytes / 512 bytes
  Disklabel type: dos
  Disk identifier: 0x6c586e13
  Device     Boot  Start       End   Sectors   Size Id Type
  /dev/sda1         8192    532479    524288   256M  c W95 FAT32 (LBA)
  /dev/sda2       532480 500118191 499585712 238.2G 83 Linux
  Command (m for help): x
  Expert command (m for help): i
  Enter the new disk identifier: 0xd34db33f
  Disk identifier changed from 0x6c586e13 to 0xd34db33f.
  Expert command (m for help): r
  Command (m for help): w
  The partition table has been altered.
  Syncing disks.
```

That’s it. Let’s verify our change using blkid:

```
$ sudo blkid

/dev/mmcblk0p1: LABEL_FATBOOT="boot" LABEL="boot" UUID="5203-DB74"
TYPE="vfat" PARTUUID="6c586e13-01"
/dev/mmcblk0p2: LABEL="rootfs" UUID="2ab3f8e1-7dc6-43f5-b0db-
dd5759d51d4e" TYPE="ext4" PARTUUID="6c586e13-02"
/dev/sda1: LABEL_FATBOOT="boot" LABEL="boot" UUID="5203-DB74"
TYPE="vfat" PARTUUID="d34db33f-01"
/dev/sda2: LABEL="rootfs" UUID="2ab3f8e1-7dc6-43f5-b0db-dd5759d51d4e"
TYPE="ext4" PARTUUID="d34db33f-02"
```

Your /dev/mmcblk0 and /dev/sda devices should now be different from each other. The SD card’s ID is 6c586e13 and the SSD’s PARTUUID is now

Update /boot/cmdline.txt

We are going to change cmdline.txt to point to the SSD for booting instead of the SD card. First make a backup of your existing cmdline.txt file with the following command:

```
sudo cp /boot/cmdline.txt /boot/cmdline.txt.bak
```

We’ve now created a backup you can restore if something goes wrong. If you need to restore your backup plug the SD card into a computer/device and replace cmdline.txt with cmdline.txt.bak that we made above. Now your Pi should boot normally again.

Open up /boot/cmdline.txt using nano or your favorite text editor:

```
sudo nano /boot/cmdline.txt
```

The existing file will look like this:

```
console=serial0,115200 console=tty1 root=PARTUUID=6c586e13-02
rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait quiet
init=/usr/lib/raspi-config/init_resize.sh
```

We are going to change the root=PARTUUID section to point to our new d34db33f PARTUUID like the following:

```
console=serial0,115200 console=tty1 root=PARTUUID=d34db33f-02
rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait quiet
init=/usr/lib/raspi-config/init_resize.sh
```

Make the change and double check the line is what it should be,then press Ctrl+X to save our changes.

Note: cmdline.txt should be one long solid line with no breaks — don’t add any line breaks or the system won’t boot and you’ll need to restore the backup we made earlier!

Test SSD

We are now ready to test booting from the SSD. Restart your Pi by issuing a

```
sudo reboot
```

The first boot with your SSD can be slow due to running fsck on the drive. If you have a really large SSD it can take surprisingly long to check all that space. Once the check completes it will mark the drive clean and skip the disk check from now on. It can take over a minute or two sometimes for really big drives so give it at least that much time before assuming it didn’t work.

After signing in we can verify that the SSD is being used like this:

```
$ findmnt -n -o SOURCE /

  /dev/sda2
```

Verify that partition has switched over as shown below to /dev/sda2 (SSD) instead of /dev/mmcblk0p2 (SD card).

Update /etc/fstab

We are now ready to edit the /etc/fstab file to point to our updated drive. To edit the file type:

```
sudo nano /etc/fstab
```

Your current file will look like this:

```
$ cat /etc/fstab
proc            /proc           proc    defaults          0       0
PARTUUID=6c586e13-01  /boot           vfat    defaults          0
2
PARTUUID=6c586e13-02  /               ext4    defaults,noatime  0
1

sudo nano /etc/fstab
```

Your current file will look similar to this (PARTUUID varies based on your Raspbian image version):

```
cat /etc/fstab
proc            /proc           proc    defaults          0       0
PARTUUID=6c586e13-01  /boot           vfat    defaults          0
2
PARTUUID=6c586e13-02  /               ext4    defaults,noatime  0
1
```

We want to change the root (/) partition (PARTUUID ending with -02) to load our SSD’s PARTUUID instead of the SD card. Replace the 2nd partition’s PARTUUID field on the last line in the file with the d34db33f label we applied earlier with fdisk. After making the change my /etc/fstab file looks like this:

```
proc            /proc           proc    defaults          0       0
PARTUUID=6c586e13-01  /boot           vfat    defaults          0
2
PARTUUID=d34db33f-02  /               ext4    defaults,noatime  0
1
```

Press Ctrl+X to tell nano to save our changes. Now type sudo reboot to restart the Pi.

Note: We want to leave the first partition (/boot) on the SD card. If you change this to the SSD then apt will update your SSD instead of the SD card so they won’t be used during boot! Remember that we are using the SD card as a bootloader and that is why the firmware updates (such as start.elf, etc) should go there instead of the SSD’s boot partition (which is never used).

Resizing Filesystem

By default the partition on the SSD / Flash drive will only be 1.8G. The Pi expands this automatically on micro SD drives but we will need to do it ourselves for a SSD / Flash drive. To do this we need to expand the partition and then resize the file system.

First let’s open fdisk and print the partitions:

```
pi@raspberrypi:~ $ sudo fdisk /dev/sda
 Welcome to fdisk (util-linux 2.33.1).
 Changes will remain in memory only, until you decide to write them.
 Be careful before using the write command.
 Command (m for help): p
 Disk /dev/sda: 238.5 GiB, 256060514304 bytes, 500118192 sectors
 Disk model: ASM105x
 Units: sectors of 1 * 512 = 512 bytes
 Sector size (logical/physical): 512 bytes / 512 bytes
 I/O size (minimum/optimal): 512 bytes / 512 bytes
 Disklabel type: dos
 Disk identifier: 0xd34db33f
 Device     Boot  Start     End Sectors  Size Id Type
 /dev/sda1         8192  532479  524288  256M  c W95 FAT32 (LBA)
 /dev/sda2       532480 4390911 3858432  1.9G 83 Linux
```

There is the line we need. Our start value for /dev/sda2 (rootfs) is 532480. Next we need to remove and recreate the partition as a larger size.

If you make any mistakes during this command just close fdisk by pressing q. The changes won’t be written to disk. If you mess up any of the commands the drive will no longer boot and you’ll have to start over again so be careful!

```
Command (m for help): d
 Partition number (1,2, default 2): 2
 Partition 2 has been deleted.
 Command (m for help): n
 Partition type
    p   primary (1 primary, 0 extended, 3 free)
    e   extended (container for logical partitions)
 Select (default p): p
 Partition number (2-4, default 2): 2
 First sector (2048-500118191, default 2048): 532480
 Last sector, +/-sectors or +/-size{K,M,G,T,P} (532480-500118191,
default 500118191): 500118191
 Created a new partition 2 of type 'Linux' and of size 238.2 GiB.
 Partition #2 contains a ext4 signature.
 Do you want to remove the signature? [Y]es/[N]o: N
 Command (m for help): w
 The partition table has been altered.
 Syncing disks.
```

If everything went well then type “w” and press enter. Otherwise press “q” to quit and try again. Once you enter “w” the changes will be permanently written to disk!

Now reboot the system. Type “df -h” to view the current disk:

```
pi@raspberrypi:~ $ df -h
 Filesystem      Size  Used Avail Use% Mounted on
 /dev/root       1.8G  1.3G  415M  76% /
 devtmpfs        1.8G     0  1.8G   0% /dev
 tmpfs           2.0G     0  2.0G   0% /dev/shm
 tmpfs           2.0G  8.5M  1.9G   1% /run
 tmpfs           5.0M  4.0K  5.0M   1% /run/lock
 tmpfs           2.0G     0  2.0G   0% /sys/fs/cgroup
 /dev/mmcblk0p1  253M   52M  201M  21% /boot
 tmpfs           391M     0  391M   0% /run/user/1000
```

We can see our disk is still 1.8G even after resizing the partition. That’s because we still have one more step! We need to resize the filesystem to fill our new partition space. For this we will use “sudo resize2fs /dev/sda2”:

```
sudo resize2fs /dev/sda2
 resize2fs 1.44.5 (15-Dec-2018)
 Filesystem at /dev/sda2 is mounted on /; on-line resizing required
 old_desc_blocks = 1, new_desc_blocks = 15
 The filesystem on /dev/sda2 is now 62448214 (4k) blocks long.
```

Now let’s check df -h again:

```
pi@raspberrypi:~ $ df -h
 Filesystem      Size  Used Avail Use% Mounted on
 /dev/root       235G  1.3G  224G   1% /
 devtmpfs        1.8G     0  1.8G   0% /dev
 tmpfs           2.0G     0  2.0G   0% /dev/shm
 tmpfs           2.0G  8.5M  1.9G   1% /run
 tmpfs           5.0M  4.0K  5.0M   1% /run/lock
 tmpfs           2.0G     0  2.0G   0% /sys/fs/cgroup
 /dev/mmcblk0p1  253M   52M  201M  21% /boot
 tmpfs           391M     0  391M   0% /run/user/1000
```

And that’s it! You will now be using all of your space on your drive.

Verify SSD Functionality / Performance

You can make sure everything is running correctly (and as fast as it should be) by running my quick storage benchmark. You can run the benchmark with the following one-liner:

```
sudo
curl https://raw.githubusercontent.com/TheRemote/PiBenchmarks/master/St
orage.sh | sudo bash
```

This will give you a score you can compare to the other Raspberry Pi Storage Benchmark results and make sure that you are getting an equivalent speed to your peers with the same device!

Update Pi using apt

Now that we’ve updated fstab it is safe (and highly recommended) to update your Pi’s software. Type “sudo apt-get update && sudo apt-get dist-upgrade” to update the system and firmware.

Your system will now be running completely from your USB drive! To verify this, run the command “findmnt -n -o SOURCE” / to ensure your root partition has switched over as shown below to /dev/sda2 instead of /dev/mmcblk0p2.

Conclusion

The Samsung 950 Pro NVMe drive in the featured picture scored a 9189 on the Raspberry Pi Storage Benchmark. The previous all-time record score on a Pi 3B+ was 3561. The performance gains are very real and very dramatic.

For me getting this performance is well worth having to waste a micro SD card just to be a bootloader. I am largely after the USB 3.0 bus and gigabit ethernet performance improvements and using this method I am able to achieve the performance I was after without waiting an indeterminate amount of time for the feature to be added back in!

Although there are ongoing compatibility issues and we lack the super easy native USB booting support we had before I’m more than willing to go through the growing pains to finally get rid that ancient USB 2.0 bus! Just make sure if you are planning to build a system you plan your adapters and parts accordingly.

Other Resources

If you want to see which Pi storage performs the fastest and get an idea of what kind of drives to look for check out my 2020’s Fastest Raspberry Pi 4 Storage Benchmarks

If you have one of the new Raspberry Pi 400 kits *then don’t miss my Pi 400 Overclocking and SSD Setup Guide

Related Posts:

Minecraft V1.12 Server - Excellent Performance Guide

Pwnagotchi WiFi Audit Tool Build / Guide

Best Working SSD / Storage Adapters for Raspberry Pi 4 / 400

Raspberry Pi 4 Ubuntu Server / Desktop 18.04.4 Image

Raspberry Pi Minecraft Server Setup Script w/ Startup Service

Goldshell BOX ASIC Miner Firmware Recovery Guide