Waveshare IO Board

- Programme for Mon 19th Feb 2024
  - 1945 Introduction
  - 1950 Waveshare IO board
  - 2010 Storage media
  - 2030 Formatting an NVME drive
  - 2050 Dual boot machine
  - 2100 Build it yourself
  - 2110 Question & answer
  - 2120 Close

Introduction

- Aim is to promote questions & discussion
  - Advantages of the Waveshare IO board;
  - Storage media : SATA vs. NVMe;
    - Preparing a multi-partition NVMe drive.
  - Interfacing to the media (PCI express):
    - Pinebook Pro, CM4 IO, Titanium, ARMX6.
  - Missing items on IO boards
    - How to do fan control & add real time clock.

Who am I?

- I have been interested in computer programming, in BASIC, since 1972
- My name is Chris Hall
  - Chartered Mechanical Engineer
  - Career in Nuclear Safety, now retired
  - Have used RISC OS to publish several books
  - Maintain a web site
  - Volunteer on a Heritage Railway

Notes of the talk

- I shall be using a printed set of notes for the talk which are available at:
  - http://www.svrsig.org/RTalk.pdf
- Advantages of Waveshare IO board:
  - Small footprint, uses CM4
  - On-board socket for fast storage (NVMe M.2);
  - Can add extra sockets: 2 USB + 1 HDMI
  - Fan control & real time clock can be added

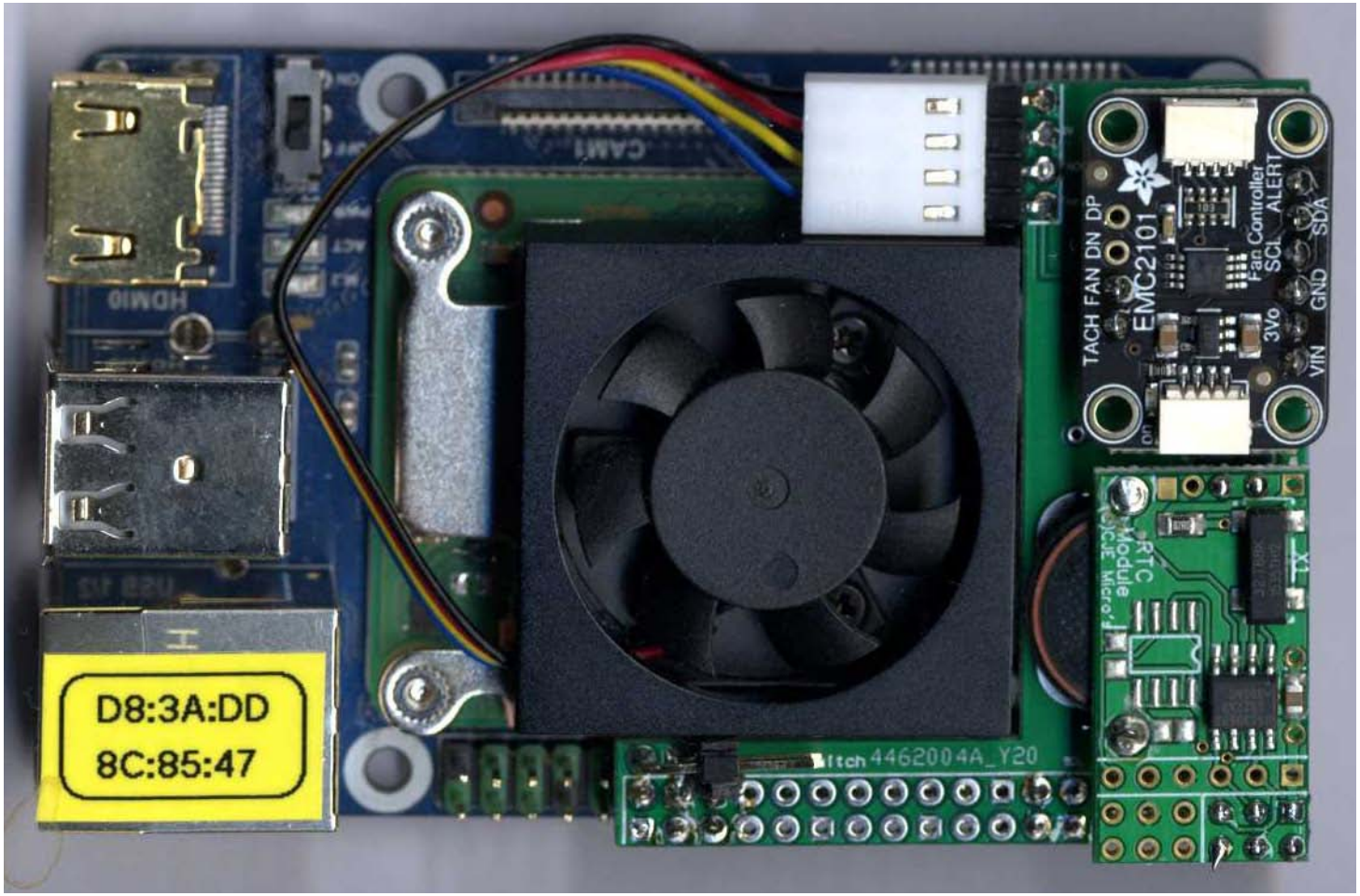
Waveshare Mini-A or Mini-B

- Difference: on-board RTC + fan controller – but RISC OS cannot ‘see’ these
- Uses CM4 - can be clocked to 2000 MHz – so long as cooling is provided
- Just room to add RTC & fan on header
- Linux can use on-board RTC and fan

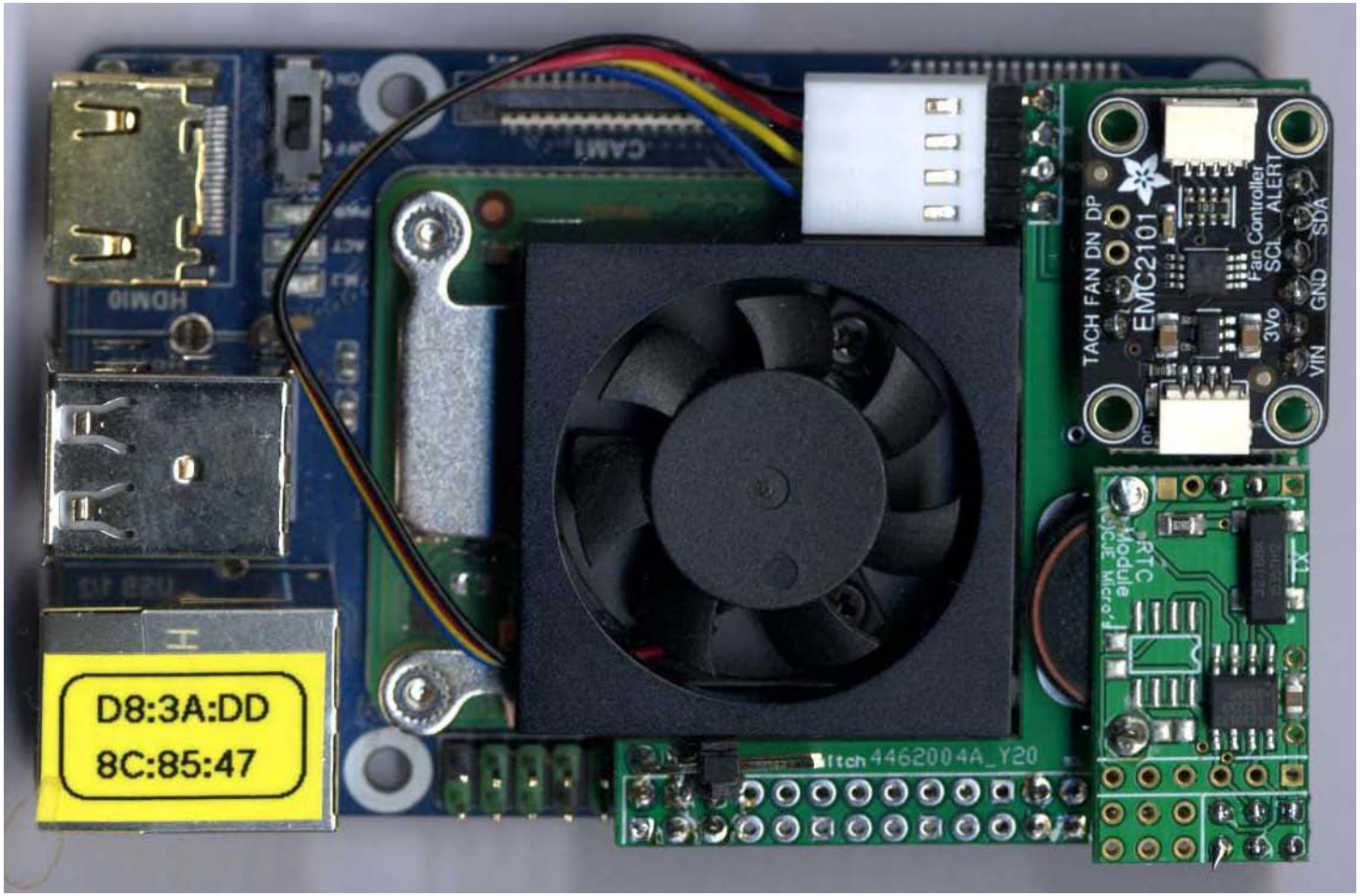
Other IO boards

- Pi Foundation
  - Access to ‘RUN’ pin (can add ‘reset’ switch)
  - Can choose storage medium for PCIe slot;
  - Easier to control speed of 12V fan.
- GeekPi DeskPi Mini (aka PiRO Qube)
  - Similar to Waveshare Mini-A/Mini-B;
  - !CPUClock can control on-board fan;
  - Nice case

Waveshare board – with RTC & fan complete



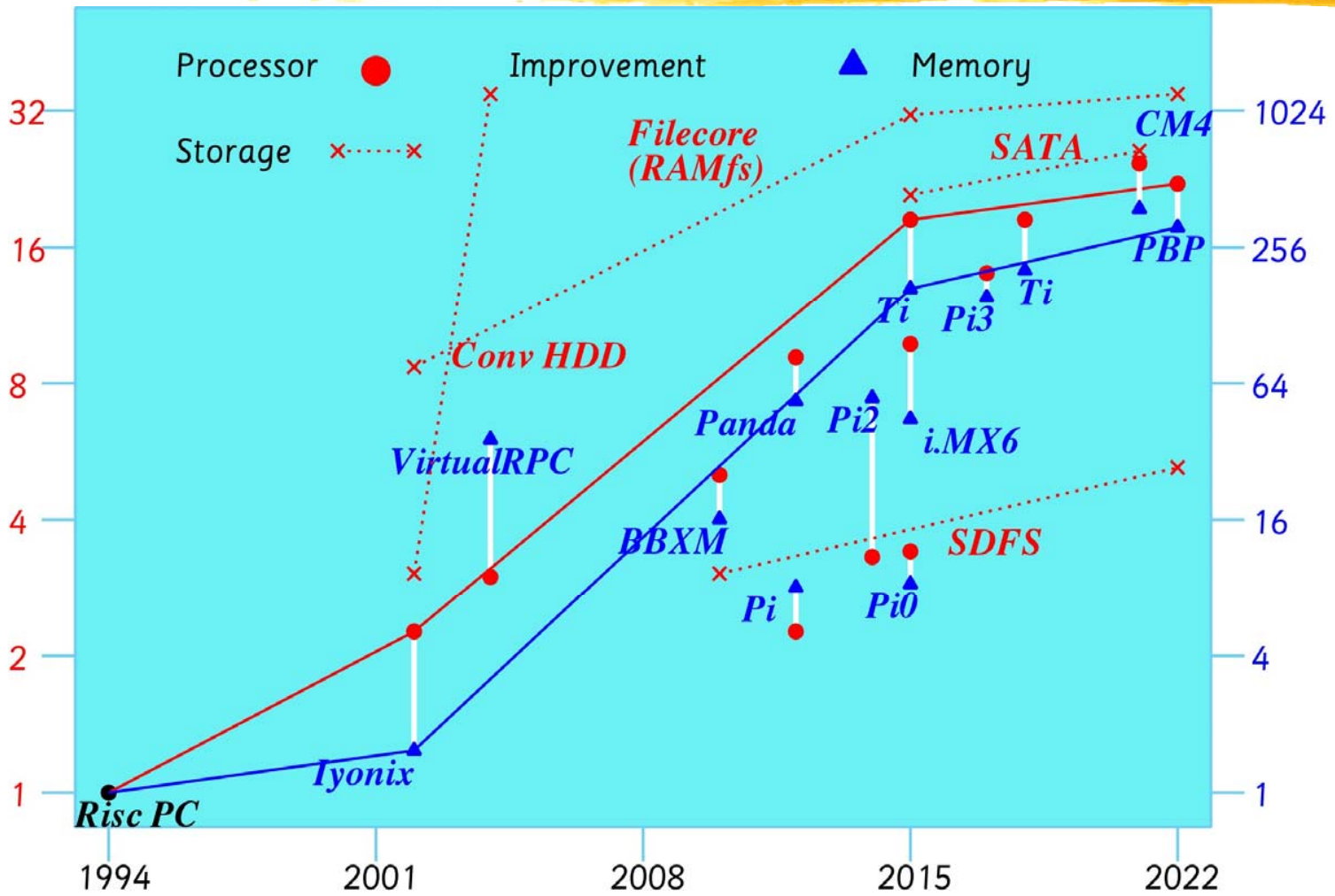
Waveshare board – (pass around the table)



Limitations: storage media

- Storage speed is limited by interface clock
  - SDIO (SDFS, eMMc) 50MHz (50MB/s)
  - PCIe Gen 1 (Titanium) 2Gb/s (200MB/s)
  - PCIe Gen 2x1 (IO board) 4Gb/s (400MB/s)
  - PCIe Gen 2x2 (PBPro) 8GB/s (800MB/s)
- And by filing system (Filecore/RAMfs)
- And by medium:
  - SATA: 540MB/s, NVMe: 7200MB/s

Performance since 1994



Different storage media

- Serial Advanced Technology Attachment
  - SATA: Serial interface – up to 540MB/s
- Non Volatile Memory Express
  - NVMe: low latency and some parallelism;
  - Interface – up to 3500MB/s
- Comparison
  - NVMe has potential to offer much better random read/write speed than SATA

Different storage media

Drive	Basic model	RISC OS	HD Read	HD Write	FS Read	FS Write	HD Read	HD Write	FS Read	FS Write
Sabrent 512GB NVMe over USB		RISC OS	9%	3%	3%	3%	35929	37236	134	136
Sabrent 512GB NVMe over USB		Linux	9%	3%	135%	33%	37000	39800	6645	1672
NVMe Sabrent 512GB	W/s IO Mini-A	5.29 12-Jun-2023	102%	29%	749%	181%	406000	395000	36900	9287
SanDisk 128GB SSD	Pi IO ADFS::5		36%	12%	31%	27%	141784	162217	1543	1374
SanDisk 240GB SATA	Titanium :4	5.24 16-Apr-2018	31%	7%	51%	49%	121663	91022	2524	2536
V series 240GB SATA	Pi IO ADFS::5	RISC OS	90%	25%	30%	31%	356879	337317	1489	1603
V series 240GB SATA	Pi IO ADFS::5	Linux	98%	28%	453%	109%	391000	386000	22300	5609
RAMfs 1500MB			100%	100%	100%	100%	397433	1362629	4928	5134
							Results from Linux shown thus			

- Variability between different SATA drives
- NVMe speed bounded by RAMfs and Linux
- HD read/write bounded by interface 400MB/s
- NVMe FS read/write offers more potential
- Important for copying files etc.

Different storage media

- Detailed discussion SATA vs. NVMe
  - Potential limits with RISC OS (filecore/RAMfs)
  - How I did the testing (using Linux for NVMe)
  - How NVMe looks to RISC OS over USB
  - Notes available www.svrsig.org/SSDs.pdf
- Conclusion
  - NVMe has potential to offer RISC OS much better random read/write speed than SATA

Conclusion

- NVMe offers more potential than SATA
- More compact
  - Waveshare Mini-A/Mini-B – build it yourself;
  - GeekPi DeskPi Mini – halfway house;
  - PiRO Qube – ready built.
- USB 3 n/a but RISC OS can’t drive it
- What can we do with NVME so far?

RISC OS and NVMe

- Work in progress
- What can we do now?
  - An NVMe drive in a USB caddy works
  - ... but only at USB speeds!
- Need both RISC OS and Linux to format it
  - Dual-boot machine makes it easier
- Once prepared, it can be used now for Linux and later for RISC OS

Formatting an NVMe drive

- The RISC OS two-partition solution:
  - Has a FAT partition (1) which can be accessed from RISC OS using Boot:Loader
  - Has a filecore partition (4) for RISC OS files
- Add partitions using Linux:
  - Large FAT partition (2) – for file sharing;
  - Large ext4 partition (3) for Linux to use.
- Tweak the partition table in RISC OS

Formatting an NVMe drive

- HDForm can only do SDFS/SCSI/ADFS
  - Put NVMe drive in a USB caddy
  - Use HDForm to format SCSI::4
  - Use SystemDisc to create FAT partition
  - That's the RISC OS bit done
- Reboot into Linux
  - Add 20GB FAT partition
  - Add 20GB ext4 partition

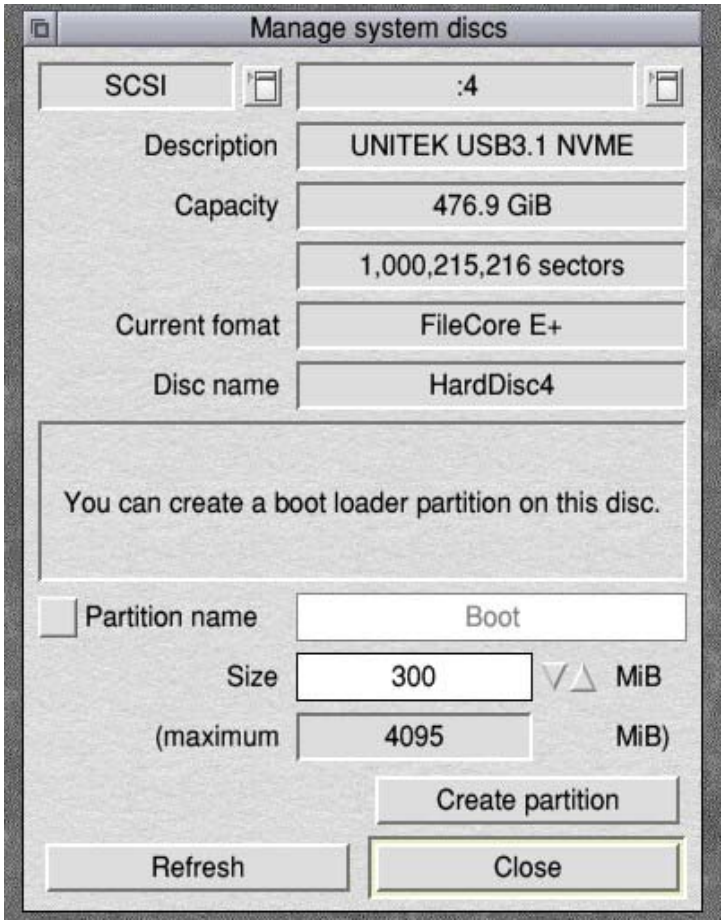
Formatting an NVMe drive

```
FILECORE HARD DISC FORMATTER 2.76 (10-Oct-20)

Is your drive connected to SCSIFS or SDFS (S/M) ? S
Searching for devices...
SCSI::4 : 0:0.0  Direct-access      477 Gbytes UNITEK   USB3.1 NVME      1.00
Format which drive (0 - 7) ? 4
Disc capacity    : 488386 Mbytes
Drive capacity too great. Limiting to 262144 MB
Suggested shape  : 62660 cylinders, 136 heads and 63 sectors/track

Dismounting drive
This drive does not currently have a valid FileCore format.
Heads ? 136
Sectors per track ? 63
Cylinders ? 31000
Parking cylinder ? 62661

A: no more changes to defect list
B: add defect by cylinder, head, byte/sector
C: add defect by disc address
D: remove defect
A,B,C or D ? a

Disc will be formatted as :

          31000 Cylinders, 136 Heads and 63 Sectors/Track
          Parking cylinder 62661

Format or just initialise the drive (F/I) ? I
Soak test the hard disc for defects (Long/Short/None) ? N
Do you wish this to be a bootable disc (Y/N) ? Y
Do you want long filenames on this disc (Y/N) ? Y
Large file allocation unit ? 4096

Are you SURE you want to do this to drive SCSI::4 (Y/N) ? y

Writing defect list
Creating map
Writing map
Writing root directory
Bytes free &0000001FA9341000 =   132,795,652 Kbytes
Bytes used &0000000000827000 =         8,348 Kbytes
```

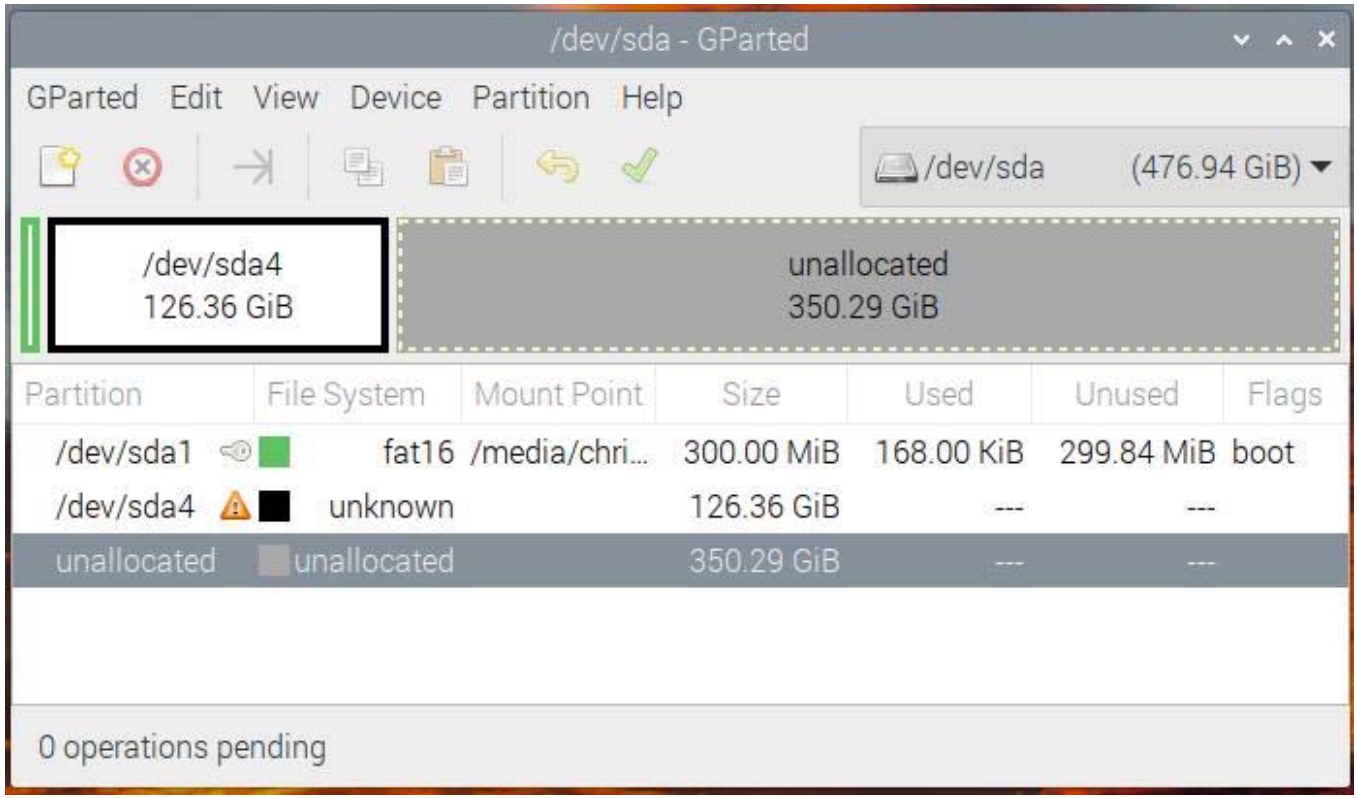
- Using Utilities.Caution.HDFormat

Formatting an NVMe drive

- Now add Loader
  - Check the drive is SCSI::4
  - Increase the size to 300MB
  - Ignore ‘capacity’ as the filecore partition has already been formatted to use just 110 GB
  - ‘Create partition’ will do just that: A file !Boot.Loader that is 300MB with the two partitions otherwise empty but formatted.

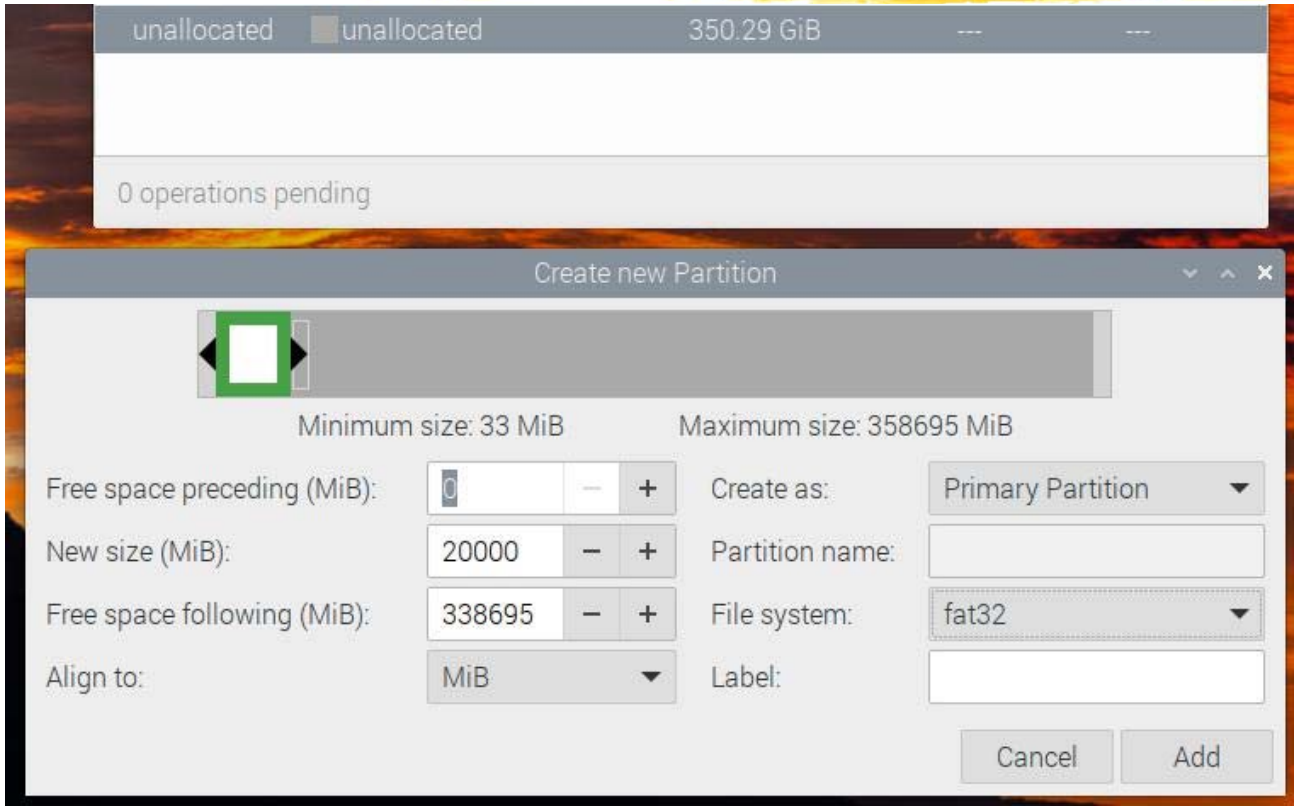


Formatting an NVMe drive



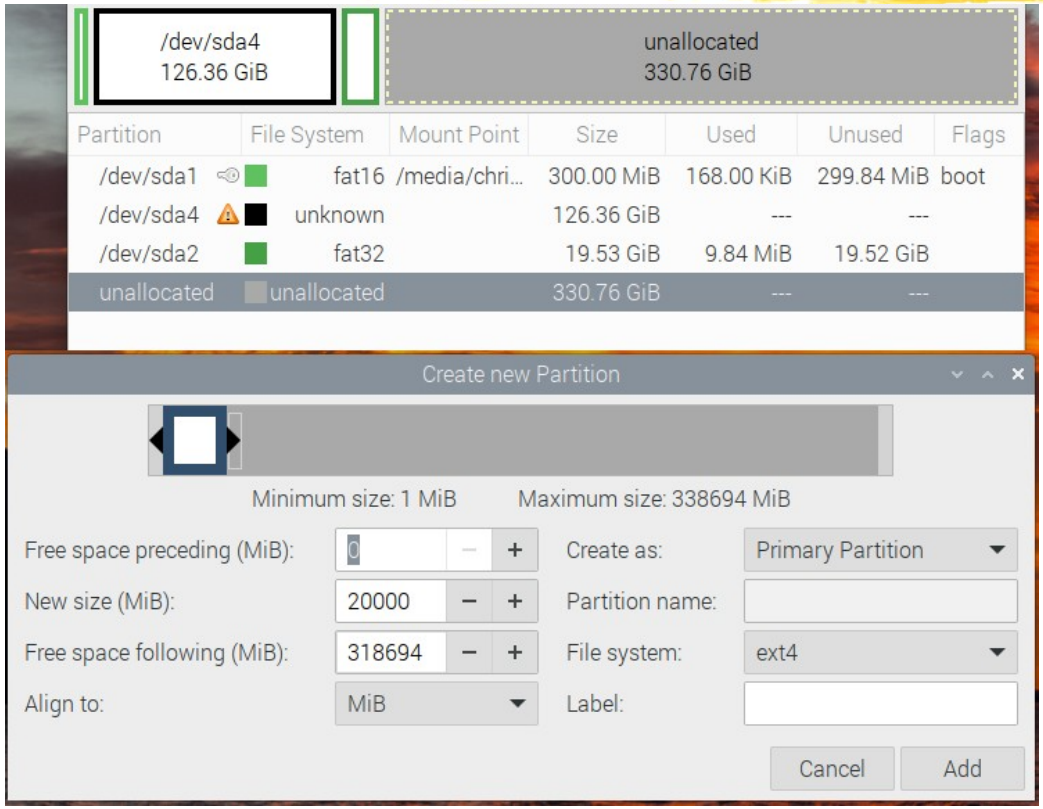
- So far, we have two partitions, 1 and 4

Formatting an NVMe drive



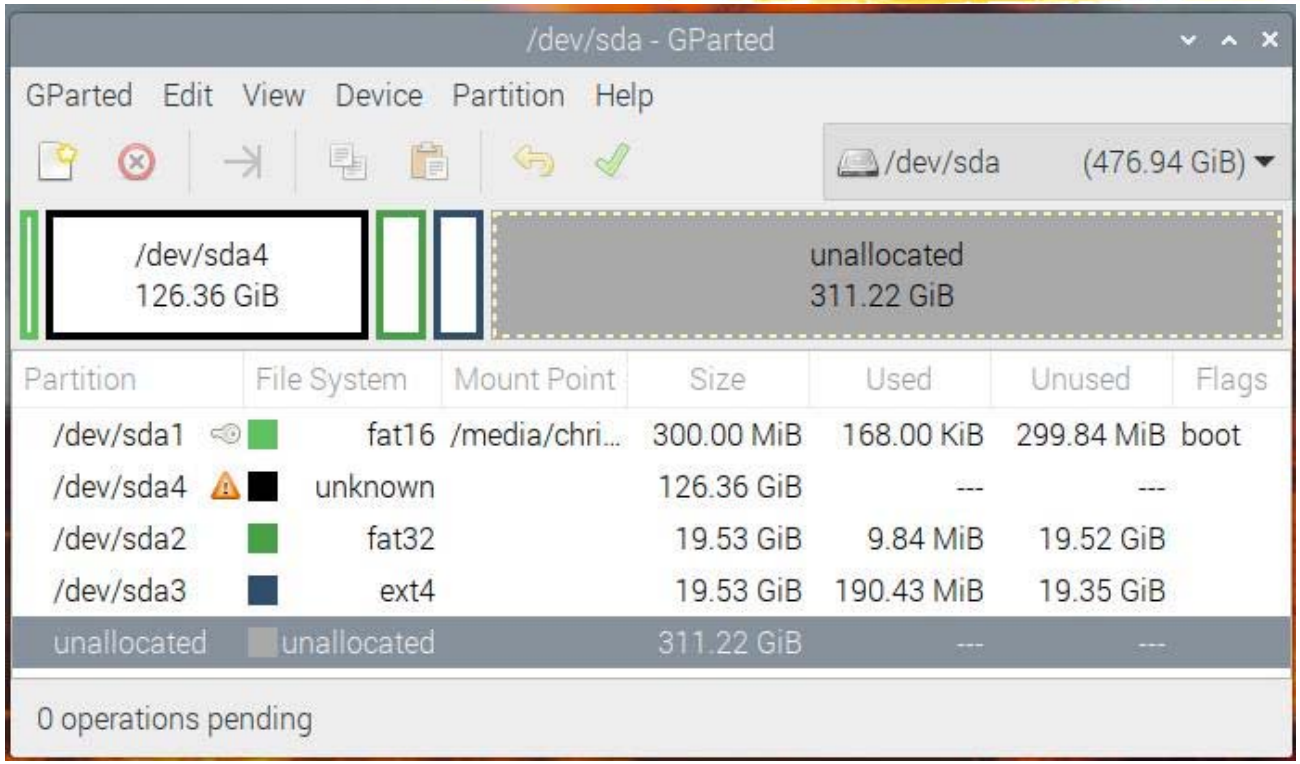
- Adding a 20GB FAT partition 2 after partition 4 using GPartEd
- This could be used for sharing files between RISC OS and Linux

Formatting an NVMe drive



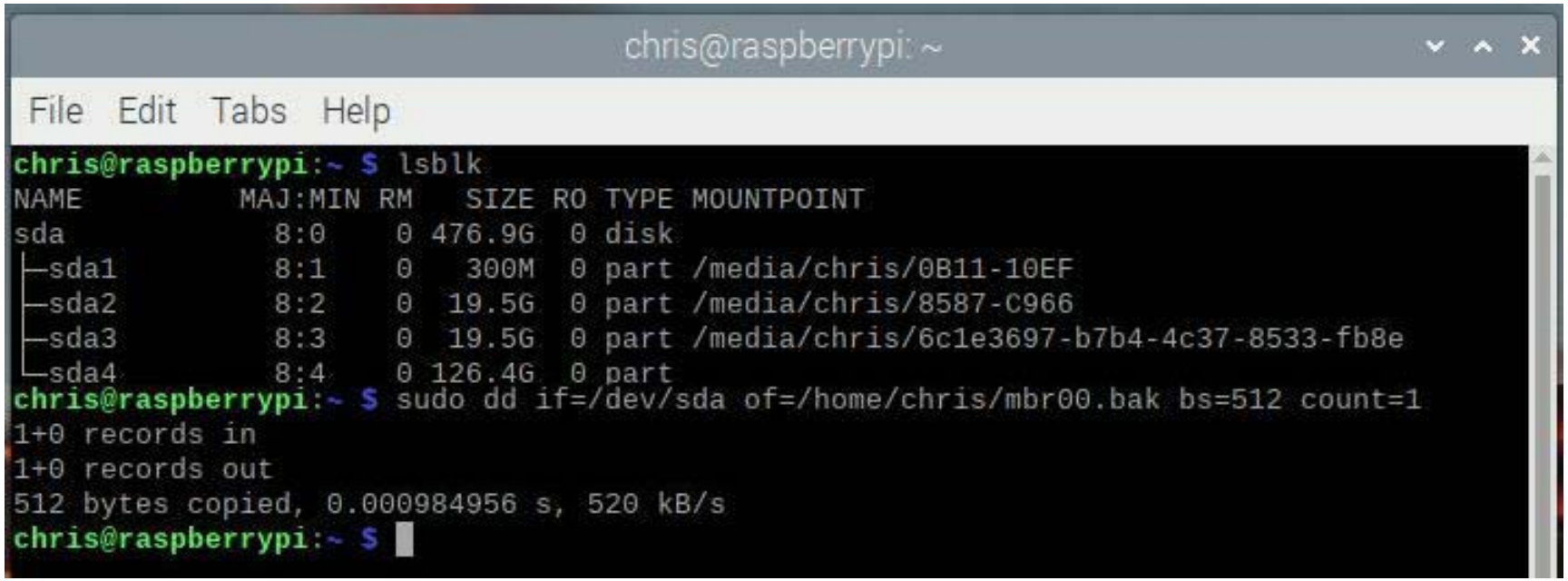
- Adding a 20GB ext4 partition 3 after partition 2 using GPartEd
- This partition will be read only by Linux

Formatting an NVMe drive



- It looks odd and will cause problems
- The RISC OS filecore partition (126GB) is second on disk but listed as fourth in the partition table

Formatting an NVMe drive

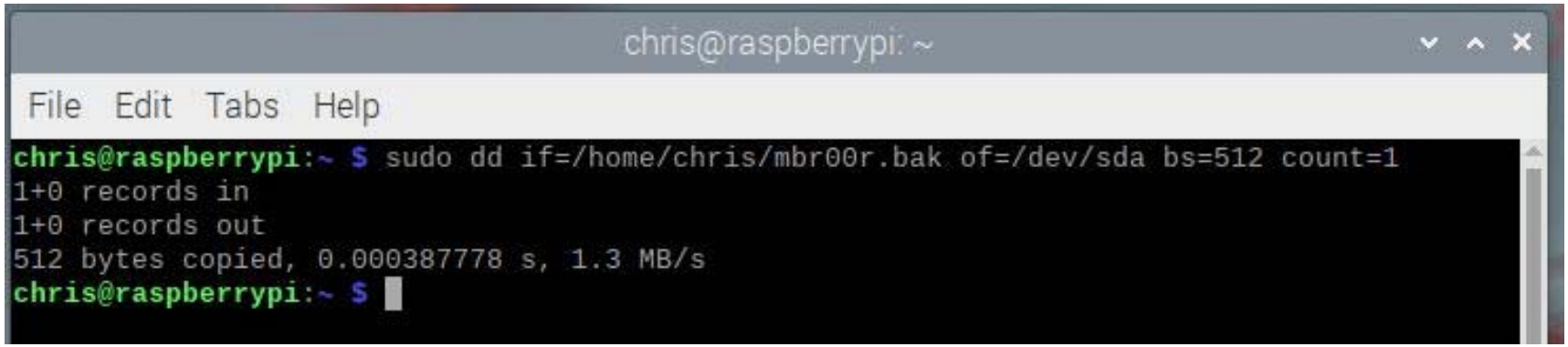


- /dev/sda/ is the NVMe drive in the USB caddy
- The ‘dd’ command is writing its Master Boot Record (MBR) to disc – this contains the 64 byte partition table
- We can copy it from /home/chris/ to the eMMc FAT partition

Formatting an NVMe drive

- What is in the partitions?
  - 1. 300MB FAT (empty FAT 16)
  - 4. 110GB filecore (contains !Boot.Loader)
  - 2. 20 GB FAT (empty and not formatted)
  - 3. 20 GB ext4 (empty and not formatted)
- So use ‘dd’ to write partition table to disc
  - Copy from /home/chris to /dev/sda1 (eMMc)
- Reboot into RISC OS

Formatting an NVMe drive

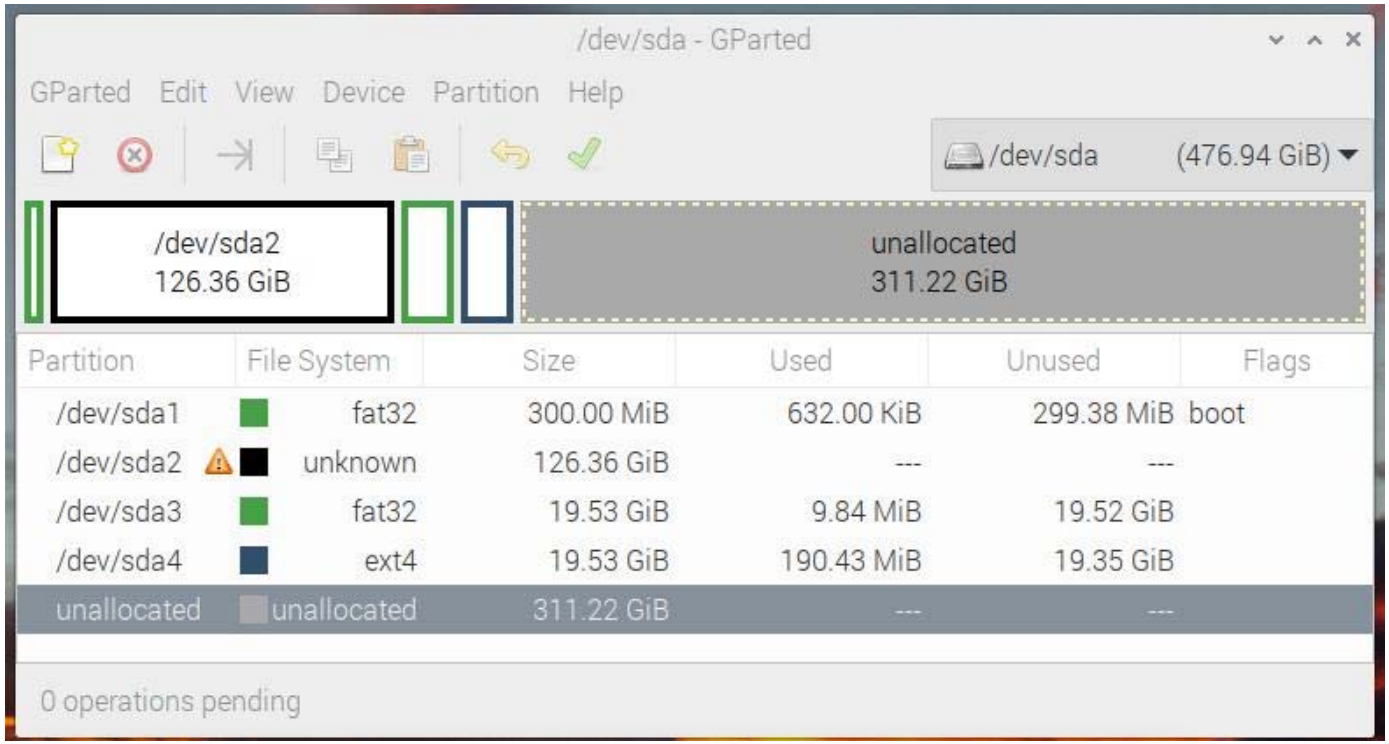


- First we reboot into RISC OS and run a BASIC programme which takes the MBR (mbr00.bak) and shuffles its partition table into the same order as on the disc and saves it as mbr00r.bak
- Now in Linux we write this corrected MBR back to disc
- It is essential now to reboot immediately

Formatting an NVME drive

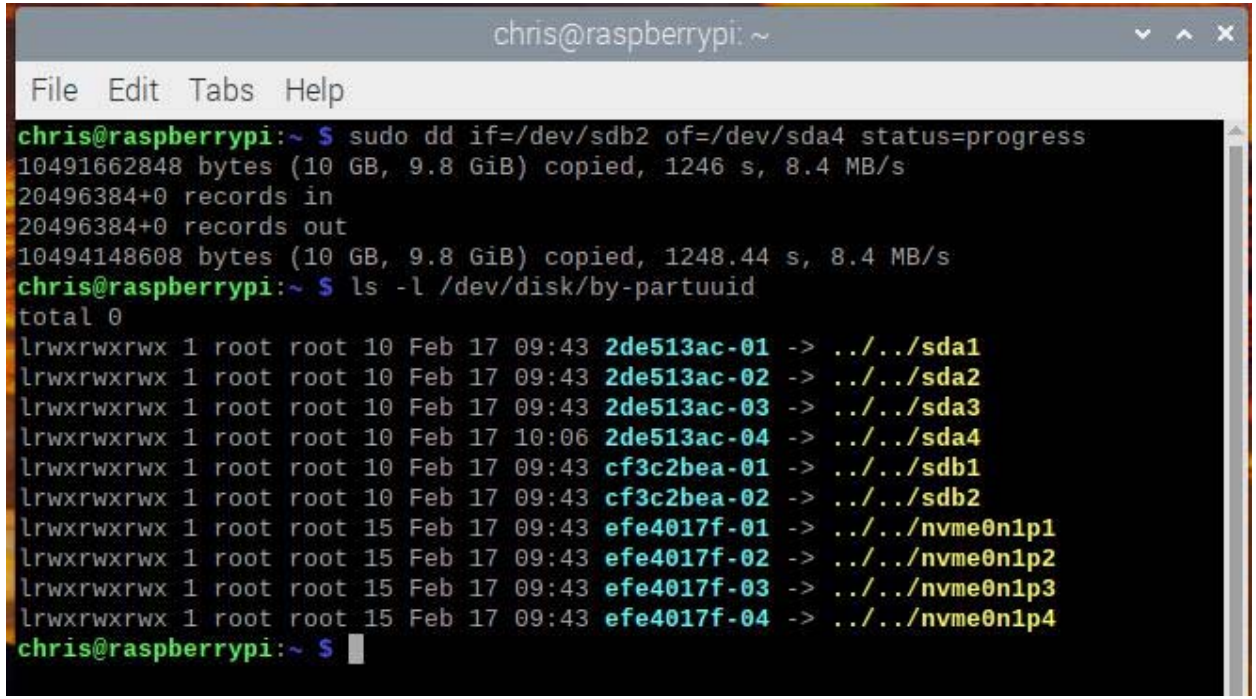
- Use small BASIC programme
  - Shuffle partition table from 1423 to 1234
- Reboot into Linux
  - Write partition table and **reboot Linux**
  - Format partition 1 as FAT32
  - Format partition 3 as FAT32
  - Format partition 4 as ext4
  - Copy Linux distro to partition 4 and **Check**

Formatting an NVMe drive



- Now all looks well on Linux – the partitions are in the right order
- Partitions 1 and 3 are now formatted FAT32 (and empty)
- Partition 4 is now formatted ext4 (and empty)

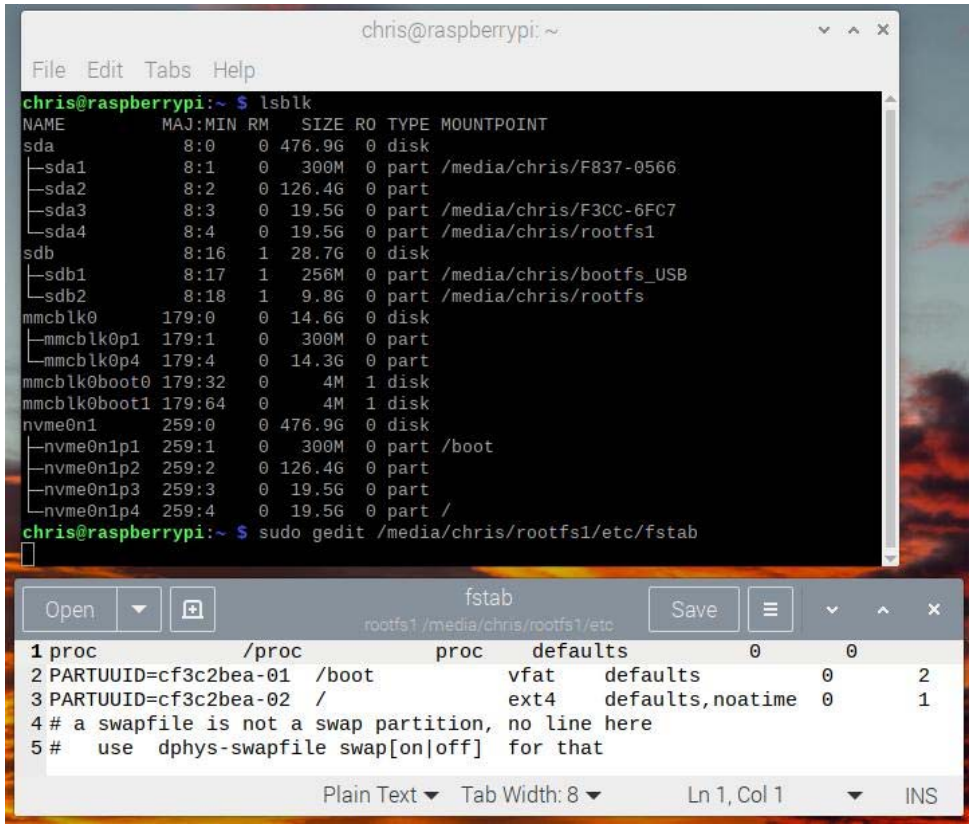
Formatting an NVMe drive



- Now we copy a standard Linux distro from a USB stick into the ext4 partition (it will have very little free space at the moment)
- We display the UUIDs (2de513ac and cf3c2bea) for use later
- Use the ‘Check’ command in GPartEd to resize ext4 to 20GB

Formatting an NVMe drive

- Linux has a ‘root’ filesystem (/) at a place identified by its UUID in CMDLINE.TXT on the eMMc storage (SDFS)
- It loads its kernel from the UUID specified in /etc/fstab
- The standard Linux distro (Bullseye) uses cf3c2bea so we edit that to 2de513ac (noting that root is now on partition 4 not 2)
- If we specify 2de513ac in Boot.Loader then Linux will boot from the NVMe drive



Formatting an NVMe drive

- Copy Linux distro
  - *dd if=/dev/sdb2 of=/dev/sda4 status=progress* (takes 15 min to copy 10GB)
  - Reboot Linux
  - Use *gparted* to ‘Check’ partition, extends free space to size of partition)
  - Use *ls –l /dev/disk* to show UUID
  - Edit */etc/fstab* (*sudo gedit*) to update UUID

Dual boot machine

- Linux can see both FAT partitions and use partition (1) for its boot drive
- Linux uses the ‘ext4’ partition for its files
- With the drive in a USB caddy, RISC OS can see FAT partitions and filecore
- Makes it easy to switch between OS’s
- Update CMDLINE.TXT with new UUID

Dual boot machine

- Easy to do
  - Requires bigger FAT partition on SD card
  - Awkward to reformat eMMc ‘on the fly’
  - Detailed instructions on !Store ‘How To ...’
  - Also www.svrsig.org/HowToNVMe.pdf
  - Clever bit is in CONFIG.TXT using GPIO5
  - Linux instructions sit in CMDLINE.TXT
- Simple push button selects boot OS

Dual boot machine

- With button pressed, Linux now boots
  - Using the NVMe drive in the USB caddy
  - for / and /boot (partitions 4 & 1)
  - If NVMe drive is plugged in to the IO board, Linux will use it natively
- With no button pressed, RISC OS boots
  - Using the eMMc as its selected boot drive
  - Can see three of the four partitions via USB

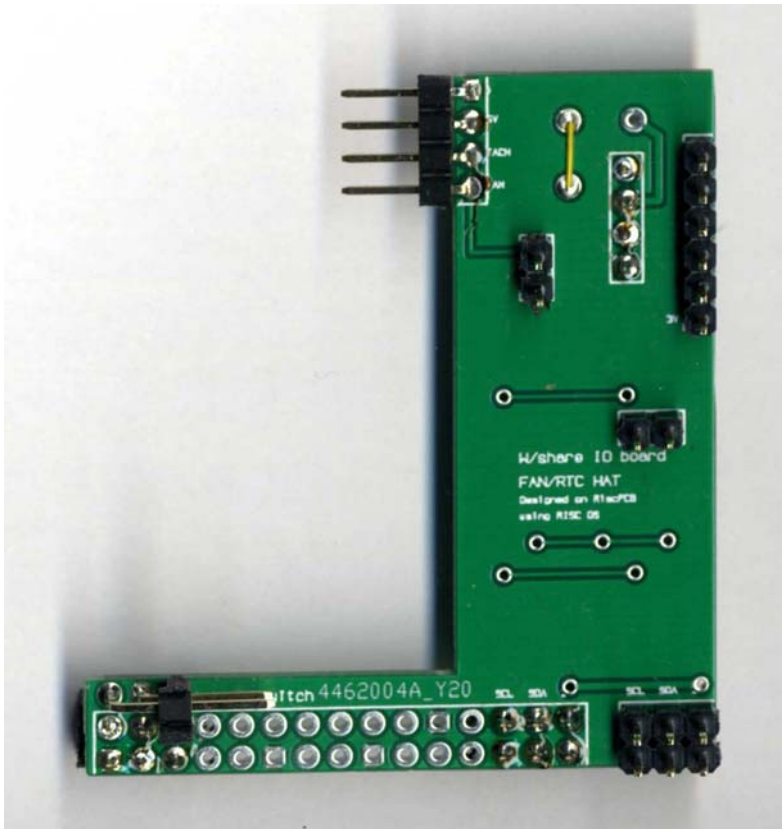
Build it yourself

- It is simple to put together, the most difficult bit is to get hold of a CM4!
- A 5V power supply (USB-C) is required
- The ‘eMMc’ version of CM4 is better:
  - Current RISC OS ROMs can’t see large SD cards on CM4 Lite – bug #611
  - A ‘dual boot’ solution is a good idea
  - NVMe is the future!

Build it yourself

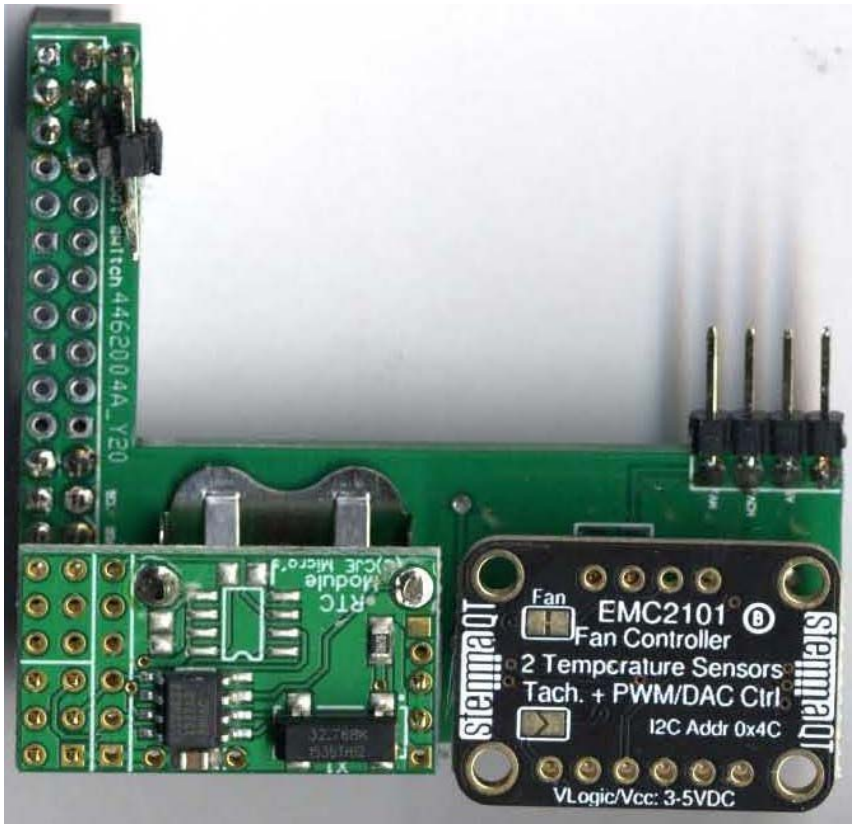
- Adding a fan/heatsink
  - Waveshare 5V fan bolted to CM4 board
  - Has integral heatsink and 4-pin lead
  - Simple to fit
  - DeskPi Mini already has a fan
- Need RTC and fan control
  - Purpose-built PCB to fit space available
  - Instructions www.svrsig.org/ClockFit.pdf

Build it yourself



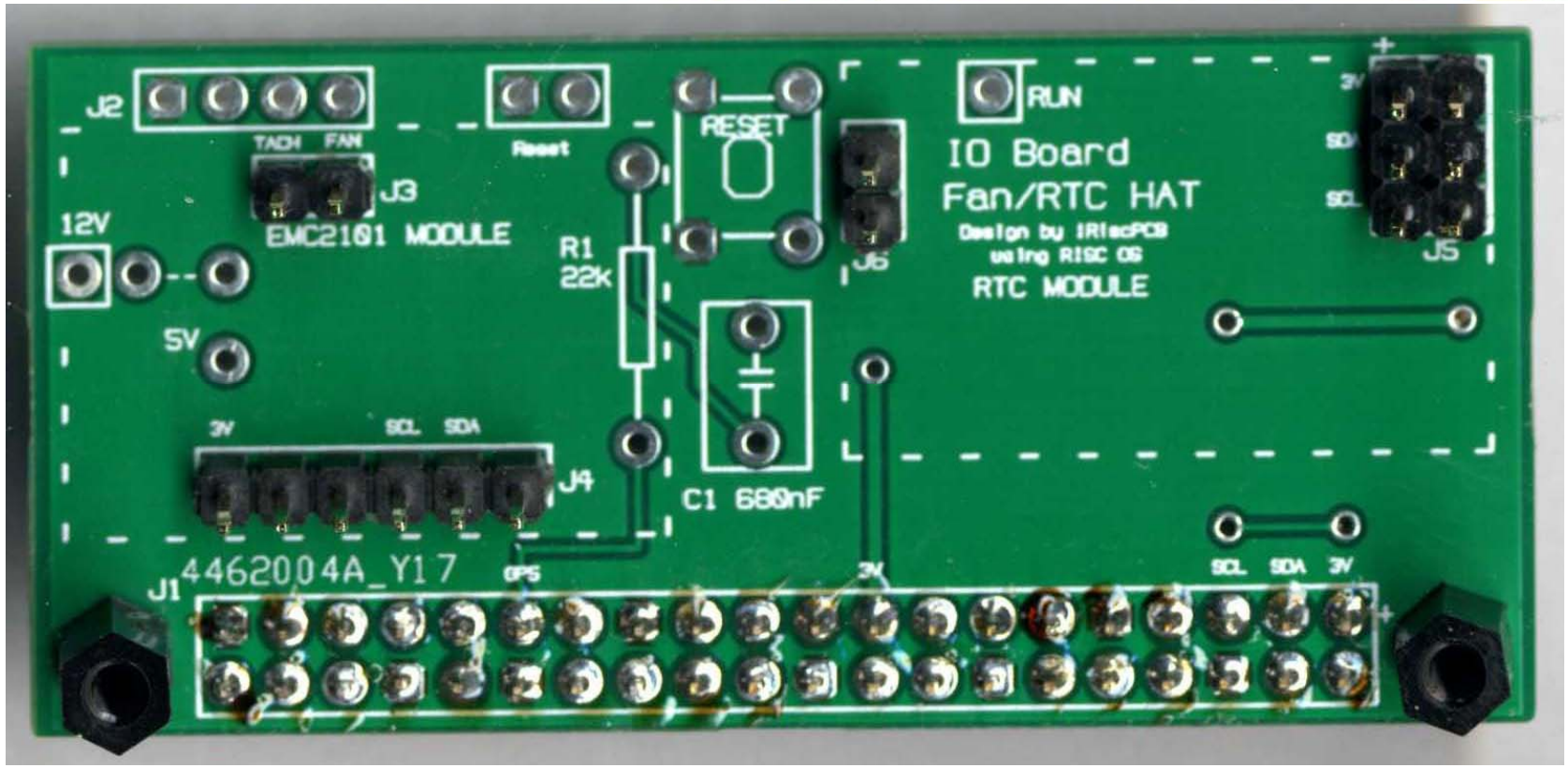
- PCB for Waveshare IO board

Build it yourself



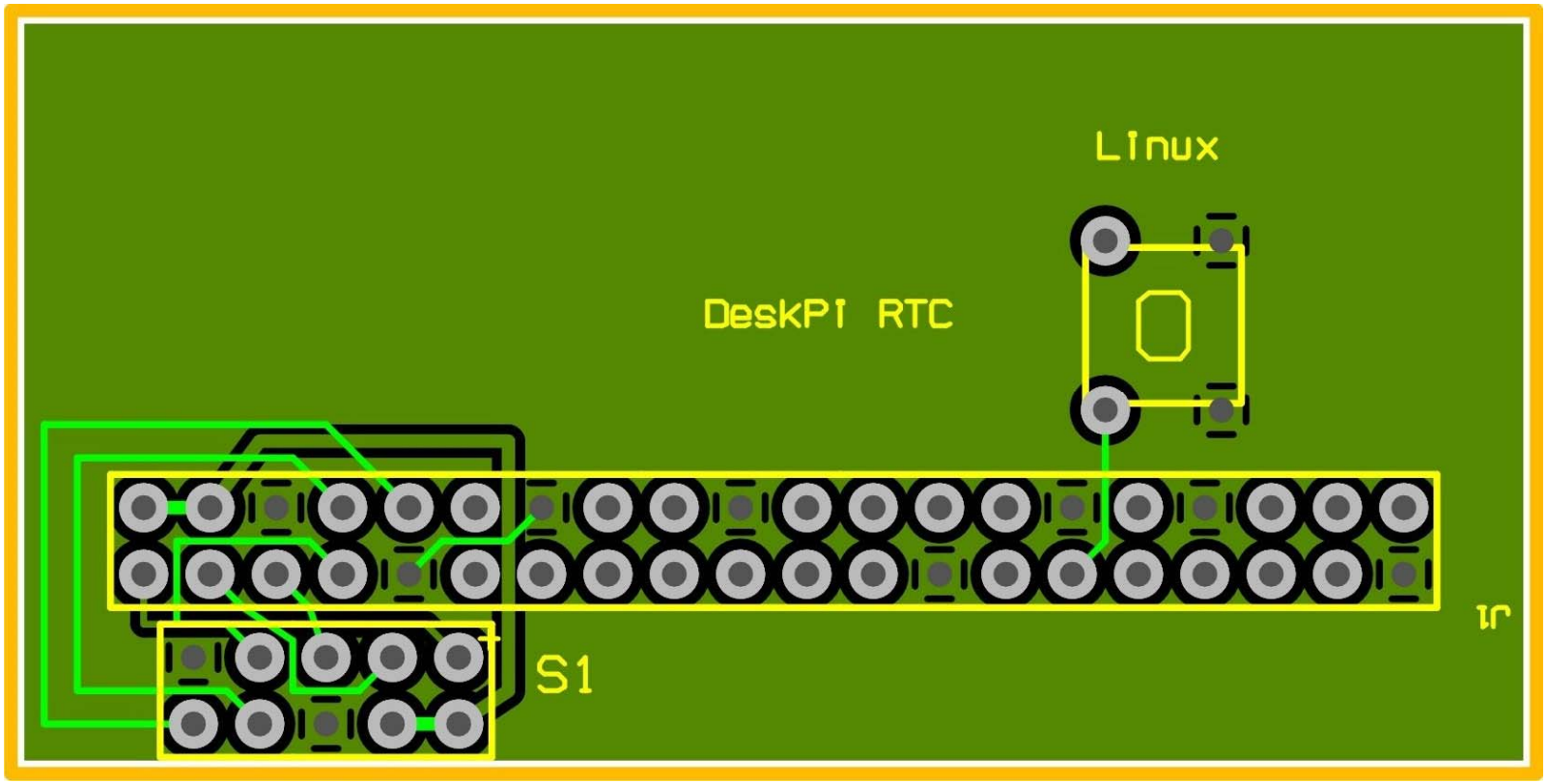
- PCB for Waveshare IO board

Build it yourself



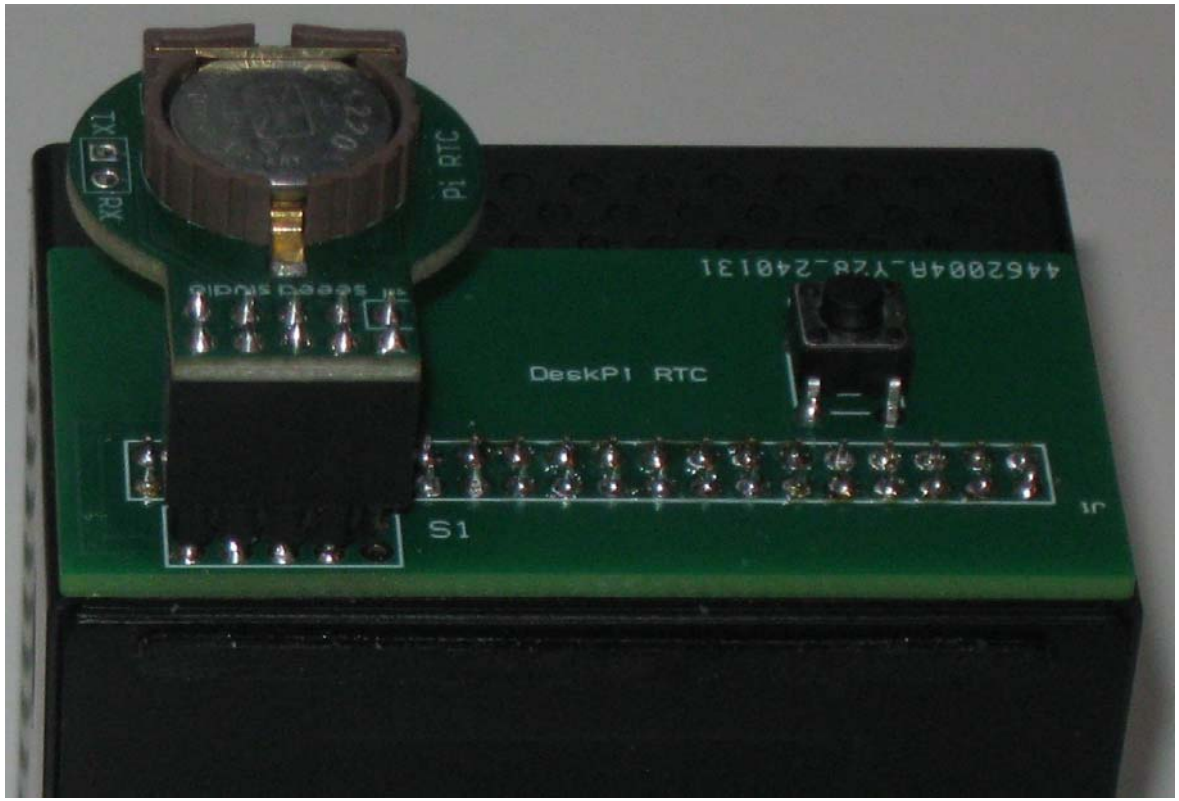
- PCB for Pi Foundation IO board

Build it yourself



- PCB for DeskPi Mini (PiRO Qube)

Build it yourself



- PCB for DeskPi Mini (PiRO Qube)

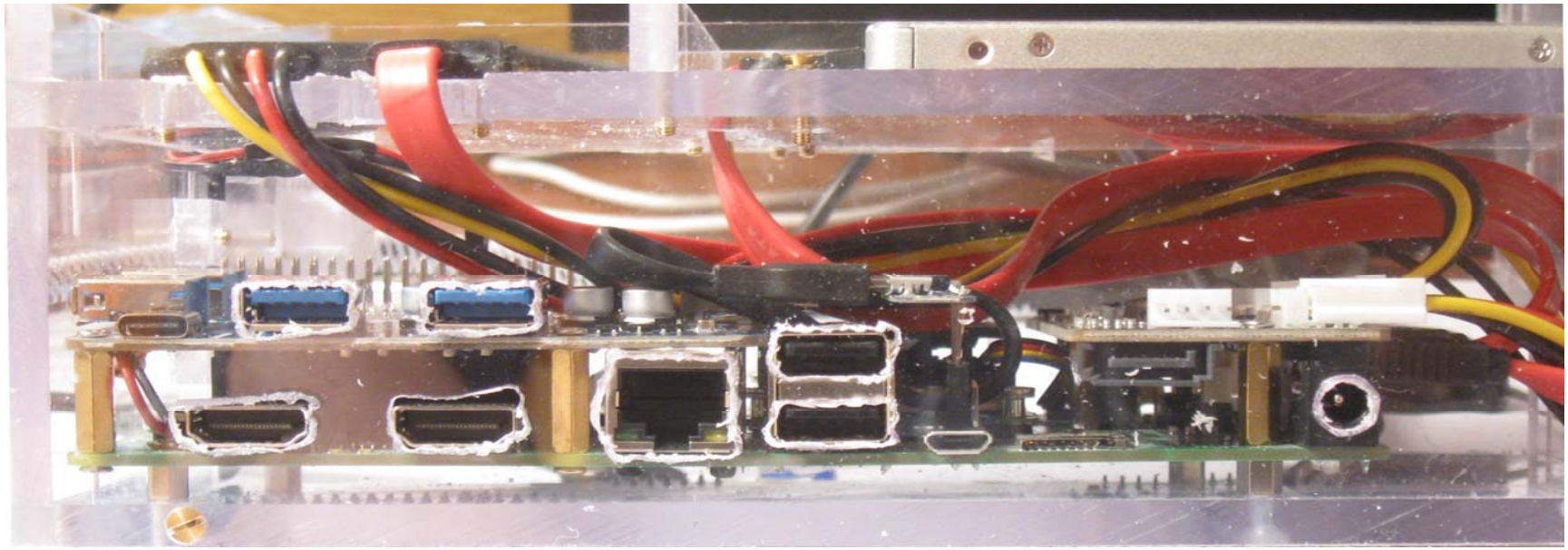
Build it yourself

- What does it add?
  - DeskPi – RTC and dual-boot switch
  - W/share – RTC, fan control boards + switch
  - Pi IO – RTC, fan control, boot/reset switch
- Simplest solution is PiRO Qube
  - Fit NVMe drive (easy to do)
  - Fit RTC using purpose-built PCB on back
  - All sockets on front

Build it yourself

- Case with Waveshare IO board and Fan

Build it yourself



- Pi Foundation IO board with SATA
  - 2 x HDMI; 4 x USB; 1 x Ethernet; 12V power
  - Drive bay for second SSD drive (cold plug)

Question & Answer

- I will do my best to answer – fire away!