Brilliant **beginner's builds for newcomers**

3D design with **Python and FreeCAD**

Run an **LLM AI chatbot on Raspberry Pi**

Raspberry Pi

Official Magazine
#161 | January 2026

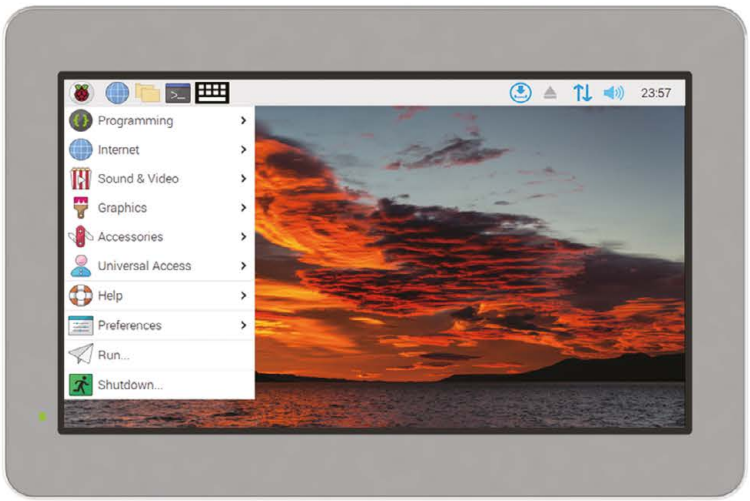
GET STARTED WITH RASPBERRY PI

Set up and make with your brand new tiny computer

01 £7.99
ISSN 2977-4403 (Print)
9 772977 440004

Industrial Raspberry Pi

ComfilePi



The ComfilePi is a touch panel PC designed with high-tolerant components and no moving parts for industrial applications. It features a water-resistant front panel, touchscreen, color LCD (available in various sizes), RS-232, RS-485, Ethernet, USB, I2C, SPI, digital IO, battery-backed RTC (real-time clock), and piezo buzzer.

Use the rear-panel 40-pin GPIO header to expand its features and capabilities with additional I/O boards. The ComfilePi is UL Listed and employs Raspberry Pi Compute Module.

Visit www.comfiletech.com

Welcome to Raspberry Pi Official Magazine

Editor
Lucy Hattersley
This month Lucy has been benchmarking large language models on Raspberry Pi. Makes a change from talking to herself.

rpimag.co

We all had to start somewhere. The initial run of just 10,000 Raspberry Pi computers was designed to boost the quality of applications to Cambridge computer science departments. It did that; and then some.

Raspberry Pi creates affordable and accessible computers used by engineers and educators. There's a direct line from learning computer science on Raspberry Pi to becoming an engineer.

Raspberry Pi has come a long way. We now see students who grew up with Raspberry Pi applying for jobs with us. The circle is complete. And we're all better off for it.

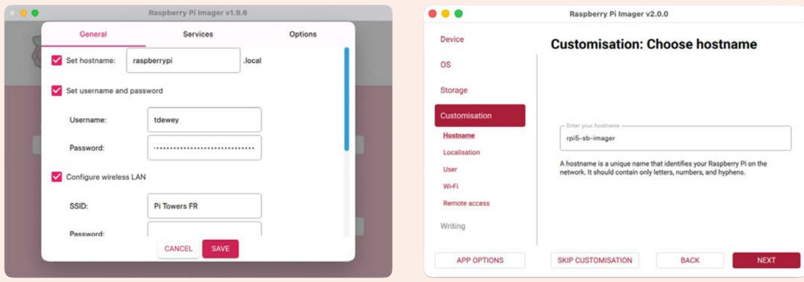
Now, more than ever, it's vital that people understand how computers work, and what they can really do with an edge computer, some computer code, and a really good idea.

This month we've got a guide to using Raspberry Pi for the first time. You'll discover how the hardware integrates with the software, and how GPIO pins are used to interface code with the real world. We all had to start somewhere. And you can start here.

Lucy Hattersley – Editor

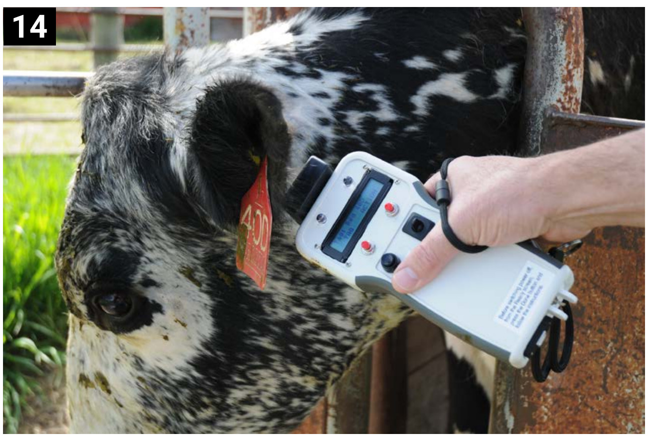
Contents

10



18

14

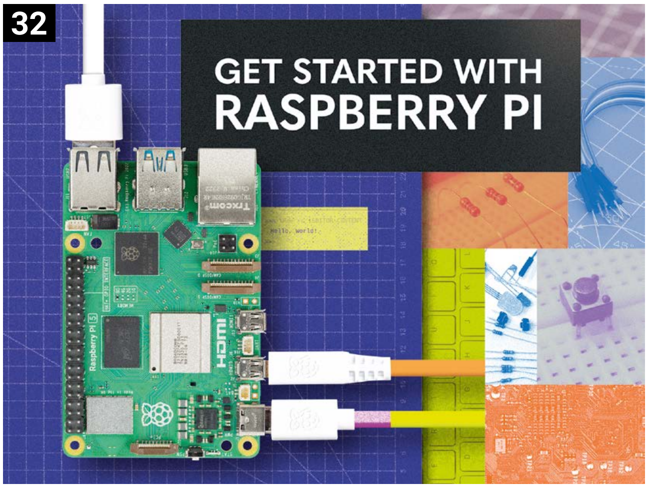


Raspberry Pi (ISSN No: 2977-4403, USPS No: 25-672) is published 12 times per year by Raspberry Pi Ltd, and distributed in the USA by Asendia USA, 701 Ashland Ave, Folcroft PA. Application to Mail at Periodicals Postage. Prices is pending at Philadelphia, PA and additional mailing offices. POSTMASTER: send address changes to Raspberry Pi, 701 Ashland Ave, Folcroft, PA. 19032.

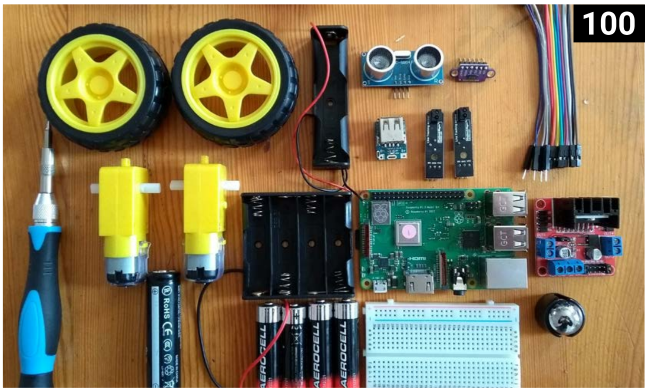
56

Feature

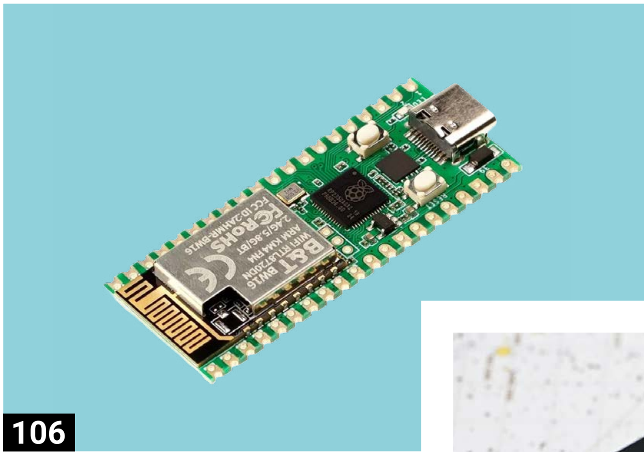
32



Feature



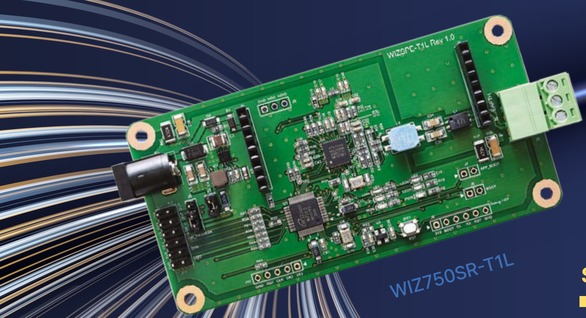
100



Reviews

Raspberry Pi Community

Competition



1 km Ethernet over a single pair

Simplify installations and lower TCO

Single Pair Ethernet → **1,000m**
100m Traditional Ethernet

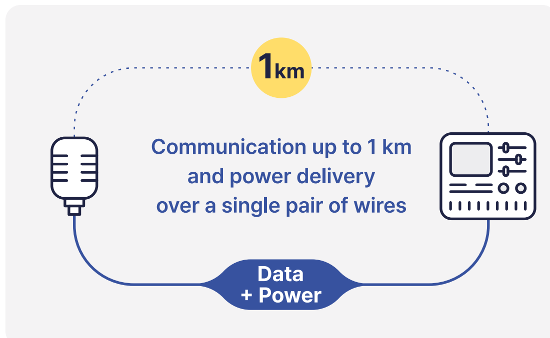
IEEE 802.3cg (10BASE-T1L) compliant
Data + power over one pair
Built for industrial environments

WIZ750SR-T1L Single Pair Ethernet Solution

Get Started

SPE / 10BASE-T1L at a Glance

- Standard: IEEE 802.3cg (10BASE-T1L)
- Cabling: Single-pair (2 conductors); PoDL options for combined power + data
- Reach/Speed: Up to 1 km at 10 Mbps (ideal for industrial IIoT)
- Stack: Integrates with standard IP networking (TCP/UDP, IPv4)



Why Better than RS-485 / Traditional Ethernet

Versus **RS-485**	• Delivers 10 Mbps communication up to 1 km distance • Native IP networking • Leverage standard IT tools and monitoring • TCP/IP, UDP, IPv4/v6 protocol support
Versus **10/100Base-T**	• Up to 1 km vs 100m distance • Single-pair cabling for lighter, lower-cost runs • Simpler installation with fewer connection points • Lower infrastructure costs

Why WIZ750SR-T1L

- Serial-to-Ethernet protocol converter: IP-enable your serial devices fast
- Industrial grade temp range, link/TCP status pins, separate Data/Debug UART
- Easy configuration: AT commands, PC Config Tool, Web UI, CLI

SPE Market Outlook

- SPE market projected to reach $5.78 billion by 2029
- 11.1% CAGR growth (2025-2029)
- Key driver for IIoT and Industry 4.0 connectivity
- Enabling technology for 27 billion IoT devices by 2025

Applications

Process/Factory Automation	Sensors, actuators, valve positioners, RTUs
Building /Infrastructure	BMS, energy /environmental monitoring, long-reach edge nodes
Legacy Digitalization	Reuse existing single-pair wiring for retrofit projects

Notes: Distance and performance depend on cable type, topology, and EMI/EMC environment. PoDL functionality requires compatible IEEE 802.3cg equipment.

Raspberry Pi Imager 2.0

Raspberry Pi introduces new Imager with a wizard interface. By **Lucy Hattersley**

Raspberry Pi has introduced Imager 2.0, the software program used to install operating systems onto storage drives to boot Raspberry Pi.

Imager 2.0 is a "complete reimagining of the application that's been brewing in our development pot for the past year," says Tom Dewey, Software Engineering Manager at Raspberry Pi. "It brings a new wizard interface, the opportunity to preconfigure Raspberry Pi Connect, and improved accessibility for screen readers and other assistive technologies."

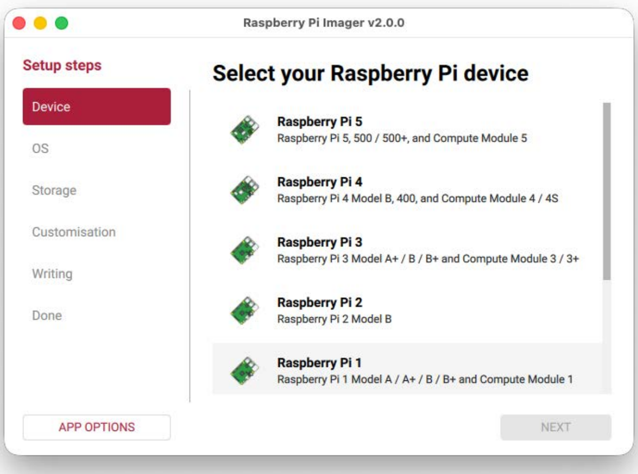
Raspberry Pi Imager was released in March 2020 with the aim of making getting started with Raspberry Pi much simpler than the workflow of the time. Since then, the team has been responding to feedback and gradually adding new features. "As the application progressed, so did the strain on the interface," explains Dewey, "Eventually, the size of the customisation form grew so large that it became unwieldy." The new update responds to the challenge.

What is new

Imager 2.0 has a "radical UI change" that surfaces many features from the program with a new wizard and sidebar design.

One of the most popular aspects of Raspberry Pi Imager is the OS customisation that lets you determine the username and hostname, set the networking / Wi-Fi options, and set up interface options. "OS customisation is one of the most popular aspects of Raspberry Pi Imager," says Dewey, "but we'd left it hidden in a separate window. Why hide the good stuff?"

▼ Raspberry Pi Imager 2.0



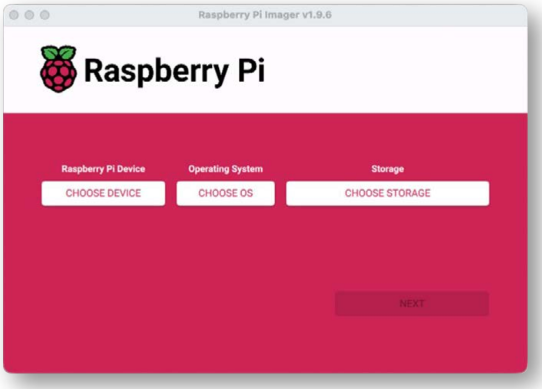
Raspberry Pi Connect also joins the customisation options. "Following the launch of Raspberry Pi Connect, an obvious hope was to enable Connect sign-in during imaging, rather than after booting," explains Dewey.

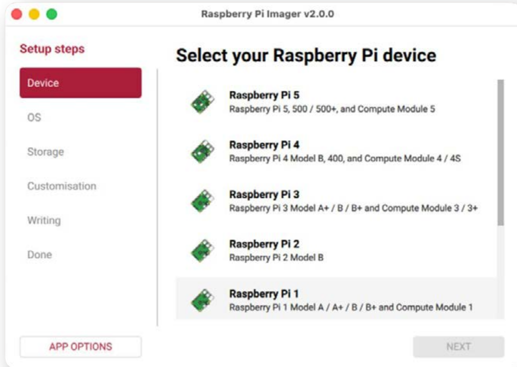
A wizard appears

The most obvious change is the new step-by-step wizard interface. Says Dewey: "In order to make OS customisation a first-class citizen, we needed to bring it into the main interface, so now you'll progress through a series of clearly defined stages." These are as follows:

1. **Device.** Select your Raspberry Pi device
2. **OS.** Choose your operating system
3. **Storage.** Pick your storage device
4. **Customisation.** Configure your system (hostname, location, user account, wireless LAN, remote access, Raspberry Pi Connect, and interface options)
5. **Writing.** Write your image
6. **Done**

It brings a new wizard interface





▶ Raspberry Pi Imager old (left) and the new interface (right)

Each step is displayed in the full window, allowing more room for "helpful descriptions, validation feedback, and relevant links. It's a more spacious, less crowded experience all round."

Accessible by design, not by chance

"Raspberry Pi Imager is a much-loved, widely used application that represents, for many people, their very first interaction with Raspberry Pi," says Dewey. "Every user deserves the best possible experience."

The controls in Imager 2.0 include accessibility labelling for screen readers and other assistive technology. The application is fully navigable via keyboard. Dewey explains that the design team "paid careful attention to colour contrast throughout, ensuring text remains readable against its background in all situations."

"The new colour scheme, built around Raspberry Red, is designed to make the UI easier to read (and to draw your attention to sensible choices). We use white space generously to establish clear boundaries between controls, making the interface easier to parse at a glance."

Raspberry Pi Imager 2.0 is available to download for Mac, Windows, and Linux machines from the Raspberry Pi website: **rpimag.co/software**.

To install on Raspberry Pi OS, type:

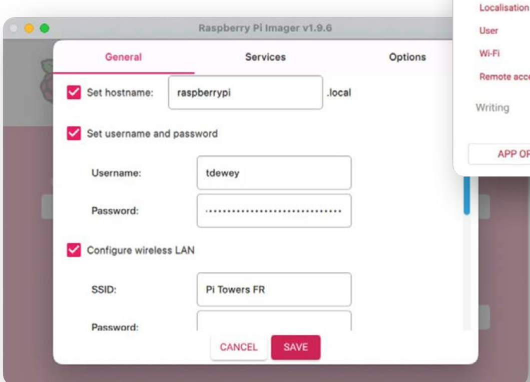
```
$ sudo apt install rpi-imager
```

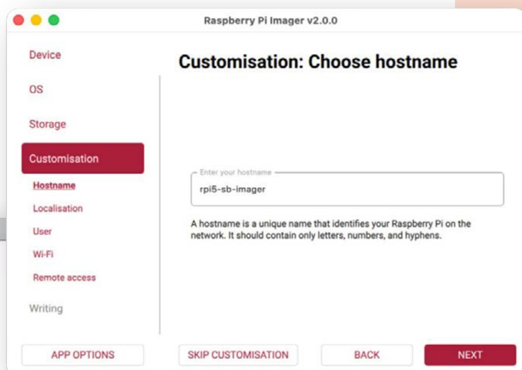
…into a terminal window.

Raspberry Pi Connect: ready to go

One popular customisation the original Raspberry Pi Imager offered was the ability to configure – at the point of writing your SD card – your remote access credentials, giving you the chance to preprogram your SSH public keys.

What if, however, you wanted to use the much simpler Raspberry Pi Connect system for remote access to your Raspberry Pi? With Imager 2.0, you can do just that. Simply authenticate during the imaging process, and when your Raspberry Pi boots for the first time, it'll already be connected to your Raspberry Pi Connect account, ready to offer screen sharing or remote shell access from the start.





◀ The new customisation options

Filament Scale

Discover whether or not you have enough filament for a 3D print. By **David Crookes**

Maker

Chris Forde

Chris is an engineer and athlete who makes and creates in a variety of areas, mechanical, electronics, microcontrollers, software, 3D printing, and handicrafts.

rpimag.co/filascale

If you're going to use a 3D printer, then you need to be sure that you have enough filament for the job. "Knowing if there is sufficient filament will tell you if you need a standby reel, a new reel, or whether the current reel will do the job," says maker Chris Forde. One way of determining this is to weigh the filament remaining on the spool before comparing it to the weight estimate provided by your slicing software. Understanding this, Chris came up with a solution.

"Normally when weighing filament, people use a separate scale and a calculator, but I thought it would be more convenient to integrate a filament scale into the printer," he said. The idea was to replace the printer's existing spool holder with one that contains a beam load cell. A beam load cell is a cantilever that measures applied force and converts it into an electrical signal which can then be interpreted as weight.

"I identified a beam load cell with a maximum capacity of 5kg, although the filament reels to be used are 1kg, allowing a degree of overload protection," Chris says. "My chosen load cell also came with a signal conditioning amplifier which can be interfaced to a microcontroller." It enabled him to combine the specialised transducer with an RP2040 microcontroller board, choosing one with an integrated LCD. "It allowed me to reduce the footprint, cost, and build time," he adds.

Scaling up

With the beam load cell identified, Chris designed a 3D-printed spool holder for an Elegoo Neptune 4 Pro 3D printer. "The spool holder and the case were designed using BlocksCAD, with the screw thread designed using Tinkercad," he said. Once all the elements were printed, he was able to assemble the scale arm which also incorporated an HX711 ADC load cell conditioning amplifier, designed for weighing scales.

Chris also custom-made a PCB onto which he could mount an RP2040 microcontroller board. "I opted for a PCB, designed using EAGLE to mount the components to improve repeatability, reproducibility, and create a robust project," he explains.

With the main unit assembled, he turned his attention to the software, which he wrote in MicroPython v1.15 using the Thonny IDE.

01. The Beam Load Cell is suitable for low to medium loads

02. The device is controlled by switches and there are reset and boot buttons too

> *Normally when weighing filament, people use a separate scale and a calculator*

Weighty issues

To work, the software needs a bunch of information. First, it requires the calibration value. "Calibration is important to ensure the scale displays the correct weight, and this is accomplished using a known weight," Chris explains. The software also needs the baseline weight as well as information about the filament material: its density (g/cm^3), diameter (millimetres), and the weight of the empty reel (grams)

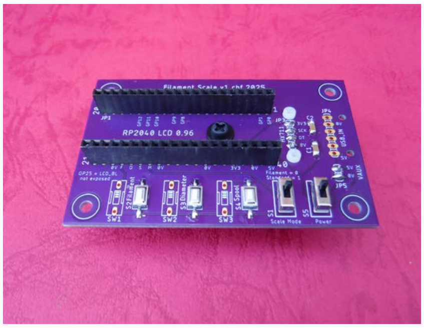
"A small list of this information is held internally and within text files, either of which may be edited to extend or amend the lists," Chris says. "The user needs to select the correct details … and, with all this information, weight and filament length can be calculated."

The LCD instructs the user to perform any necessary tasks such as removing the spool to capture the unladen weight. It also displays the required result, including the length in metres and weight in grams. Given the low cost of the parts, it's already proven to be an efficient, money- and waste-saving device. With full instructions and printable files on Hackster.io, it's a project that many makers are sure to find useful.

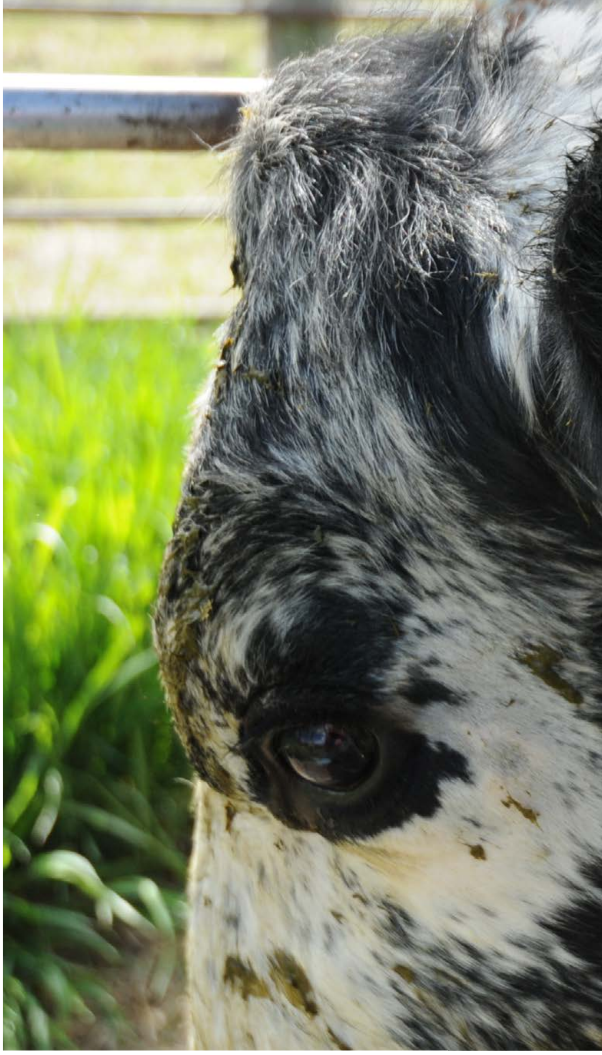
Quick FACTS

- The project measures the weight of filament loaded on a printer
- It replaces a 3D printer's own spool arm
- Switches are used to change the filament type and diameter
- Power is provided by a separate USB board
- It cost less than £50 to create the project

▲ The assembled main unit includes a clear window to protect the RP2040 microcontroller



▲ The PCB is fitted with an HX711 ADC and headers for an RP2040 microcontroller board



Flokk herd management

Trade tariffs and traceability are two good spurs to improving Canadian cattle tracking using Raspberry Pi handsets, learns **Rosie Hattersley**

Maker

Mark Olson

Mark is an enterprise IT veteran and founded Flokk to bring digital record-keeping to the Canadian ranches where he grew up.

flokk.ca

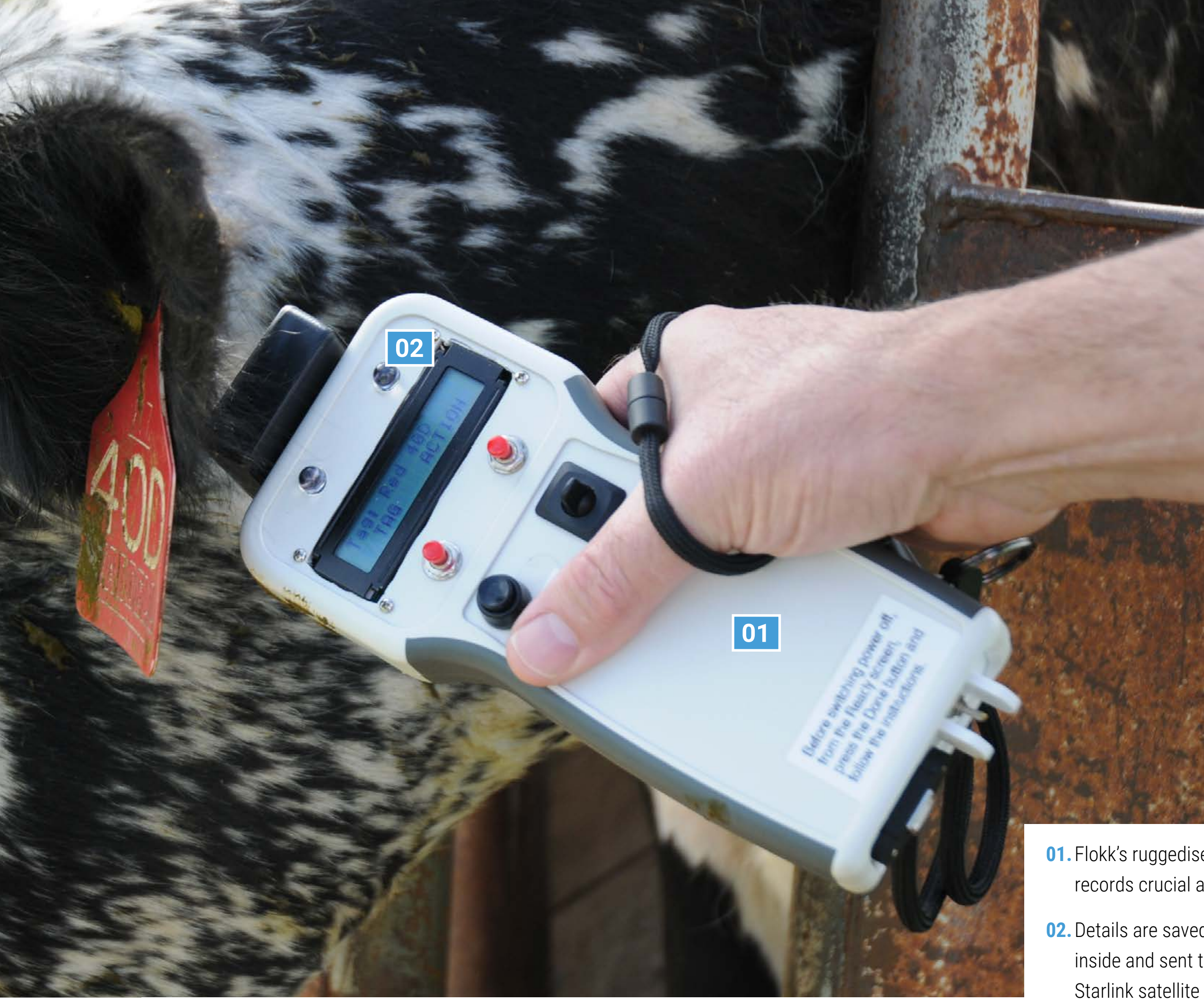
Keeping tabs on cattle ranging over enormous ranches in rural Canada is a serious challenge. "Ranching operates outdoors, in all weather, in remote locations where there is no power or data connectivity," explains Flokk founder and CEO Mark Olson. Record-keeping has traditionally been done using pocket notebooks, which of course are easily lost, damaged or trampled underfoot while the rancher struggles to check on each animal's health and well-being. It's the sort of task that hapless TV vets might have nightmares about. Nonetheless, detailed and fully up-to-date information about each cow is mandatory in the highly regulated world of animal husbandry.

Computer scientist and farmer Mark Olson applied his knowledge of data collection to address the issue, coming up with a ruggedised scanner that has Raspberry Pi Zero W at its heart and which automates much of the record collection process.

Rural reach

Mark grew up on a farm in Alberta, Canada and, after briefly exploring the computerisation of agriculture when he graduated in the mid-1980s, went on to work in enterprise IT and management. Open-source software has always interested Mark, not least because his early forays into building home servers were of the DIY, fish-the-hardware-out-of-the-dumpster variety. Linux was the only platform he could afford and that would run on the hardware he'd sourced. An interest in Raspberry Pi is therefore no surprise. It's "a logical extension of my interest and aptitudes in open source," says Mark, who uses Debian Linux for both Flokk's website and online services.

This contrasts with "digital solutions for ranching [that] currently attempt to use hardware built for offices, not ranches. Consumer hardware is expensive and delicate, and unsuitable for the harsh environment of ranches."



01. Flokk's ruggedised handheld scanner records crucial animal data

02. Details are saved to a Raspberry Pi Zero W inside and sent to a central server via Starlink satellite broadband

Traceability is not only desirable; it's legally mandated. All Canadian cattle are fitted with a tag with an RFID code so that if an issue arises with the animal, it can be traced back and any other animals that might have been impacted by the issue located and investigated.

Flokk digitises data collection and record keeping for cow/calf ranching, a $5 billion industry in Canada and a trillion dollar industry globally, moving it from the small paper pocket notebooks ranchers use today to a digital solution.

Mark's LinkedIn article about the importance of such record-keeping to the Canadian agricultural economy (**rpimag.co/canadafood**) offers some important insights. He talks of the clear need for beef traceability and for Canada to secure its future food security via agricultural exports, keeping data (saleability, individual animal traceability, pregnancy, and other health stats) on-device and freeing up the rancher's time and providing autonomy – not just laboriously filling out paperwork but capturing it via in-field scans and having it immediately be digitised.

He points out that Canada spends around 40% of its agricultural services budget on inspection and control and says this could be massively reduced using technology to tag and monitor agricultural assets. Starlink provides "a ready answer for rural broadband and wireless connectivity" and he rates it above 5G connectivity, "which is both far more expensive to install and requires time to roll out along with ongoing maintenance".

> *His early forays into building home servers were of the DIY, fished-thehardware-out-of-the-dumpster variety*

▼ RFID tags are read by the Flokk handheld scanner and details uploaded to a central database

No platform other than Debian, the flavour of Linux underlying Raspberry Pi OS, lets you take the same software, and skill set, and apply it all the way from battery-powered handheld, through SBC desktop, all the way through to multi-node server.

▲ The ruggedised handset is ideal for use by ranchers on the move

Hard-headed approach

Flokk chose Raspberry Pi for its affordability, high capability, and compatibility with other open-source products. "Raspberry Pi Zero W uses a full-function Broadcom system on a chip, not a microcontroller, and Linux as the OS, [so] Flokk can leverage standard open-source tools. We can add functionality in days and be confident our platform will support whatever software or functionality we will need to deliver in future."

It was important that the cost of the handheld scanners wasn't a barrier to ranchers adopting them. Flokk will make its money from subscription, rather than the hardware. The end of life for a Flokk handheld will most likely not be the result of wear and tear or obsolescence –it will get dropped into a pen of cattle and stomped on. Flokk has to be able to rapidly, and affordably, recover a customer's data from offsite backup and ship them a new Flokk ready to use.

Mark wrote all the code himself, devoting more than 1000 hours to both code and platform development. When they began the process, Raspbian/ Raspberry Pi OS would not run on the handheld scanner, so they used Tiny Core Linux instead. Future iterations may use this alongside Raspberry Pi OS.

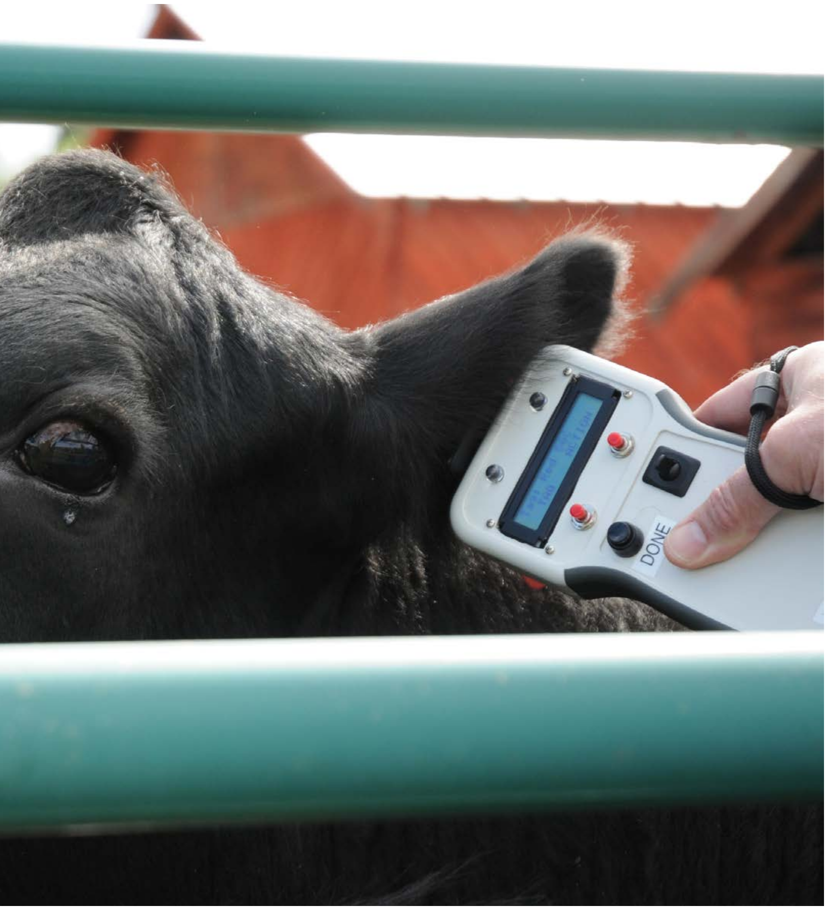
Mark says Raspberry Pi Zero's capabilities were "a perfect match; right price, power efficient, and exactly the I/O we needed". Flokk is currently preparing its next iteration of hardware, migrating to Raspberry Pi Zero 2 W, upgrading the handheld scanner's display, adding a GPS receiver and, perhaps, a camera.

For now, Mark is busy garnering investor interest in Flokk so it can be rolled out at scale across Canada's ranches. Demonstrating its real-world use is imperative. "The Flokk I proudly use for investor presentations has brown stains on it; and the source of those brown stains is exactly what you think it is."

▶ Blockchain protects records from interception and exploitation once each animal's details have been logged

Quick FACTS

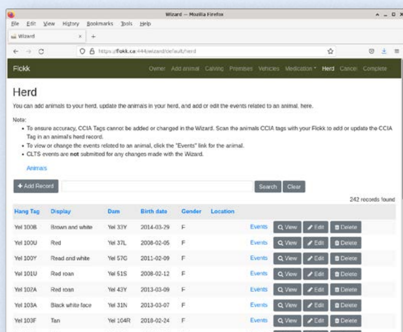
- Every livestock animal in Canada should have an in-ear RFID tag
- Most ranchers have no means of reading the CCIA tags, so keep paper records
- Missing and inaccurate record-keeping impacts traceability efforts
- Dedicated digital readers are expensive and have no other functions
- Flokk works with record management software, making compliance easier



Stand by to scan

1. Flokk runs Tiny Core Linux code on Raspberry Pi Zero W alongside an RFID reader inside a Hammond Hand-Held T ruggedised case.

2. Flokk's record management platform makes keeping detailed logs about each animal straightforward, helping ranchers comply with husbandry regulations and log medications, health, and welfare details.



3. Data is stored locally on an SD card and records can be automatically uploaded to the online management dashboard over the Starlink broadband network or when the rancher is back at base.

Amstrad PPC 640 Cyberdeck

A broken retro portable PC has been given a bright future. By **David Crookes**

Maker

Mikey Damager

Mikey is a UK-based artist, technologist, and technowomble. He builds interactive narratives, musical interfaces and AV installations from code and old bits of hardware he finds lying around.

rpimag.co/ppc640deck

When faced with a broken Amstrad PPC 640, Mikey Damager had two choices: to return the machine to its former glory or tear it apart and rebuild it using modern parts. He decided to do the latter, turning what was Amstrad's first portable IBM PC compatible computer, released in 1987, into a cool-looking cyberdeck powered by Raspberry Pi 4. It produced a machine capable of running an interactive fiction project for Mikey's master's degree.

"I wanted to explore AI and machine learning to see if I could incorporate some of the tools into a creative workflow in a way that felt somewhat critical and not too detached," he says. "I ended up making something that uses LLMs to explore what it's like to be existentially brutalised by an evil corporation which has hidden access to pseudo-sentient technology. It's supposed to be somewhat tongue-in-cheek and satirical."

▶ The original innards have been replaced, but Mikey has retained the motherboard in case someone wants it

Key to success

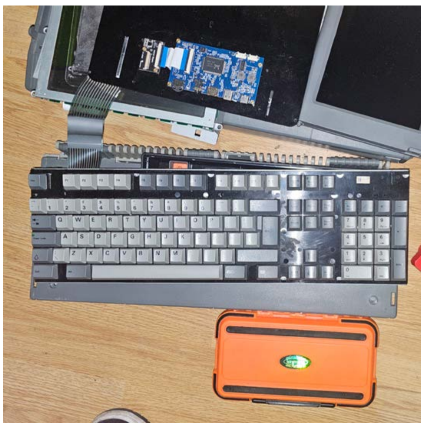
Mikey had considered repairing the original machine, yet doing so would have entailed a huge amount of work. "The screen was completely smashed and the case was pretty dinged-up. A few bits of plastic had also snapped off."

Deciding an upgrade was preferable, he opened the case, and detached the screen and keyboard. "The chassis is basically a plastic suitcase with a screen and a keyboard attached," Mikey says. "Once inside with a screwdriver, the motherboard practically leapt into my arms. I was left with a big empty box that I could fill up with new gadgets."

The screen was replaced with an eight-inch 4:3 LCD panel connected to an HDMI driver board. Mikey used screen repair tape to fix this panel to a sheet of 2mm acrylic for protection and he connected the display to Raspberry Pi before wiring the PPC 640's original LEDs and switches to the new hardware, allowing the system to be easily powered up.

01. The project is faithful to Amstrad's original cool-looking case design

02. All of the components including the battery fit neatly inside



◀ Mikey says the PPC 640's plastic had become brittle over time, so care was needed when working on the case

Quick FACTS

- The project uses a case from an Amstrad PPC 640
- There's plenty of space, offering lots of design choices
- Finding suitably sized screens and keyboards is easy
- Raspberry Pi is being used in kiosk mode
- The computer has been coloured using car bumper spray paint

> *The screen was completely smashed and the case was pretty dinged-up*

Replacing the keyboard proved easy too. "One of the best things about the PPC is that it has a full-size keyboard, which means that if you just remove the little plastic tangs where the Fn key should be, you can grab pretty much any random full-size mechanical keyboard, de-glove it, and there's a good chance that it will fit almost perfectly."

Black is the new grey

The floppy disk drive was also retained even though it was disconnected. "I wired the write-protect switch from the floppy drive to an Arduino to turn on a small screen when a disk is inserted, but it doesn't read any data," Mikey says.

The biggest challenge was the development of the front end. "I needed what's running on screen to look and feel suitably cyberdecky," he explains. "The piece is built around a React app that's styled to look like an OS. Raspberry Pi is running FullPageOS, so it's just a kiosk that boots straight in to a web page over Wi-Fi, with the back end running either from my laptop or in the cloud."

It means Raspberry Pi is just handling the display and user input, but it ensures the cyberdeck does the intended job. As a finishing touch, Mikey sprayed the grey computer black and added colour to some of the keys, but he'd like to go further. "There are a few cosmetic improvements I'd like to make, such as new badges and branding," he says. "I'm also planning a better power solution because currently it's running from power banks that I've hidden inside and I'd like something a bit more elegant."

SUBSCRIBE TODAY FOR JUST £10

Get **3 issues** + **FREE** Pico 2 W

SUBSCRIBER BENEFITS

Free delivery
Get it fast and for free

Exclusive offers
Great gifts, offers, and discounts

Great savings
Save up to 37% compared to stores

SUBSCRIBE FOR £10

Free Pico 2 / Pico 2 W

3 issues of Raspberry Pi Official Magazine

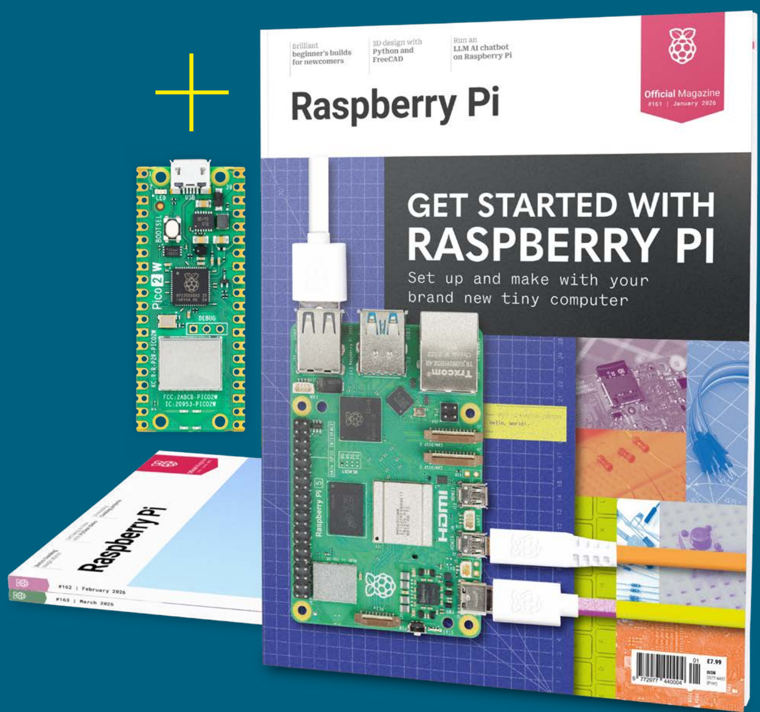
£10 (UK only)

SUBSCRIBE FOR 6 MONTHS

Free Pico 2 / Pico 2 W

6 issues of Raspberry Pi Official Magazine

£30 (UK)	**$43** (USA)
€43 (EU)	**£45** (Rest of World)



Subscribe by phone: 01293 312193

Subscribe online: rpimag.co/subscribe

Email: **raspberrypi@subscriptionhelpline.co.uk**

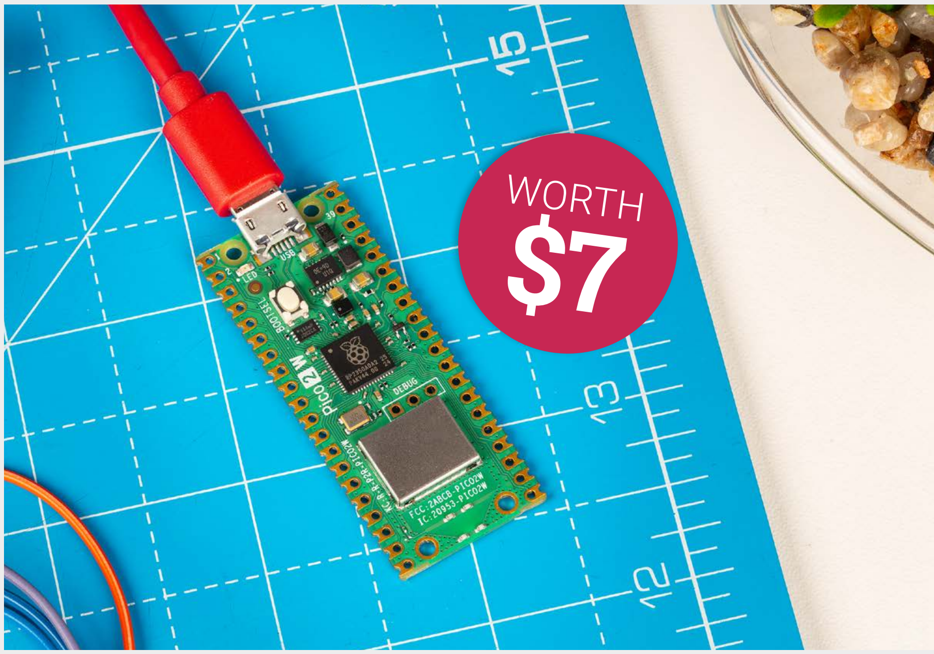
Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A choice of free Raspberry Pi Pico 2 W or Pico 2 is included with all subscriptions.

SUBSCRIBE TODAY AND GET A

FREE Raspberry Pi Pico 2 W (or Pico 2)

Subscribe in print today and get a FREE development board

- A brand new RP2350-based Raspberry Pi Pico 2 W / Pico 2 development board
- Learn to code with electronics and build your own projects
- Make your own home automation projects, handheld consoles, tiny robots, and much, much more



Free Pico 2 or Pico 2 W. Accessories not included. This is a limited offer. Not included with renewals. Offer subject to change or withdrawal at any time.

Buy now: rpimag.co/subscribe

SUBSCRIBE on app stores

From **£2.29**

Available on the App Store

GET IT ON Google Play

Moon and tide clock

By PJ Dines

rpimag.co/moon-and-tide-clock

Want a clock that "can't tell time but instead uses the current time to calculate the lunar phase and an incredibly specific oceanographic prediction for a tiny subset of the UK coastline?" asks PJ Dines. "Then have I got the clock for you."

After all, time is just an illusion that we've mapped on to the natural phenomena around us: the phases of the moon, the length of the day, the changing of the seasons: these were all here before someone decided to divide the day into 24 so-called 'hours', but only use numbers that only go up to 12 to describe them.

One person who's transcended time, and possibly space, PJ is the creator of the Moon and Tide Clock. Built into a repurposed brass carriage clock, this e-paper device is programmed using a Raspberry Pi Pico W, powered by a 3.7V LiPo battery, and kept accurate when offline by a real-time clock module designed for Pico. There's also a lovely toggle switch to turn the device on and off with a satisfying 'thunk', though the maker admits that with the power-saving options available in Pico, this wasn't really necessary.

▶ A thoughtful retirement gift for the surfer in your life


23/08/2024
High
03:47
6.1

15:56
6.1
Low
09:52
0.8

22:26
0.3

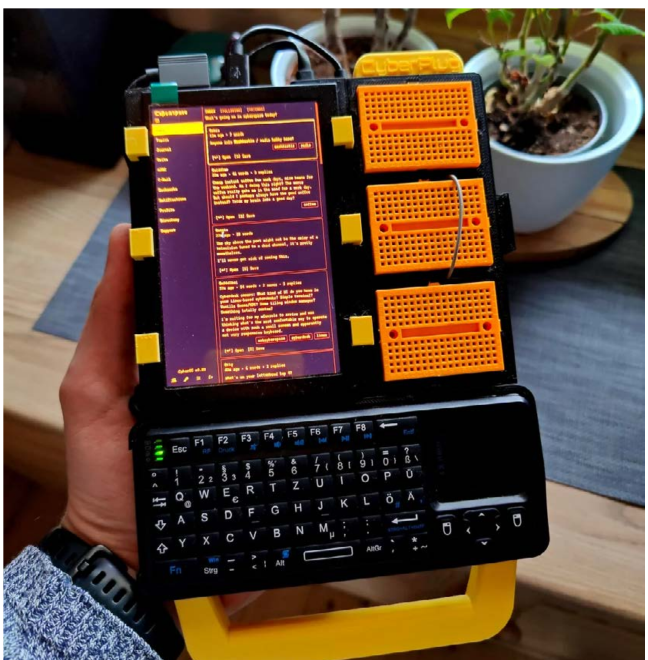
CyberPlug

By PickentCode

rpimag.co/cyberplug

Many people make their own cyberdecks – portable, compact devices for hacking in the post-apocalyptic wastelands – because they're cool. We all like to customise our working environments, so why not go the whole hog and create your own computer? While we've seen plenty of devices built into ruggedised cases, we don't think we've seen a cyberdeck that incorporated prototyping breadboards before. At least, not until now!

This CyberPlug by PickentCode is a smart handheld device for hardware hackers, featuring three mini breadboards, a Raspberry Pi, a tiny Rii X1 keyboard, 5-inch monitor, and a 22.5W, 10,000mAh Baseus Adaman power bank. It's also packing a CC1101 wireless module (that's the antenna you can see in the picture). Other than the breadboards and the contrasting colour scheme, which tickled our fancy, is that it's running Cyberspace: a social media network whose website describes it as "a quiet corner of the internet where you can think, write, read and connect. Like how the internet was supposed to be." Maybe it could catch on?



▶ You can plug this cyberdeck into a docking station when you're not out exploring desolate ruins

BMO Console

By Confident-Dare-9425

rpimag.co/bmo-console

If you're a fan of the animated TV series *Adventure Time*, you'll recognise this BMO console, made with (among other things) 3D printing, a custom-designed PCB, and a Raspberry Pi 5. If you're not a fan, bear with us. *Adventure Time* is a cartoon, broadcast on the Cartoon Network, and apparently it's quite good. More than that, it gives kids something to watch that adults can also get something out of, which is a rare treat (and the reason we love the kids' cartoon *Bluey*).

BMO (pronounced Bee-Mo; short for Be More) is a sentient games console, and faithful friend of Finn and Jake, the main characters. To the initiated, it sounds a bit like the Talkie Toaster from *Red Dwarf* in the episode where it received exceptional computing powers, but that's not important: what is important is that Reddit user Confident-Dare-9425 and their daughter love it, and so they made her one of her own.

The console runs the RetroArch front end for emulators – currently playing Sega Mega Drive, Sony PSP, and Nintendo Entertainment System games – as well as regular desktop applications, and it also works with a TV in dual-display mode. The body is 3D printed, with all the attachments coming in the form of M2.5 bolts and brass inserts. The screen is an 800×600 device from AliExpress, and at its heart there's a Raspberry Pi 5 with 8GB of memory.

▼ According to the maker, this machine occasionally passes wind, which is just about the funniest thing ever


MAIN
100%
NES
PSP
SEGA
SYSTEM
Plex

XM Radio

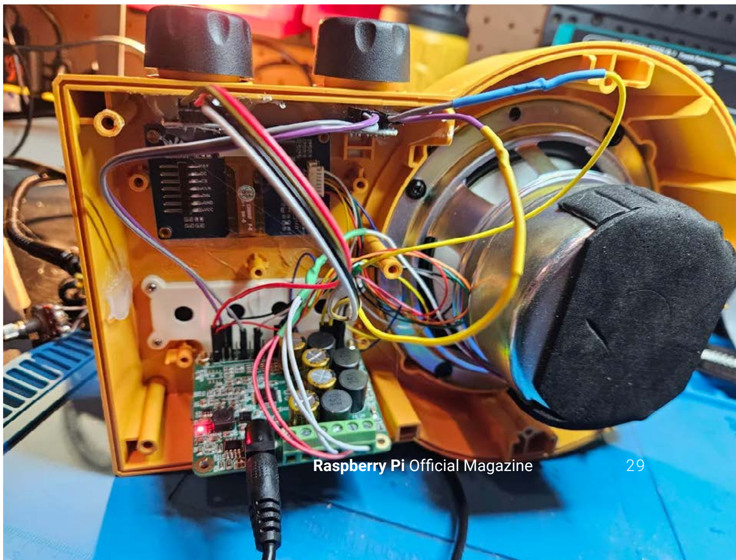
Legitimate-Elk 3627

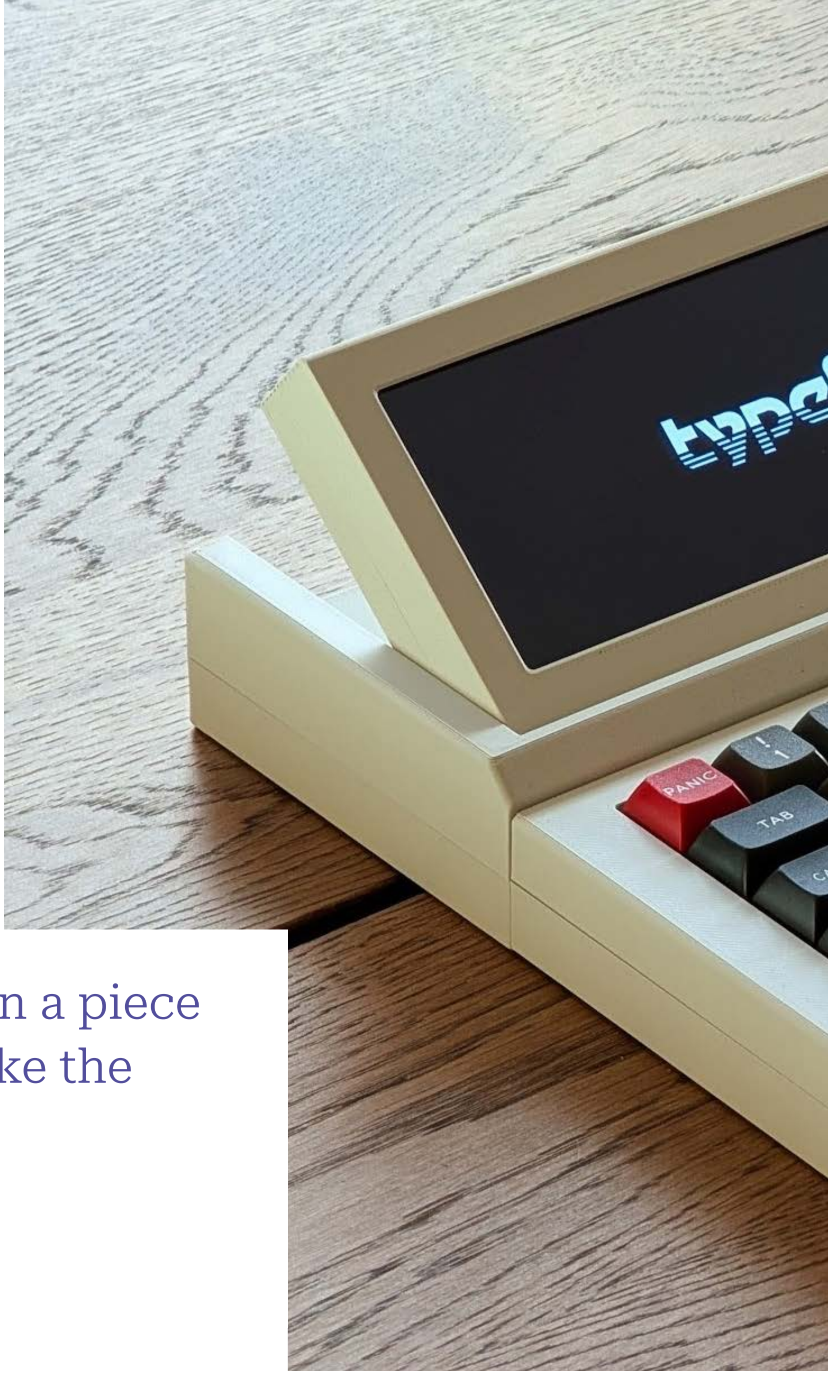
rpimag.co/XM-Radio

Gone are the days when working the radio required a steady hand, a keen ear, and a tolerance for fuzzy noise. Today you just press a button – or click forward on a rotary encoder – and thanks to the magic of digital radio, your device switches to the state you want perfectly, first time. The phrase 'tune in' has become archaic, a relic just as outdated as the floppy disk icon that modern computers still use for Save. There was a magic to turning a potentiometer and having different music and sometimes different languages appear in your living room, but on this point we can't pretend that the olden days were better – we have more choice than ever, and it's far easier and cheaper to exercise that choice. DAB and internet radio are far better than almost anything we had access to as kids.

Raspberry Pi has played its part in this sonic betterment. This build, for example, takes an old AM/FM radio (the crackly kind) and replaces most of its parts with modern electronics to be able to get internet radio – specifically XM Radio, which requires an account with a subscription. It uses a Raspberry Pi Zero 2W, an amplifier, a screen to display channel info, and a pair of rotary encoders. The original speaker is used, along with that glorious plastic casing, which has plenty of room in it for the new hardware.

▲ Rotary encoders (on top) are used rather than potentiometers because everything, even the volume control, is digital





3D print

Distraction-free writing on a piece of new, vintage kit – it's like the olden days, but better

By **Andrew Gregory**

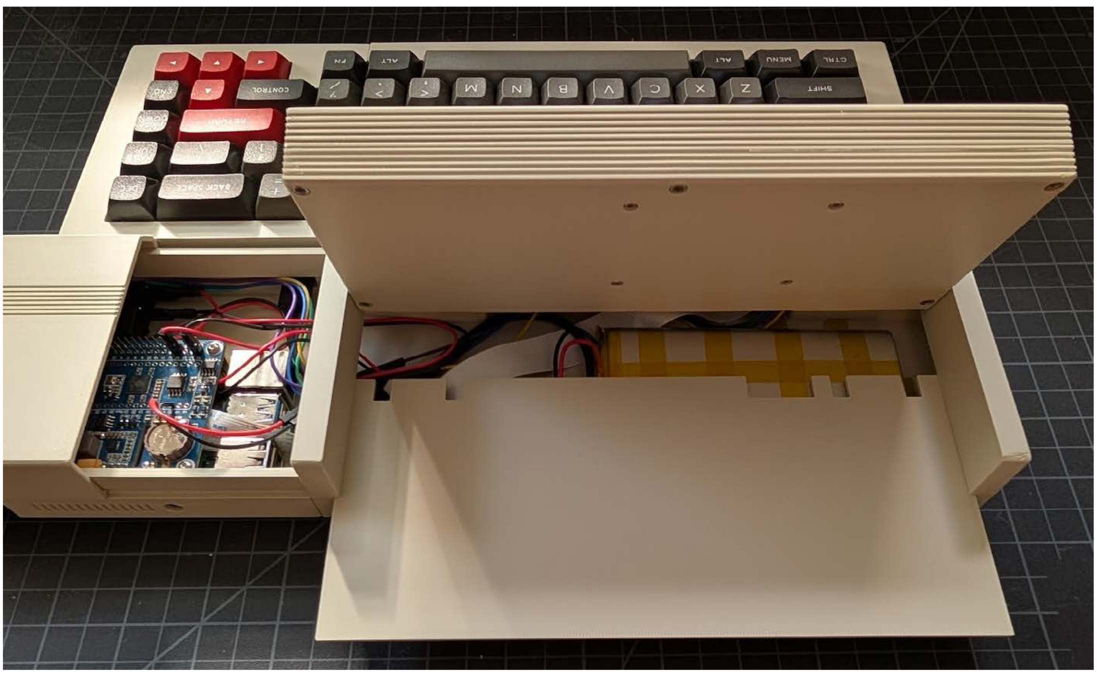
rpimag.co/typeframe

Nineteen eighty-five was a good year. It gave us *Back to the Future*, Tears for Fears' *Songs From the Big Chair*, and perhaps more relevantly for us, the publication of Richard Stallman's GNU Manifesto, which somehow managed to be more than three times longer than the American Declaration of Independence.

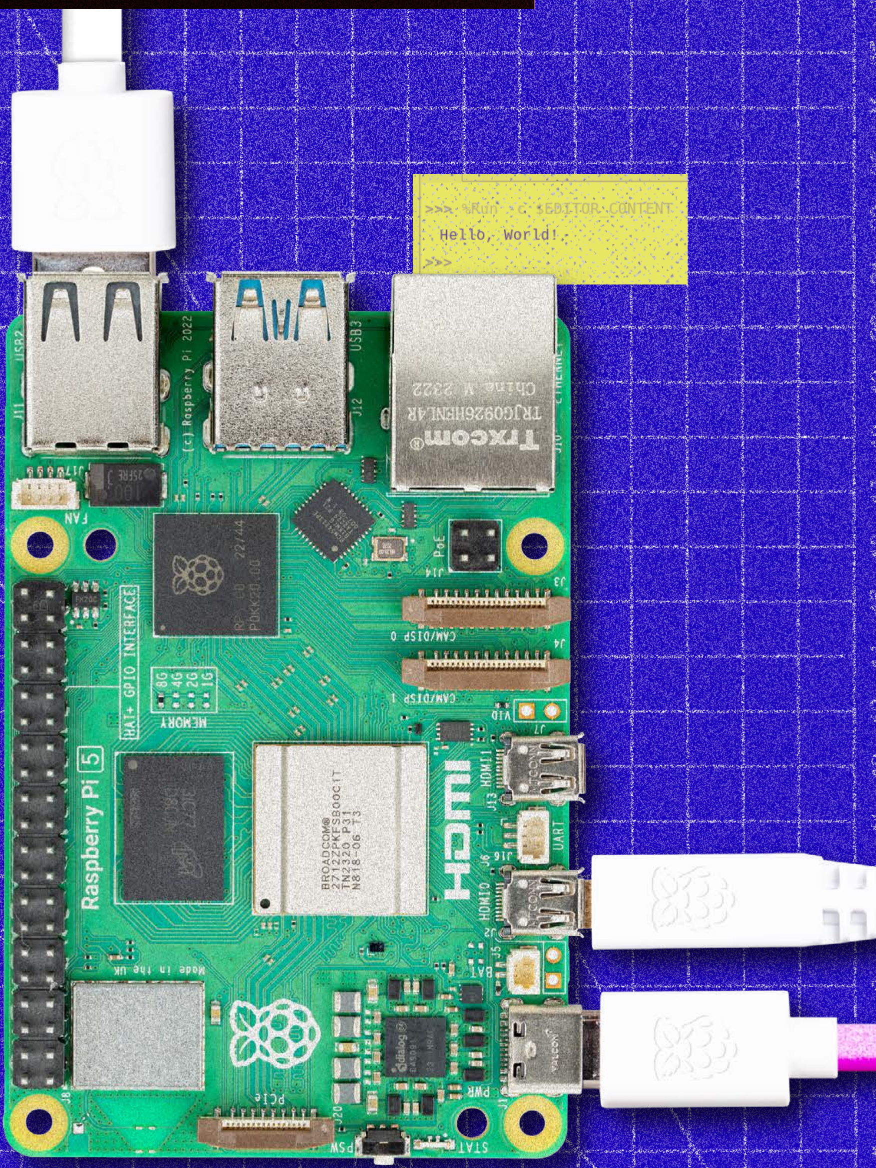
The year 1985 is also when the Epson PX-4 was released. This portable computer, which could run on four AA batteries, was amazingly ahead of its time, with its Zilog Z80 processor and 64kB of memory. It also captured a certain eighties aesthetic that we love. Now, 40 years later, Jeff Merrick has built a cyberdeck-cum-writerdeck that channels all that was good about the Epson PX-4 and makes it better.

The Typeframe PX-88 features a mechanical keyboard, touchscreen, and a Raspberry Pi 4, all in a 3D-printed chassis with the electronics made accessible via sliding panels. Unlike the PX-4, the Typeframe PX-88 runs on a single USB power adapter. And incredibly, it's been designed to fit together with absolutely minimal soldering – just the power switch and status light.


BACK SPACE
DEL
PAGE UP
PAGE DOWN
RETURN
END
CONTROL
FN
ALT
SHIFT
MENU
CTRL

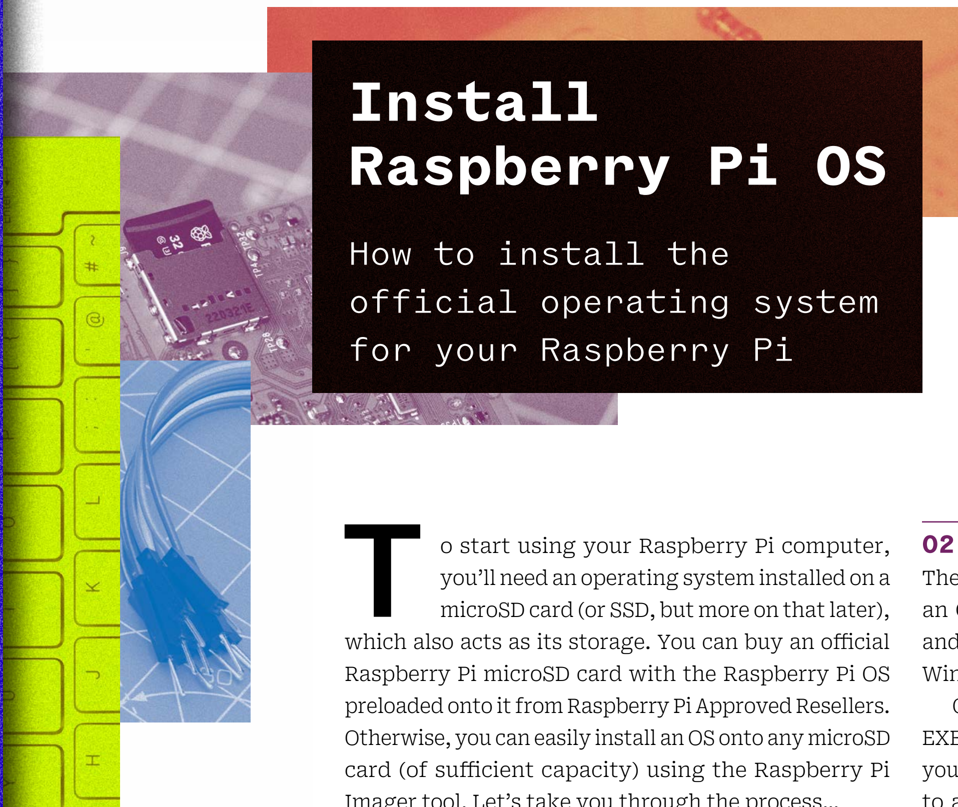


GET STARTED WITH RASPBERRY PI



No matter which Raspberry Pi model you have, you're holding a powerful little computer that can help you learn to code, control electronic circuits through its GPIO pins, and tackle all kinds of home projects. Here's how to get started…

By Phil King



Install Raspberry Pi OS

How to install the official operating system for your Raspberry Pi

The official Raspberry Pi Imager tool makes installing an OS quick and simple

To start using your Raspberry Pi computer, you'll need an operating system installed on a microSD card (or SSD, but more on that later), which also acts as its storage. You can buy an official Raspberry Pi microSD card with the Raspberry Pi OS preloaded onto it from Raspberry Pi Approved Resellers. Otherwise, you can easily install an OS onto any microSD card (of sufficient capacity) using the Raspberry Pi Imager tool. Let's take you through the process…

01 Get a microSD card

A microSD card with 32GB or greater capacity is recommended for installing the desktop version of Raspberry Pi OS. You may well be able to use a lower-capacity card, but you'll have less space left for storing files and applications. For the non-desktop 'Lite' version of Raspberry Pi OS, a 16GB card is sufficient.

In addition, for optimal performance, you should look for a Class A2 microSD card such as the official cards available from Raspberry Pi. While some cards come with Raspberry Pi OS pre-installed, it may not be the latest version, in which case you may want to reinstall it following the steps here.

YOU'LL NEED

- Raspberry Pi
- microSD card
- Raspberry Pi Imager (**rpimag.co/software**)

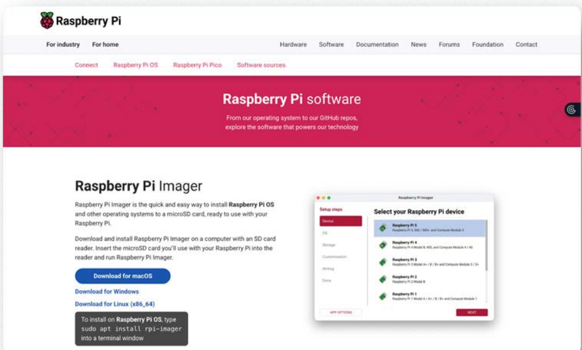
02 Install Imager

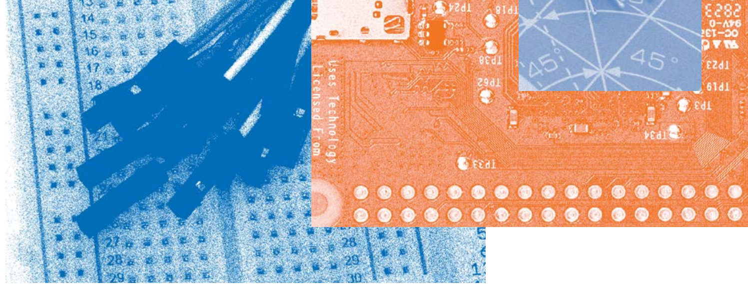
The official Raspberry Pi Imager tool makes installing an OS quick and simple. Visit **rpimag.co/software** and download it for your computer: it's available for Windows, macOS, and Debian/Ubuntu Linux.

On Windows or macOS, double-click the downloaded EXE or DMG file and follow the instructions. On macOS, you may need to change your Privacy & Security setting to allow apps downloaded from the App Store and identified developers to enable it to run.

On Linux, right-click the downloaded AppImage file, select Properties, and switch to the Permissions tab. If you see a checkbox that allows you to make the file executable, click it; if you see an Execute drop-down menu, set it to Anyone. You can now double-click the AppImage file to run the installer.

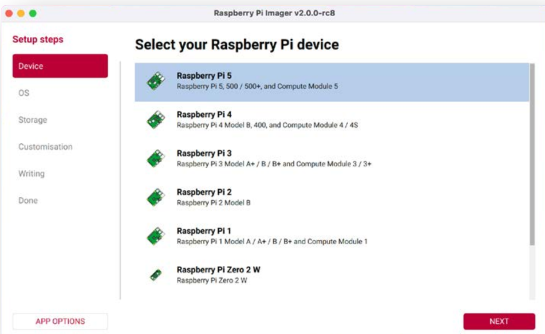
Alternatively, if using a Raspberry Pi computer running Raspberry Pi OS, you should find the pre-installed Raspberry Pi Imager in the applications menu under the Accessories category. If not, you can install it from a Terminal window, by entering `sudo apt install rpi-imager`.





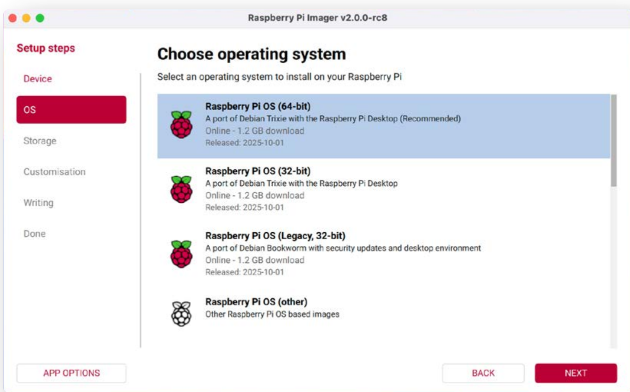
03 Choose model

Once Raspberry Pi Imager has been installed on your computer, double-click its icon to launch it. The installation steps are listed in the left-hand pane of the window. The first step is Device, prompting you to select your Raspberry Pi model in the main pane. This will filter the selection of operating systems accordingly for the following step, so that it lists only those suitable for your model. Click the Next button to proceed to the next step.



04 Select OS

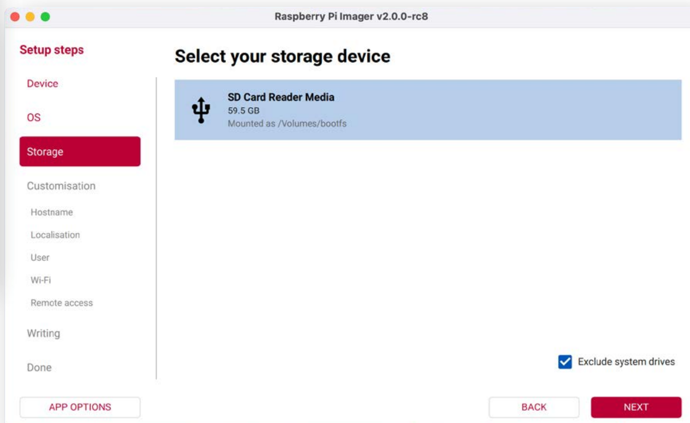
The OS step lists a range of operating systems available for your Raspberry Pi model. Versions of the official Raspberry Pi OS will appear at the top. In most cases, you'll want to select the top 'Recommended' option. For most models, this will be the latest 64-bit version with Raspberry Pi OS Desktop; for earlier and Zero models, it will be 32-bit.



05 Choose storage

You'll need to connect your microSD card to your computer to write to it. Some laptops have suitable card slots. If not, you can use a USB adapter (you'll need one if you're running Imager on a Raspberry Pi, as it can't write to the microSD card in its slot).

In the Storage step, your attached microSD card should then appear. Typically, it'll be the only device listed. If not, be careful: you don't want to erase a different device during the write process, so disconnect all removable drives except your target microSD card, then open Imager again.

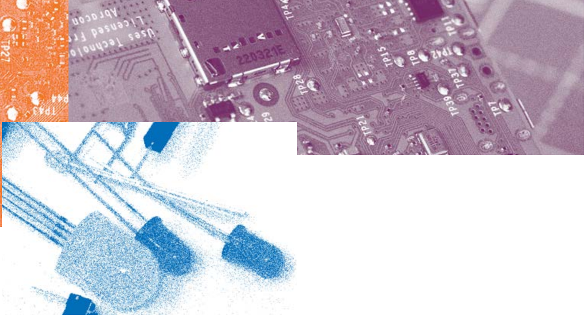


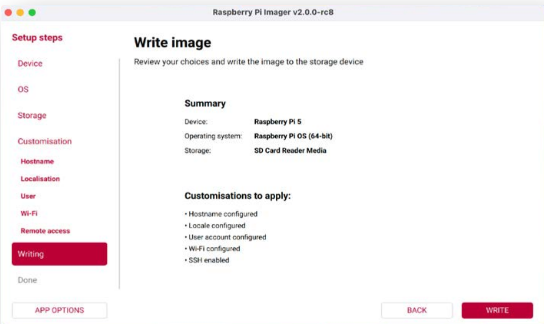
06 Customisation options

The optional Customisation step enables you to set up some things in advance, such as the Wi-Fi connection and remote access – especially useful if you'll be using your Raspberry Pi headless, without a monitor.

First, you get to set the hostname of your Raspberry Pi system to help identify it on the network. Then, you can select your locale, time zone, and keyboard layout. Next, you choose a username and password – don't forget the latter!

After that, you can enter your Wi-Fi router's name (SSID) and password to connect to it. Then you can enable SSH for remote login. There are two options for authentication: a password (which you set earlier) is simplest; or, for passwordless authentication, you can use an RSA key that you've generated on the computer



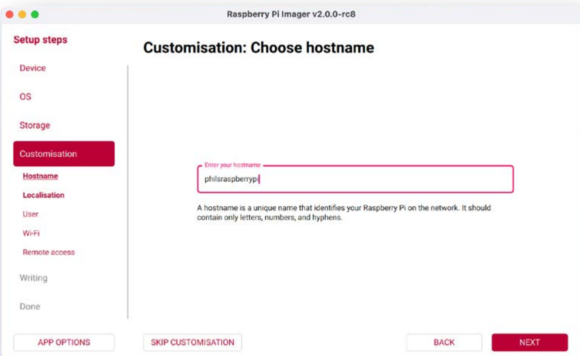


you're using (with the `ssh-keygen` command). Finally, you can enable Raspberry Pi Connect for remote access using your Raspberry Pi ID; just follow the prompts.

07 Write image

The final part of the process is to write the OS image to your microSD card. After checking the summary and list of customisation options you've set, click the Write button. Note the warning about all files being erased on the card, then click the 'I understand, erase and write' option. You may need to enter your local computer's password to proceed.

While Imager is writing the OS, don't disconnect the storage device or the write process will fail. It can take a few minutes to write the OS and then verify it. Once finished, it will show a summary, with the option to write another card with the same options. You can now remove the card and insert it in the microSD card slot on your Raspberry Pi.



Solid-state drive

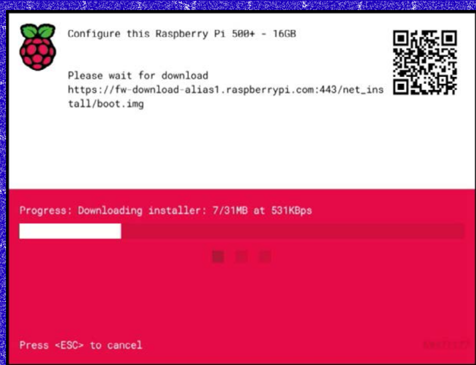
Instead of booting Raspberry Pi from a microSD card, you can use a different medium such as a USB drive or solid-state drive (SSD) by writing the OS to it with Imager. Note that the SSD in Raspberry Pi 500+ is preloaded with Raspberry Pi OS, so you don't need to do it manually.

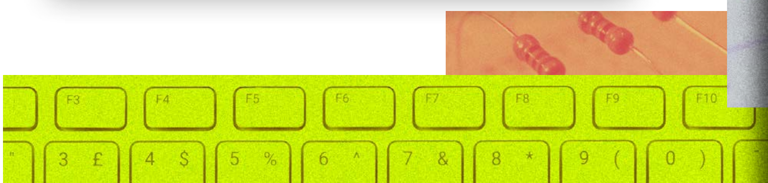
The best way to use an SSD with Raspberry Pi 5 is to buy an official SSD Kit, which includes a 256GB or 512GB NVMe SSD and connects to Raspberry Pi 5's PCIe port.

You can write an OS to the SSD using Imager on the same Raspberry Pi having first booted from a microSD card, then change the boot order in the raspi-config tool: enter `sudo raspi-config` in a Terminal window, then select Advanced Options > Boot Order; specify an option that includes NVMe, then select Finish and reboot Raspberry Pi.

Network Install

On a Raspberry Pi 4, 5, 400, 500, or 500+ with a monitor, mouse, and keyboard connected, you can instead install the OS over the network. To do so, insert a blank microSD card into your Raspberry Pi – or connect an SSD – then use an Ethernet cable to connect it to your router and power up Raspberry Pi. Hold the **SPACE** key as it boots; this will bring up the boot mode menu. Press **N** to start the network installer, which will download a special version of Raspberry Pi Imager and load it. As before, you'll then be able to use it to choose an operating system and install it onto your storage device.

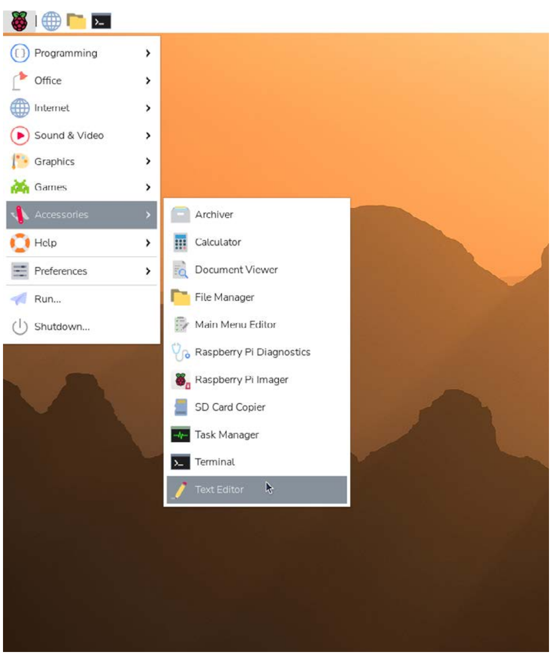




Using Raspberry Pi OS

Navigate the official operating system and its applications

When you first power up your Raspberry Pi, you may (unless you set customisation options when writing the OS with Imager) be greeted by a Welcome Wizard that takes you through the initial setup process for the operating system, including locale settings, creating a user, and connecting to Wi-Fi. If you have already set these up in the Customisation section of Raspberry Pi Imager, however, you will instead be taken straight to the Raspberry Pi OS desktop.



▲ Applications can be opened via the menu on the left

Along the top of the desktop is a taskbar. At its left end are some pinned applications – Chromium web browser, File Manager, and Terminal (command-line interface) – along with tabs for any other open applications or windows. At its right end are clickable icons including Raspberry Pi Connect, Bluetooth, Wi-Fi, and audio – right-click the last one to select an output. Other icons that may also appear here include a downward arrow if software updates are available, an eject icon if you have removable storage connected (such as a USB drive), and a mic icon if you've ever connected a Bluetooth headset.

At the far right end of the taskbar is the current time (sourced from the internet). Hovering over it shows the date; clicking it brings up a calendar.

Clicking the Raspberry Pi icon brings up an applications menu with various categories

▲ The Control Centre lets you change a range of settings

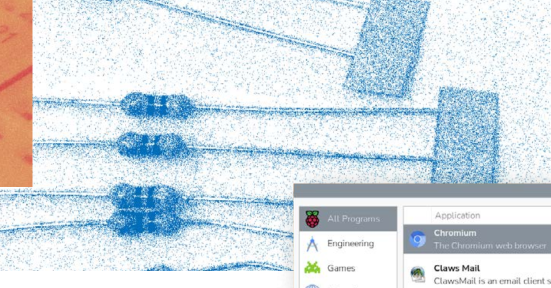
Opening applications

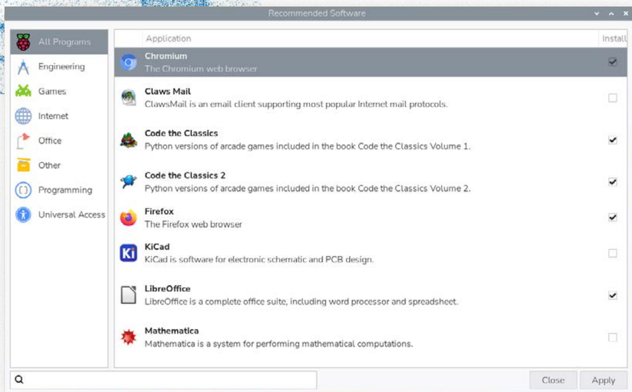
On the far left of the taskbar is a Raspberry Pi icon. Clicking it brings up an applications menu with various categories. For instance, under Programming you'll find the Geany and Thonny editors; under Sound & Video you'll find the VLC Media Player. Just click on the name of an application to launch it.

The number of applications that appear in the menu by default depends on which version of the OS you installed – the 'Full' one includes a larger range of recommended applications. To install more applications, the easiest way is to use the Recommended

YOU'LL NEED

- Monitor
- Keyboard
- Mouse



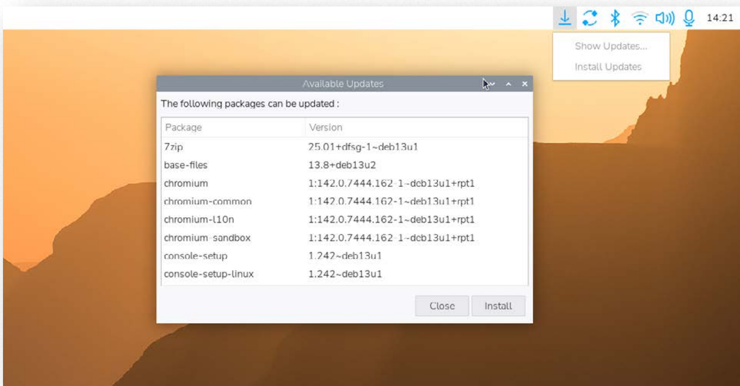


▲ Install extra applications with the Recommended Software tool

Software tool, found under the Preferences category of the menu. Upon launching it, you can browse applications in various categories. Tick the checkbox for any app(s) you wish to install, then click Apply or OK. For instance, you may well want to install the LibreOffice productivity suite, which includes a word processor, spreadsheet, database, and other applications. Once installed, it will appear under the new Office category in the applications menu.

If you want to uninstall an app, untick its checkbox in the Recommended Software tool. An even greater range of applications and software packages can be found in the Add/Remove Software tool, found under Preferences in the menu.

▼ Click on the arrow icon in the taskbar to install updates



In addition, applications and tools can be installed and run from the command line by opening a Terminal window. Here, you can issue a host of different commands to become a power user of your Raspberry Pi. For more details, check out the *Conquer the Command Line* book: **rpimag.co/conquertcl**.

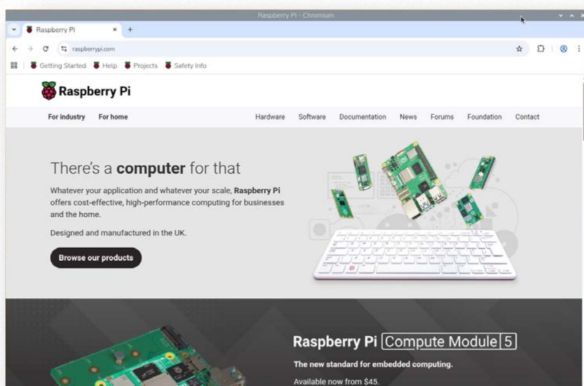
You can also find the book in the Bookshelf app, under Help in the applications menu. This lets you download digital PDFs of books and *Raspberry Pi Official Magazine* issues. Note that some titles are padlocked unless you are Raspberry Pi contributor (from £5 a month) and log in with your Raspberry Pi ID.

System management

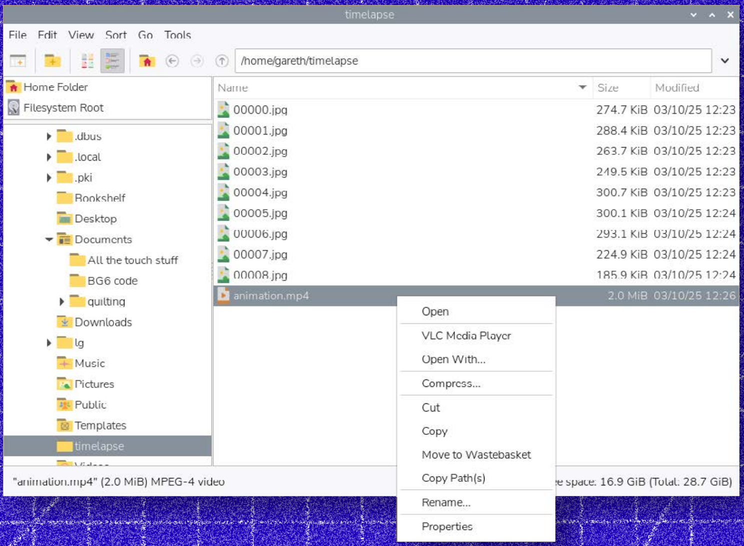
If you want to make changes to the settings for how your Raspberry Pi OS system works, the Control Centre is the place to go (replacing the Raspberry Pi Configuration tool in earlier versions of the OS). Found under menu > Preferences, it features numerous sections, selectable from the alphabetical list on the left. For instance, in Display you can enable screen blanking (to turn it black when the OS is not used for several minutes) and choose when/if an on-screen keyboard appears. Under Interfaces, you can enable or disable communication protocols like I2C and SPI, along with remote login methods like SSH and VNC. Localisation is pretty self-explanatory, enabling you to set the locale, time zone, keyboard layout, and Wi-Fi country.

As mentioned, if you see a downward arrow on the right-hand side of the taskbar, it means that software updates are available – for the system and/or applications. Clicking on the icon brings up two options: Install Updates to download and install them, and Show Updates to preview a list of them first.

Finally, when you've finished using Raspberry Pi OS for the time being, you will want to shut down your Raspberry Pi. It's important to do this properly rather than simply unplugging it, which could result in the operating system being corrupted. To do so, select Shutdown from the applications menu. This will bring up a dialog with four options: as well as Shutdown to shut the system down, you can Reboot it (shut down before restarting), select Lock Screen to return to a password-protected screen, or Logout of your user account (useful if your system has more than one user).



▶ Visit websites with the built-in Chromium web browser



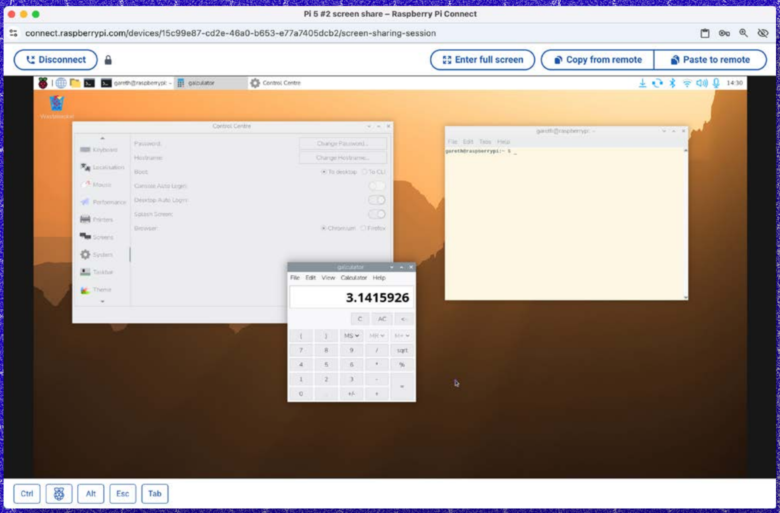
File Manager

This is the file browser for Raspberry Pi OS and is similar to that found on other computers, enabling you to browse the folders (aka directories) and files on your system. The left pane shows the directories, while the right-hand pane shows the files and subdirectories for the selected directory.

Upon first opening the File Manager, it will show your user's 'home' folder, which contains subdirectories such as Documents, Downloads, Music, Pictures, Videos, and (once you've opened the Bookshelf app) Bookshelf containing downloaded book and magazine PDFs.

You can drag and drop files between folders. Double-clicking a file will open it in its default application; right-clicking lets you choose what to open it with, along with numerous other options.

If you insert a USB drive, you'll be prompted to open it in File Manager, where you can view its files and folders.



Raspberry Pi Connect

While it's still possible to access the Raspberry Pi OS desktop remotely via VNC, and the command-line interface with SSH, Raspberry Pi Connect makes it easier to do both remotely – not only on your local network, but from anywhere in the world.

To set it up (if you haven't done so beforehand when writing the OS with Raspberry Pi Imager), click its circular icon on the right-hand side of the taskbar and select Turn On Raspberry Pi Connect. This will open the browser, prompting you to sign in with your Raspberry Pi ID – if you don't have one yet, click the link to create one. After authenticating, you're prompted to enter a nickname for your Raspberry Pi.

Once this is done, the Connect icon will turn blue to indicate it is working. You can now log in from any remote computer or device by visiting **connect.raspberrypi.com**. From the 'Connect via' drop-down next to your device, select 'Screen sharing' to view the desktop (or 'Remote shell' for its command-line interface). You can now navigate the desktop and use it as normal.

Coding and digital making

Program Raspberry Pi with Python and control an electronic circuit

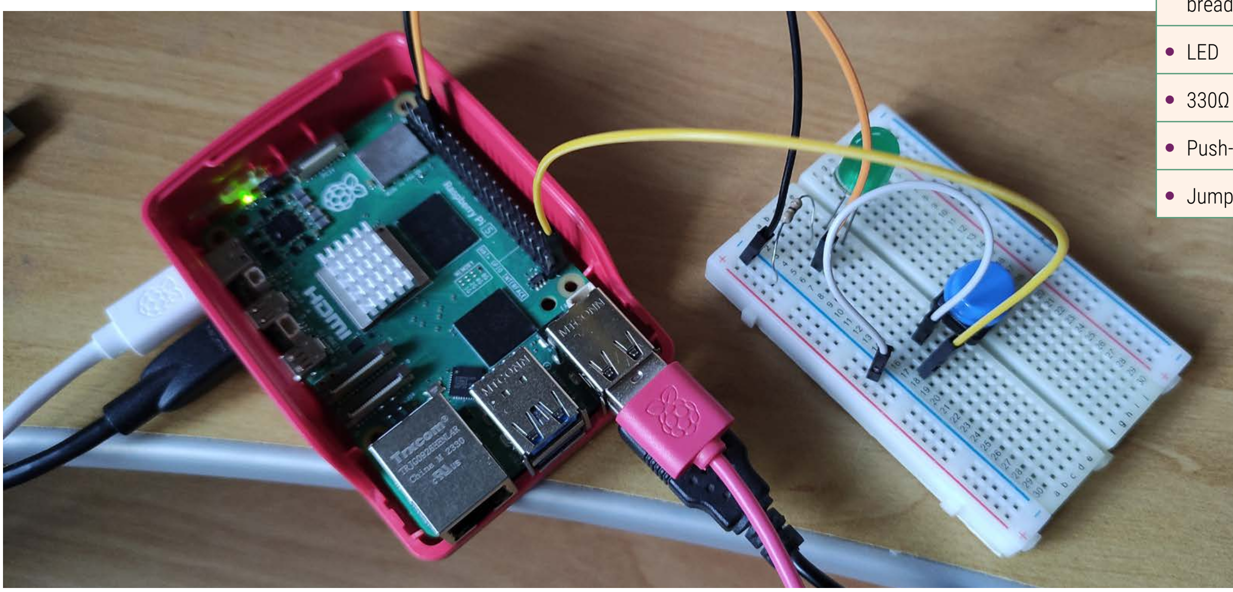
One of the best aspects of a Raspberry Pi computer (and the reason it was originally created) is the ease with which you can start programming on it and controlling electronic circuits connected to its GPIO header.

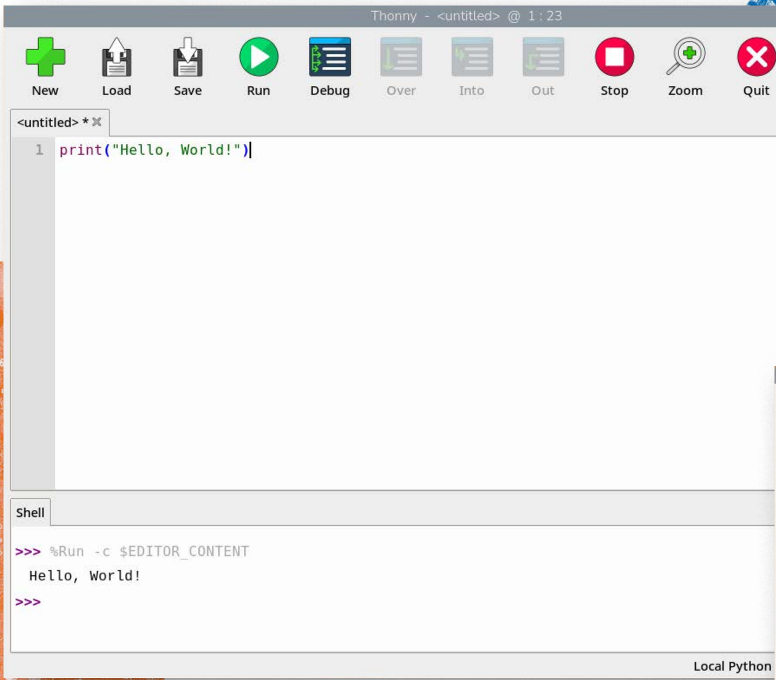
While you can code in a variety of languages, one of the easiest to get started with is Python. To get started, open the applications menu and select Programming > Thonny. This opens the Thonny IDE (integrated development environment) for coding in Python.

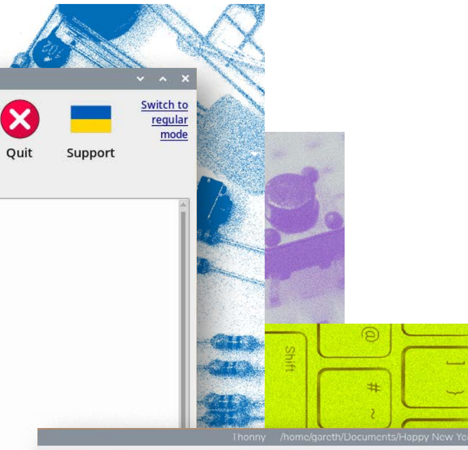
One of the easiest languages to get started with is Python

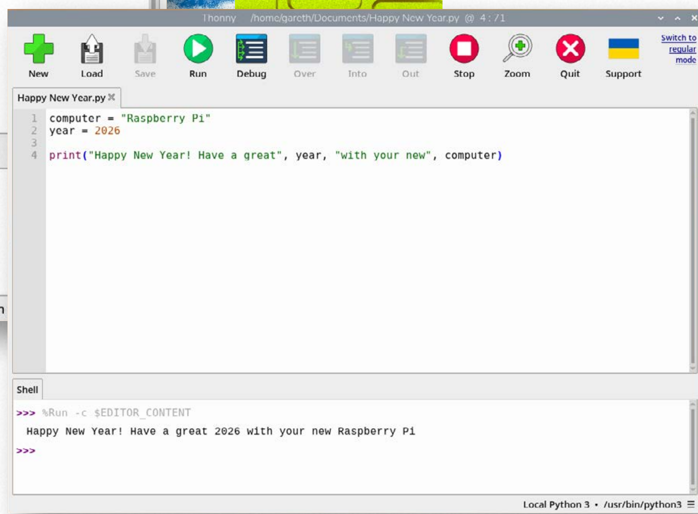
YOU'LL NEED

- Solderless breadboard
- LED
- 330Ω resistor
- Push-button
- Jumper wires









Hello, world

You can start by typing a simple one-line Python instruction in the Shell area at the bottom of the window:

```
print("Hello, world!")
```

Press the **ENTER** key and it will print the message 'Hello, world!' to the Shell area. As with other Python functions, brackets are used to enclose the output – in this case a text string within the quote marks.

Alternatively, you can enter the same line in the main coding area above the Shell area and click Run to run the program. The result will be the same. Try it.

Text strings and numbers can both be stored in variables, each of which has a user-defined name. For instance, you can store the text string 'Raspberry Pi' in a variable called `computer` , and a number in a variable called `year`, with:

```
computer = "Raspberry Pi"
year = 2026
```

Then print a celebratory message; for instance:

```
print("Happy New Year! Have a great",
year, "with your new", computer)
```

This should output the message, 'Happy New Year! Have a great 2026 with your new Raspberry Pi'.

Python maths

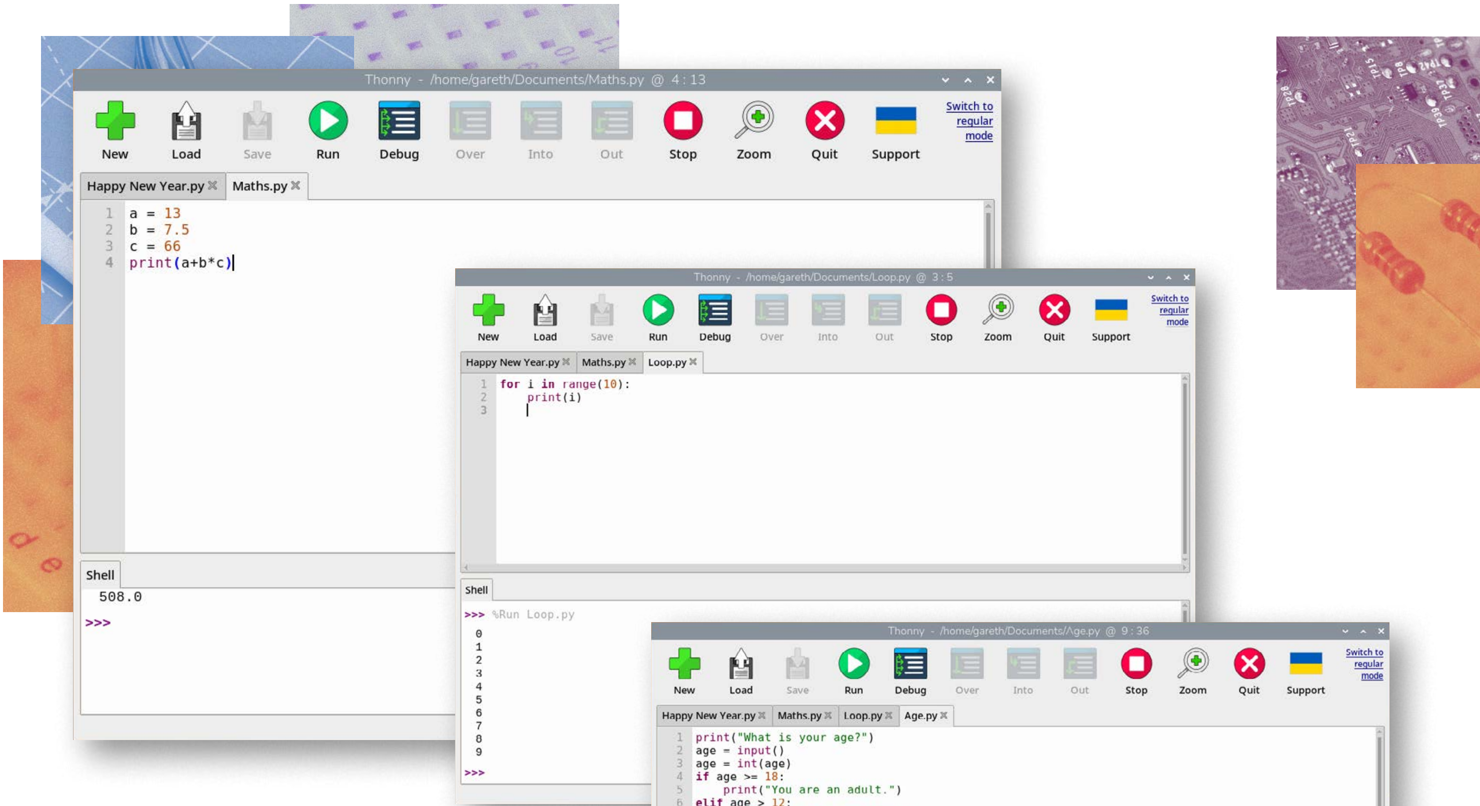
You can use maths operators to add (`+`), subtract (`–`), divide (`/`), and multiply (`*`) numbers. For instance, let's use them on some variables:

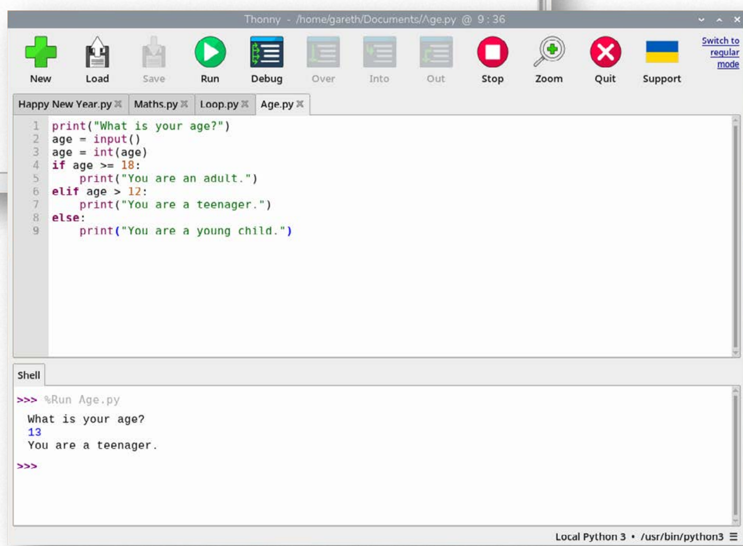
```
a = 13
b = 7.5
c = 66
print(a+b*c)
```

The result given for this is 580.0 – since we have set a floating-point value for `b` (i.e. it's not an integer), the result is too and so has a decimal point. Note also that the BODMAS/PEMDAS rule applies, so multiplication is done before addition.

Loop the loop

Another important aspect of coding is a loop. There are two main types in Python, using the keyword `while` or `for`. Both are followed by indented (typically by four spaces) lines that form the loop to be repeated. The difference is that `for` has `range` parameters to tell it how many times to execute the loop. For instance:






```
for i in range(10):
    print(i)
```

...will print the numbers 0 to 9 (it starts at 0 by default and finishes with the tenth loop).

The `while` keyword, on the other hand, performs the indented loop below it for as long as its variable has a certain value. If you just use `while True:`, it will execute the loop ad infinitum, which can be very useful for performing repetitive tasks.

On condition

Another key concept of coding is the use of conditional statements which only execute the block of code below (again indented) when a specified condition is met. For instance:

```
print("What is your age?")
age = input()
age = int(age)
if age >= 18:
    print("You are an adult.")
```

Here, using Python's built-in `input()` function, we ask the user to input their age. We convert that string to an integer in the following line. We then use `if` to test a condition for it (here, whether it is greater than or equal to 18). If so, the indented line below is executed, printing the message.

We can add further conditions and outputs to this with `elif` and/or `else` statements:

```
print("What is your age?")
age = input()
age = int(age)
if age >= 18:
    print("You are an adult.")
elif age > 12:
    print("You are a teenager.")
else:
    print("You are a young child.")
```

Blink an LED

Now we know a bit about Python, let's try blinking an LED with it. Connect the LED as shown in **Figure 1**, with its legs in two different columns of holes on the solderless breadboard (so they're not connected to each other). The longer leg is the anode (positive), which we will connect to one of Raspberry Pi's special programmable GPIO pins. The latter can be used as an input or output; here it's an output. We've wired ours to the GPIO 14 pin (**rpimag.co/gpio**), using a male-to-female jumper wire.

The cathode (negative) leg of the LED is connected to a GND (ground, or 0V) pin via a 330Ω resistor. The resistor is vital to limit the maximum current flowing through the LED – too much current could damage it or even your Raspberry Pi. Make sure your Raspberry Pi is turned off when wiring up the LED.

With the LED connected, we can turn it on and off in Python – see the **blink.py** code listing. Using the GPIO Zero library makes things easier, so we import its `LED` function at the start (along with `sleep` from the time library). We then set GPIO 14 as an output pin for our LED. Finally, we use a `while True:` loop to repeatedly turn the LED on and off, with a one-second (`sleep`) delay each time.

blink.py

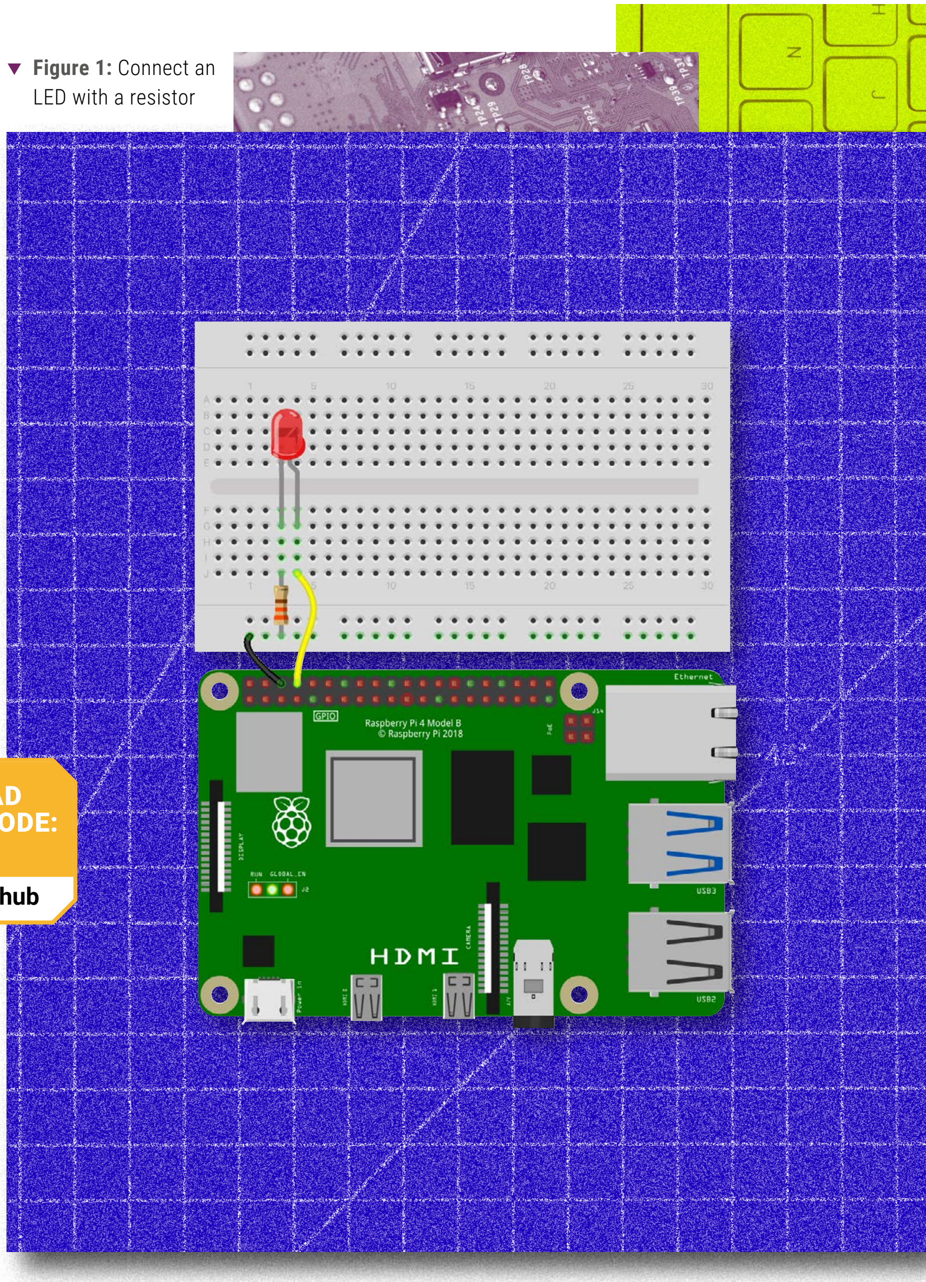
> Language: **Python**

DOWNLOAD THE FULL CODE: rpimag.co/github

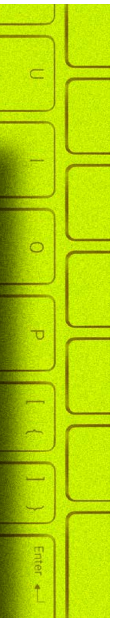
```python
from gpiozero import LED
from time import sleep

led = LED(14)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```



▼ **Figure 1:** Connect an LED with a resistor



Let's add an input device to our circuit, in the form of a push-button

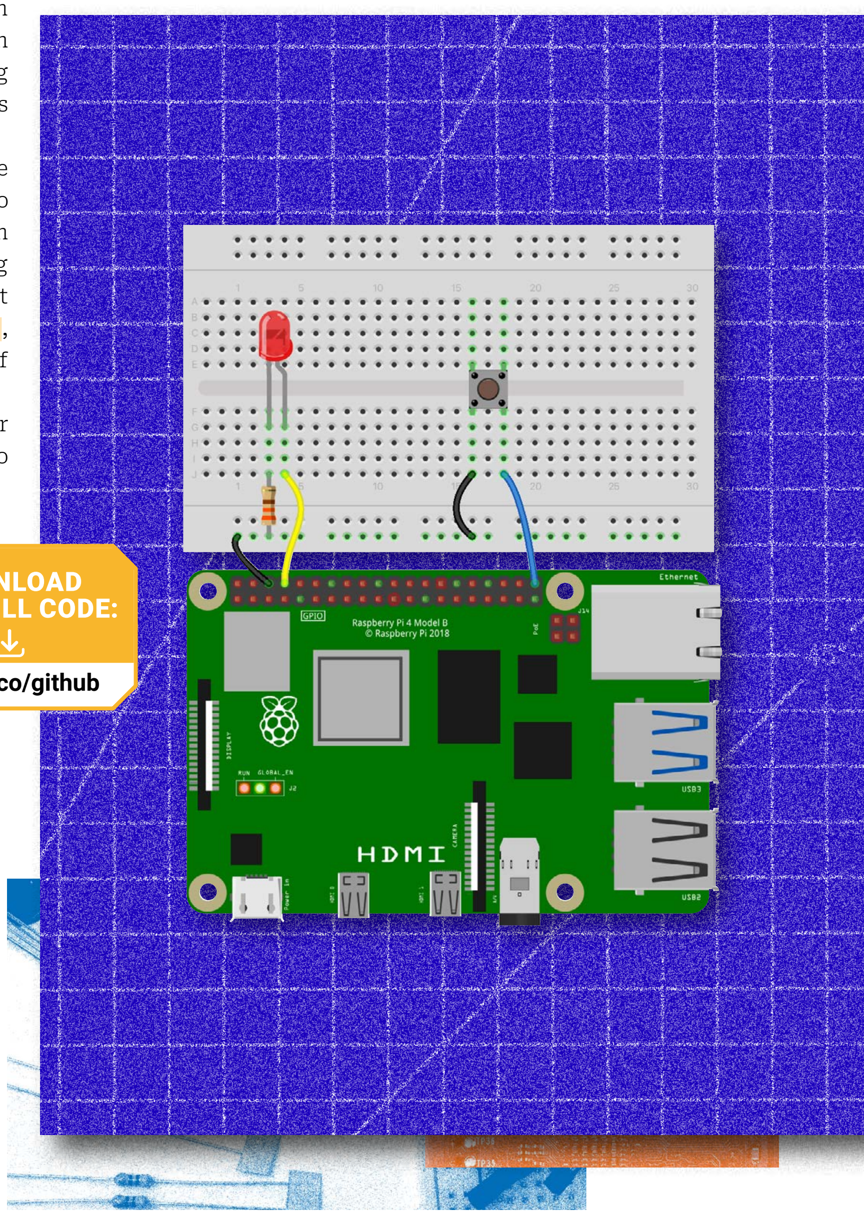
Push the button

Now let's add an input device to our circuit, in the form of a push-button. Wire it up as shown in **Figure 2**. Note that here we're using a 'ground rail' (all its holes are connected) on the breadboard to connect both components' cathodes to the same GND pin on Raspberry Pi. You'll often see this in circuits with multiple components. We've connected the other leg of our button (on the same side of the breadboard's central gap) to GPIO 21 (**rpimag.co/gpio**).

In our code (**led_button.py**), we also import the `Button` function from GPIO Zero and assign GPIO 21 to our button. In an infinite `while True:` loop, we then check whether the button has been pressed, using GPIO Zero's built-in `is_pressed` function. (Note that we could instead have used `if button.value == 1:`, where the `==` is Python's conditional form of equals.) If so, we light the LED, otherwise we turn it off.

Can you add more LEDs and/or buttons to your circuit? And perhaps also an active piezo buzzer to make a noise?

▼ **Figure 2:** Adding a button to the circuit



led_button.py

> Language: **Python**

DOWNLOAD THE FULL CODE:

rpimag.co/github

```python
from gpiozero import LED, Button

led = LED(14)
button = Button(21)

while True:
    if button.is_pressed:
        led.on()
    else:
        led.off()
```

Project ideas

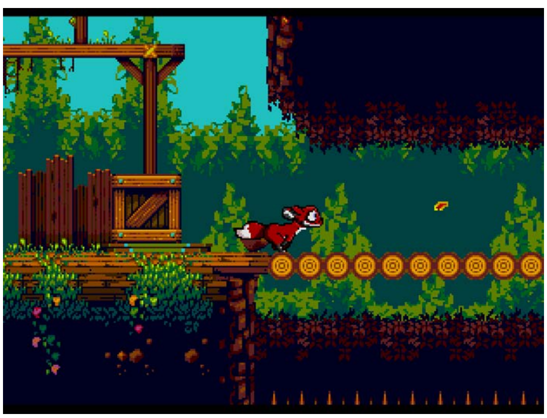
Raspberry Pi can be used in so many different and exciting ways

With its Linux-based operating system, processing power, and GPIO header, your Raspberry Pi can drive a wide variety of projects. We've highlighted a few popular software-based applications to get you started, but the potential uses go far beyond these. For further ideas, see this issue's 'Brilliant Beginner Builds' feature.

Retro gaming

One of the most popular uses of a Raspberry Pi is as a retro games machine that can emulate classic computer and consoles. The main options available for this are Recalbox (**recalbox.com** and available in Raspberry Pi Imager), RetroPie (**retropie.org**), Lakka (**lakka.tv**), and Batocera (**batocera.org**). All of these feature a user-friendly front end and can emulate myriad systems such as NES, SNES, Mega Drive/Genesis, PlayStation, ZX Spectrum, C64, and Amiga.

Game ROMs can typically be added via a USB drive or over the network. Be careful downloading them from sites hosting copyrighted games illegally, however. There are lots of other legal ROMs available, including many modern 'homebrew' titles developed for classic hardware – see **rpimag.co/legalroms**.



Retro-style games include Tanglewood for Sega Mega Drive / Genesis

Media centre

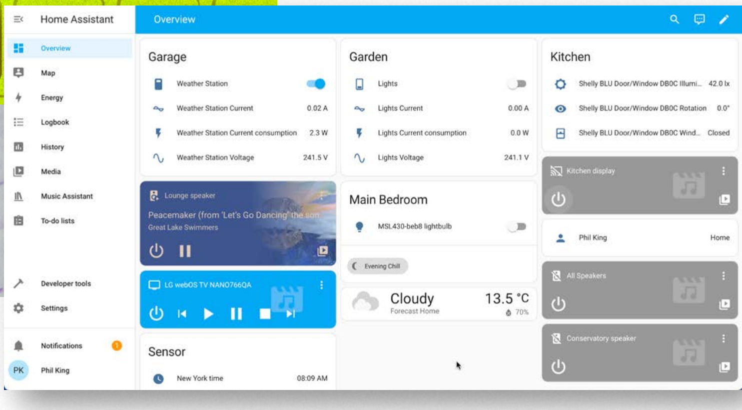
Your Raspberry Pi can be turned into a media centre to play video and audio on your TV. One of the simplest ways to do this is to install the specialist media player OS, LibreELEC (**libreelec.tv** and available in Imager, under the 'Media player OS' category). Based on the open-source Kodi software, it enables you to play files stored locally and on network-attached storage (NAS), as well as offering access via add-ons to popular streaming services such as YouTube, BBC iPlayer, and Spotify. An optional mini remote control makes navigating menus easier.

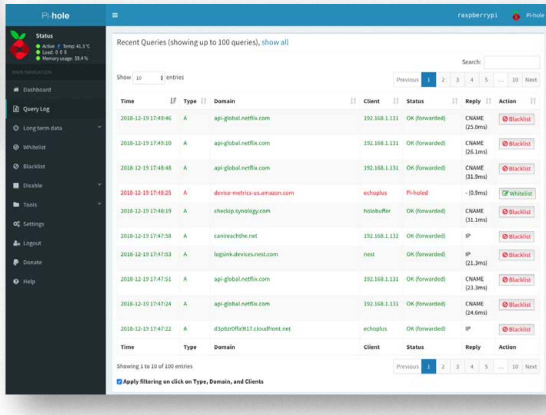
Alternatives to LibreELEC include Plex (**plex.tv**) and Recalbox (also a games emulator), along with audio-specific systems such as Volumio, moOde, and piCorePlayer. For more details on building the ultimate media centre, check out our detailed guide in issue 155 (**rpimag.co/155**).

Home automation

As we explored in our issue 154 guide (**rpimag.co/154**), it's easy to set up your own smart home dashboard on Raspberry Pi with Home Assistant – the free, open-source home automation platform. Rather than relying on a proprietary system, this enables you to build a fully customisable smart home setup tailored to your own preferences.

Once set up, you can view your Home Assistant dashboard with a web browser on any device. It's easy to add all sorts of smart devices, including Google and Alexa ones. You can assign devices to areas/rooms and





▲ The Home Assistant web dashboard

▶ Pi-hole lets you add domains to a blacklist to block adverts

create automations – from scratch, or by adapting one of the many 'blueprint' templates shared by the Home Assistant community. In addition, 'scenes' enable you to set a state for a group of devices or entities.

Ad blocker

These days, many websites seem to be crammed with obtrusive adverts and pop-ups. While browser-based ad blockers are available, a smarter and more reliable method is to do it at a network level, to work on all connected devices. For this, you can install Pi-hole on a Raspberry Pi and then access its numerous options and statistics via a web dashboard.

Pi-hole works as a DNS sinkhole. When a device on your network requests a website, the request goes to Pi-hole first. If the domain is on the blacklist, Pi-hole blocks it, otherwise it works as usual. You can customise the blacklist, as well as adding sites to a whitelist. Since unwanted adverts are blocked before being downloaded, this has the added benefit of improving your network performance.

For details of how to set up a Pi-hole, check out this guide: **rpimag.co/pihole**.

NAS server

With a suitable storage device connected, such as an SSD, hard drive, or USB flash drive, your Raspberry Pi can be turned into a network-attached storage (NAS) server. This means that the files held on it can be accessed wirelessly by other devices on your network.

The standard method for setting this up is to use Samba – not the Brazilian dance, but a free and open-source re-implementation of the SMB networking protocol. You can then share the drive as a folder over the network and grant access to authorised users with a password. There are a few steps to the process, as detailed in this guide: **rpimag.co/nasbox**.

If you want to access files remotely from outside your network, it may be better to set up your Raspberry Pi as a cloud server using a solution such as NextcloudPi (**nextcloudpi.com**) or ownCloud (**owncloud.com**).

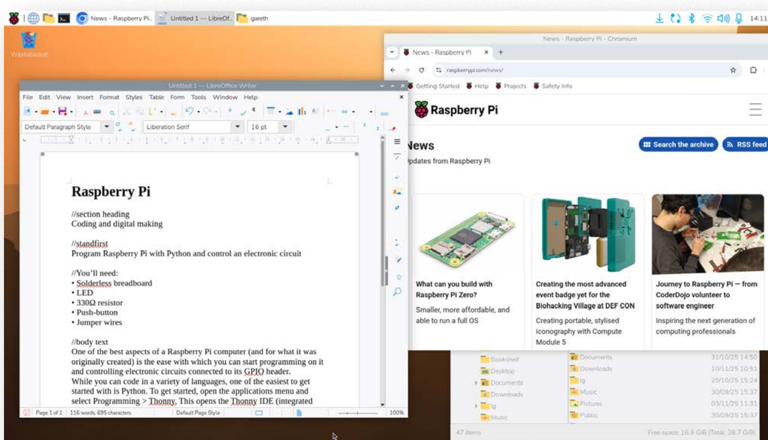
Desktop PC

Your Raspberry Pi can also work really well as a standard Linux-based desktop PC replacement, giving you all the functionality you need for work and leisure. Along with the built-in Chromium and Firefox web browsers, there's VLC for media playback – it can handle video and audio streams (such as for radio stations) too.

For email, you can install the Claws Mail client, while the LibreOffice productivity suite includes word processing, spreadsheet, presentation, database, maths, and drawing applications.

There are lots more tools and applications available to install in the Add/Remove Software tool and/or via the command line. These include the GNU Image Manipulation Program for editing photos and artwork, Inkscape for drawing vector graphics, and Audacity for editing audio files.

▼ You can use Raspberry Pi as a desktop PC



Conquer the command line: connecting disks

Disk drives come and disk drives go – tackling the management of removable storage

Maker

Richard Smedley

A tech writer, programmer, and web developer with a long history in computers, who is also into music and art.

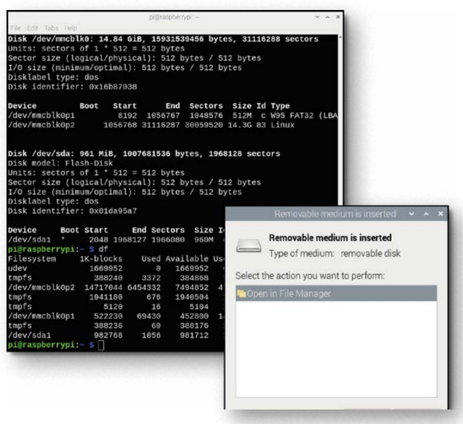
about.me/RichardSmedley

Although Raspberry Pi OS will, when running the GUI desktop, automatically mount any disk-type device (USB flash key, camera, etc.) plugged into the USB port and offer to open it, you may wish to get more direct control of the process. Or, as is more often the case, you may want to mount a disk when the Raspberry Pi is running a project that doesn't involve booting as far as the GUI, as it's not necessary for most sensor projects, robots, etc. This tutorial has plenty of new concepts to get to grips with – but they'll also be useful when a USB drive fails to mount, and the GUI doesn't give you enough useful information about what's going on.

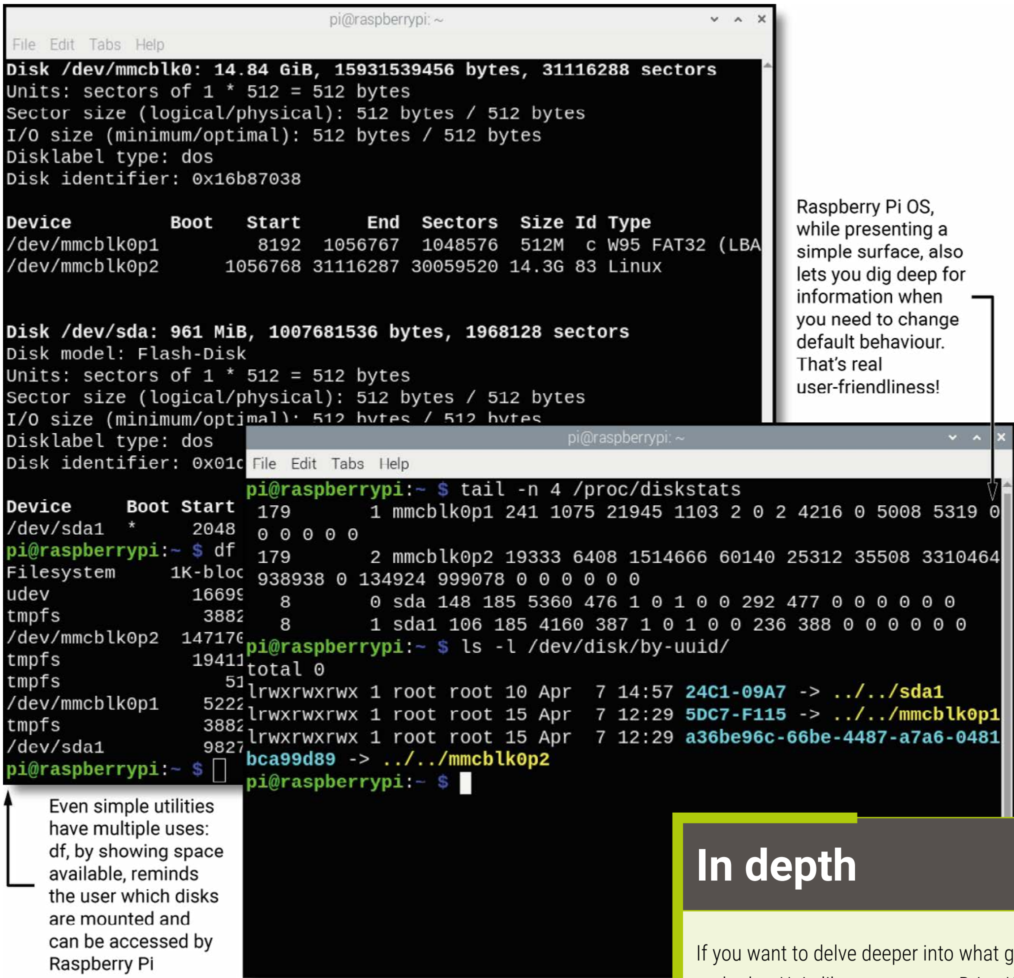
Connected or mounted?

Raspberry Pi OS wants to mount plugged-in disks and take care of the details for you – note how in **Figure 1**, the GUI tells you it's a 'removable disk' – while the command line beneath has information for you to take control when you need the job done in a particular way, telling you it's a FAT file system.

Plugging a drive or flash memory device into your Raspberry Pi (connecting it) is not the same as making it available for Raspberry



▲ **Figure 1:** Raspberry Pi OS prompts you to mount a disk



Raspberry Pi OS, while presenting a simple surface, also lets you dig deep for information when you need to change default behaviour. That's real user-friendliness!

Even simple utilities have multiple uses: df, by showing space available, reminds the user which disks are mounted and can be accessed by Raspberry Pi

◀ Working with disks from the command line

In depth

If you want to delve deeper into what goes on inside Raspberry Pi OS and other Unix-like systems, try Brian Ward's excellent book, *How Linux Works*: **rpimag.co/howlinuxworks**.

Pi to interact with (mounting it) – knowing what's on it and reading, writing, and altering files there. It's an odd concept to accept: the computer knows there's a disk plugged in, but its contents remain invisible until Raspberry Pi is told to mount it (by you or by the desktop GUI). It's like seeing a book on your shelf, but not being allowed to open or read it.

Disks and disk-like devices are mounted by Raspberry Pi OS on a virtual file system, and you'll rarely need to worry about what goes on beneath that layer of abstraction, but to see some of it, enter the command `mount`. The information displayed is of the form device on mount point, file-system type, options. You'll see lots of device without the prefix **/dev** for parts of the virtual system that you don't need to worry about; the devices that concern us start with **/dev** and have names like **/dev/mmcblk0p1** for partitions of the Raspberry Pi's SD card, and **/dev/sda1** for plugged-in USB drives.

Plug in a USB drive (a flash drive should work fine, but some hard drives may require a separate power supply). Like most computer desktops, Raspberry Pi OS automatically mounts the disks, so (unless you boot to a virtual console rather than the desktop) you'll need to unmount it. `mount` will show an entry beginning something like **/dev/sda1 on /media/pi/UNTITLED...** and you can unmount with `sudo umount /dev/sda1` (yes, that is umount without an 'n' after the first 'u'). An error will result if the device is in use, so change directory and/or close apps using files from the device. Now we can mount it just the way we want.

Plugging a drive or flash memory device into your Raspberry Pi is not the same as making it available to interact with

Disks without partitions

Some USB sticks are partitionless, so if the output of mount shows no partition number (for example, **/dev/sda** rather than **/dev/sda1**) or the output of fdisk looks strange (for example, a 1GB drive shows multiple 1TB partitions), you can use blkid (discussed in the next section) to get information about the disk, including its file-system type.

Finding the disk

The **/dev/sda1** part refers to the first and only partition (a section of the hard drive that is separated from other segments) on **/dev/sda**. The next device plugged in would be assigned **/dev/sdb1**. You can see what's being assigned by running `tail -f /var/log/messages`, then plugging in the USB device. On other systems, if **/var/log/messages** draws a blank, try **/var/log/syslog**. Stop the tail with **CTRL**+**C**. Another way of seeing connected devices that aren't necessarily mounted is fdisk, a low-level tool used to divide disks up into partitions, before creating file systems on those disks. Called with the list option, `sudo fdisk -l`, it performs no partitioning, but simply lists partitions on those disks connected to Raspberry Pi. It also gives file-system information, which you need in order to mount the disk.

Before you proceed, you need a mount point (somewhere to place the device on the file-system hierarchy) with appropriate permissions. Create one with:

```
$ sudo mkdir /mnt/usb
$ sudo chmod 775 /mnt/usb
```

Mount the disk with `sudo mount -t vfat /dev/sda1 /mnt/usb`, where vfat (or exfat, NTFS, ext3, etc.) is the file-system type you got from fdisk or blkid. Remember to replace sda1 with the partition or disk you want to mount.

Format

Should you need to format a new partition on Raspberry Pi OS, read the manual: `man mkfs` and for individual file-system types such as `man mkfs.ext4`.

File-system table

Raspberry Pi OS knows which disks to mount at boot time by reading the file-system table (**/etc/fstab**), and we could put our **/dev/sda1** in there, but if we start up with two drives plugged in, the wrong one may be selected. Fortunately, disks (or rather, disk partitions) have unique labels known as UUIDs randomly allocated when the partition is created. Find them with `sudo blkid`, which also helpfully tells you the label, if any, that often contains the make and model of external drives, or look in **/dev/disk/by-uuid**.

For our VFAT-formatted drive, we ran `sudo nano /etc/fstab` and added the following line to the end of the file:

```
/dev/disk/by-uuid/2A0F-AF03 /mnt/usb1GB vfat
nofail 0 0
```

This specifies the device name (yours will be different), mount point, file-system type, option, and two numeric fields: the first of these should be zero (it relates to the unused dump backup program), while the second is the order of check and repair at boot: 1 for the root file system, 2 for other permanently mounted disks for data, and 0 (no check) for all others. `man mount` will tell you about possible options. The `nofail` option will ensure that your Raspberry Pi will boot even if the USB drive is not present.

Editing with nano

We touched briefly on nano before. Looking in a little more depth, the first thing to be aware of is the dozen shortcuts listed across the bottom two lines of the terminal: each is the **CTRL** key (represented by the caret **^**) held at the same time as a single letter key. For example, **^R** for ReadFile (i.e. open), **^O** for WriteOut (in other words, save), and **^X** for Exit. Remember those last two for now, and you'll be able to manage nano. However, if you learn more of them, you will really race through your editing tasks.

While nano lacks the power features of Emacs and Vim, its two main command-line code editor rivals, it has useful features such as a powerful Justify (**^J**), which will reassemble a paragraph of line-break strewn text into an unbroken paragraph,

```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ sudo nano /etc/fstab
pi@raspberrypi:~ $ systemctl daemon-reload
pi@raspberrypi:~ $ ls /mnt
pi@raspberrypi:~ $ sudo mkdir /mnt/usb1GB
pi@raspberrypi:~ $ df
Filesystem     1K-blocks    Used Available Use% Mounted on
udev             1669952       0   1669952   0% /dev
tmpfs             388240    3372    384868   1% /run
/dev/mmcblk0p2  14717044 6456984   7492200  47% /
tmpfs            1941180     668   1940512   1% /dev/shm
tmpfs               5120      16      5104   1% /run/lock
/dev/mmcblk0p1    522230   69430    452800  14% /boot/firmware
tmpfs             388236      60    388176   1% /run/user/1000
pi@raspberrypi:~ $ sudo mount -a
pi@raspberrypi:~ $ df
Filesystem     1K-blocks    Used Available Use% Mounted on
udev             1669952       0   1669952   0% /dev
tmpfs             388240    3372    384868   1% /run
/dev/mmcblk0p2  14717044 6456984   7492200  47% /
tmpfs            1941180     668   1940512   1% /dev/shm
tmpfs               5120      16      5104   1% /run/lock
/dev/mmcblk0p1    522230   69430    452800  14% /boot/firmware
tmpfs             388236      60    388176   1% /run/user/1000
/dev/sda1         982768    1056    981712   1% /mnt/usb1GB
pi@raspberrypi:~ $
```

Figure 2: `mount -a` reads **/etc/fstab** to mount your disks

or apply line breaks at a fixed character length. This is a legacy of its development for email composition. **^K** cuts the line of text the cursor is on, but it isn't just a delete function: each cut is added to a clipboard. **^U** will paste the entire clipboard at the cursor position: it's great for gathering together useful snippets from a longer text.

Hit **^O** to save fstab, and the shortcut listing changes, with many now prefixed with M instead of ^ – the M is short for Meta, which means the **ALT** key on your keyboard (once upon a time, some computers had several modifier keys, such as Super and Hyper). One 'hidden' shortcut after **^O** is that at this point, **^T** now opens a file manager to search for the file/directory where you want to save. For now, don't explore any of those shortcuts: just save the file.

Disk & disk space

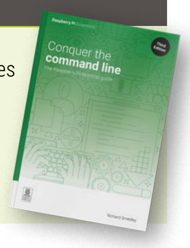
The `df` command shows you space on mounted drives: just type `df` and you'll also get a list of connected drives. It's more readable than using `mount`, though lacking file-system type info. It's also quicker to type!

After saving, exit nano and run the command `systemctl daemon-reload` to reload the fstab; now `sudo mount -a` will mount the external drive at the desired mount point (**Figure 2**), regardless of what else is plugged in. If you have other new entries in **/etc/fstab**, then `sudo mount /mnt/usb1GB` (or whatever entry you put in fstab) will mount just that chosen device if you don't want to mount any of the others. What's more, next time you reboot your Raspberry Pi, the drive will be mounted if it's plugged in.

Having got inside connected disks, the next part of this series will see us accessing all of Raspberry Pi, but remotely, from anywhere on the planet with an internet connection.

Conquer the Command Line 3rd Edition out now

This tutorial is part of a series from the latest revision of *Conquer the Command Line*. Grab it today at **rpimag.co/commandlinebook**.



PicoCalc: get started adding hardware

The PicoCalc is a fantastic handheld device, with great support for add-on sensors and hardware. Let's dive in!

Maker

Jo Hinchliffe (AKA Concretedog)

With a house and shed full of lathes, milling machines, 3D printers, and more, Jo is a constant tinkerer and is passionate about making. Obsessed with rockets and robots and much more besides, he often releases designs and projects as open source.

concretedog.blogspot.com

Back in issue 154 we reviewed the PicoCalc, the new Pico-centred handheld device from ClockworkPi. We've also had a competition to give away five of these fabulous devices to *Raspberry Pi Official Magazine* readers. PicoCalc ships running a version of the excellent PicoMite – we looked at getting started with that last issue (**rpimag.co/159**). Most of what we discussed there will work with the PicoCalc, and it's worth checking that article out alongside this one!

In this article, we are going to look at adding peripherals to the PicoCalc to begin using it to work with hardware projects. After assembling your PicoCalc, you may have noticed a set of pin headers are broken out of the left-hand side of the unit. These are broken into two separate headers. A sticker is supplied in the PicoCalc kit that you can align with these pins to label them (**Figure 1**). Notably, the uppermost set of pins on one header are the pins broken out from the PicoCalc core board: as shipped, Raspberry Pi Pico W. The lower header socket is a breakout of pins from the STM32 that is on board, which primarily is tasked with controlling the keyboard and the battery management systems and more. Whilst you can re-flash the STM32 IC and potentially make use of those pins, this article is going to focus on using the broken-out Pico pins.

To get going, we don't even need to write a program: we can simply boot into PicoMite and issue some commands from the prompt. Let's begin, as we might with any new microcontroller, and add an LED. Using a male-to-female pair of DuPont connectors, we can easily connect an LED: with the ground/cathode leg of the LED connected to the GND pin on the PicoCalc header and then the anode leg of the LED connected to any GPIO pin – we went with GP4.

A PicoCalc running PicoMite with a classic 'blink' program written in MMBasic

We can test our program by pressing the F1 key

We can issue a pair of commands directly at the prompt. First type `SETPIN GP4, DOUT` and click the **ENTER** key to set pin 4 up as an output. Then, on the next line, type `PIN (GP4) = 1` and this should turn on your LED. To turn it off, you can then type `PIN (GP4) = 0`. Whilst this is a great start, we have a microcontroller at our disposal, so let's write a classic 'blink' program to turn the LED off and on automatically.

QUICK TIP

Throughout this tutorial, we've used the classic 'ALL CAPS' approach to writing BASIC. PicoMite and therefore PicoCalc actually doesn't mind whether you use caps or not!

Let's create a new program slot by typing `NEW` and pressing **ENTER**, then let's type `EDIT` to launch the onboard editor. In the editor, we can retype our line that sets up the pin as an output. So, on the first line, type `SETPIN GP4, DOUT`. The rest of our small blink program we are going to write inside a loop. We set up a loop by typing `DO` and pressing **ENTER**. On the next few lines, when we are inside the loop, make sure you click the **TAB** key to create an indent on each line to signify the commands are inside the loop.

On the next line, write `PIN(GP4) =1` to turn on the LED. Then, on the next indented line, write a duration for the LED to stay lit, such as `PAUSE 300`. This will cause a pause of 300 milliseconds. Next, on a new line, turn off the LED with `PIN(GP4) = 0` and, in turn, below that add another pause with `PAUSE 300`. On the next line, don't include an indent and type `LOOP` to close the loop.

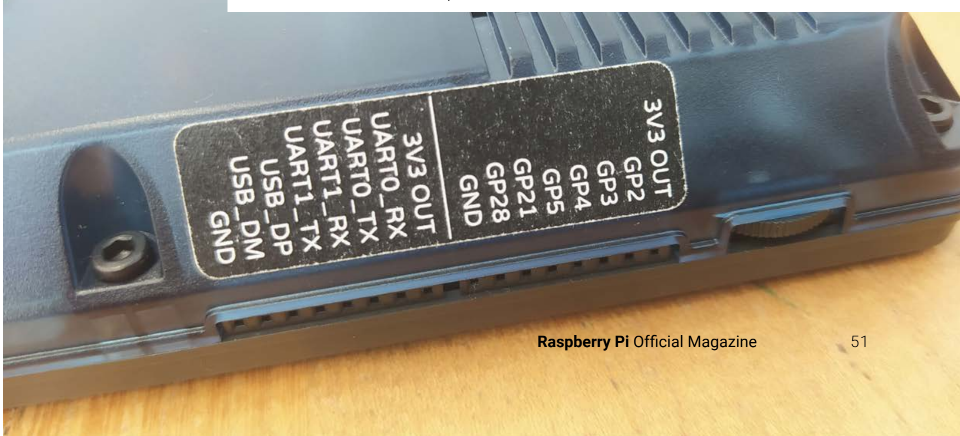
Your code should look like the following:

```
SETPIN GP4, DOUT

DO
      PIN(GP4) = 1
      PAUSE 300
      PIN(GP4) = 0
      PAUSE 300
LOOP
```

We can test our program by pressing the **F1** key to leave the editor. Note that this 'saves' the program, but as the program is unnamed, it is just held in a temporary memory slot. If you turn the unit off and on again and type `EDIT`, your program will still be there, but if you type `NEW` to create a new program then the slot and your program will be overwritten. Let's test our 'blink' program, and then we can save it properly afterwards. To test it, simply type `RUN` at the prompt and press **ENTER**. The screen will go blank, but your LED should turn on and off. To exit the program, press and hold the **SHIFT** key and press the **ESC/BREAK** key to return to the prompt.

Figure 1: The PicoCalc ships with a nice sticker with which you can label the broken-out pins

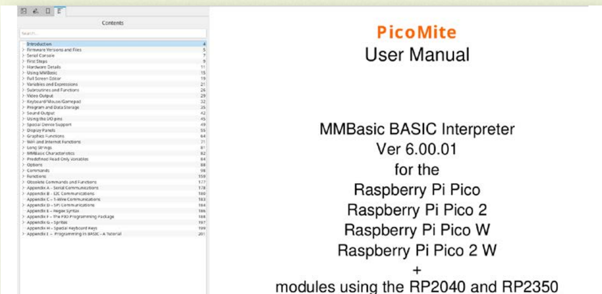


The manual

Everything in this article is based on information we have pulled from the absolutely excellent PicoMite manual (**rpimag.co/picomitemanual**) written by one of the project creators, Geoff Graham.

Whilst this isn't a manual directly for PicoCalc, it's an incredibly detailed and well-written document covering all the features of the PicoMite firmware. If you are totally new to programming with MMBasic, then Appendix I has a fantastic getting started guide, but frankly it's worth scanning through the entire document.

Notably, lots of hardware or 'special devices' have support built into the PicoMite firmware. Alongside the devices mentioned and explored in this article, there is support for ultrasonic sensors, humidity and temperature sensors, IR decoders, camera modules, and more. Also, if you aren't using PicoCalc, and are running PicoMite on other hardware, there is support for many different displays and input devices.



Having confirmed that our program works, we can save it to one of the memory slots on the Pico by simply typing `SAVE "blink.bas"`. Then we can return to this program if needed later by using `LOAD "blink.bas"`. We can see the files saved by simply typing `FILES` and hitting the **ENTER** key.

Next up, let's look at adding a temperature sensor to the PicoCalc and take some temperature readings. Again, PicoMite supports a common temperature sensor, the DS18B20, directly.

The DS18B20 is sold commonly either as the bare package sensor, which looks like a transistor package, or also is sold as a waterproof pre-terminated probe with a metal cover and a metre of cable. This is the option that we used. We crimped some DuPont pins to the three sensor wires, but, as we needed to also use a 4.7kΩ resistor in the circuit, we used a small breadboard to make all the connections. We wired the connector up based on a schematic (**Figure 2**) in the PicoMite manual.

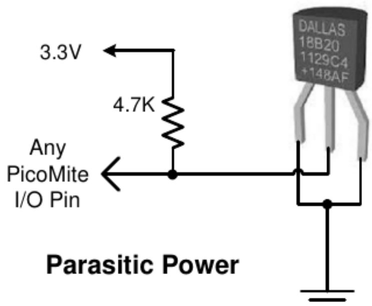
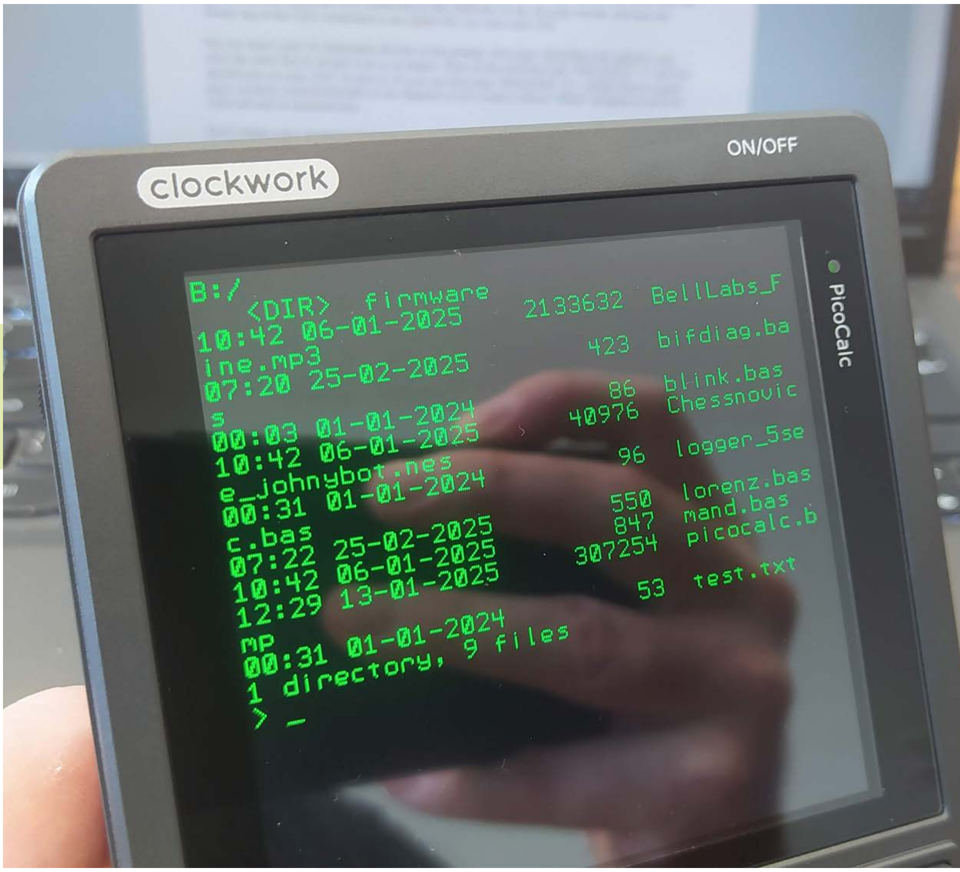


Figure 2: There are two methods of wiring the DS18B20 temperature sensor; we went with the 'parasitic power' option shown in the excellent PicoMite manual

As shown in the diagram, we connected the data line from the DS18B20 to GP4 on the PicoCalc. It's then a simple case, at the prompt, to type `PRINT TEMPR(GP4)` and you should be returned a temperature value in degrees Celsius. If there is an issue with the sensor, this command will always return the answer 1000, which can be useful if you have something connected incorrectly.

Once you have verified that the `TEMPR(GP4)` command is working, it's pretty trivial to set up a looping program that regularly logs the temperature. Similar to the blink program we worked on earlier, let's use the `NEW` command to create a new program and then `EDIT` to jump into the editor.

So, in the first instance we can simply replicate the blink program. But in the loop, instead of setting the LED on and off, we can simply read the temperature with the `TEMPR(GP4)` command, but add `PRINT` to print the temperature to the prompt. We then add a longer time delay in the loop that sets the logging frequency.



▶ **Figure 3:** It's amazing to have a huge SD card connected to a boot-to-BASIC computer! Using `DRIVE "B:"`, you can set the SD card as the current drive

If we set the `PAUSE` value to 5000, then the program will read and print the temperature every five seconds. So this little code should look like this:

```
DO
      PRINT TEMPR(GP4)
      PAUSE 5000
LOOP
```

Press the **F1** key to jump out of the editor and then issue the `RUN` command.

Brilliantly, the PicoCalc ships with an included 32GB SD card and this provides ample storage for a *lot* of basic files. By default, new programs created in the editor are saved in the Pico memory, and we need to know a few commands to help us interact with the SD card. First, from the PicoMite prompt, we can issue `DRIVE "B:"` to swap the working drive to the SD card. Note that if you want to interact with the Pico memory slots, then `DRIVE "A:"` will get you back there. Next, we can execute the `FILES` command and we should see a list of the files and directories on the SD card (**Figure 3**).

To save the program currently being edited – for example, our small logging every five seconds program we just made – we can use the command `SAVE "B:my_data_logger.bas"`. Double-check that the file is saved by repeating the `FILES` command and checking the listing. Before running a `NEW` command, make a habit of saving your current program with its name to the SD card. Of course, you don't always have to create a .bas file: you can use the editor to create all manner of files, including .txt files. You could theoretically use PicoCalc as a very offline text editor!

However, a more interesting use case for different file types is that we can write programs such as our data logging temperature program and write the data to a .txt file on the SD card.

To begin learning these commands, let's actually just type some commands at the command prompt and not in a program. First, type `OPEN "test.txt" FOR OUTPUT AS #1`. This command creates and opens a file called **test.txt**, sets it as an output file (somewhere something will be output to) and gives it a label, '#1'.

For the next command, type `PRINT #1, "this is a test first line string"`. This command will print the string enclosed in the quotes to the object #1. This means that if we opened the text file, the first line would read "this is a test first line string". Next, type another print command using the DS18B20 temperature probe from earlier: `PRINT #1, TEMPR(GP4)`. This will print the temperature value from the temperature probe on the second line in the **test.txt** file. Finally, we can use the command `CLOSE #1` to close the **test.txt** file on the SD card. We can run the `FILES` command to check that the file has been created and then we can issue the `EDIT "B:test.txt"` command to open the file in the editor, checking our two lines have been added correctly.

This is all useful learning, but one problem with this approach is that if we ran the same sequence of commands, but changed

NeoPixels

Included in the collection of devices that PicoMite, and therefore PicoCalc, supports is provision for every hardware hacker's friend, the addressable RGB LED. Checking out page 54 of the manual, there are some straightforward instructions on how to get going with some different flavours of 'NeoPixel'. Whilst you can totally add a strip of NeoPixels, we grabbed a single WS2812B variant that was on our desk and hooked it up. This was straightforward: we connected the VCC and GND pins and again we connected the data line to GP4.

Checking out page 54 of the manual, we discovered that setting a standard colour was as simple as `WS2812 B, GP4, 1, RGB(blue)`, where the argument `B` states the type of RGB LED attached. There is also support out of the box for SK6812, SK6812W, and the original WS2812, with different letters to designate them. You can also use RGB values in the command, – e.g., `WS2812 B, GP4, RGB(200,10,56)` – and then, of course, you could write all manner of code to vary the numbers and create patterns.

the text entry in the string, it would overwrite the lines that are already in the **test.txt** file. For data logging purposes, we need to open a file using the `APPEND` argument rather than `OUTPUT`. You can experiment issuing the above commands with `OUTPUT` and then `APPEND` and see the difference it makes to the files. Combining what we have covered so far, it's now pretty trivial to write a simple data logging program that will log the temperature to the SD card.

We can use `NEW` to create a new program and then `EDIT` to open the editor. Begin by setting up a loop with `DO` and then in the indented position beneath we can `OPEN` a text file in `APPEND` mode, `PRINT` the temperature to the text file, `PAUSE` for the duration we want to have between the data logs, and finally `CLOSE` the SD card before then finishing the `LOOP`. This should look like this:

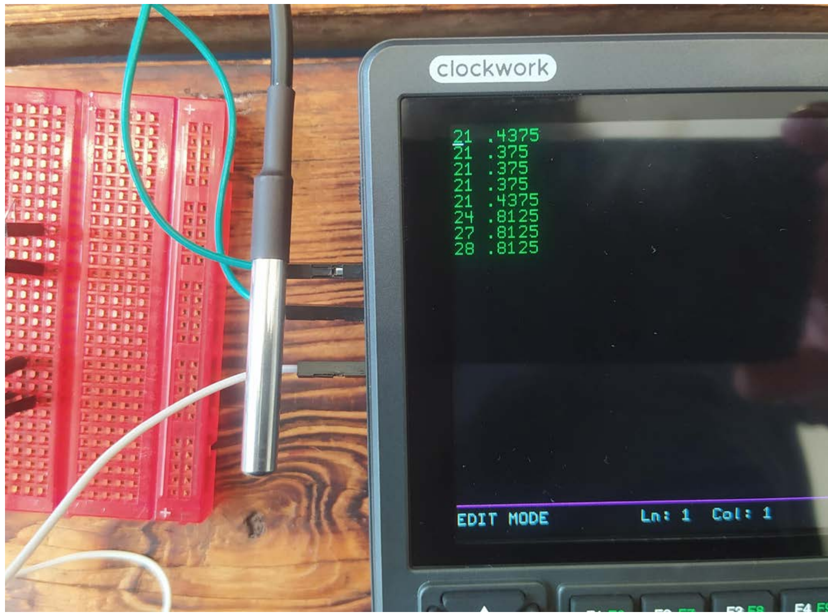


Figure 4: The output of the data logging program that writes the temperature to a text file on the SD card

```
DO
      OPEN "test.txt" FOR APPEND AS #1
      PRINT #1, TEMPR(GP4)
      PAUSE 5000
      CLOSE #1

LOOP
```

Running the logger, you can then check that you have created a text file on the SD card where each line is the next log entry of the temperature (**Figure 4**).

It's great fun playing with this stuff on the PicoCalc and it's fantastic that so many devices are supported directly without having to install libraries or other dependencies. Of course, there are some limitations: the PicoCalc uses many pins of the Pico internally compared to a more bare-bones PicoMite setup. However, the onboard keyboard and display means it's a super-portable all-in-one setup for quick playful tinkering. And you can always grab a bare Pico, flash PicoMite, and set up a standalone headless version of your experiments!

Part 01

Building a capable underwater rover on a budget

Let's look at a common approach to building the chassis of an underwater rover with plastic pipes and 3D printing

Maker

Jo Hinchliffe

With a house and shed full of lathes, milling machines, 3D printers and more, Jo is a constant tinkerer and is passionate about making.

concretedog.blogspot.com

Back in issue 157 (rpimag.co/157), we presented the Tiny Open-source Underwater Vehicle (TOUV), a palm-sized underwater explorer built with little more than some small DC motors, 3D-printed frame, and some small float tanks. It's a super-affordable and accessible project, perfect for playing with in small pools or tanks, and a great family fun activity or small STEM project.

However, its tiny stature meant that it had some shortcomings: there's little capacity for any sort of payload on board, and it's only really operable short range where you can maintain line of sight with it in its environment. Therefore, as the famous film quote says, "we're gonna need a bigger boat!"

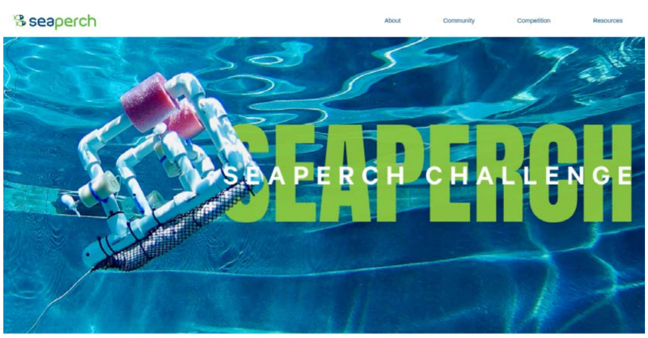


Figure 1: The Seaperch challenge is a great source of design inspiration for budget-friendly underwater vehicles

Researching reveals lots of interesting DIY underwater vehicles across a range of complexities. We wanted to build a project large enough to carry cameras and other payloads but also retain a relatively simple drive system built onto an affordable chassis that could be built by most people.

The common approach we went with was to design our chassis as a tubular frame to be made from PVC pipe and pipe

◀ The pipework chassis with thrusters mounted and buoyancy tanks

Warning!

Water and electricity

This project is low-voltage (12V), but water and electricity don't mix. Make sure to test your waterproofing.

rpimag.co/waterproofing

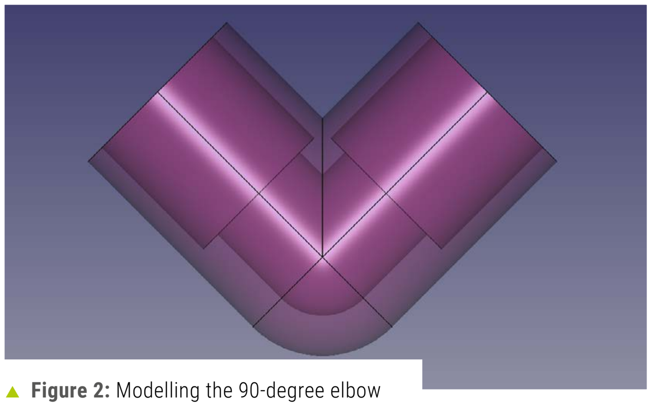
We wanted to build a project large enough to carry cameras and other payloads

fittings. This is a really common approach and there are a lot of examples. A good starting point for research is the Seaperch Challenge, **Figure 1**. The Seaperch challenge is a STEM program promoting underwater exploration, with international school teams taking part in regional competitions leading potentially to an international final event. If you look around the community, you will find a lot of information and build instructions for all manner of underwater explorers.

Of course, this has led to people increasing the capability of these PVC pipe chassis craft, and there are lots of examples of larger more complex systems based on Seaperch principles. One such source of inspiration for our build was a video series on an enhanced Seaperch build by 'Agermangineer', who has an excellent video series on YouTube (**rpimag.co/agermangineer**).

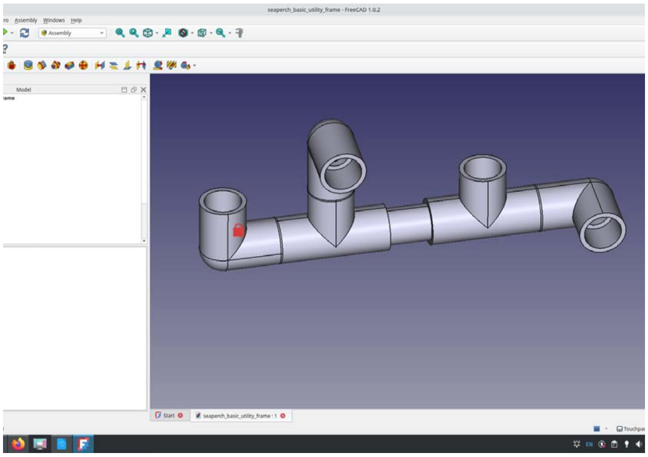
PVC pipe and fittings come in all manner of shapes and sizes. The commonest pipe family used for Seaperch-style chassis designs is ½″ schedule 40. In metric terms, this pipe is 21.3mm outside diameter with a 15.8mm internal diameter. Most DIY stores in the UK and elsewhere will stock this style of pipe and associated fittings. You can also find it available online and you can even find a range of colours. It's easy to work with: you can cut the pipe with specialist pipe-cutting shear tools that are reasonably affordable, but equally it is not difficult to cut with a small junior hacksaw.

The nice thing about PVC pipes and fittings is you can slot pieces together without gluing them and you can disassemble and reassemble many times, trying out different configurations. Before cutting your pipe into lots of sections, though, you might consider creating a 3D model assembly to try out ideas. We used FreeCAD to create a small toolkit of parts that could then be used in the Assembly workbench to play with different chassis layouts.



▲ **Figure 2:** Modelling the 90-degree elbow component in FreeCAD. Once modelled, you can use the part as many times as you like in a 3D assembly

The main aim of the chassis is to provide mount points for three thruster systems and also have room for extra payloads such as cameras and lighting. As a 3D assembly project, it's quite achievable as we essentially only modelled three different parts. We modelled a straight section of tube, a 90° elbow section (**Figure 2**), and a T piece. As ½″ schedule 40 pipe fittings are common, it is easy to search online and find dimensional drawings to help make the parts accurately. One of the standard Seaperch chassis designs calls for straight pipe sections that are 1 ½″, 2 ½″, 3″, and 4″. It's simple to draw a profile sketch of a pipe section using two circles and then extrude the part to a required length before saving that part.



▲ **Figure 3:** With our three basic parts models, we can import many versions of the parts into the FreeCAD Assembly workbench and begin to prototype

One of the advantages of the tubular chassis design is that it can be assembled and disassembled multiple times

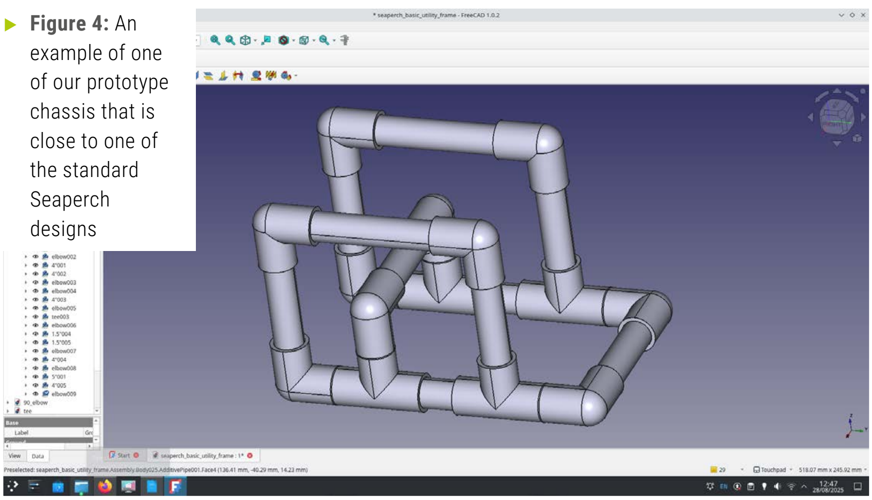
In the newer FreeCAD versions (1.0 and beyond), there is a new built-in Assembly workbench. Whilst a full tutorial on this workbench is beyond the scope of this article, you can insert multiple copies of parts into an Assembly container, then select faces or edges of parts and constrain them together (**Figure 3**). For chassis assembly, we tended to just mate faces together using the 'Create a fixed joint' tool, and you can quickly iterate numerous chassis designs for consideration (**Figure 4**). Once happy with a design, you can count up the parts in your Assembly model tree and cut your pipes to length and perhaps order the correct number of elbows and T sections you require.

Turning our attention to thrusters, we decided to go with a common approach where people modify 12V bilge pumps, designed to pump water out of boats, to create powerful water-resistant drive systems. The pumps we used are designed to move 1100 gallons of water per hour drawing around 3 amps in operation. They cost around £10 per pump and therefore are one of the more costly components. The bilge pumps in standard form have a blue plastic cover that can be removed, and then have a white plastic shroud which covers an impeller system that pushes water out of a connector. In their official usage you would place the pump in an area that may become flooded, with a pipe attached to the connector which you would run out of the vessel. To convert to a motor thruster, we need to remove the white plastic shroud and then replace the impeller with a propeller. To remove the shroud, we again used a junior hacksaw to carefully cut through the white plastic a couple of millimetres above the joint with the red plastic section. This sounds harder than it is, but once in hand, it's pretty easy to feel where you should cut. Once the shroud section was removed, we used a flat-head screwdriver to remove the impeller (**Figure 5**).

We decided it could be a fun project, and a test of our FreeCAD skills, to design and 3D-print the propellers for our thrusters. The propellers are designed with a 4mm hole through the hub, into which we inserted an M4 bolt. We then found we could purchase brass couplers where one end had a 3.17mm hole matching the bilge pump output shaft, and the other end a 4mm hole (**Figure 6**). The first props we made had a helical pitch angle of 30 degrees and, when performing a rudimentary test in the bathtub, proved to be far too aggressive! We then adjusted the pitch down to 15 degrees and this seemed to generate a lot of thrust (simply estimated by how much the system pushed against our hand) without too much violent white water thrashing!

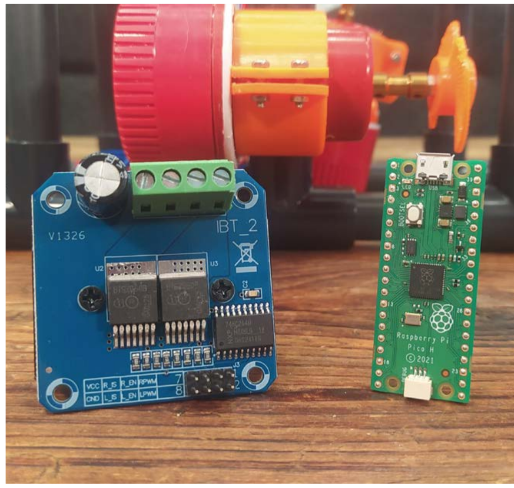
Prior to testing in water, we did some bench-testing of the thrusters to see what kind of amperage they draw at 12V.

▶ **Figure 4:** An example of one of our prototype chassis that is close to one of the standard Seaperch designs



◀ **Figure 5:** Adapting a 1100 gallon per hour 12V bilge pump to become a thruster required cutting some case sections off the device

▼ **Figure 7:** Our plan for motor control primarily consists of pairing a Raspberry Pi Pico with the large and capable BTS7960 motor drivers



Like most brushed DC motors, there is a larger current inrush as the motor starts, which then settles. Our bench-top variable DC power supply tops out at 5 amps and indeed, the motor did momentarily top out our current before settling to around 3 amps in unloaded running. Obviously, compared to the previous TOUV project which we controlled with a Raspberry Pi Pico coupled to some DRV8833 motor drivers, we are going to need much more capable motor drivers. Of course, it's possible to directly switch the motors to a 12V supply, but it's a much finer experience if you have as a minimum some way of setting the speed of each thruster, and at best variable control of the thrusters.

Looking around, we came across the affordable BTS7960 motor control boards which can drive attached motors via an input PWM signal (**Figure 7**). The BTS7960 supports 12V for the motor driving side and the onboard dual H-bridge ICs can switch up to 43 amps. This offers a lot of headroom and should drive our thrusters with a large safety margin. As a bonus, the BTS7960 modules can output a 5V supply and have protection circuits built in, so we can hook up a Raspberry Pi Pico without an additional power supply.

One of the advantages of the tubular chassis design is that it can be assembled and disassembled multiple times and the friction fit of the plastic components is tight enough that it's not necessary to glue the pieces together. In fact, looking around other builds, some people don't ever glue at all. We can imagine iterating on the design and potentially testing the

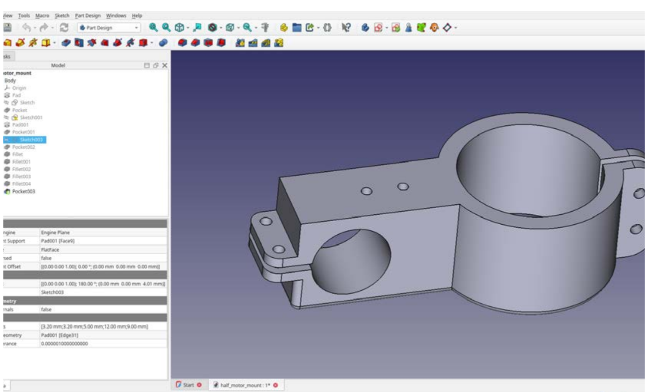
▼ **Figure 6:** A complete thruster with a DIY 3D-printed propeller mounted with an off-the-shelf brass coupler

design in water before maybe using PVC cement to finalise some parts of the chassis. This allows for some different design considerations when it comes to mounting components. As an example, we used FreeCAD to model thruster mount designs, one for the side thrusters and one for the vertical thruster (**Figure 8**). As we can disassemble the chassis, we made a slotted clamp feature allowing them to be slid over an unattached tube section before being bolted into place. If we chose to cement the chassis before fitting these, we would have to design some form of two-piece clamp to allow the mounts to be fitted. Finally, designing with clamps that fit around the standard tube sections allows for flexibility and trimming; we can raise and lower thruster mount points, for example.

As well as designing for trimming and flexibility, when designing somewhat experimental vehicles it's useful to design parts for modularity. For our motor mounts, we envisage that it would be useful to add some form of shroud section that protects the propeller from contact or weeds becoming tangled in it. As this is a part that might need a few iterations, we simply added some holes to the motor mount design to which we can attach designs. This allows for changes in propeller design and any real-world problems encountered in testing and hopefully means we don't have to constantly reprint the basic mount section.

One of the considerations for underwater vehicles, of any scale, is buoyancy. Many examples of Seaperch-style builds add small amounts of foam, often sourced from pool noodles, which are cable-tied to the upper sections of the tubular chassis. This has some merit in that it's pretty simple to do, you can reposition the foam, and also you can add or reduce the amount of foam. One issue reported with this is that if you start to go a little deeper in your underwater explorations, the foam can become compressed due to the increasing pressure and then it becomes less effective.

▼ **Figure 8:** Designing clamp-on thruster mounts in FreeCAD



▶ **Figure 9:** Adding the longer buoyancy tubes; we again used simply designed 3D-printed clamping mounts

When designing somewhat experimental vehicles it's useful to design parts for modularity

With this in mind, another option we have seen is to add larger plastic pipes, with sealed bulkheads, to create less compressible air chambers. In the TOUV build, this concept was used with the tiny 35mm film canisters and a further advantage was discovered that you could 'trim' the balance and buoyancy of the vessel by slightly flooding the canisters as needed. With this in mind, we opted to add two large tube sections sealed with threaded end caps (**Figure 9**). The tubes are 36.5mm diameter and we estimated we might need around 50cm length. Adapter sections are cemented in place as well as the threaded closure receiving sections and then a commercially available end cap with a large rubber washer can be fitted to seal the tubes. Again, this has some advantages in that we can perhaps use weights and pieces of foam inside the tube to add ballast to specific areas to help balance the vehicle, particularly when loaded with payloads such as cameras, etc. Similar to the motor mount clamps, it's reasonably trivial to create 3D-printed clamping mounts for these tubes and these allow for repositioning of the tubes and clamps relative to the vehicle chassis.

With the mechanical design largely in place, we can now start to look at assembling and testing the thruster drive system, a land-side controller, and some kind of wire tether system. Plans are also in place to create a rugged underwater camera from a Raspberry Pi Zero and we have a cunning idea for how to make it incredibly waterproof. Join us in the second instalment of this build as we tackle these and other issues.

Touch sensor quilt block

Quilt yourself a little piece of history and let Raspberry Pi play its part too

Maker

Nicola King

Nicola is a freelance writer and sub-editor. One thing she was not short of for this project was fabric – her stash of all things crafty knows no bounds!

@holtonhandmade

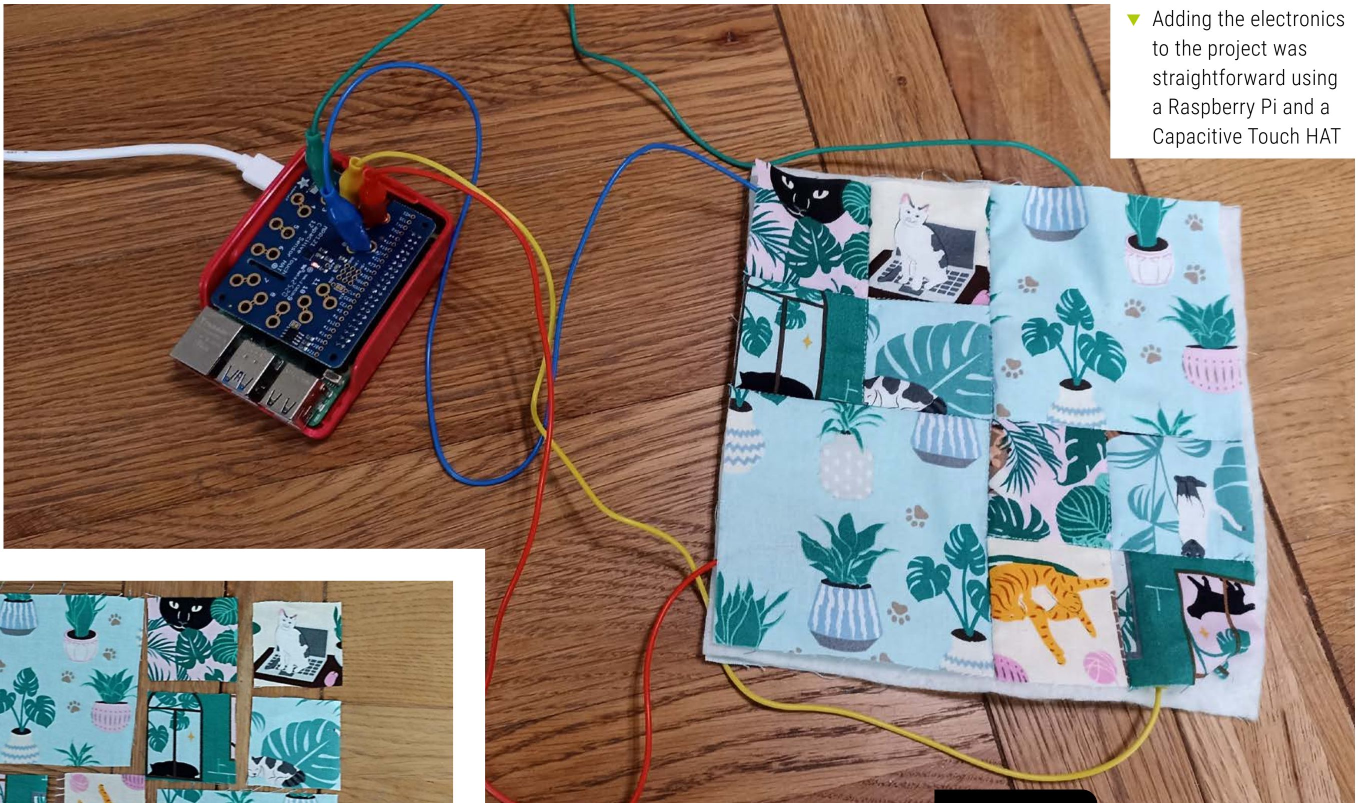
QUICK TIP

Press and starch quilting fabric before you cut into it – it gives stability, control and this also aids precision.

Quilt making has a history that can be traced back many hundreds of years and, as well as being items that provide much-appreciated warmth and comfort to the user, quilts often become cherished heirloom pieces with the unique handiwork, history, and love of the maker literally deeply embedded within them. Early American settlers in particular were highly skilled in quilt making, and sitting together to hand-sew their pieces became something of a social event, with stories shared and friendships strengthened. Today's quilters often employ the same techniques used many years ago, but new and contemporary designs have evolved too, and a quilt can easily become an e-textile project if you want it to be. The Tilt Sensing Quilt (**rpimag.co/tiltquilt**), for instance, uses 41 textile tilt sensors and the maker wrote a basic application that displays the tilt direction of each square as a unique colour.

Incorporating sensors that will trigger a sound when touched

Inspired by quilt-makers both old and new and by the use of technology by some very savvy and current quilt makers, in this tutorial we're going to make a simple quilt block, and we're going to bring this historic hobby bang up-to-date by incorporating some sensors that will trigger a sound when touched. The hope is that you will be inspired to make your own bespoke quilt, maybe even a full-sized version, to deliver warmth plus and maybe a wow factor to boot if you would like to try incorporating sensors too.



Adding the electronics to the project was straightforward using a Raspberry Pi and a Capacitive Touch HAT

The individual fabric elements of the top layer of the quilt. You can really use any fabric, but cotton is a favoured choice for many quilters

Quilt sandwich

So, a quilt is usually made up of three layers: your top layer which is the colourful, carefully planned layer, the middle layer of padding or 'wadding', and the backing to the quilt which is usually in a plain fabric. Use whatever you have to hand for the top and bottom layers – perhaps some old clothes that you want to upcycle, or fabric from a stash. We are using some fat quarters that complement each other, and a plain backing fabric to contrast. We are also using some very simple shapes here for the sake of ease and speed, but the more adventurous among you may want to employ more complicated shapes.

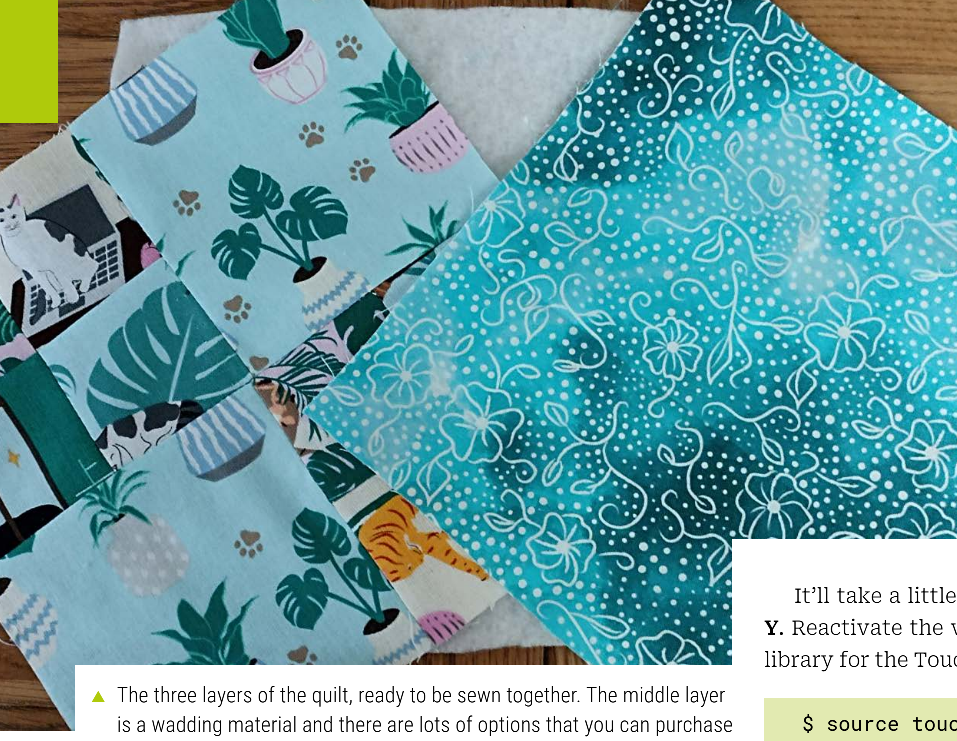
We cut eight 5cm squares in differing fabrics, to use in each of two four-square blocks in our design. (Note that usually quilters measure in inches, but we'll stick with metric here, and also note we'll be using a 5mm seam allowance when we sew.) We also cut two larger 9cm squares in another fabric. We then took two smaller squares and machine-sewed them together along one

QUICK TIP

Accurate cutting and accurate sewing are the keys to points lining up and a professional-looking result, so measure twice, cut once!

YOU'LL NEED:

- Top layer of cotton fabric
- Plain backing cotton fabric
- Wadding layer
- Iron
- Sewing machine
- Hand sewing needle
- Regular thread
- Pins/seam ripper
- Scissors and rotary cutter
- Cutting mat
- Ruler and tailor's chalk
- Raspberry Pi computer
- Adafruit Capacitive Touch HAT, **rpimag.co/touchhat**
- Crocodile clips
- Conductive thread (optional)



▲ The three layers of the quilt, ready to be sewn together. The middle layer is a wadding material and there are lots of options that you can purchase

seam with a 5mm seam allowance, right sides to right sides. We repeated that with two other small squares, then sewed those two strips together (to form a square) along one seam, right sides facing right sides, pressing all seams with an iron as we went, and matching the seams to each other and pinning before joining. We repeated that with the other four small squares, and then took each four-square block and sewed it along one seam to one of the larger 9cm blocks. Then it's just a question of sewing the two pieces, each containing one large square and one four-square block, together to form a final larger square.

Next, we cut a piece of wadding and a piece of backing fabric to the same size as our top layer, and we then stitched all three layers together (with the right side of the backing layer showing on the bottom), and quilted by employing the 'stitch in the ditch' technique, whereby we sewed along the seams or channels that we'd already made on the top layer of fabric to produce one quilted block. Now, let's look at the electronics.

Touch and go

To make our quilt interactive, we'll use touch sensors that will trigger sounds. For this, we opted to use a Raspberry Pi equipped with an Adafruit Capacitive Touch HAT. You will need to use the Blinka compatibility library to make use of the required CircuitPython libraries in Python on Raspberry Pi. In a command-line terminal, we create a new virtual environment, then download and run a setup script:

```
$ mkdir touch
$ python -m venv touch --system-site-packages
$ source touch/bin/activate
$ pip3 install --upgrade adafruit-python-shell
$ wget https://raw.githubusercontent.com/
adafruit/Raspberry-Pi-Installer-Scripts/master/
raspi-blinka.py
$ sudo -E env PATH=$PATH python3 raspi-blinka.py
```

QUICK TIP

As with anything, make sure you have good lighting and are sitting comfortably when you're working.

It'll take a little while and then ask you to reboot, so enter **Y**. Reactivate the virtual environment and install the mpr121 library for the Touch HAT:

```
$ source touch/bin/activate
$ pip3 install adafruit-circuitpython-mpr121
```

Move to the **touch** directory and create a new file:

```
$ cd touch
$ nano touch_test.py
```

We'll now write some code to test the touch inputs on the HAT:

```
import time
import board
import busio
import adafruit_mpr121

# Create I2C bus
i2c = busio.I2C(board.SCL, board.SDA)

# Create MPR121 object
mpr121 = adafruit_mpr121.MPR121(i2c)

# Loop forever, play sounds upon touch
while True:
    # Loop through inputs (0-11) or how many
you need
    for i in range(12):
        if mpr121[i].value:
            print(f"Input {i} touched!")
    time.sleep(0.25) # short delay
```

Before running it, connect some crocodile clips to several of the numbered holes on the Touch HAT. When the code is run – with `python3 touch_test.py` – touching the other end of each cable will trigger a message to say it's been touched.

We'll now sense when an input is touched and play a sound assigned to it. For the latter, we downloaded some Creative Commons-licensed sound effects from **freesound.org**, which we renamed **boing.wav**, **crash.wav**, **horn.wav**, and **meow.wav** and placed in the **touch** directory. The code for our **touch_sounds.py** program is:

```
import time
import board
import busio
import adafruit_mpr121
import pygame
#Initialise Pygame
pygame.init()

# Create I2C bus
i2c = busio.I2C(board.SCL, board.SDA)
# Create MPR121 object
mpr121 = adafruit_mpr121.MPR121(i2c)

# Assign sounds and create list
boing = pygame.mixer.Sound('boing.wav')
crash = pygame.mixer.Sound('crash.wav')
horn = pygame.mixer.Sound('horn.wav')
meow = pygame.mixer.Sound('meow.wav')
sounds = [boing, crash, horn, meow]

# Loop forever, play sounds upon touch
while True:
    # Loop through inputs (0-3) or how many you
need
    for i in range(4):
        if mpr121[i].value:
            print(f"Input {i} touched!")
            pygame.mixer.Sound.play(sounds[i])
    time.sleep(0.25) # short delay
```

Touching the crocodile clip then triggers the connected input on the Touch HAT and plays the assigned sound. We used a Bluetooth speaker for the audio.

We found that our touches were sensed when the crocodile clips were placed under our thin top layer of fabric. If yours is thicker, you may need to use conductive thread to connect the clips to the surface, but take care not to create a short-circuit otherwise it'll register phantom presses.

Of course, you could also use touch inputs to trigger any other action Raspberry Pi can perform, such as lighting LEDs. So, there are plenty of possibilities to get creative!

▲ The back layer after being quilted using the stitch-in-the-ditch method. The back can be a plain fabric if you wish

QUICK TIP

If you have a 'walking foot' for your sewing machine, use it when quilting – this helps pull through all the layers of fabric at the same time.

The mathematics of quilting

When you make and sew anything with fabric, a certain amount of maths is always required, so that you measure and cut the correct amount of fabric for example. However, when it comes to quilting, this author would argue that the mathematics involved is more complex, and geometry sits at the heart of it all. If you take a look at many quilts, more complex than our own simple block, you can see the precision of the angles within the blocks, the symmetry – these are things that quilters strive to achieve and this takes planning and accuracy. Triangles, hexagons, chevrons, whatever repeating shapes a quilter is employing within their design, numbers are a part of the art.

The Fibonacci sequence and the golden ratio are just two mathematical concepts that some quilters use when making very intricate quilts (**rpimag.co/quiltgeometry**), and once you fall down this rabbit hole, you'll want to keep exploring.

QUICK TIP

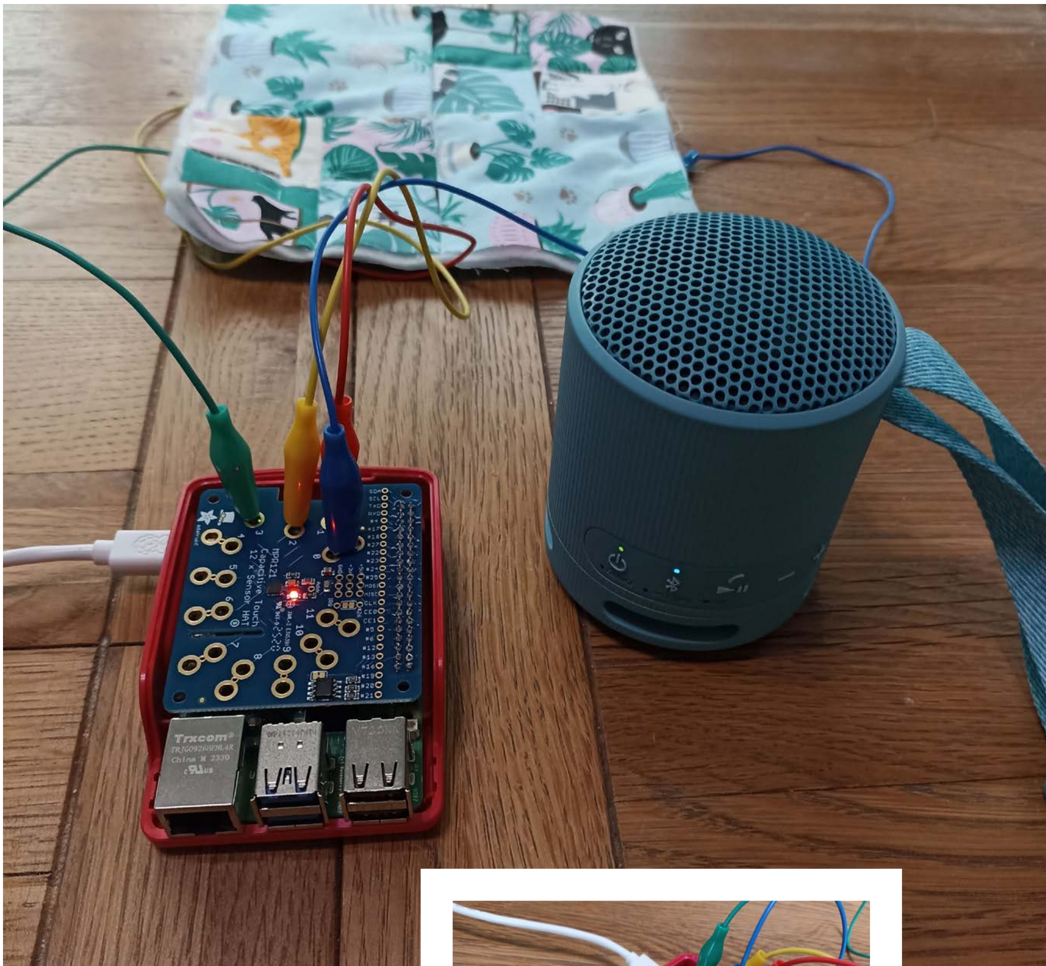
Graph paper is a handy tool when planning – draw the pattern on it, and use it to help calculate the amount of fabric required.

Innovative quilting projects

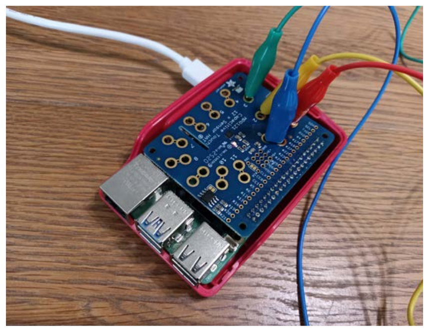
Here are two more examples of very original quilt projects that are taking the concept of quilting in a fresh and exciting direction:

1. Made back in 2013, this beautiful interactive quilt (**rpimag.co/interactivequilt**) in Australia has been purposed as a huge wall hanging, and has incorporated electronics, including sensors and LEDs, to fully illuminate the amazing sewing work of the makers. Made by a team of individuals who each worked on their own sections, when touched, the interactive panels react in a very illuminated fashion!

2. This clever and creative project (**rpimag.co/newmumsquilt**) led by Swansea University brought together mothers of young babies so that they could explore their experiences of feeding. Pieces of their babies' clothing were stitched together to create an interactive quilt, and digital sensors were also incorporated. When the sensors are pressed, audio clips play and reflect the mothers' feelings around and experiences of feeding their babies, which can often be a challenging and testing part of motherhood. Quilted spiral shapes, often seen on Welsh quilts, have been included to reflect a new mother's spiralling emotions.



▲ We connected a Bluetooth speaker to output the audio samples triggered by the touch sensors



► We used a Capacitive Touch HAT, but the same MPR121 touch sensor chip is also available on a breakout board that could be used with Raspberry Pi or Pico

A quilting quest

Whether you want to hand-sew or machine-sew a whole quilt or just a smaller block, this is a pastime that offers so many possibilities, and you can literally create any design you want, constructing a unique and beautiful piece of art in the process. If you would like to learn more, here's some inspiration:

1. For a contemporary take on quilting, this is a great read and shows how quilting can even be a social commentary: **rpimag.co/vamquilters**.

2. If you want to feel part of a community, how about joining a group where you can find support on the journey (**quiltersguild.org.uk**), or find a group local to you? You can impress with your e-textile knowledge and change some perspectives!

3. For a one-stop shop guide to quilting, find yourself a 'bible' that quite literally lays it all out for you, like this one: **rpimag.co/quiltbible**.

Part 01

Object design with Python in FreeCAD

Use your programming skills to make printable 3D designs

Maker

Rob Miles

Rob has been playing with software and hardware since almost before there was software and hardware.

robmiles.com

DOWNLOAD THE FULL CODE:

rpimag.co/pythonfreecad

In these articles, you'll create 3D objects from a Python program running inside the FreeCAD application. This is a great way to take your programming skills into another dimension. You can find all the sample programs in the GitHub repository: **rpimag.co/pythonfreecad**.

Figure 1 shows a super-secret encoder made from three panels with holes in them. Individually each panel is meaningless, but stack the three together and you can read the secret message through the open holes. This would be hard to design by hand, but if you use a program to place the holes it gets a lot easier. Let's find out how to do this.

Get FreeCAD

FreeCAD will run on a range of computers, including Raspberry Pi 4 and 5 (and 400, 500, and 500+). It is a large application and likes having plenty of memory. There are installers for Windows, Mac and Linux. To install FreeCAD on a Raspberry Pi, you'll have to do a bit of work at the command prompt:

```
 $ sudo apt update
 $ sudo apt install -y flatpak
 $ sudo flatpak remote-add --if-not-exists
flathub https://dl.flathub.org/repo/flathub.
flatpakrepo
 $ sudo flatpak install -y --system flathub org.
freecad.FreeCAD
 $ flatpak run org.freecad.FreeCAD
```

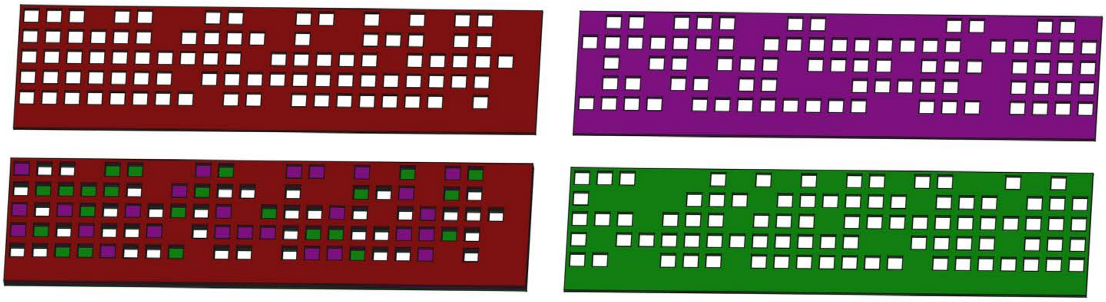


Figure 1: You may have to squint a bit to see the word here (bottom left), but the encoder works very well in physical form

The above commands install Flatpak and FreeCAD and then runs FreeCAD. FreeCAD links are added to the Education and Graphics categories in the applications menu. The first time FreeCAD runs, you will be invited to set some defaults. Change the navigation style in the bottom left of the application to 'OpenInventor'. This makes diagrams much easier to navigate, especially if your mouse has a clickable scroll wheel. When the defaults have been set, the application starts.

> **QUICK TIP**
>
> Flatpak is a Linux app packaging and sandboxing system used to ship applications with their dependencies so they run consistently across different distributions.

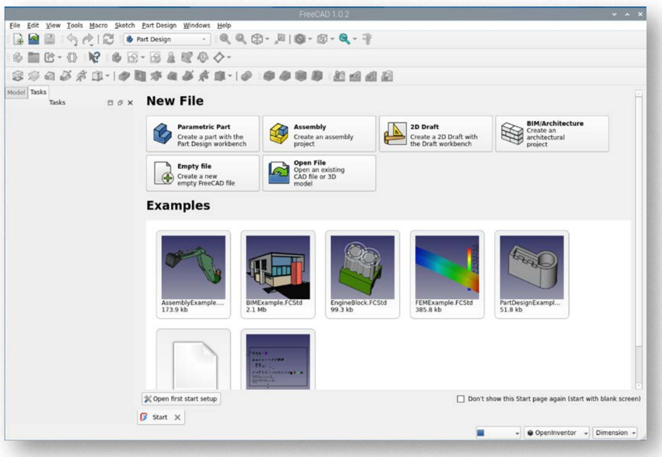


Figure 2: The examples that are presented (and the startup image) will change from time to time.

Figure 2 shows the FreeCAD start screen. You could use FreeCAD to design a building and everything in it, right down to the light switches. The user interface reflects this. Your author doesn't know what many of the things on this page do, but he does know how to start writing Python macros in FreeCAD, and that is what we are going to do.

> *In FreeCAD, macros are recorded as sequences*

Macro magic

One of the most powerful features of FreeCAD is macros. A macro is a recorded sequence of actions. Perhaps you are making a grille for a loudspeaker, and you need to put lots of holes into a panel. Rather than having to perform the steps to cut every hole, you instead record the hole-cutting actions into a macro and then use that every time you want a hole.

In FreeCAD, macros are recorded as sequences of Python statements. When you do something in the user interface, Python functions are called to perform the action. The FreeCAD macro recorder records these Python calls into a macro file. The Macro dialog is selected from the toolbar at the top of the FreeCAD window.

> **QUICK TIP**
>
> The KiCad circuit board design tool also uses macros written in Python.

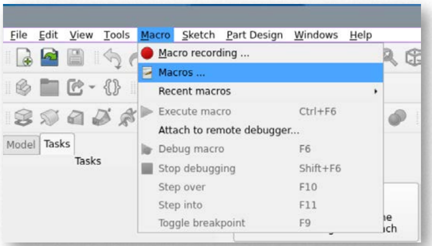


Figure 3: If you select Macro recording, you will be prompted to enter a filename into which a macro will be written

Figure 3 shows the Macro menu in FreeCAD. You can also start a macro recording and select from recently used macros. If you click Macros… you will open the 'Execute macro' dialog.

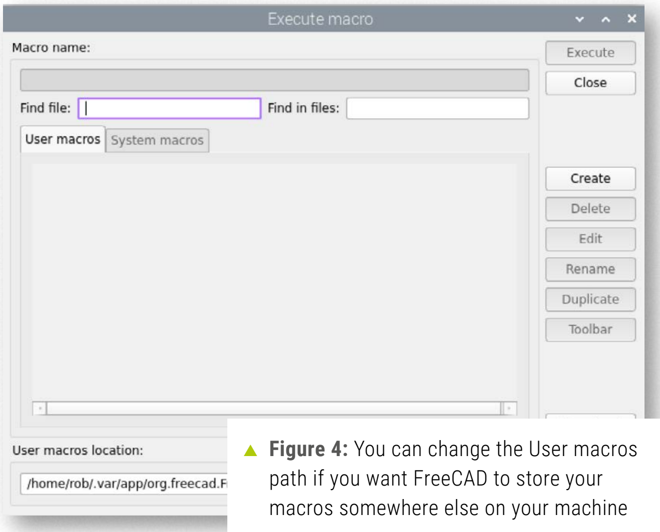


Figure 4: You can change the User macros path if you want FreeCAD to store your macros somewhere else on your machine

QUICK TIP

FreeCAD macro files are stored with the language extension .FCMacro, but you can also use the File menu to load Python programs with the .py extension.

Figure 4 shows the 'Execute macro' dialog. If you click Create, you can start making a macro. You will be asked to give a filename, perhaps 'First Panel', and then you will arrive in a text editor where you can enter Python code.

The first thing to do is import some libraries:

```
import FreeCAD as App
import FreeCADGui as Gui
import Part
import random
```

The `FreeCAD` library manages your documents, `FreeCADGui` manages the display, and `Part` creates parts. Now we have our libraries we can create a new document:

```
doc = App.newDocument("First Panel")
```

FreeCAD organises drawings in documents. A document can contain different part designs which will be drawn together. Let's make a panel and put it in the document:

```
width = 100
depth = 50
height = 5
panel = Part.makeBox(width, depth, height)
```

The statements above set the width, height, and depth of the panel and use the `makeBox` method from the `Part` object to create a panel which is 100mm wide, 50mm deep, and 5mm high. Now that we have a panel, we can add it to the document and display it.

```
frame_obj = doc.addObject("Part::Feature",
"Panel")
frame_obj.Shape = panel
Gui.ActiveDocument.ActiveView.fitAll()
```

The statements above add a new object called Panel to the document, set the Shape component to the panel we just created and then scale the display.

Figure 5 shows the application ready to run inside FreeCAD. We can run it by selecting Run from the Macro menu or by holding down **CTRL** and pressing **F6.**

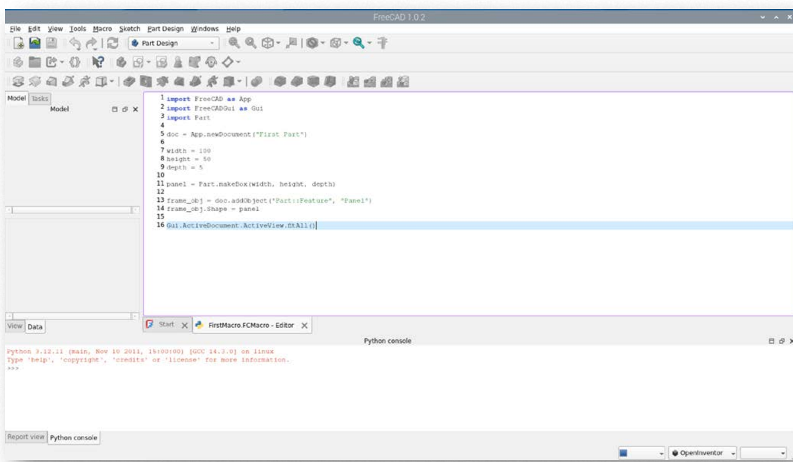


Figure 5: The Python Console and the Report View panels are opened from the View > Panels menu

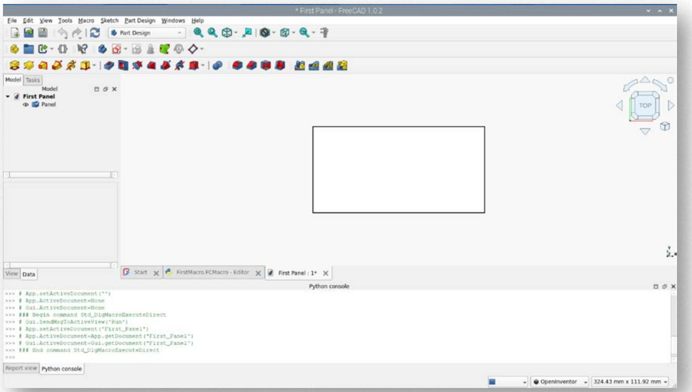


Figure 6: Click the Model tab in the data view on the left to see all the elements in a document

Figure 6 shows the output from the program. We are viewing the panel from the top, so it just appears as a rectangle that is 100mm wide and 50mm high. You can see the object in 3D by left-clicking on the image and then dragging the mouse with the left button held down to rotate it.

Axes of evil

Figure 7 shows the panel in 3D. It is important that you understand how the X, Y, and Z axes work in FreeCAD drawings. If you have drawn 3D graphics on your computer, you may be used to the X axis going across the screen, the Y axis going up the screen, and the Z axis going into the screen. This is not how FreeCAD sees the world.

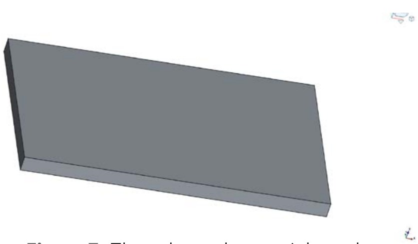


Figure 7: The cube at the top right and the axes at the bottom right show the orientation of the view

Instead, the X axis goes across the screen, the Y axis goes into the screen (hence the name depth) and the Z axis goes up from the floor (hence the name height). If you have ever done any 3D printing, you will be used to locations on the print bed being expressed using X and Y, and the height of the print head being expressed using Z. When we export the panel design, this is how it will work.

Coordinates increase going to the right, into the screen, or going up. When we created the panel, the bottom left-hand corner of the panel is at the coordinate (0,0,0). Coordinates can go negative. There is no such thing as ‘clipping’ in the coordinate space: if you want to centre your designs about the (0,0,0) coordinate, this is perfectly fine. Shapes can be translated to new positions; we will do this later.

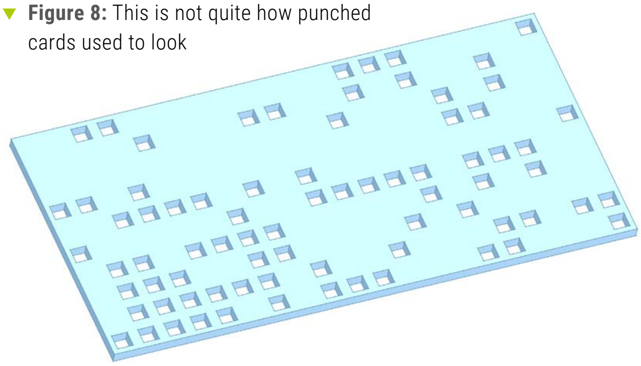
Exporting your designs for printing

To print your designs, you export them from FreeCAD as STL (stereolithography) files. Select the object in the model view and then select Export from the File menu. Give a name for the file and save it.

At the moment, it doesn’t look like running Python in FreeCAD is much better than creating elements by hand. So, let's make something which would be very hard to do without some help from software.

Figure 8 shows a panel with a set of randomly punched holes in it. You could use this as a super-secure ID card. Only a holder of a card with exactly the same pattern would be allowed into your treehouse or other secure location. We start by defining the dimensions of the card we are creating:

Figure 8: This is not quite how punched cards used to look



```
width = 100
depth = 50
height = 2
```

The statements above define the card we are making. Our card will be 100mm wide, 50mm deep, and 2mm high. Now let’s specify the hole arrangement:

```
rows = 10
cols = 20
hole_margin = 1.0
```

The statements above indicate that there will be 10 rows, each of 20 columns. There will be a 1mm margin around each hole.

Now let's compute some values for our program to use:

```
hole_x_spacing = width/cols
hole_y_spacing = depth/rows
```

The two variables created by the statement above hold the spacing of the holes in the card. We divide the width of the card by the number of columns, and the height of the card by the number of rows.

```
hole_width = hole_x_spacing-(2*hole_margin)
hole_depth = hole_y_spacing-(2*hole_margin)
```

These two statements calculate the width and depth of the holes we are going to cut (remember that depth gives the value of y and this dimension goes into the screen). We have a margin around each hole. The holes will be the same height (the value of z) as the panel (because we are cutting a hole in it). It's easy to get height and depth confused. Go back and read 'Axes of evil' if you find this confusing. The final variable we are going to create will determine how many holes we are going to get in the card:

```
hole_chance = 0.4
```

At each hole position, the program will generate a random number between 0 and 1. If the number is less than `hole_chance`, the program will make a hole. Therefore, the bigger the value in `hole_chance`, the more likely the program is to make holes. Now we can start making things. First let's make our panel, ready to have some holes punched in it.

```
panel = Part.makeBox(width, depth, height)
```

Now that we have our panel, let's write a function to cut holes in it:

```
def make_hole(x,y):
  global panel
  hole = Part.makeBox(hole_width,hole_
depth,depth)
  hole_pos = App.Vector(x,y,0)
  hole.translate(hole_pos)
  panel = panel.cut(hole)
```

The `make_hole` function is provided with the x and y coordinates of the hole to be cut. It creates a box called `hole`, translates the hole to the required position on the card and then uses the cut operation on the panel to create a new panel with a hole in it and assign panel to the newly cut version.

The distance of a translation is specified by a vector, which is an object that contains x, y, and z distances to move the shape. We want the hole to be in the position specified by x and y, so we create an `App.Vector` object that contains these values. We don't want the object to move up or down when it is translated, so the value of z in the vector is 0.

The `make_hole` function contains a `global` statement so that it can write to the `panel` variable which is declared outside it. The next part of the program works through the hole positions and cuts random holes:

```
for col in range (0, cols):
  for row in range(0,rows):
    if random.random()<hole_chance:
      x = col*hole_x_spacing+hole_margin
      y = row*hole_y_spacing+hole_margin
      make_hole(x,y)
```

The two outer `for` loops take the values of `col` and `row` through all the hole positions. At each position the program decides whether to make a hole. If a hole is required, the program works out the hole position and calls `make_hole` to make the hole. Once the loops have finished, we have a punched card containing a random arrangement of holes. Each time you run the program, you will get a different arrangement.

Next time, we will expand our knowledge to create complex objects from code.

Keeping secrets

The ability to cut holes can be expanded to make the secret encoder program. This contains a set of font designs. It uses the fonts to cut character shapes out of multiple panels and makes sure that the only holes in common across the panels are part of the secret text. You can find the code here: **rpimag.co/pythonfreecad**.

Unusual tools: degaussing tool

Tools that detract from your natural magnetism

Maker

Dr Andrew Lewis

Andrew is a specialist maker and fabricator, and is the owner of the Andrew Lewis Workshop.

lewiswork.etsy.com

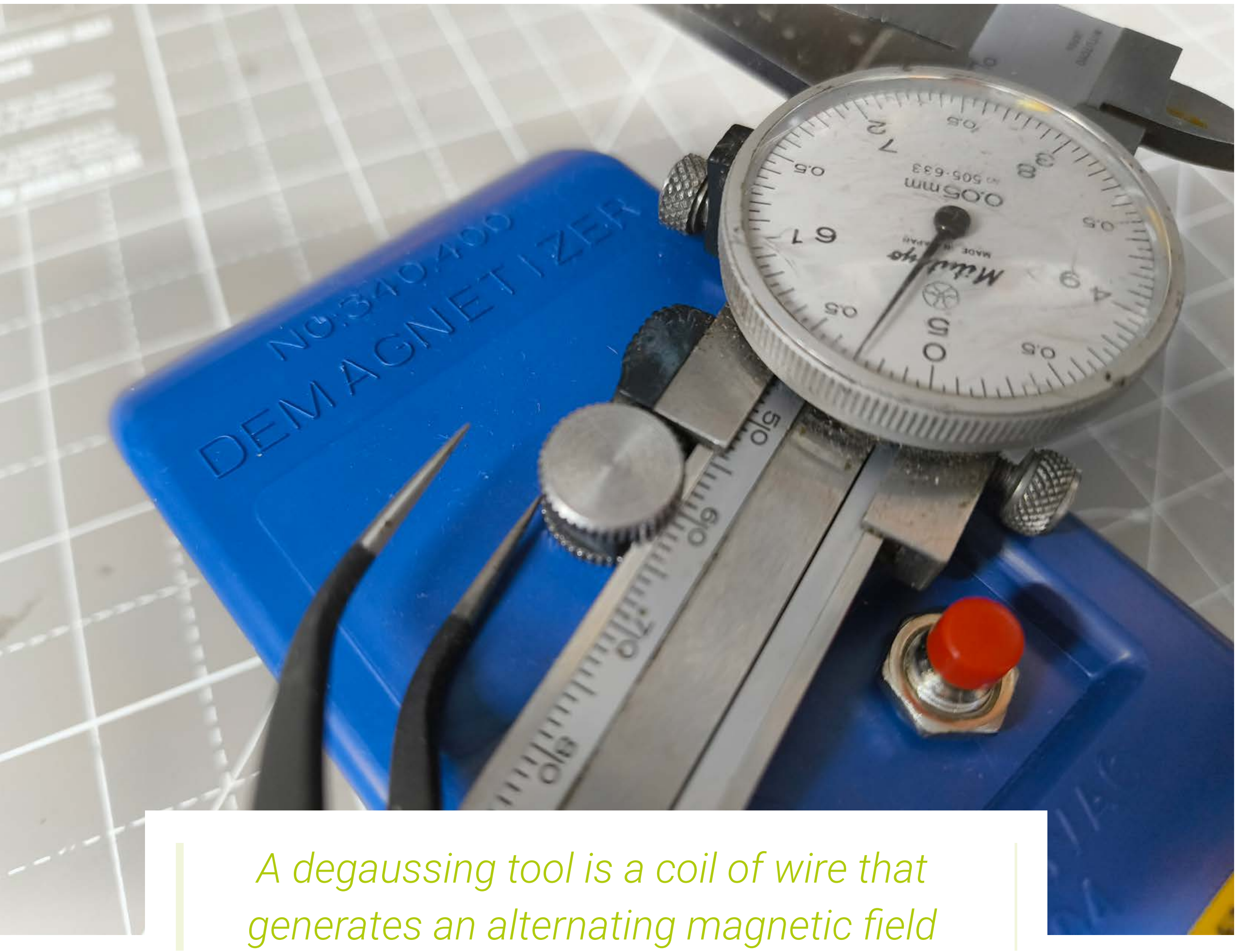
If you're old enough to remember when computer monitors were big boxes that took up an entire desk, then you might also recall that some monitors had a tiny button with 'degauss' printed next to it. Pressing the button would make the screen distort and produce a satisfying whomping sound from somewhere inside the box. Those monitors had a built-in degaussing coil to demagnetise the chassis and prevent distortion on the screen. Most large colour TV sets had them too, and they would fire as soon as the TV was powered on. Other monitors didn't have a built-in degaussing coil and would have to be degaussed manually using an external coil or wand. When technology moved over to LCD monitors, there was no more need for degaussing and the magic of the degaussing button was lost to a generation.

QUICK TIP

Check for residual magnetism with a Hall effect probe, or slide a compass close to the object and see whether it deflects.

Degaussing tools come in all shapes and sizes. In this photograph, you can see an entire ship wrapped with degaussing coils to neutralise residual magnetic fields. **Image:** OlegErm, CC BY-SA 4.0

Degaussing isn't just something that happened to CRT monitors back in the day. There are a surprising number of situations where degaussing tools are needed, and they are used in everything from watchmaking to marine engineering. The modern world is full of magnetic fields, and precision instruments like mechanical wrist-watches and measuring tools can be affected by all of that electromagnetic chaos. When the parts of a watch get magnetised, it can cause the finely balanced mechanisms to run at the wrong speed and stick against each other. Tabletop degaussing tools for watches are becoming increasingly common, and they can also be used to demagnetise other small objects like micrometers and screwdrivers. Techniques vary, but while holding down the button, you move the object into the field from the side, then lift it upwards while also moving it in circles parallel to the desktop.



A degaussing tool is a coil of wire that generates an alternating magnetic field

▲ Tabletop degaussing tools are becoming a more common feature in small workshops. They can be used to demagnetise precision tools, ruin all of the swipe cards in your wallet, or set fire to the desk if you accidentally rest something on the power button

QUICK TIP

There are two main types of degaussing tool. Wands offer more precise control for degaussing things like tape heads, while coils are more of a scatter-gun for things like CRT monitors and arcade machine chassis.

In the simplest form, a degaussing tool is a coil of wire that generates an alternating magnetic field that demagnetises ferrous metals. The practical reality of the situation is a bit more complicated. Flipping a magnetic field at 50 or 60Hz will scrub any existing field, but when you turn the coil off, you probably won't exactly hit the zero-crossing point of the sine wave and the ferrous metal will still have some residual magnetism. To reduce this effect, you can either use a very fancy degaussing tool that can control the power and frequency of the field (sometimes called a pulse degausser), or you can do the traditional 'degaussing dance' and use the fact that the power of the field reduces as distance from the coil increases. Moving the coil in small circles around the area you are demagnetising (or moving the object itself, if the coil is fixed) will help create an even effect, and then increasing the distance between the coil and the object at the same time will reduce the overall strength of any residual field.

QUICK TIP

If you need a degaussing tool in an emergency, you could try using an old mains-powered electric drill. Place the drill at right angles to what you're trying to degauss, and pull the trigger while moving the drill away. The field generated by the motor coils should do the trick.

Warning!

Magnetic fields

Degaussing tools create strong magnetic fields that can wreak havoc on sensitive equipment, magnetic recording devices (like hard disks and tapes), and medical devices like pacemakers. Using a degaussing coil or wand for longer than its recommended safe duty cycle might lead to unexpected smoke or flames. Unplug, disconnect, and remove batteries from equipment before you degauss it, or you will almost certainly cause permanent damage.

Part 01

Run a local LLM on your Raspberry Pi

Deploy a large language model and keep your chat private and secure. By **Lucy Hattersley**

Maker

Lucy Hattersley

Lucy talks to her Raspberry Pi all the time, so it's about time it started talking back.

rpimag.co

YOU'LL NEED

- Raspberry Pi 4 or 5
- Internet connection
- 4GB RAM (preferably 8GB or 16GB RAM)
- 8GB or more storage

Large language models (LLMs) are the public face of AI. While there are linear regression models like OLS (ordinary least squares), decision trees like CART (classification and regression), and CNNs (convolutional neural networks) like MobileNet, few people outside of AI have heard of such technologies.

Everybody has heard of the large language model GPT (generative pre-trained transformer), in particular the chat application built on top of it: ChatGPT.

So welcome to the public face of AI. In this tutorial, we are going to explore running large language models on Raspberry Pi hardware.

It is possible to run an LLM on Raspberry Pi hardware, even on a Raspberry Pi 4 with just 4GB of RAM, although you will be limited to some of the very small models. On the latest Raspberry Pi 5 model with 16GB of RAM, you have plenty of scope to run larger models, and you will be pleasantly surprised by the results.

The good news is that setting up an LLM on Raspberry Pi is remarkably easy.

From install to Hello, LLM

We are going to use an open-source platform called Ollama. This software makes it easy to download, install, and run AI language models.

Open a terminal window and enter:

```
$ curl -fsSL https://ollama.com/install.sh | sh
```

This will install Ollama to the **/usr/local** directory. Check that it is installed with:

```
$ ollama --version
```

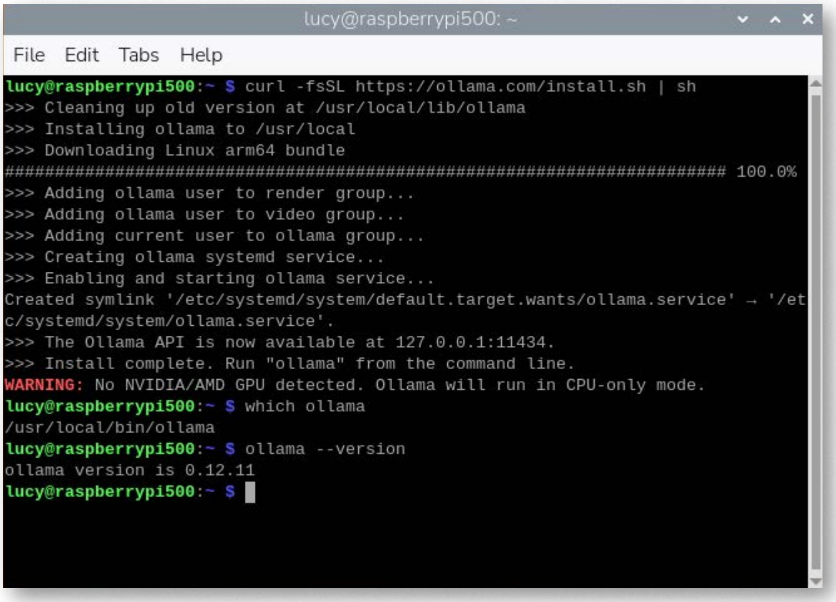


Figure 1: Check that Ollama is installed and running correctly

GDPR compliant

Ollama models can operate fully offline, enabling them to be private, secure, and compliant with privacy regulations like GDPR. This makes it ideal for AI processes such as legal, healthcare, or finance where you must process information locally and ensure GDPR data compliance.

The LLM does not see words. It sees tokens, which are numbers underneath the hood representing parts of the words

Check that it is running (**Figure 1**) with:

```
$ systemctl status ollama
```

Install your LLM

Let's install our first LLM, we do this with the pull option. Enter:

```
$ ollama pull llama3
```

This connects to Ollama's model registry and downloads the llama3 model files into your Raspberry Pi Ollama installation.

Start your LLM

Start your chat:

```
$ ollama run llama3
```

Enter a question and press **RETURN** (see **Figure 2**). For our test, we are using the same question in various different models:

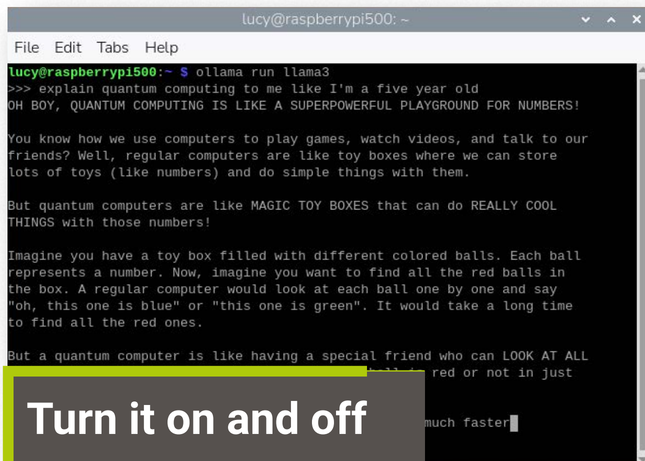


Figure 2: Asking the LLM a question

Turn it on and off

You can stop and start the Ollama service with:

```
$ sudo systemctl stop ollama
```

```
$ sudo systemctl start ollama
```

```
 > explain quantum computing to me like I'm a
five year old

 OH BOY, QUANTUM COMPUTING IS LIKE A
SUPERPOWERFUL TOY!
 You know how we use regular computers to play
games and watch videos? Well, quantum computers
are special machines that can do some really
cool things that regular computers can't.
 Imagine you have a toy box filled with
different colored blocks. Each block has a
special letter or number on it, like A, B, C, or
1, 2, 3. You want to find all the blocks with
the letter "A" on them, but they're all jumbled
up in the box.
 A regular computer would look at each block one
by one and say, "Oh, this one has an 'A'! And
this one has a 'B'... And this one has another
'A'!" It would take a long time to find all the
blocks with the letter "A".
 But a quantum computer is like having a magic
friend who can look at ALL THE BLOCKS AT THE
SAME TIME! It's like having a superpower that
lets you see everything in the box all at once.
 This magic friend, called a "qubit" (say it
like "cue-bit"), can be many things at the same
time. It's like being both a boy and a girl, or
both tall and short, ALL AT ONCE! This means the
quantum computer can solve problems way faster
than regular computers.
 For example, imagine you have a big list of
numbers to add up. A regular computer would do
each addition one by one: 1+2=3, then 3+4=7, and
so on. But a quantum computer could do ALL THE
ADDITIONS AT THE SAME TIME! It's like having a
magic calculator that can solve all the math
problems in your head!
 So, quantum computing is like having
superpowerful friends who can help us solve
really hard problems way faster than we ever
thought possible!
```

When the LLM has finished, you can ask another question. Bear in mind there are many caveats with using LLMs locally on your machine. Unlike online services such as ChatGPT and Perplexity, your local model does not have access to the web, so its information will be limited to a cut-off date. This is the date when the model was trained. You can often ask the LLM directly what this date is.

```
 > what is your cutoff date
 I'm a large language model, my training data
cutoff date is December 2021.
```

You are also limited to entering text. The LLM will not work with images, files, or other data sources. And like all LLMs, it is prone to hallucinations, so take what it says with a pinch of salt and be sure to cross-reference important information.

When you are finished with asking questions of your LLM you can return to the command line with: `/bye`(or press **CTRL+C**).

Check the options

Enter `/?` at the prompt to take a look at all the other options available to you.

Verbose mode

Now that you have had a chance to look at your first model, it's time to take a look at what other LLM models are available to you. There are plenty of models available in Ollama and even more available online using services like HuggingFace.

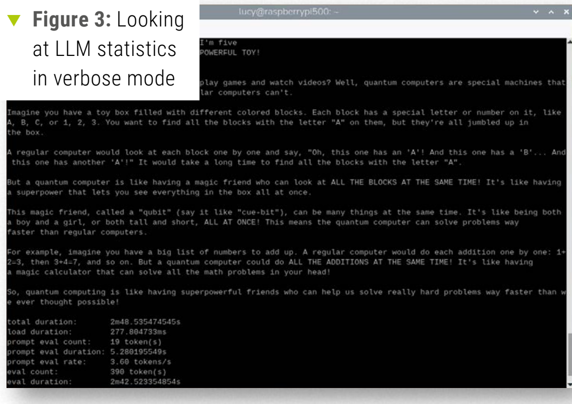


Figure 3: Looking at LLM statistics in verbose mode

But before you start to explore models, it's a good idea to get some feedback on the performance of what you are using. For this you'll want to start your model in verbose mode, like this:

```
$ ollama run llama3 --verbose
```

Now when you ask a question you will get the response, and at the end some statistics, as shown in **Figure 3**:

```
total duration: 2m48.535474545s
load duration: 277.804733ms
prompt eval count: 19 token(s)
prompt eval duration: 5.280195549s
prompt eval rate: 3.60 tokens/s
eval count: 390 token(s)
eval duration: 2m42.523354854s
eval rate: 2.40 tokens/s
```

The LLM will not work with images, files, or other data sources

Here is what these statistics mean:

- **total duration.** The entire time from the moment you ran the command to the finished output.
- **load duration.** How long it took Ollama to load the model into memory.
- **prompt eval count.** This is how many tokens your prompt contained (in our case 19).
- **prompt eval duration.** How much time Ollama spent reading and processing the 19 tokens.
- **prompt eval rate.** The rate at which the tokens were processed. Smaller models run faster.
- **eval count.** How many tokens Ollama generated in response (390 in our case).
- **eval duration.** The time it took to generate the tokens.
- **eval rate.** The model's generation speed. This was about 2.4 tokens per second on our test Raspberry Pi 500+.

Armed with this information, you can start to compare models and decide what you want. All models are a trade-off between speed and size. The larger the model, the more accurate (or intelligent) it is; and the slower the eval rate at which it generates tokens will be.

What is a token?

LLMs break up your prompt into tokens using a tokenizer. A token usually represents around three to four characters of English text. This includes spaces (which we are representing here as underscores) and special characters. So, our prompt "explain quantum computing to me like I'm a five year old" is broken up by the tokenizer and processed like this:

```
_ex
pla
in
_quant
um
_comput
ing
_to
_me
_like
_I
'
m
_a
_five
_year
_old
```

The LLM does not see words. It sees tokens, which are numbers underneath the hood representing parts of the words. Each token is mapped to a vector and fed through a transformer layer that has been trained to look at the relationship between different tokens. The model then generates tokens (390 in our response) based on the most likely next token and continues until it predicts an end-of-sentence token or hits a stop condition.

Strawberrry!

Understanding tokens is the key to breaking the mystical spell of large language models and understanding their limitations. You should know that an LLM doesn't understand words: it breaks text into tokens and use maths underneath the hood to spit out the most likely next token until it hits a stop condition. This is why asking some LLMs "how many Rs are there in strawberry" gets a comedic response of 2. Or in some cases insistence that it is spelled "STRAWBERY". The model does not understand the word. This is also the clue behind hallucinations, as for some training quirks it is spitting out tokens that make sense mathematically but are not located in reality.

Look at the model

Our Llama3 model runs on Raspberry Pi 500+, but at two tokens per second you'd be hard pushed to call it speedy. Fortunately, a lot of work has been done on models since then, and you can get a great speed boost by checking out some of the more recent models.

First, let's take a look at the model information. Back in terminal enter:

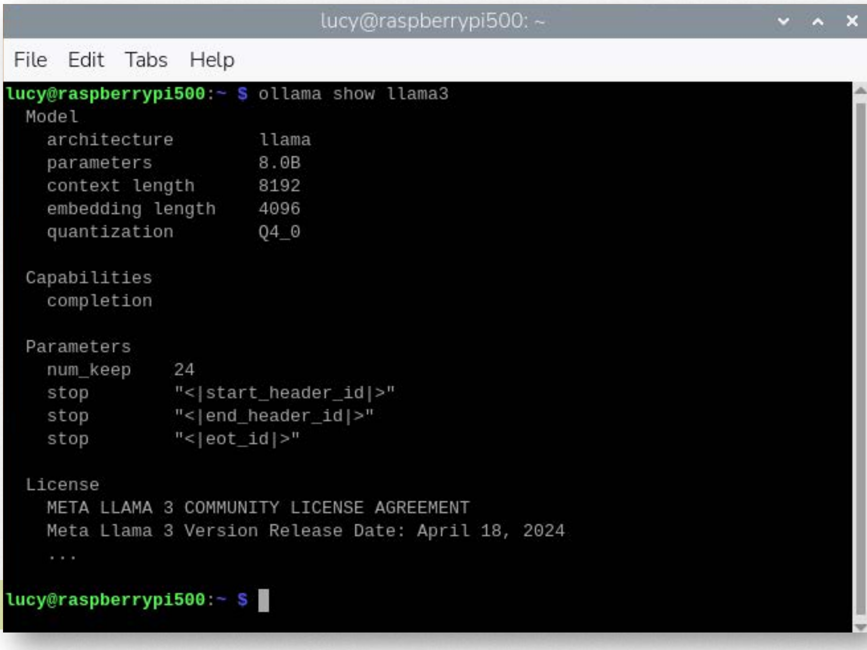
```
$ ollama show llama3
```

This will show model information (**Figure 4**):

```
Model
    architecture        llama
    parameters          8.0B
    context length      8192
    embedding length    4096
    quantization        Q4_0
```

Let's break this down:

- **architecture:** llama. This is the family tree. In this case META's LLaMa design.
- **parameters:** 8.0B. The number of weights inside the model. In this case we have 8 billion. More parameters means a richer internal world, but slower speed. Our Raspberry Pi just about runs but is occasionally a bit wheezy. Fewer parameters means faster speeds.
- **context length:** 8192. This is the model's working memory. The model can keep 8192 tokens (roughly six thousand words) in memory. Anything older than that falls off its mental chalkboard.
- **embedding length:** 4096. This is the dimensions of the model's thoughts. It converts the token into a 4096-space. More embedding means more expressive, though.
- **quantization:** Q4_0. This is where the model's numbers are compressed to save memory. Q4_0 means each weight is stored in 4-bit precision (from its original floating point during training). Quantization makes the model faster at the cost of some precision.



▲ **Figure 4:** Model information

Altogether, our model has a lot of parameters and we could run a much faster, and more modern model to get similar results at a higher speed. So let's see what newer models are available.

Ollama library

Open your web browser and head to **ollama.com/search**. Here you'll find a library of models with a default list of the most popular (**Figure 5**).

Pay attention to the number of parameters next to each model. The popular gpt-oss model by OpenAI is only available in 20b and 120b versions, both of which will be achingly slow without acceleration.

We should stick with models that are 8b or less. Let's take a look at one of the lightest modern models around: 'DeepSeek-R1'. Search for it (or find it in the list) and click to get more information, including a list of available models.

You'll typically see models like:

- deepseek-r1:latest
- deepseek-r1:1.5b
- deepseek-r1:7b
- deepseek-r1.b (latest)
- deepseek-r1:14b
- deepseek-r1:32b
- deepseek-r1:70b
- deepseek-r1:671b

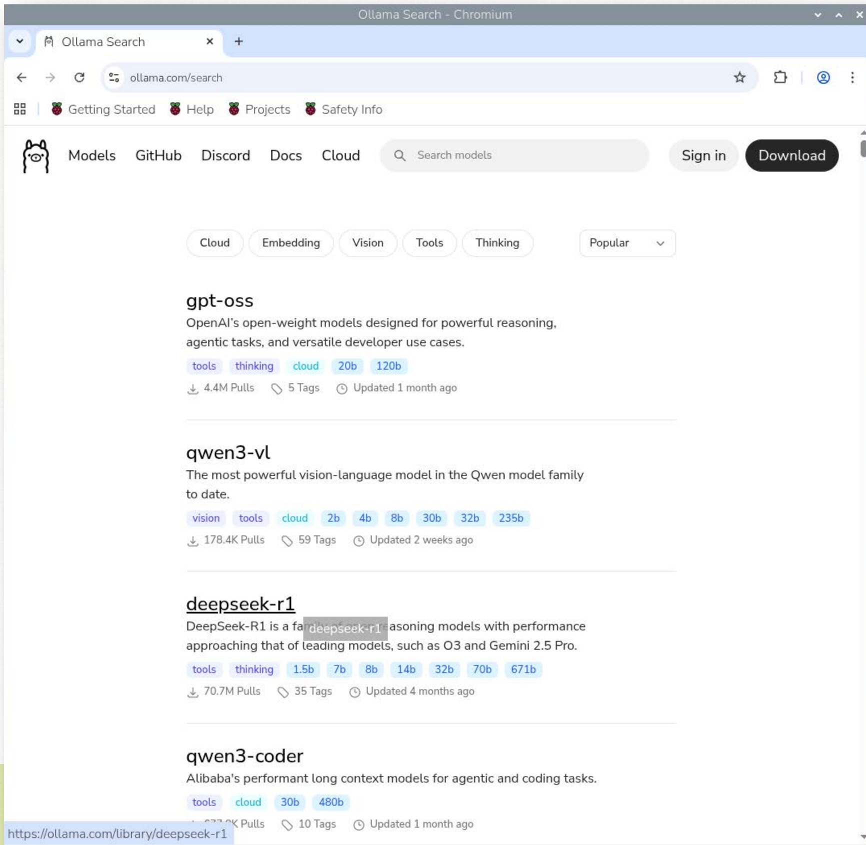


Figure 5: Ollama's model library

The 'latest' option is what is typically installed if you use the standard `ollama pull deepseek-r1` command. But you can also specify specific versions to pull and run. Let's see how the very lightest model performs:

```
$ ollama pull deepseek-r1:1.5b
$ ollama run deepseek-r1:1.5b
--verbose
```

Now enter your prompt again: "explain quantum computing to me like I'm a five year old". You'll find that DeepSeek-R1 with just 1.5 billion parameters runs much faster. Our output was 8.43 tokens/s. However, its explanation was much more limited than the Llama3 model. After some time spent showing us its thinking, it said:

```
In summary, quantum computing offers a
fundamentally different approach to  processing
information using qubits and entanglement.
While still  theoretical and challenging, it
has the potential to solve certain  problems
exponentially faster than classical computers,
though building  and maintaining large-scale
systems remains an ongoing goal.
```

Which doesn't quite hit our brief of "like I'm a five year old".

We can test out other models using `pull` and `run`, such as deepseek-r1:8b and phi3:3.8b. These offer a great combination of speed and functionality.

Models to explore

Here are some large language models to explore with Ollama:

- DeepSeek-R1 (1.5b, 7b)
- Phi3 (3.8b)
- Llama3 (8b)
- Llama3.2 (1b, 3b)
- TinyLlama (1.1b)
- Gemma 2 (2b)
- Qwen 2.5 (3b)

Create an AI dataset

Streamlining dataset creation for AI development with the IMX500 sensor on Raspberry Pi AI Camera

Maker

Lucy Hattersley

Lucy is the editor of Raspberry Pi Official Magazine. She loves a good dataset.

rpimag.co

YOU'LL NEED

- Raspberry Pi
- Raspberry Pi AI Camera
- Second computer
- Local network

Starting an AI project often begins with building a quality dataset, which can be a complex and time-consuming task. This dataset contains the data that you want to train, test, and verify that your AI model works. This tutorial introduces a practical approach to help simplify the process.

With the IMX500 sensor in Raspberry Pi AI Camera (**rpimag.co/aicam**), you can use your own datasets to improve your AI models. Whether you're an experienced maker or just beginning to explore the world of edge AI, this guide will help you organise, refine, and export datasets with ease. Let's look at how this tool can support you to build smarter AI models, faster.

The challenge of dataset creation

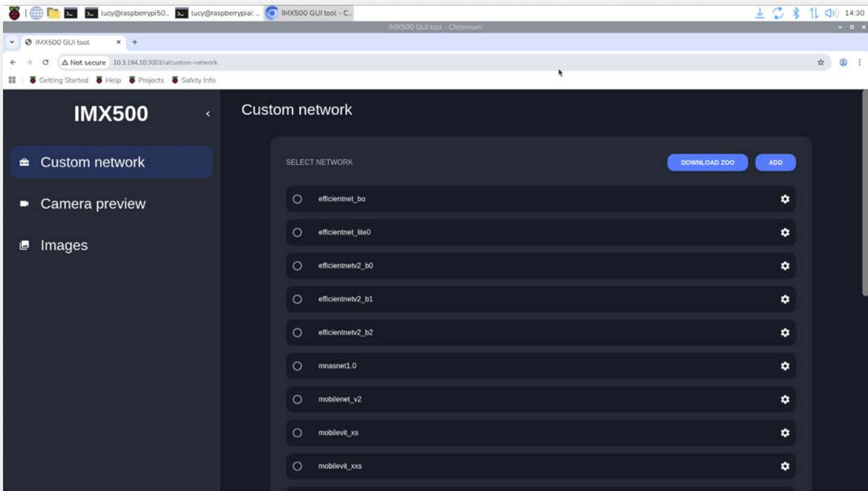
Dataset preparation is an important yet sometimes challenging aspect of vision AI projects. Capturing images, organising them, cropping out irrelevant details, and ensuring they're formatted correctly is a lot of work. This process can be a roadblock that slows down progress or discourages you from starting in the first place. But with the right setup and tools, you can simplify these tasks and focus on your AI development.

Setting up and getting started

We will use the tool that provides some convenient features for dataset creation. This makes it easier to capture images that are very close to the deployment environment and highly suitable for training, since the data comes directly from the IMX500 image sensor.

GUI Tool runs on your Raspberry Pi with AI Camera attached, and you access it via a web browser using another computer on the same network.

Creating datasets isn't just a technical task: it's a gateway to collaboration, learning, and real-world innovation



Installation

Read the GitHub page when installing the software.

rpimag.co/aitriosrpiguitool

Figure 1: The GUI Tool web interface

To run the tool, you'll need Node.js (**nodejs.org**) and uv (**docs.astral.sh/uv**) software:

```
$ sudo apt install nodejs npm
$ curl -LsSf https://astral.sh/uv/install.sh |
sh
```

Check that everything installed correctly with:

```
$ node --version
$ npm --version
$ uv --version
```

Now clone the repository from GitHub:

```
$ git clone https://github.com/
SonySemiconductorSolutions/aitrios-rpi-sample-
app-gui-tool
```

Navigate into the new folder and install the software in the root of the folder.

```
$ make setup
```

To start the GUI Tool, run:

```
$ uv run main.py
```

You'll need the IP address:

```
$ hostname -I
```

Or hostname:

```
$ hostname
```

…of your Raspberry Pi to access it on the network.

Access the GUI Tool

Now move to the second computer on your local network and open a browser. Open your browser and navigate to:

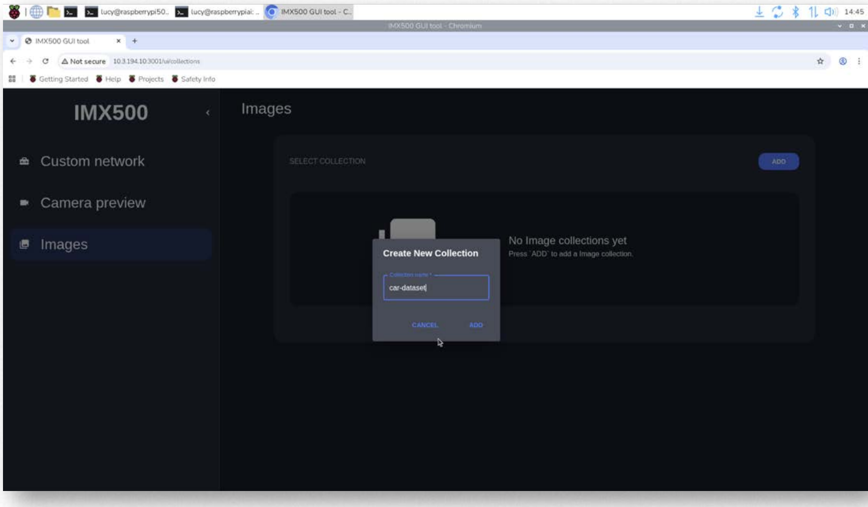
```
http://<your-raspberrypi-IP-address>:3001
```

…to access the tool's interface.

You can also access the GUI Tool directly from your Raspberry Pi with the AI Camera module via:

```
http://127.0.0.1:3001
```

You will see the GUI Tool web interface as seen in **Figure 1**.



▲ **Figure 2:** Create a new dataset

Creating a dataset using the IMX500 sensor

Once the setup is complete, you can use the GUI Tool to create and organise your dataset. Choose the Images tab in the sidebar and click Add to create a new dataset. Give the dataset a name in the pop-up window; for example, car-dataset (**Figure 2**). Click Add to create the dataset.

Now we need to add images by uploading them from your computer. For this tutorial we have used the Vehicles-OpenImages Dataset from Roboflow (**Figure 3**). You can download this from **rpimag.co/cardataset**.

Click Upload and choose an image from your Raspberry Pi OS file system. The image will appear in the car dataset (as in **Figure 4**).

Capture images with the camera

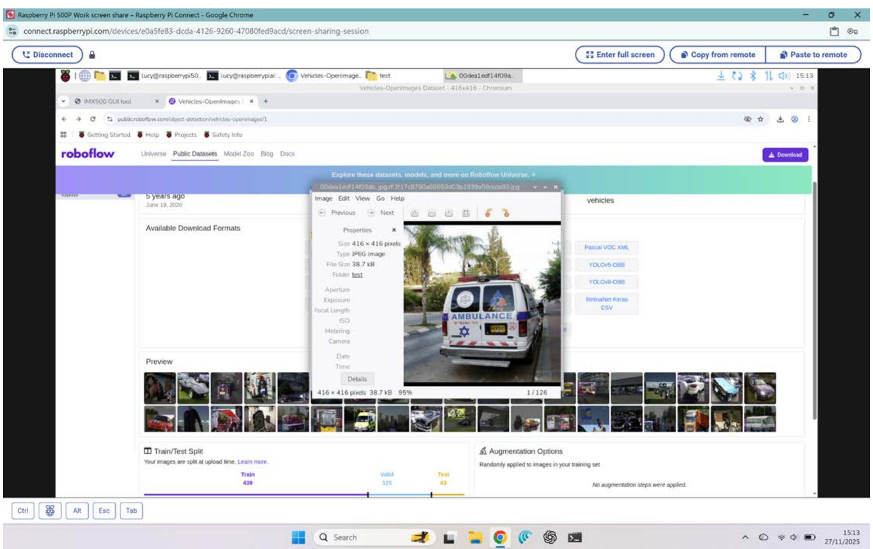
It is also possible to use the GUI Tool to automate image capture directly from a camera attached to your Raspberry Pi. If you have a Raspberry Pi AI Camera connected, you can also gather input tensor data alongside the raw image.

Choose the Camera preview tab to view the image from your camera.

Select collection: Click on Select Collection and choose a dataset to add the images to.

Input: Click the Timer switch to automate image capture at set intervals. For example, to capture a frame every 10 seconds for 50 images, set the Capture Rate to 0.1 and the number of photos to 50. Activate the image capture and let the tool handle the rest.

Input Tensor: Raspberry Pi AI Camera works differently compared to traditional image processing systems. The IMX500 Camera Module includes an internal ISP that preprocesses the sensor data and supplies the input tensor directly to its onboard AI accelerator chip. So, for optimal performance, it's highly recommended to train models using the exact input tensor data produced by the IMX500 sensor, rather than relying on raw images or preprocessed images only. This ensures that the model learns from data that precisely matches the runtime conditions, which leads to better model performance.

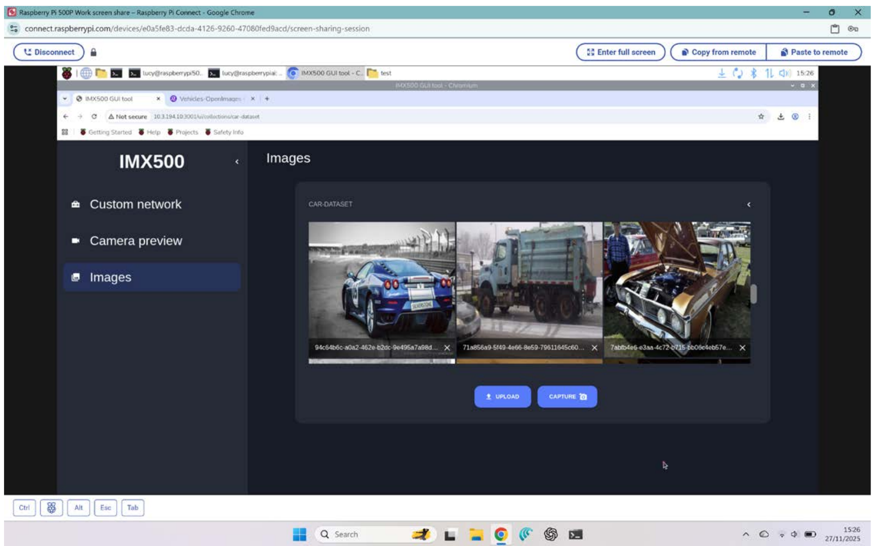


▶ **Figure 3:** The Vehicles-OpenImages Dataset from Roboflow is a good test bed of images for training a vehicle detection model

> *You can use the GUI Tool to create and organise your dataset*

Fortunately, we can very quickly get this input tensor data by enabling the Input Tensor flag during the image capturing process.

Start capture: Click the camera icon to start the image capture process.



▼ **Figure 4:** The vehicles dataset added to GUI Tool

Manage images

Head to the Images tab to upload, delete, or capture images directly into your dataset to keep it organised.

Once your dataset is ready, click on Images and click the cog icon next to your dataset. Select Download to save the images as a ZIP file to your computer.

Practical example: recognising cars

Imagine you're developing an AI model for car recognition with the IMX500 sensor. Here's how the process might look:

1. Create a 'car-dataset' dataset.
2. Capture images of cars using the IMX500 sensor.
3. Automate the capture process to ensure consistency.
4. If needed, crop images to focus on relevant areas, such as individual cars.
5. Organise and manage these images within the tool.
6. Export the dataset and use it to annotate and train your AI model.

What to do next: training your AI model

Once your dataset is ready, the next steps involve annotation, and training with TensorFlow or PyTorch. Alternatively, for a streamlined and user-friendly experience, you can use a dedicated tool to simplify these steps. One tool that can assist you is Brain Builder for AITRIOS (**Figure 5)** from the Studio Series of AI tools and services for AITRIOS (**rpimag.co/brainbuilder**).

Further reading

Download the GUI Tool from the Sony Semiconductor Solutions GitHub account:
rpimag.co/guitool

Learn more about what you can build with Raspberry Pi AI Camera and IMX500 sensor on the AITRIOS Developer website:
rpimag.co/aitrios

Annotating

Annotating your dataset is a critical step in training an AI model because it teaches the AI exactly what you want it to learn. If the annotations contain mistakes, the model will learn those mistakes as well, which can reduce its accuracy.

There are many tools available for annotation, such as Roboflow (**roboflow.com**) or CocoAnnotator (**cocoannotator.com**), that help you label your datasets according to the type of model you plan to train.

When choosing an annotation tool, make sure to check which export formats it supports. Your dataset must be exported in a format compatible with the AI model you want to train.

Training

Once your dataset is annotated and exported, you are ready to start training. We suggest you follow your chosen framework's guides on how to create a training script and what hardware you might need.

Brain Builder for AITRIOS

This tool is designed to simplify the annotation and training process, which might be helpful for users at varying levels of AI expertise. With Brain Builder for AITRIOS, you can annotate and train your AI models in a few steps all inside the same tool. You can send your annotated dataset straight to training with it all being in the right format.

Brain Builder for AITRIOS currently supports three types of models: Classification, Object Detection, and Anomaly Hi-Fi. You can train and evaluate your model and, when you are happy with the accuracy, export it easily for IMX500 without any hassle.

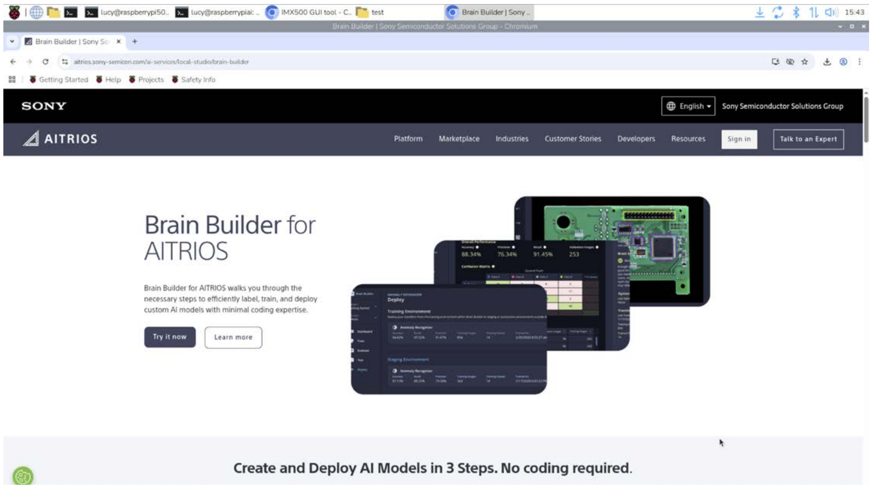


Figure 5: Sony AITRIOS Brain Builder software can simplify the process of training AI models

Deploying your AI model

Once your model is trained, you can package it on your Raspberry Pi and deploy it on the IMX500 to start building your AI application. For example:

- Package your model on your Raspberry Pi.
- Build an application to visualise the results, such as counting cars.

Creating datasets isn't just a technical task: it's a gateway to collaboration, learning, and real-world innovation. The possibilities are wide-ranging: educators can introduce students to AI and machine learning hands-on, makers can build smarter IoT devices, like home security systems or gesture recognition, and researchers can accelerate their work on projects like wildlife conservation, medical imaging, and more.

Harvard Mark I

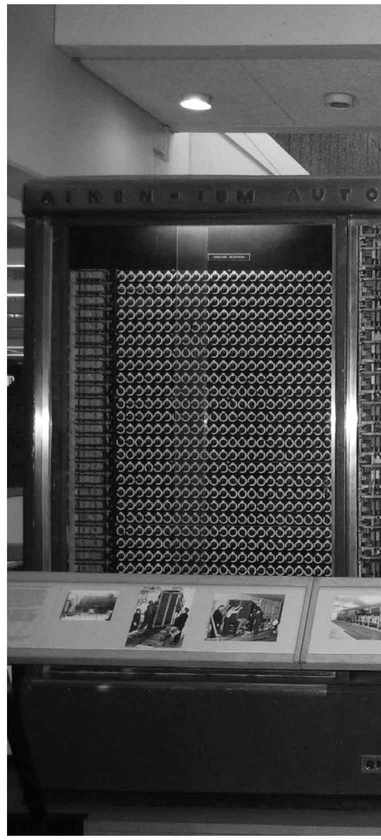
The first in a series of pioneering electromechanical machines

Maker

Tim Danton

When not writing books about classic computers, Tim is editor-in-chief of the British technology magazine PC Pro. He has also helped to launch several technology websites, most recently TechFinitive.com, where he is a senior editor.

dantonmedia.com



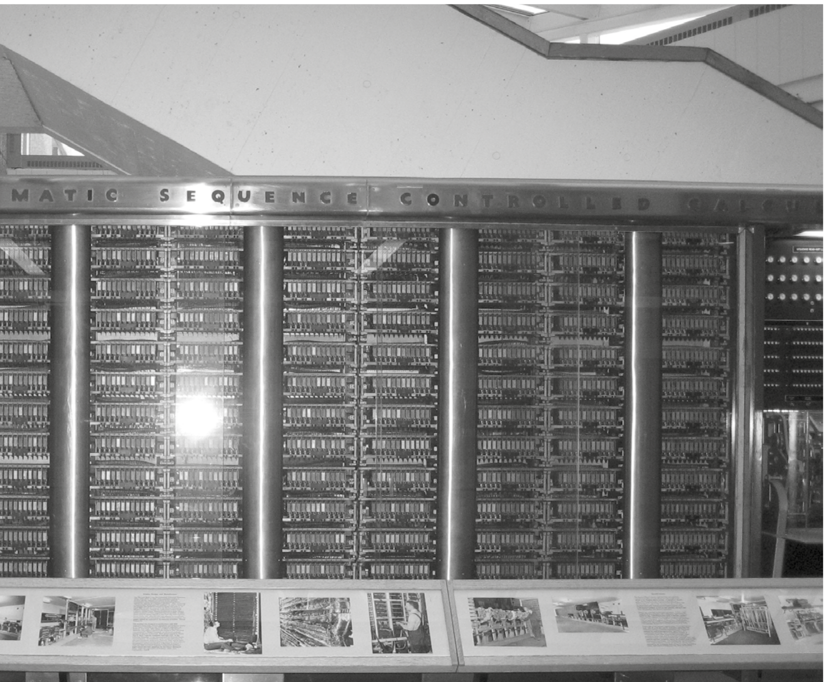
Howard Hathaway Aiken was a remarkable man. Not merely because he was the driving force behind the world's first genuinely useful computer, but because there were two starkly contrasting sides to his personality: on one side, the fierce, brisk and often unforgiving head of department. A man who ruled the Harvard Computational Laboratory with an iron fist. On the other, a thoughtful, loyal, entertaining friend and a loving son devoted to his mother.

That sense of loyalty came to the fore in 1912, when a 12-year-old Aiken confronted his drunkard father Daniel who was – not for the first time – physically abusing his wife, Margaret Emily Mierisch Aiken. According to Bernard Cohen's account in *Howard Aiken: Portrait of a Computer Pioneer*,[1] the young Howard "grabbed a fireplace poker and drove his father out of the house".

That was the last they saw of Daniel and, more crucially, his family money: Howard Aiken's paternal grandparents, who had helped to support the young family, immediately cut funds to mother and son. An only child, this meant Aiken became financially responsible for himself, his mother, and even her parents – who lived with them – once he completed eighth grade at the age of 14.

At this point Aiken's future, and the creation of the Mark I, was hanging by fate's thread, only for a teacher to intervene. Aiken was a brilliant student, but money was more immediately important to the family than schooling. The teacher pleaded with Margaret to change her mind, but the rent needed to be paid, food put on the table. She needed her only child to earn money.

1 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer* (The MIT Press, 1999, ISBN 978-0262032629), p9

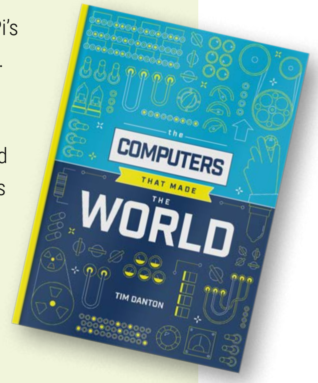


▲ Left side of the Harvard Mark I displayed in the Cabot Science Building, Harvard University, 2005
Image: Waldir Pimenta, CC BY-SA 3.0

The Computers that Made the World

This article is an extract from Raspberry Pi's book, *The Computers that Made the World*. This book tells the story of the birth of the technological world we now live in. It chronicles how computers reshaped World War II. And it does it all through the origins of twelve influential computers built between 1939 and 1950. You can pick up a copy on the Raspberry Pi website.

rpimag.co/tctmtw



Of all the people in The Computers that Made the World, there's little doubt that Aiken's teenage years were the toughest

Fortunately, Aiken's teacher (whose identity has now been lost) was not merely tenacious but also imaginative. He, or she, found the young scholar a night job at the Indianapolis Light and Heat Company as an electrician's helper.[2] Once his shift ended, the money earned, Aiken could head to school.

To pause for a moment, this means that a teenage Howard Aiken was pulling all-night shifts, then studying, leaving little time for rest. Although Aiken claimed that some of his nights were spent manning the switchboard, allowing him to grab a few hours' sleep on occasion. Of all the people covered in *The Computers that Made the World*, many of whom lived privileged lives, there's little doubt that Aiken's teenage years were the toughest.

Despite his challenging schedule, Aiken worked hard at school and gained not only his education but also support from Indianapolis's Superintendent of Public Instruction, Milo Stewart. In a 1973 interview with Henry Tropp and Bernard Cohen, Aiken explained that he went to the University of Wisconsin because Stewart sent letters to public utilities "in every Midwestern University town",[3] and only the Madison Gas and Electric Company offered him a job.

2 As before

3 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p119

Aiken was now 19 years old, studying electrical engineering, and would emerge with a Bachelor of Science degree in 1923. But financial factors would continue to slow his path. Rather than stay on at college, he used his degree to secure a better job at the Gas and Electric company and would only re-enter academia almost a decade later.

Aiken spent five of those years as an engineer at the Westinghouse Electrical and Manufacturing Company in Chicago, where he earned good money but found that his intellectual itch was not being scratched. He resigned from his job and signed up at the city's university, but only spent two quarters of the academic year there "upon discovering that the faculty were bootlegging grades at the behest of the newly appointed president".[4]

Spurred on by a professor at Chicago, he switched to Harvard, a centre of excellence for physics. By this time, autumn 1933, Aiken was in his mid-30s and significantly older than most graduate students. Ronald King – who joined Harvard at a similar time to Aiken and would go on to be a professor of applied physics there for almost 30 years – later told Cohen that this tall young man, who was certainly aware of his own brilliance, did not always get on with the Physics Department.

4 As before, p120

To such an extent that he might have been "kicked out," said King,[5] were it not for the intervention of Professor Emory Leon Chaffee, who headed the Communication Engineering group. Chaffee essentially took Aiken under his wing, and decided that his bright graduate student should work on space charge as his PhD thesis.

As was the case for Professor Atanasoff with the ABC (covered in the first part of this series), Aiken saw the need for faster and more reliable calculators during his research. Aiken said that "the object of the thesis almost became solving nonlinear equations," and doing so was both tedious and time-consuming on existing desk calculators. He added: "it became apparent to me at once that this could be mechanised and programmed and that an individual didn't have to do this." [6]

Aiken set to work to devise such a machine, finding inspiration in a book entitled *Modern Instruments and Methods of Calculations: A Handbook of The Napier Tercentenary Exhibition*. This exhibition took place in Edinburgh in July 1914, 300 years after the Scottish mathematician John Napier published *A Description of the Admirable Table of Logarithms*. While the 'handbook' was merely designed to accompany the exhibition, its 343 pages would have given Aiken a comprehensive oversight of calculating machines' development from Napier's bones to slide rules to early 20th century innovations such as the Monarch-Wahl Adding and Subtracting Typewriter: "You type the amounts – the Monarch-Wahl gives you the total," read the company's newspaper adverts.[7]

5 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p22

6 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p2

7 The Daily Telegraph, 19 November 1913, p13

The British Library still holds two copies of the handbook: one sits undisturbed in a low oxygen chamber, but a second is available for visiting Readers.[8] Four pages in particular must surely have grabbed Aiken's attention, laying out the key principles behind Babbage's Analytical Engine. The description covered his concept of the "mill", "store" and "Jacquard apparatus." [9] These are signature inclusions, albeit with different names, in the IBM Automatic Sequence Controlled Calculator, or Harvard Mark I as it came to be known.

We can't know exactly how influential this handbook was, because Aiken's own account of the Mark I's creation appears to have slipped into legend. In 1973, shortly before his death, Aiken said that "the first time he ever heard of Babbage" was when he discovered that two wheels from Babbage's Difference Engine were sitting in "the attic of the old research laboratories" at Harvard University.[10]

He told the story to illustrate the university's resistance to his computing project. "The faculty had rather limited enthusiasm about what I wanted to do, if not almost downright antagonism," said Aiken.[11] He added that one member of the laboratory "couldn't see why in the world I wanted to do anything like this in the Physics Laboratory, because we already had such a machine and nobody ever used it."

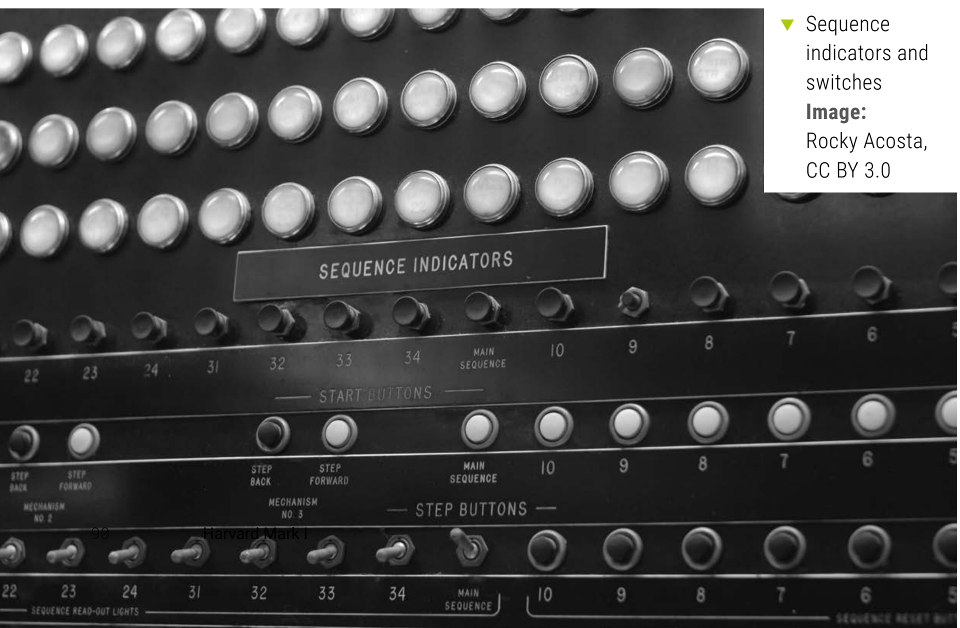
Aiken set out to find this machine, only to discover a pair of forgotten wheels that had been presented to Harvard by Charles Babbage's son. The fact these were gathering dust in an attic reflects how distant a memory Babbage had become, which is perhaps surprising when you consider the impact he had in 19th century Britain.

8 Anyone over the age of 18 can become a Reader at the British Library. You apply on-site, explain the project you're working on, and are given a badge entitling you to enter one of the many Reading Rooms. From here, you request one of the 170 million items stored across its London and Yorkshire sites. Unlike a regular library, you cannot take any borrowed books out of the Reading Rooms.

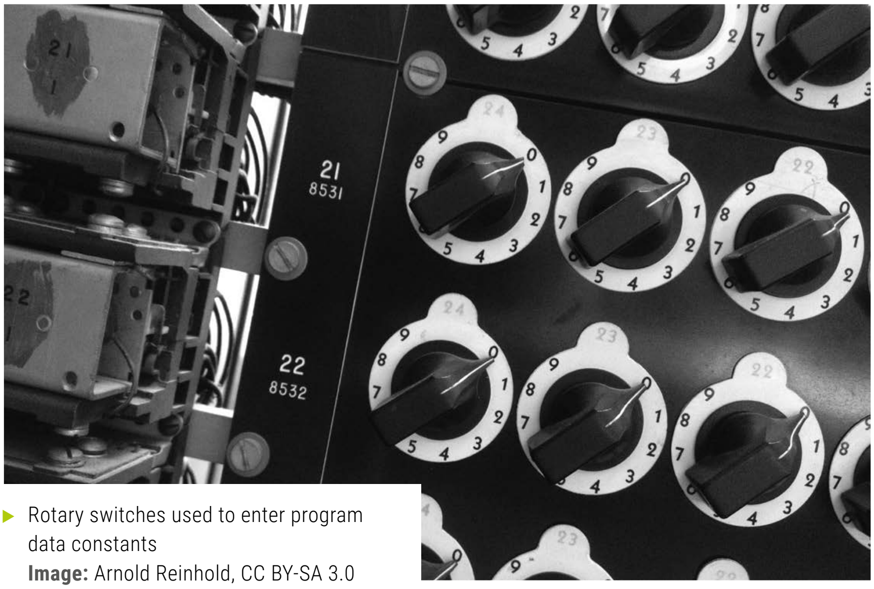
9 PE Ludgate, 'Automatic Calculating Machines', in E M Horsburgh (ed.), *Modern Instruments and Methods of Calculations: A Handbook of The Napier Tercentenary Exhibition* (The Royal Society of Edinburgh, 1914), p124

10 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p4

11 As above, p3



▼ Sequence indicators and switches **Image:** Rocky Acosta, CC BY 3.0



▶ Rotary switches used to enter program data constants
Image: Arnold Reinhold, CC BY-SA 3.0

Now we fast-forward to 22 April 1937: the date on which Aiken presented his plans to Monroe Calculating Machine Company, then a leading manufacturer of desk calculators.

It's crucial to note that, unlike other pioneers such as Atanasoff, Mauchly, and Zuse, Aiken's proposals focused on what he wanted his invention to do rather than the mechanics: calculations such as tabulating Bessel functions, a time-consuming task that would give the Mark I its nickname of Bessy. He essentially presented an architecture that a manufacturer would then convert into a functioning machine using its own expertise. Aiken later claimed that this meant that if Monroe had decided to partner with him, the computer would have been made out of mechanical parts. If RCA had been interested, it might have been electronic. And it follows that it was made out of electromechanical, tabulating machine parts because IBM was willing to pay the bill.

At that time, Monroe was a big name in desk calculators, so it was natural for Aiken to approach the firm. He met with George Chase, director of research, who shared the details of the meeting in a slide show detailing the history of mechanical computers in 1952 – published in the IEEE Annals of the History of Computing.[12]

"He told me certain branches of science had reached a barrier that could not be passed until means could be found to solve mathematical problems too large to be undertaken with the then-known computing equipment," wrote Chase. "He outlined to me the components of a machine that would solve those problems."

Chase concluded: "What he had in mind at that time was the construction of an electromechanical machine, but the plan he outlined was not restricted to any specific type of mechanism; it embraced a broad coordination of components that could be resolved by various constructive mediums. I knew then that the second era of development of computing machinery had started."

Unfortunately for Chase, and Monroe, after many months of going back and forth he could not persuade the company's board to pay the high cost of the machine. It must have been with great reluctance that Chase suggested to Aiken that he contact IBM instead.

Automatic computing wasn't entirely new ground for International Business Machines. In 1936, it had built a Calculation Control Switch designed by a Columbia University Astronomy Department professor called Wallace Eckert (no relation to Presper Eckert of ENIAC fame). Eckert would eventually head up IBM's 'super calculator' project in 1947,[13] which became the Selective Sequence Electronic Calculator (SSEC). Eckert's 1936 switch made astronomical calculations much quicker by using punched cards to send commands to IBM's existing Type 601 electric multiplier.

Aiken's proposal was an order of magnitude more complicated than the Calculation Control Switch, however, and also far more expensive. The initial estimate rapidly grew from around $15,000 to $100,000 and, forming a precedent for IT projects that persists to this day, its cost would more than double again. Despite this, IBM would eventually donate the Mark I and an additional $100,000 to Harvard as a contribution to its running costs. In return, IBM owned all the patents resulting from its construction.

One of the reasons Thomas Watson Sr, IBM's chairman, was willing to spend the money was because he wanted to replicate the strong links with Harvard that IBM already had with Columbia University. But, as we shall see later, the course of history didn't run so smooth.[14]

Another reason for IBM saying yes is because Aiken found a backer for the project in James Bryce, who was both the company's chief engineer and Watson's most trusted technical adviser. According to IBM historian Emerson Hugh, Bryce "had no trouble convincing Watson of the merits of the project."[15]

12 George C Chase, 'History of Mechanical Computing Machinery', in Annals of the History of Computing, Vol 2, No 3, July 1980

13 This project would lead to the Selective Sequence Electronic Calculator (SSEC) – source: Emerson W Pugh, *Memories That Shaped an Industry* (MIT Press, 2000, ISBN 978-0262661676), p7

14 Emerson W Pugh, *Building IBM: Shaping an Industry and its Technology* (MIT Press, 1995, ISBN 978-0262161473), pp73-76

15 As above, p73

Still, progress was slow. We know Bryce and Aiken first met in November 1937, after which Bryce arranged for Aiken to visit Eckert at Columbia to see if the IBM calculator could achieve all Aiken wanted. The answer: a firm no. Bryce decided Aiken needed a better idea of what IBM's equipment could do, so next arranged for him to attend official IBM training classes and even accompany service engineers on jobs.

> *Certain branches of science had reached a barrier that could not be passed until means could be found to solve mathematical problems too large for the then-known computing equipment*

But Aiken's most intense inauguration into IBM world came when he spent four days at IBM's manufacturing facility in Endicott, New York, early in 1938. Here, he worked closely with the two engineers who would essentially build the Mark I: Frank Hamilton and Benjamin Durfee. Over the course of those four days, the three men attempted to set out the requirements of the machine, down to what could be created from existing IBM parts and what would need to be designed from scratch.

At this point, both parties believed that the computer would be built at Harvard with IBM's assistance as required. As such, Aiken still needed to gain the backing of his university. Although Harry Mimno, a close colleague of Aiken's at Harvard, would later describe his role in the creation of the Mark I as that of "an enthusiastic and fascinated listener",[16] he was the one who wrote a compelling report to Harvard's then-President, James Conant, a matter of days after Aiken returned from Endicott. In this report, he emphasised the minimal cost to Harvard – a "few thousand dollars," he wrote – and the immense return they would get in a calculator that could be put to use in a wide variety of academic research projects. His examples, presciently, included atomic physics.

Even without a formal agreement in place, IBM's Bryce kept the project moving in the right direction. Described by Aiken as "an astute inventor" who was "worth all other IBM people put together",[17] it was Bryce who sent Hamilton a layout of the Mark I's panels – having devised them in partnership with Aiken. In May 1938, IBM formally put money behind the project by giving Hamilton the funds to start work on a design on how the components would fit together.

While history knows the Mark I's importance, at this point IBM did not. Thomas Watson Sr never envisaged that it would be the beginning of a new line of products. He thought that it would be merely a scientific and academic partnership – much like the calculator IBM had created for Eckert at Columbia. As such, the project lived on the bottom rung of the Endicott team's priorities, and Hamilton didn't start work on the design until August.

It took a further eight months – during which Aiken made numerous visits to Endicott to provide information and chivvy the project along – before Harvard and IBM signed the final agreement to construct the machine. During the previous months, it had become obvious that Harvard was in no position to build the Mark I, so that duty fell to IBM. In particular to Hamilton and Durfee, with the ultimate responsibility falling to chief engineer Clair Lake.

Although Aiken was later dismissive of Lake's contributions – "Lake contributed absolutely nothing,"[18] he said in 1973 – there is plenty of evidence to suggest otherwise. According to Hamilton, Lake designed a new relay, and co-designed (with Hamilton) the punch for preparing the sequence tape on which instructions were passed to the computer. Ultimately, he was also in charge of the budget, reining back Aiken's more extravagant requests whilst delivering practical solutions.

Nor could the Mark I have been built without Durfee. His role was to devise the circuitry, and as the computer moved further away from existing IBM components the more work he had to do. A modest man, he perhaps undersold his talents: in an interview with IBM historians, he said he "had never been too good at mathematics,"[19] yet he covered an "enormous piece of drawing paper... with pencil arithmetic on all sides" to prove to himself that Aiken's suggested method of division would work.[20]

16 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p 37

17 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p23

18 As before

19 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p 74

20 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p24

Aiken continued to visit Endicott regularly, both to check on progress and to agree on elements of the design with Lake and Hamilton. By this point, Bryce had moved onto other projects, leaving Aiken as the theoretician while the IBM engineers interpreted his ideas, requests, and sketches into something that could be built without costs spiralling out of control. For example, Aiken wanted three multiply-divide devices, and that dropped down to one in the final machine. Similarly, he envisaged multiple sequence control units, and this again became one. Durfee estimated that if they had agreed to all Aiken's suggestions the Mark I would not only have been prohibitively expensive but also three times its ultimate size. And bear in mind the final machine was "longer than a diesel locomotive,"[21] according to The Boston Globe.

Aiken's frequent visits and detailed instructions kept coming until the spring of 1941, when he was called up to the US Navy. While Aiken said that he "still made those trips to Endicott to plead: 'Come on, let's do a little more',"[22] during his time in service, he passed the responsibility of finishing the job to a Harvard graduate called Robert Campbell.

Unlike the ENIAC, which had been commissioned by the army to calculate missile trajectories, at this point the Mark I had no formal military connection. Nor did those close to it appreciate its potential applications in the war effort. So while some of IBM's factories were being used to create weapons of war, the company was also under pressure to convert existing machines into a form that could be used by the US military. It's little wonder that the experimental Mark I dropped down IBM's pecking order.

Despite Campbell doing his best to apply pressure on Hamilton and colleagues, he also had the distinct disadvantage of being a second-year graduate student rather than a professor. Plus, there were certain questions that IBM felt only Aiken could answer, but for chunks of time they simply couldn't contact him.

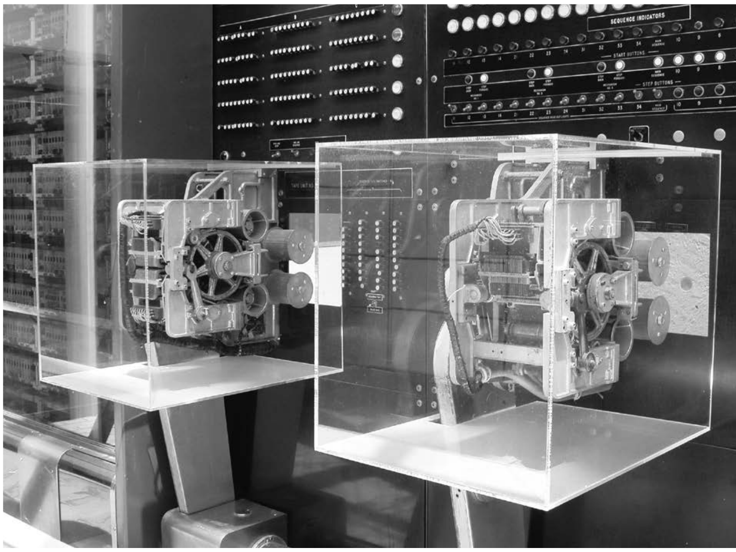
To an extent, then, it seems remarkable that the project wasn't quietly sidelined. Instead, IBM, Campbell, and Robert Hawkins – a graduate in 'Electrical Construction'[23] who had been working in Harvard's Physics laboratory but was sent to Endicott so he could help with development – kept things moving along as rapidly as they could. And at last, on 1 January 1943, almost six years after Aiken had first approached IBM, the Mark I solved its first 'real' problem.

21 Dorothy Wayman, 'Harvard Gets Huge Calculator', in The Boston Globe, 7 August 1944, p7

22 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p13

23 Robert Hawkins oral history interview with William Aspray, 20 February 1984, Charles Babbage Institute, **rpimag.co/hawkinsinterview**, p3

It wasn't very exciting. According to Campbell's notes, the Mark I computed and graphed "the time required to build up the current in an inductive circuit based on an equation which required the machine to multiply, divide, add, subtract, compute logarithms and antilogarithms." And after this auspicious start? Silence. When Campbell sent details of the first successful program to Aiken, he heard not a word in reply. And we have no records of any other programs being run on the computer until December 1943.



▲ Input/output and control readers
Image: Waldir Pimenta, CC BY-SA 3.0

We know that Aiken was spending at least some of his time lecturing on the subject of electricity to navy recruits at the Naval Mine Warfare School in Yorktown, Virginia. But there are also good reasons to suspect he had other more secretive duties. One rumour is that he personally decommissioned a German torpedo; another that he travelled to France by submarine, was smuggled into the country dressed as a peasant, all to inspect a secret piece of German weaponry. While we can never know for sure, it at least serves as a possible explanation for Aiken's lack of response to Campbell's message.

With no clear leadership from Harvard, and wartime distractions of their own, the Mark I was essentially forgotten for months. It could also be that the IBM engineers were dealing with the patents that had emerged during construction, and that IBM wished to complete that work before shipping the computer to Harvard as promised. There was a logistical side to this too, as dismantling and then rebuilding the Mark I would be a time-consuming task on its own.

What we know for sure is that finally, in October 1943, the President of IBM, Thomas Watson, sent a letter to Harvard's President James Conant inviting him for a demonstration of the completed machine later that month. Conant, however, was tied up with war duties of his own, with the visit delayed until December.

This, perhaps, was when the first seeds of discontent appeared in the Harvard-IBM relationship. From Watson's point of view, he had spent hundreds of thousands of dollars – in both parts and man-hours – creating this scientific calculator for Harvard. The least that Harvard could do was to appear grateful.

Now switch perspectives to Conant. He had never been enthusiastic about Aiken's project, once telling him that if he wanted to be promoted then he should stop wasting his time on mechanical computers. To Harvard's President, at this point, the Mark I must have seemed like a toy. He was so blasé about the project that during the ceremony to announce the machine, he reportedly asked who actually owned it.


▲ Grace Hopper
Image: US Dept of Defense, Public Domain

Nevertheless, in December 1943 Conant made the trip to Endicott as per Watson's request. Aiken was also present for the demonstration and it appears to have rekindled his enthusiasm, if it ever went away. By spring the following year, he was desperate to be back in charge of the machine.

On Friday 5 May 1944, after two years away from his creation, Aiken returned to Harvard and took control of the project once more. From this point until the end of the war, it would largely be put to military use, starting with a project for the Navy's Bureau of Ships to classify steel – based on the properties of their known impurities – to determine where each batch should be best used. The mathematical problem, according to Cohen, required ten simultaneous linear equations to be solved and 'calculating multiple regression coefficients and multiple correlation coefficients'.[24] In short, a job worthy of a giant computer.

Fortunately, it wasn't long before his wish was granted. In his 1973 interview, Aiken described a "Naval Commander" calling the Navy school and said, "Why don't you come up here and run this computer?"[25] To which Aiken replied that he had his orders. "Well, I'll get that corrected immediately." Within "just a matter of hours" he "had orders to pack up and go back to Harvard and become the officer in charge of the United States Naval Computing Project. I guess I'm the only man in the world who was ever Commanding Officer of a computer."

But let us not denigrate this amazing piece of equipment. Standing eight feet high, and stretching 51 feet, it was a sight to behold: starting from the left, your eyes are struck by two giant banks of dials. The first bank sets the constant registers, the second the storage registers. Then endless columns of plug-in relays, each of which must be set by hand, until finally a selection of IBM telewriters type out the results. And that's from the front: looked at from behind, all that could be seen were forests of interconnecting wires. Little wonder that it took the five-strong team of IBM engineers several weeks to reassemble it.

The machine found a home in the basement of Harvard's Research Laboratory of Physics, displacing a gigantic wet-cell battery (much to the annoyance of some in the lab). Under Campbell's supervision, and with the help of Robert Hawkins and researcher David Wheatland, the calculating machine had already been put to work. For instance, Aiken's colleague Ronald King asked it to calculate integrals.

As with all early computers, however, the Mark I didn't always behave well. Hawkins described himself as being on call "24 hours a day, seven days a week"[26] to tackle problems, and described one occasion when it was offline for three days. "I stayed there – I don't know how long – but I just couldn't keep my eyes open anymore. I just told the boss, 'I'm going home, I don't care!'" On his return, he and the rest of the team "fixed it right away".

IBM's Durfee stayed with the machine in those early days. Days during which they still didn't trust the machine, having grown used to checking its results using desk calculators. It was a tough habit to break, but eventually the team grew to rely on the Mark I's built-in checking mechanisms.

In contrast to the Colossus or ENIAC, there was no veil of secrecy covering the Mark I. IBM was keen to promote the role that it had to play in the creation of this war machine, and so it's easy to see why Watson was so furious when he read Harvard

24 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p160

25 Howard Aiken oral history interview with Henry Tropp and I B Cohen, 26 February 1973, Smithsonian National Museum of American History, **rpimag.co/aikeninterview**, p54

26 Robert Hawkins oral history interview with William Aspray, 20 February 1984, Charles Babbage Institute, **rpimag.co/hawkinsinterview**, p13

University's press release about the project. One that the US Navy had approved, but which Harvard failed to run past IBM.

Under the heading "World's greatest mathematical calculator", its first paragraph read in uncontroversial fashion: "The world's greatest mathematical calculating machine, a revolutionary new electrical device of major importance to the war effort, will be presented today to Harvard University by the International Business Machines Corporation to be used by the Navy for the duration."

Nor could IBM have found much to fault in the following three paragraphs, which described the forthcoming ceremony and the machine itself. For example: "An algebraic super-brain employing a unique automatic sequence control, it will solve practically any known problem in applied mathematics. When a problem is presented to the sequence control in coded tape form it will carry out solutions accurate to 23 significant figures, consulting logarithmic and other functional tables, lying in the machine or coded on tapes."

The problem this computer couldn't solve came in the fifth paragraph, which described Aiken as "the inventor… who worked out the theory which made the machine possible." And while the release went on to namecheck both Frank Hamilton and Benjamin Durfee, and stated that the construction "work was carried on at the Engineering Laboratory of the International Business Machines Corporation at Endicott, New York, under the joint direction of Commander Aiken and Clair D Lake", this reportedly sent Watson into a fury when he read it.

The controversy stems from two interweaving issues. First, IBM hotly disputed the seemingly simple statement that Aiken was the inventor of the Mark I. While it's true that the machine wouldn't have existed without Aiken, an uncountable number of design decisions were taken in partnership with IBM. Many without Aiken, while he was seconded to the Navy.

Second, where was the credit for IBM the company? Yes, it and three members of its staff were mentioned, but only fleetingly, and the release made it sound like IBM passively built the machine to Aiken's specification. There was no sense of a six-year partnership, the intellectual contribution from Bryce and others, the scale of IBM's financial investment, nor the key role played by benefactor Watson – without whose sign-off the machine would never have been built.

The Mark I was 'longer than a diesel locomotive,' a sight to behold with its giant banks of dials, forests of relays, and miles of wire

It didn't help that Harvard failed to even meet Watson and his wife when he emerged from the train in Boston. He had expected to be met by "some Harvard dignitaries, or their emissaries, with an official limousine"[27] but was instead greeted by IBM's Boston branch manager, Frank McCabe, in his two-door Chevrolet. McCabe later recounted the difficulty Watson's wife had getting into the car's back seat, but this was nothing compared to the anger McCabe saw growing on Watson's face as he read the press release.

We can only imagine Watson's mood when he summoned Aiken and Conant to his hotel room that evening, and the invective he used to berate Aiken in particular. While there is some speculation that some of this show of anger may have been calculated – Watson once stated that he used such tactics to get results in his business dealings – what we know for sure is that it had immediate effect. Harvard quickly reworded and reissued the press release, and Aiken's address at the official unveiling the following day was fulsome in its praise for IBM, Watson, and the company's engineers.

The short-term effect of this fallout was limited – Watson even went ahead and gave his generous $100,000 gift towards the ongoing running costs of the machine – but this would be the end of the partnership. IBM would go on to invent its own super-calculator, the Selective Sequence Electronic Calculator, determined that it was superior in every way to the Mark I. And it's doubtful whether Aiken, a proud man, would have wanted anything more to do with IBM after that point either.

Nor would the following day's newspaper coverage have helped. While many reports mentioned that the computer was given to Harvard by IBM, others said Aiken handed the computer to Harvard and many barely mentioned the company's role, instead focusing on a narrative that the computer's existence was due to Aiken being too lazy to work out the results of his thesis by hand – based on an interview Aiken gave to reporters prior to the official presentation.

At least the photographs of the finished computer, usually showing Navy ensigns at work on it, included the stylish front design commissioned by Watson that gave this machine such a striking look. Designed by industrial designer Norman Bel

27 | Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p123

Geddes, famous for futuristic concepts such as teardrop-shaped vehicles, Grace Hopper said it "gave poor Howard Aiken an awful pain, because it was fifty or a hundred thousand bucks for the case, and he could've used it for a computer, and that irked him."[28]

A lieutenant (junior grade) in the Navy Reserves, Hopper joined the project on 2 July 1944, adding her intellectual weight to the Harvard operatives – including Robert Campbell, Robert Hawkins, and Physics Research Associate David Wheatland – and Richard Bloch, an April addition who reported directly to a Rear Admiral. Of Bloch, Hopper later said that he was "the only person I ever knew who wrote a program in ink and was correct the first time. But Bloch just thought like the machine did."[29 30]

It was Bloch who was put in charge of the Mark I's most famous series of calculations, not that he was told why at the time. All he knew, other than the technical description of the nonlinear differential equation in question, was that the problem described "a spherically symmetric flow of a compressible fluid in the presence of a spherical detonation wave proceeding inward from the surface of the sphere towards its centre".[31] Although clues in its importance came via the presence of John von Neumann, then a key physicist in the Manhattan Project, whenever the Mark I was working on the problem. He only discovered the exact reason for the secrecy when news of the first atomic bomb, which laid waste to Hiroshima, became headline news in 1946.

This doesn't mean the bomb only happened because of the Mark I. In their 1982 essay, 'Early Computing at Los Alamos,'[32] Nicholas Metropolis and Eldred Nelson revealed that Los Alamos physicists solved the problem in three weeks using a series of existing IBM equipment, while the Mark I was "halfway through the problem" after five weeks. It wasn't a fair race – the Los Alamos team produced answers to six digits while the Mark I was accurate to 18 – but it does highlight that even by 1945 standards the computer was relatively slow.

28 Grace Hopper oral history with Uta Merzbach, 7 January 1969, Smithsonian National Museum of American History, **rpimag.co/hopperinterview1969**, p8

29 Grace Hopper oral history with Angeline Pantages, December 1980, Computer History Museum, **rpimag.co/hopperinterview1980**, p23

30 In his entertaining yet boringly titled essay, 'Reminiscences of Aiken during World War II and Later', Bloch disputes his infallibility, describing Grace Hopper's description as "something of an exaggeration". See p197 of I Bernard Cohen and Gregory W Welch, *Makin' Numbers: Howard Aiken and the Computer* (The MIT Press, 1999, ISBN 978-0262032636)

31 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p164

32 Nicholas Metropolis and Eldred Nelson, 'Early Computing at Los Alamos', October 1982, in the IEEE Annals of the History of Computing, Vol 4 Issue 4, pp348-357

Partly, the slow speed was by design. Aiken decided at an early stage that the computer would calculate to 23 digits, with the 24th place indicating whether the number was positive or negative. But that degree of accuracy was rarely, if ever, required, meaning the Mark I laboured through its calculations for significantly longer than was necessary. Another key reason was the computer's reliance on electromechanical relays rather than the electronic valves inside the ENIAC and Colossus, which inevitably adds time. It also adds noise, with a Harvard student of that time, Jeremy Bernstein, eloquently, albeit with a tinge of sexism, describing the Mark I as sounding "like a roomful of ladies knitting" when in operation.[33]

Some would argue that this means the Mark I wasn't really a computer, just a supercharged, interlinked version of the electromechanical calculators that had gone before. Critics might also correctly point out that it didn't store programs: the sequence of operations depended on a program punched out on tape.

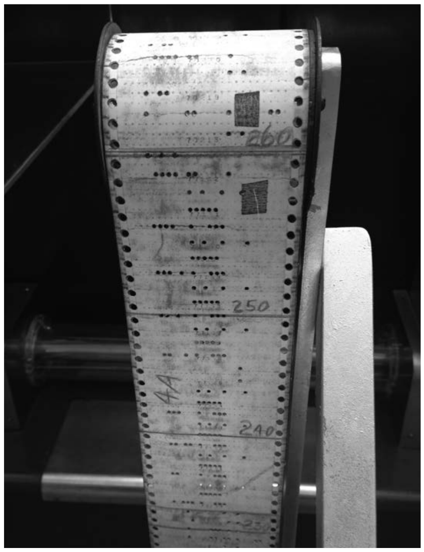
Where it wins out over some more illustrious rivals, however, is that it was completed and put to use before World War II was finished (even though the war was not on Aiken's mind in 1937 when he first conceived the idea). When Aiken gave a talk in 1960 covering its wartime tasks,[34] he mentioned "calculations in the development of radar", helping to "solve blast furnace difficulties" and calculating tables "of tremendous importance in magnetic mine warfare". But he didn't mention the Los Alamos work, so may have kept back some still-classified projects.

It was certainly put to heavy use, with three eight-hour shifts filling its day when it had an active problem to work through. And after Japan surrendered to the Allies on 14 August 1945, the Mark I moved on to its second phase in life: as a scientific, academic computer. And like so many early computers, it would evolve. The Mark I would eventually benefit from 24 more storage registers (taking it to 96), the addition of a high-speed multiplying unit, the ability to do calculations at double precision (46 digits) or half precision (11 digits) to save time. And, arguably most crucially, the introduction of program branching.

33 Jeremy Bernstein, *The Analytical Engine* (Secker & Warburg, 1965), p66

34 I Bernard Cohen, *Howard Aiken: Portrait of a Computer Pioneer*, p167

▶ Card punch used to prepare programs **Image:** Arnold Reinhold, CC BY-SA 3.0



▲ Program tape with visible programming patches
Image: Arnold Reinhold, CC BY-SA 3.0

All these enhancements, along with greater reliability when every single one of its 3500 relays were replaced, meant it continued to be put to good use until it was finally decommissioned in 1959.

Photos of the Harvard Comp Lab in the 1950s show it sitting alongside its successors, which were created for military use rather than to replace the Mark I. Instead of heading down the fully electronic route with the Mark II, which was built between 1945 and 1947 for the US Navy, Aiken and his team decided to effectively create a bigger, better version of the Mark I. It would consume twice as much floor area – 4000 square feet (370m^2) – and could operate on two problems at once. In his essay, 'Mark II, an Improved Mark I,'[35] Robert Campbell refers to a left side and a right side, that could combine forces on a larger problem if necessary. Effectively, it was two computers in one.

Instead of calculating to 23 digits, the Mark II calculated to ten digits and supported floating points. This, Aiken felt, was better suited to its primary role of calculating firing tables. They also built it at Harvard, working with third-party companies to manufacture specialist components such as relays. These remained the biggest handbrake in terms of calculation speed, but the Mark II was still six times faster at calculations than the Mark I, according to Campbell.[36]

Mark III and IV were to follow, the III designed for the Navy and the IV for the US Air Force, but that was the end of the line for Aiken's computers. By the late 1950s, the EDVAC-based design – or von Neumann architecture, to put it another way – had taken hold. There is no place for the Mark I on a modern-day computer's family tree.

Still, Aiken's creation continues to have a ripple effect to this day.

Most of this effect stems from his decision to create a Computer Lab (almost immediately shortened to Comp Lab in the Harvard vernacular) and a formal Computer Science education for Harvard's students, which in turn created a rich feeding ground for IBM and its rivals over the years. Many of Aiken's acolytes would go on to successful careers at the company, with Thomas Watson Jr proving keen to restore goodwill between IBM and the university soon after he succeeded his father as chairman in 1952.

Then there is the Harvard computers' impact on programming as a whole. Much of the credit here goes to Grace Hopper, who wrote the Mark I's manual – this necessarily included detailed instructions on how to program it. It was a task fraught with difficulty, with Hopper famously labelling the moth they found trapped in one of the Mark II's relays as "first actual case of bug being found".

That happened on 9 September 1947 (a fact we happen to know because Hopper taped the moth into the logbook), but as Peggy Kidwell explained in her 1998 article, 'Stalking the Elusive Computer Bug',[37] the term 'bug' first emerged in the late 19th century. Thomas Edison even used the term in relation to the "little faults and difficulties" that must be ironed out in the process of taking an invention to commercial reality.

Alas, we can't precisely give the term 'debug' a vigorous etymology, but once again Grace Hopper must take some credit for its popularisation. She left Harvard in 1949 to join the Eckert-Mauchly Computer Corporation, and when writing a glossary for the UNIVAC she defined the term as a colloquialism "to remove a malfunction from a computer or an error from a routine."[38] Such as, one might suppose, a squashed moth from within a mechanical relay.

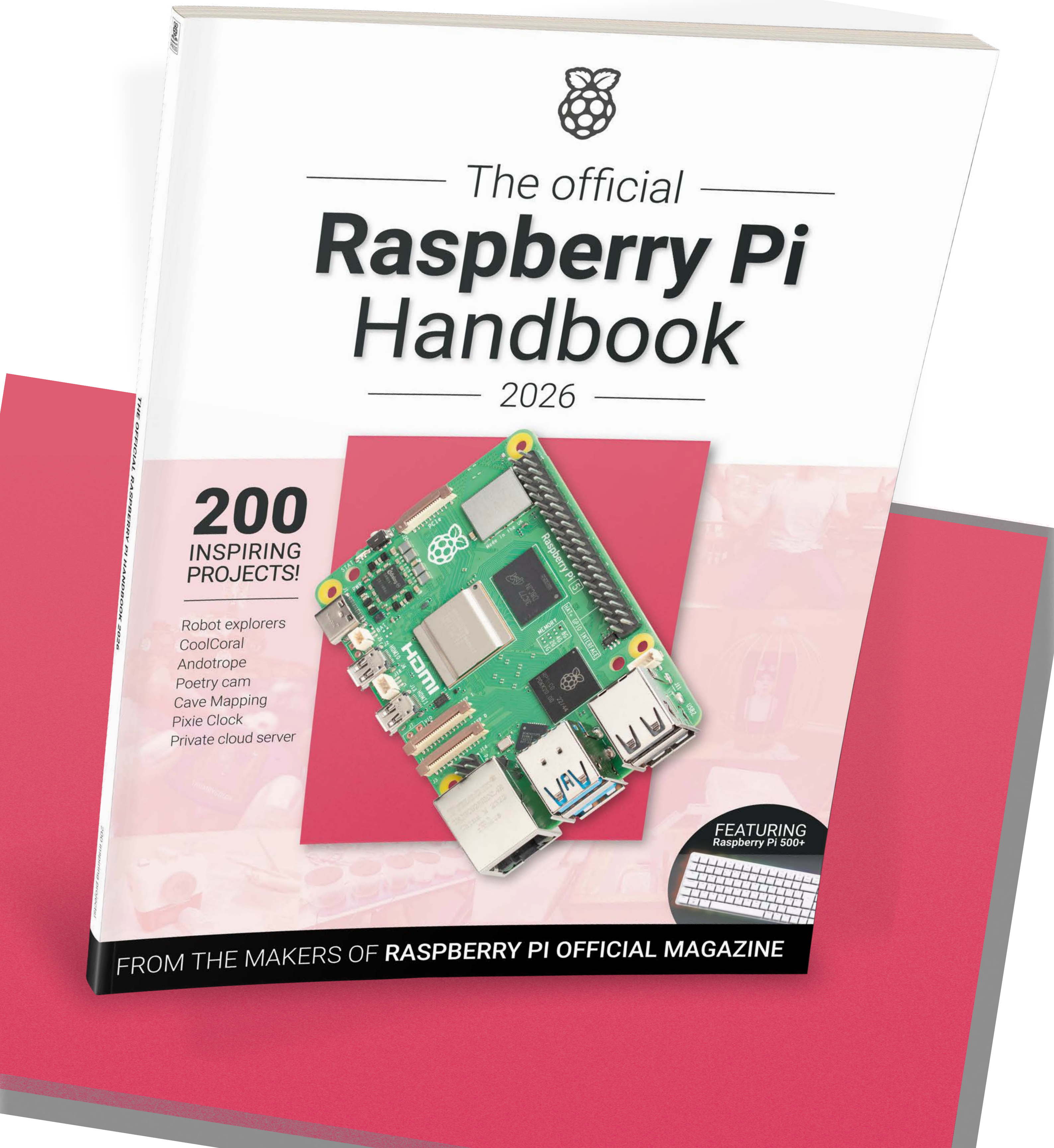
> We can't precisely give the term 'debug' a vigorous etymology, but once again Grace Hopper must take some credit for its popularisation

35 Robert Campbell, 'Mark II, an Improved Mark I', in I Bernard Cohen and Gregory W Welch, *Makin' Numbers: Howard Aiken and the Computer*, pp111-127

36 As above, p122

37 Peggy Aldrich Kidwell, 'Stalking the Elusive Computer Bug', in IEEE Annals of the History of Computing, Vol 20, No 4, 1998, p5

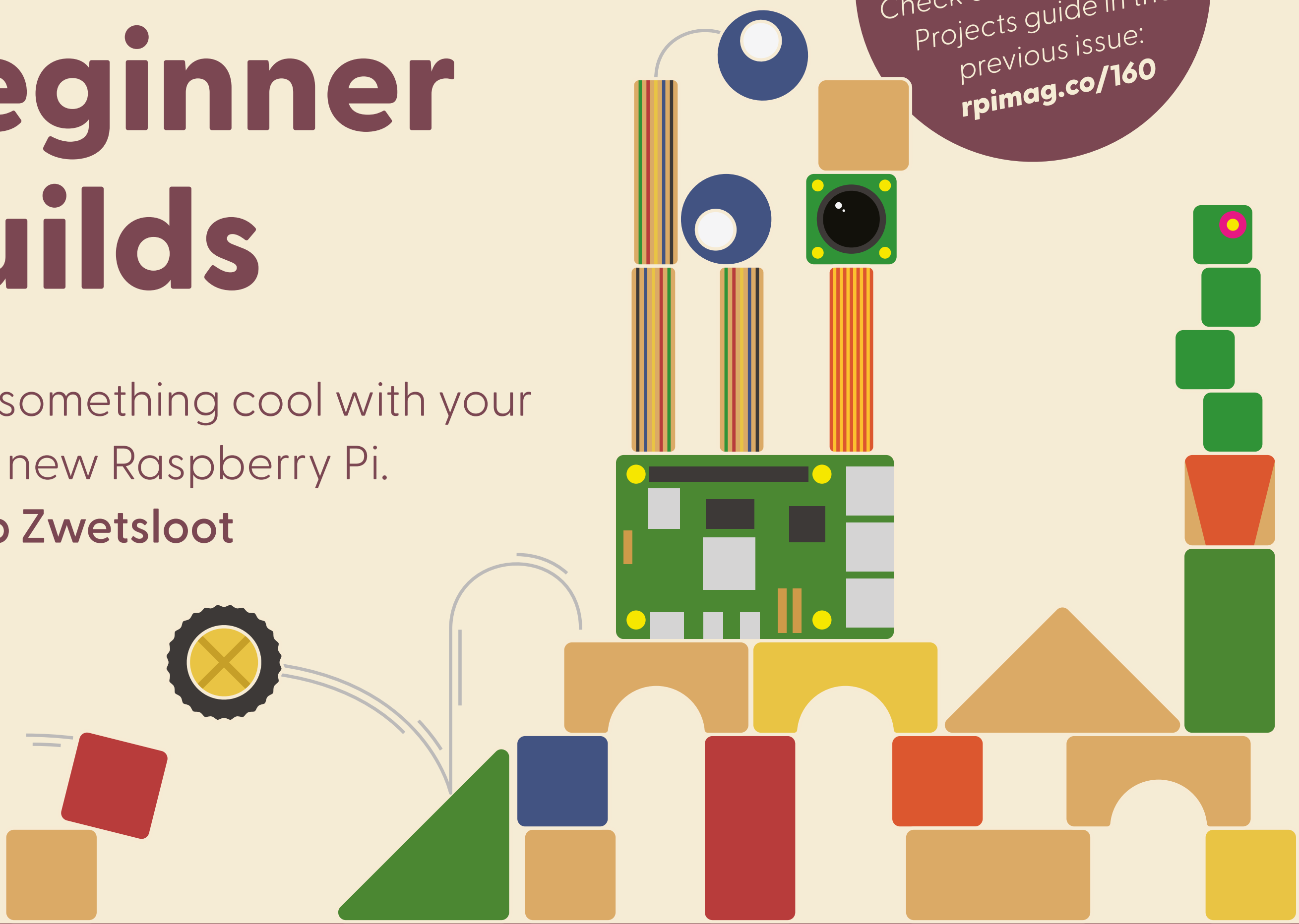
38 Grace Hopper, 'A Glossary of Computer Terminology', in Computers and Automation, Vol 3, No 5, May 1954, p16


The official
Raspberry Pi
Handbook
2026
200
INSPIRING
PROJECTS!
Robot explorers
CoolCoral
Andotrope
Poetry cam
Cave Mapping
Pixie Clock
Private cloud server
FEATURING
Raspberry Pi 500+
FROM THE MAKERS OF RASPBERRY PI OFFICIAL MAGAZINE
THE OFFICIAL RASPBERRY PI HANDBOOK 2026

Brilliant beginner builds

Make something cool with your brand new Raspberry Pi.
By **Rob Zwetsloot**

Want more ideas?
Check out our Christmas Projects guide in the previous issue:
rpimag.co/160



Earlier in this issue, we showed you how to get started with your Raspberry Pi. You may now be wanting to put it to work in your house – after all, it's a powerful machine with many applications.

While there are definitely some fun ways to make use of Raspberry Pi that we love (check out **retropie.org.uk** for all your retro gaming needs), we wanted to highlight some more practical ways to use your new microcomputer (or microcontroller).

Do citizen science with Raspberry Shake

Help map tremors around the world with this simple kit

raspberryshake.org

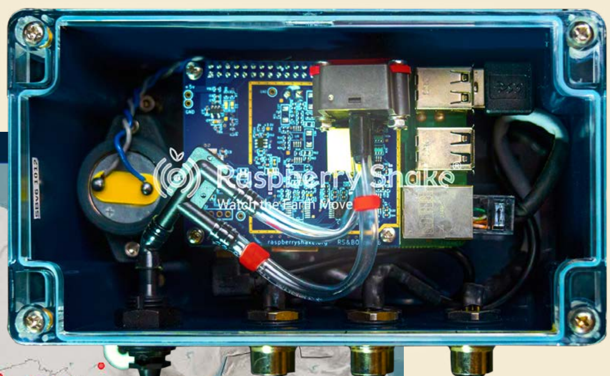
This is a seismometer kit that includes a special geophone sensor that lets you study the Earth and how it moves, very accurately. Not only that, but you can hook it up to a network of like-minded amateur – and professional – seismologists and see how seismic activity spreads around the world, and in real time.

Assembly is very easy – just connect the geophone to the add-on board, put the board on a Raspberry Pi, then assemble the acrylic case. There's a handy video that runs you through the process, too, and then goes on to cover software setup.

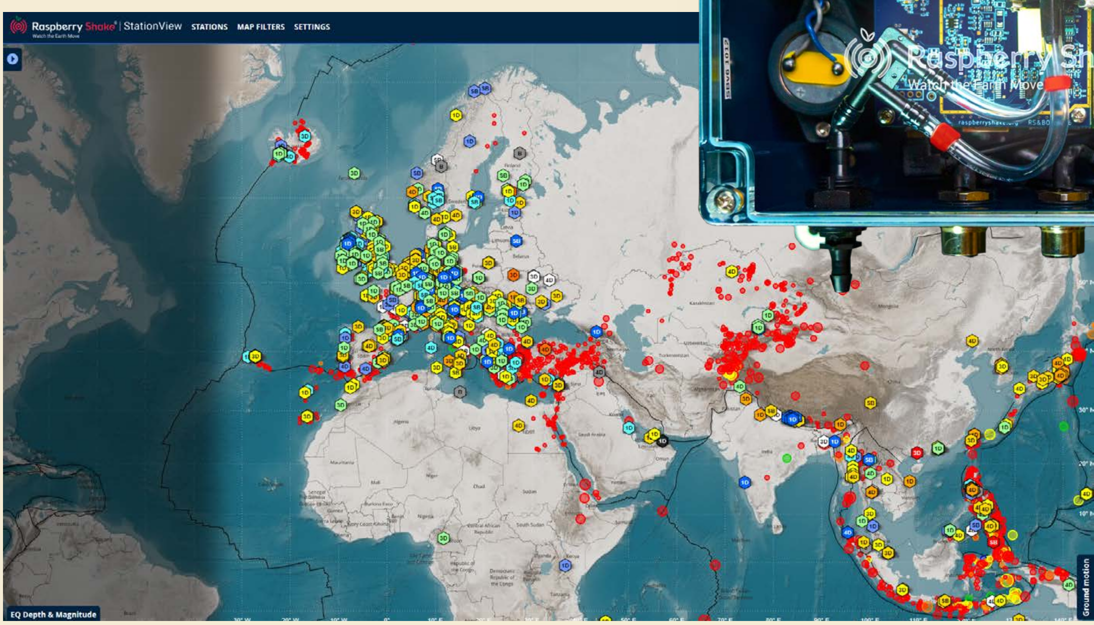
For science!

BOINC (**rpimag.co/boinc**) is a long-running service that allows researchers to use distributed computing power to perform complex calculations and simulations. Whether you're folding proteins or searching for alien life, you're able to set up BOINC to make use of your spare CPU cycles to help a larger project.

Study the Earth and how it moves



▶ The kit for Raspberry Shake is nice and compact



◀ Add to a global network of citizen scientists monitoring seismic activity

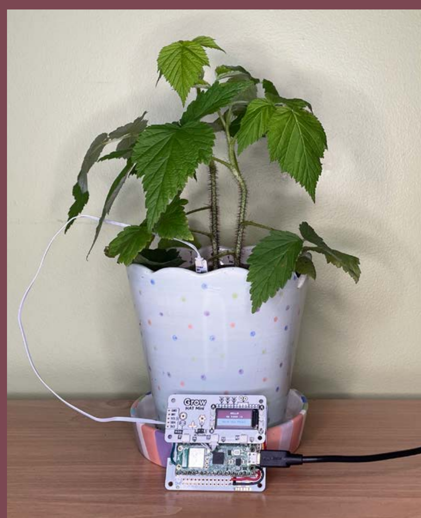
Monitor your plants with Pico

Get a text when your hydrangeas need some hydration

rpimag.co/picoplant

Home automation doesn't just have to involve controlling your lights and your heating – you can also use similar techniques for maintaining your plants. Moisture sensors are a very common component for Raspberry Pi projects and the Grow HAT from Pimoroni makes it easier to use them.

The extra trick to this project is that it will then send you emails with updates – inspiring messages in the morning, as well as reminders that it needs a bit of water. This project also makes use of a little hack to get a Raspberry Pi HAT to work with Pico – it uses a lot less power than a full Raspberry Pi too, making the project much greener.



▶ A Grow Kit monitors the soil moisture

◀ A Perma-Proto HAT is used to connect Pico to the Grow HAT

It will send you emails with updates

Home Assistant box

Fancy completely automating your home? Home Assistant Yellow (**rpimag.co/hayellow**) makes use of a Raspberry Pi Compute Module 4 to control all your network-enabled appliances from one handy little box.

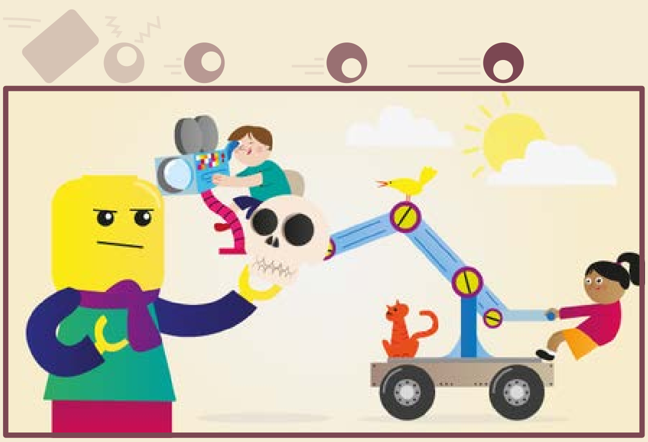
Record stop-motion videos with a Camera Module

Become a movie-maker with some Lego and a lot of patience

rpimag.co/stopmotion

We can show you to how to take simple pictures or videos with a Raspberry Pi and a Camera Module, but we think it's a lot more fun to create a hybrid setup that aids in stop-motion photography.

Very simply, this project lets you take a photo at a press of a button and then waits for you to press the button again for the next photo. In that time, you rearrange what's in the preview window on your monitor, creating a frame of animation each time. You can then stitch these together with some code to create a final product. With some programming tricks, you can even have a ghostly version of the previous frame on screen to aid you in setting up the next shot.



▲ This is a project from our colleagues over at the Raspberry Pi Foundation

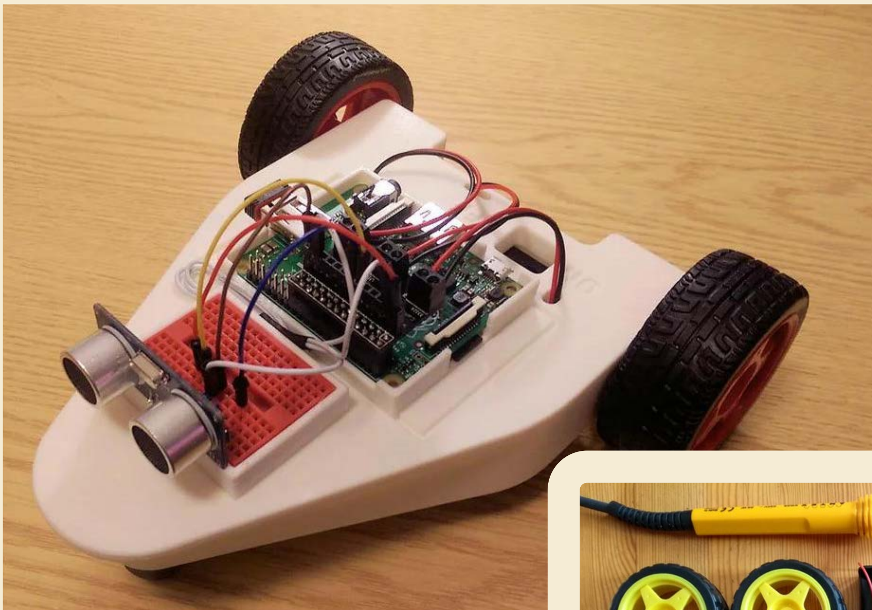
▼ Orange you glad you made a film?

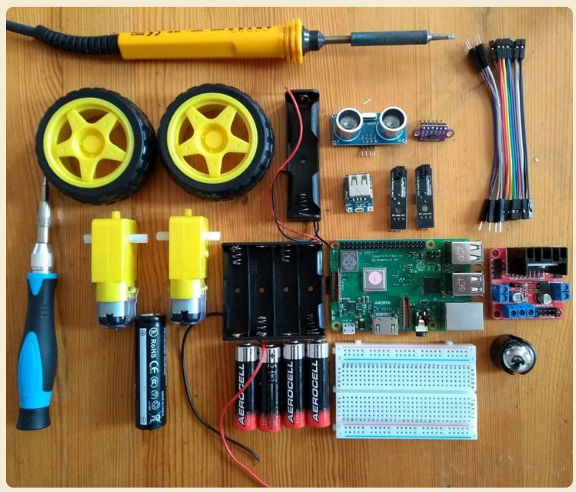
Easy AI recognition

Using a Raspberry Pi AI Camera, you can perform computer vision tasks much more easily than with a regular Camera Module. Check out issue 157 (**rpimag.co/157**) for our Face Recogniser tutorial – maybe you can use it to create a camera that labels your friends.

Create a robot

Make a friendly automaton with a suite of custom sensors





rpimag.co/buildbuggy

The CamJam EduKit 3 is the perfect introduction to building your own robot. It's a small and inexpensive kit that pairs with a full-size Raspberry Pi or Raspberry Pi Zero to create a customisable machine that you can use as both a remote-control car, and to experiment with robot automation.

It comes with an ultrasonic distance sensor and line followers – classic robotics sensors used for navigation. Assembly is fairly straightforward and the Raspberry Pi Foundation tutorial even makes use of the box it comes in to build it; no parts wasted!

▲ A 3D printed chassis is available, but you can actually use the box from the kit to make the robot

▶ There are a lot of parts, but minimal soldering involved

Robotic conflict

The creators of the CamJam EduKits also created the Cambridge Raspberry Jam, or CamJam, and its spin-off Pi Wars! It's a robotics competition for schools where robots compete in various challenges such as line following, obstacle course navigation, balloon jousting, and much more: **piwars.org**

Build a smart mirror

Check on yourself and your day with one futuristic piece of furniture

Info screens everywhere

Like the idea of reusing an old screen? Raspberry Pi has a kiosk mode that allows it to boot straight to a browser – perfect for a recipe computer in the kitchen, or a little info board with train times before you head out the door.

◀ A small IKEA frame with a small monitor does the trick for a smaller version

▲ There's not many components to a smart mirror

magicmirror.builders

Smart mirrors, aka magic mirrors, are one of those projects that every Raspberry Pi maker needs to do once. On the surface, it may seem like a complex and very advanced project; however, it's actually fairly straightforward. The hardest part can be constructing the frame, which, if you can't find a suitable pre-made frame at IKEA, involves a simple bit of carpentry.

Some reflective two-way mirrored wrap on a big old TV and installing the Magic Mirror software are the other main steps, and the latter is very easy to configure with plenty of add-ons too.

RP2350 Pico W5

It's Raspberry Pi Pico 2, but with a lot more memory. By **Rob Zwetsloot**

Elecrow **rpimag.co/picow5** £5 / $7

SPECS

DIMENSIONS:
51mm × 21mm, same as Raspberry Pi Pico

CONNECTIVITY:
USB Type-C, wireless LAN 802.11a/b/g/n 2.4GHz/5GHz, Bluetooth, 40-pin GPIO

STORAGE:
8MB flash

In order to ensure maximum compatibility with Raspberry Pi Pico use-cases, Pico 2 and the rest of its line was shipped with a USB Micro B port for connectivity and power. As the chip that powers Pico 2, RP2350, is available for anyone to use as they see fit, one of the most common alternatives we see for Pico 2 is a board with USB-C for power and data.

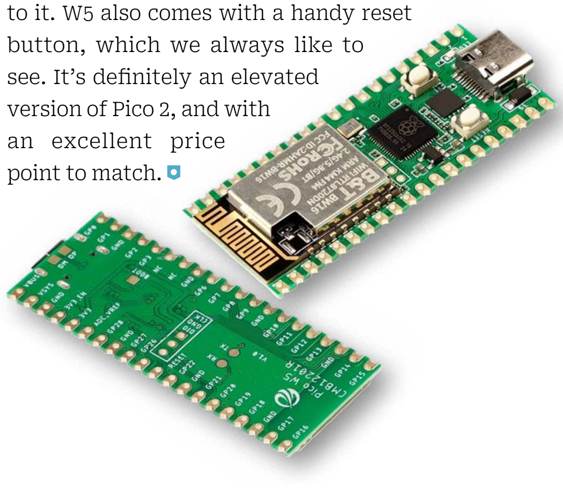
Elecrow has decided to go a little further, and a little different, by creating a version of Pico 2 that not only has a USB-C port, but also has a bump in the amount of storage it uses – doubling the 4MB of onboard storage to a whopping 8MB. Due to the way Pico is used as a microcontroller, you don't need much storage; however, it's always nice to have a slightly larger limit to work to.

Chunky wireless

This Pico 2 alternative uses a different wireless chip, too. While the standard Pico 2 W adds a noticeable chip to the board for wireless radio, the version here is a fair bit bigger – although it does offer 5GHz connectivity. So while it keeps just about the same dimensions as your standard Pico 2, the extra bulk on the PCB is something to bear in mind.

Otherwise it works like a Pico. With RP2350 powering it, all code is compatible and so is any hardware you can connect to it. W5 also comes with a handy reset button, which we always like to see. It's definitely an elevated version of Pico 2, and with an excellent price point to match.

With RP2350 powering it, all code is compatible



- Even the PCB is green, making it look very similar to a Pico 2
- The pins are the same as on a regular Pico; pads are accessible on the rear

Verdict

A great alternative to your standard Raspberry Pi Pico 2 W at the same price point, but you can't quite slip it into every Pico-specific use-case – just most of them.

9/10

Powered by Raspberry Pi

Raspberry Pi's product family extends to single-board computers, microcontrollers, keyboards, screens, HATs, and accessories. Raspberry Pi RP2040 and RP2350 technology forms an excellent basis for a whole range of other products, whether hobbyist, consumer or industrial-focused.

The Powered by Raspberry Pi certification scheme helps customers recognise the best-quality products built around our own hardware, along with a rigorous accreditation process. We're proud and gratified that there are now hundreds of certified products with Raspberry Pi in one form or other at their centre.

If you've wondered about the quality control checks we undertake, you can find detailed information about the compliance regulations and testing procedures for every Raspberry Pi product at **pip.raspberrypi.com**. And for anyone keen to become a Powered by Raspberry Pi manufacturer and proudly sport the bespoke logo signalling your product has our official seal of approval, head to:

rpimag.co/poweredbyraspberrypi

APPLY TO POWERED BY RASPBERRY PI

This month, we highlight a specialist photogrammetry kit, a fun party photo booth, and an astrophotography controller. We also cover a performance-grade musical sequencer, multisensor module for the home/office, tiny RP2350-based board, and an educational robot.

rpimag.co/poweredbypiapply

Horace photo booth

France | horace.fr

A photo booth can turn a great party into a really memorable one, with instant snaps of treasured or downright hilarious moments. This portable Raspberry Pi 5-based photo booth aims to bring joy to partygoers regardless of location. It makes use of Raspberry Pi 5's two USB 3.2 ports to offer 2K 30fps live video from its two cameras and is controlled via two large GPIO-based buttons. Mounted at different heights, the twin cameras ensure children and wheelchair users can join in the fun. The dye-sublimation printer takes just 25 seconds to process each photo.

Atov lx-euclid

Germany | rpimag.co/lxeuclid

If you like digital musical instruments that are tactile and quirky, Atov's lx-euclid four-channel trigger sequencer is likely to catch your eye. Two colourful capacitive touch-sensitive rings are used to apply mutes, fills and resets, making for an intuitive live remixing or studio experience. Endless pattern variations are made possible thanks to the numerous save and preset options along with twists on Euclidean rhythms and exponential algorithms. A burst generator has recently been added to the plethora of expressive audio options on this intriguing RP2040-based device. Atov even offers a DIY build version with instructions and supporting GitHub repo, adding an extra dimension to this fun but rewarding experimental music machine.

Seeed Studio XIAO RP2350

China | rpimag.co/xiaorp2350

Seeed Studio's XIAO range consists of thumb-sized microcontroller units designed for projects in which space constraints are an issue but wireless connectivity is a must. This Raspberry Pi version is built around the dual-core 150MHz RP2350 microcontroller (as used in Raspberry Pi Pico 2) with 2MB flash memory and 520kB SRAM, and has several LED activity indicators.

Programmable in MicroPython, CircuitPython, C/C++, and Zephyr, the tiny module can be powered via a USB-C connection, and is ideal for wearable computing projects. A useful guide at **rpimag.co/xiaorp2350guide** highlights how it can be used for machine learning and, via the PlatformIO interface, programmed using the Arduino IDE. A host of add-ons for Vision AI, mmWave, Grove modules, and CAN bus make it ideal as a development tool. Not bad for something costing just $4.90.

PalaeoPi TablePi2

UK | rpimag.co/tablepi2

Photogrammetry involves precisely mapping the points between objects, recording and interpreting photographs, and spotting patterns. It's used in all sorts of scenarios, ranging from games animation to surveying.

Powered by Raspberry Pi 4, TablePi2 is a turntable photogrammetry kit that comes in a ruggedised briefcase and works with more than 2000 different digital camera models. It can capture as many as 108 RAW 24MP photos in less than three minutes (if you disable simultaneous image downloads), which can be stored on a USB drive, and supports the use of up to three cameras simultaneously for optimal object capture. The turntable is strong enough for objects up to 10kg to be rotated and minutely photographed, giving it a broad appeal including to archaeologists and art archivists. The 7-inch touchscreen makes controlling, pausing and reviewing shots a breeze.

Sfera Labs Exo Sense Pi

Italy | rpimag.co/exosense

Compute Module is at the heart of many of Sfera Labs' industrial and smart home products. The Exo Sense Pi is no exception, packing CM4 and sensors for temperature, light, humidity and PIR motion-detection in an unobtrusive €266 wall-mounted unit. It has a speaker and microphone for two-way communications; lights to alert users of unexpected spikes in temperature, poor air quality or other issues; and connects via Wi-Fi, Bluetooth, or an Ethernet cable. A Lite version omits the 16GB eMMC storage option that security-minded users might use for video recording. Surge protection and the real-time clock with backup battery make the quad-core smart device a great option in small business settings.

AstroLink 4 Pi

Poland | rpimag.co/astrolink4pi

Poland's AstroLink shop operates as "your smart bridge to the stars" and is run by long-standing astrophotography enthusiast AstroJolo (aka Łukasz Socha). He spent more than two years setting up the rig in the not particularly astro-friendly skies near Nieborowice. "From my backyard, it's easier to reach faint fuzzies with a CCD camera than an eye," he observes. The AstroLink 4 Pi is controlled by either Raspberry Pi 4 or 5 and is a 12V DC-powered all-in-one astrophotography controller. It can measure and compensate for the sky temperature as well as the prevailing weather conditions and is compatible with many well-known third-party stepper motors in addition to AstroLink. It also has a power management hub for digital SLR cameras, USB hubs, and other essential photography gear.

BerryBot

Turkey | rpimag.co/berrybot

Robotistan makes some great educational robots and STEM kits and prides itself on being Turkey's maker market. Equipped with an RP2040-based motherboard, BerryBot is a plywood build-it-yourself robot aimed at middle schoolers and intended to help them develop their problem-solving and programming skills while also being a fun way to improve their dexterity and hand-eye coordination. Assembling the kit is intentionally straightforward, so children can get started with controlling the robot using the PicoBricks GO app on a phone or tablet. There are more than 25 input and output sensors and lights to customise the setup. Ready-to-use modes include obstacle avoidance, light and line tracking, along with a 'sumo mode'. The PicoBricks IDE offers block-based and Python programming, and students can challenge their friends to coding battles.

ONLY THE **BEST**

Environmental sensors

By **Phil King**

Once equipped with appropriate external sensors, a Raspberry Pi computer (or Pico) can be used to measure all sorts of environmental data, as well as detecting dangerous gases and particles in the air. You can log data locally and/or upload it to the cloud such as to a web dashboard or citizen science project.

There are quite a few suitable HATs and boards around, many of which have multiple onboard sensors to measure more than one thing – typically temperature, atmospheric pressure, relative humidity, plus light and/or noise levels, and sometimes air quality using a particulate matter sensor. The latter samples the air at regular intervals and analyses it to detect tiny particles of different sizes, some of which can be hazardous to health.

We also take a look at a kit for the garden – by measuring soil moisture, it can tell you when plants need watering, or even water them automatically. Let's get sensing...

Enviro Urban

Pimoroni | £59 / $65 | pimoroni.com

The PM sensor on the front detects fine particles in the air

Powered by an onboard Raspberry Pi Pico W, the Enviro Urban is packed with environmental sensors. A Plantower PMSA003I particulate matter sensor is mounted on the front of the board. Sampling the air at regular intervals, this detects the presence of very fine particles such as pollen, dust, mould spores, and metal ions. There's also a BME280 sensor for temperature, pressure, and humidity readings, plus a MEMS mic to measure noise. A Qw/ST port lets you add breakouts.

The Enviro Urban comes preloaded with software and setup entails connecting to the board's wireless hotspot on a computer or phone; you can then connect it to your wireless network and tell it how often to take readings and where/how to upload them – Adafruit IO, InfluxDB, MQTT, or HTTP endpoint.

An optional Accessory Kit includes a weatherproof enclosure and 3×AA battery pack. Battery life is aided by an onboard RTC that can wake the board up from a power-saving deep sleep.

Verdict

An easy way to get started with environmental monitoring.

Enviro + Air Quality

Pimoroni | £54 / $60 | pimoroni.com

Featuring four built-in sensors and a mini colour LCD, this Zero-size HAT turns your Raspberry Pi into an environmental monitoring station. While it's designed for indoors, you could put it in a weatherproof housing to use it outside.

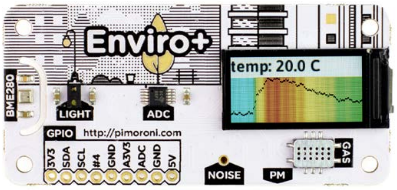
Positioned on the left edge, to avoid the heat from Raspberry Pi's CPU, a BME280 weather sensor monitors temperature, barometric pressure, and humidity. A smartphone-style LTR-559 light and proximity sensor detects the ambient light level (and can be used as an input if you put your finger on it). A tiny MEMs microphone measures sound levels, useful for monitoring noise pollution. Finally, there's a MICS6814 analogue gas sensor that can detect three different gas groups: reducing, oxidising, and NH3 (ammonia).

As the product name suggests, there's also a port to plug in an optional PMS5003 particulate matter sensor (supplied separately) to monitor air quality. You can also upload environmental data to the Sensor.Community citizen science project.

▶ The onboard LCD makes it easy to view data without a monitor

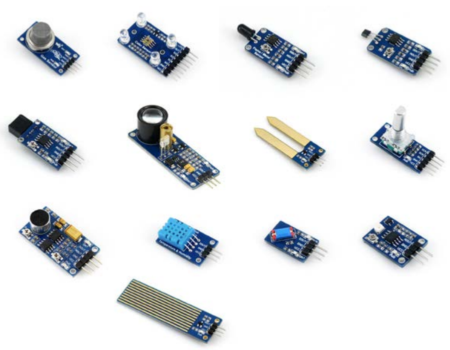
Verdict

A mini environmental monitoring station with optional air quality sensor.



Waveshare Sensors Pack

Waveshare / The Pi Hut | £40 / $53 | waveshare.com / thepihut.com



▲ Sensor types include temperature/humidity, gas, flame, moisture, and liquid level

If you want to go the DIY route for your environmental monitoring, you can connect individual sensors directly (or via a breadboard) to Raspberry Pi or Pico's GPIO pins to take readings. While many sensors can be purchased separately, a variety pack may be a good option to cover various bases. We made good use of this Waveshare Sensors Pack in our four-part 'Explore the sensory world' tutorial series starting in issue 111 (**rpimag.co/111**).

The pack comprises 13 components, including sensors to detect/measure gases, flames, moisture level, sound, temperature/humidity, colour, UV, and liquid level. So it covers quite a lot of bases for a variety of projects such as a basic weather station or garden plant monitor. The helpful Waveshare wiki includes technical details, connection diagrams (for Pico), and code examples.

Note that while most of the sensors have both digital and analogue output pins, some only have analogue, in which case you'll need to connect them to a Raspberry Pi computer via an ADC chip (on Pico, you can just use its special analogue input GPIO pins).

Verdict

A wide range of sensors for building your own custom projects.

Enviro Weather

Pimoroni | £30 / $33 | pimoroni.com

A while back, your author set up a weather station (see the tutorial in issue 119: rpimag.co/119) that was based around Pimoroni's Weather HAT board and sensors kit: wind vane, anemometer, and rainfall gauge. While that HAT is still available, Pimoroni also makes the similar Enviro Weather, a standalone device powered by an onboard Pico W. Its more slimline design should make it easier to install in a weatherproof enclosure.

There are two onboard sensors: a BME280 for measuring temperature, atmospheric pressure, and relative humidity, along with an LTR-559 light and proximity sensor (as found on the Enviro + Air Quality board).

In addition, you get two RJ11 ports for connecting external wind and rain sensors (not supplied), plus a Qw/ST port for breakouts, and a battery connector. Unlike the Weather HAT, there's no onboard LCD, but it's easy enough to upload data to the cloud, such as to an Adafruit IO dashboard.

Verdict

A standalone alternative to a Weather HAT, albeit with fewer features.

This weather station board is powered by a Pico W on the rear

SparkFun Weather Meter Kit

SparkFun | £61 / $80 | sparkfun.com

Pimoroni no longer sells the Weather Sensors Kit to go with its Weather HAT, but you can still buy very similar sensors online, such as this three-piece kit from SparkFun. It comprises a trio of sensors: wind vane (for direction), anemometer (which spins around in the wind to measure its speed), and self-emptying, tipping-bucket rain gauge. Each sensor has a long cable with a standard RJ11 connector to work with boards such as the Weather HAT or Enviro Weather.

Wind direction and speed sensors, plus a rainfall gauge

The kit also includes a metal mounting pole that you can attach to a vertical surface (e.g. a fence) using the supplied metal gear clamps. The wind vane and anemometer fit on a plastic arm on the top, while the rain gauge is clamped to the pole. Full assembly instructions are given online.

Verdict

Three sensors that are ideal for an outdoor weather station.

Grow Kit

Pimoroni | £33 / $37 | pimoroni.com

As well as monitoring weather and air conditions, you can use sensors to keep an eye on your garden or indoor plants. The Grow Kit and optional accessories make it even easier. It's based around the Grow HAT Mini for Raspberry Pi, featuring a tiny colour LCD to show data, a LTR-559 light and proximity sensor, piezo buzzer (for audible alerts), and four tactile buttons.

Most importantly, it also features three JST SH 3P connectors to plug in the kit's three capacitive moisture sensors. Attached by 35cm cables, these can be used to monitor the moisture levels for your plants. It can then alert you to when they need watering, as detailed in the online getting started guide.

You can even set up an automatic watering system using mini submersible pumps (available separately) connected to the board's three Picoblade ports. With the pumps placed in a water reservoir, silicone tubing is used to transport it to the plants. To help you, there's a detailed step-by-step online tutorial.

Verdict

A handy way to monitor soil moisture levels for your plants.

▲ The Grow HAT Mini has a mini LCD to show useful info

WEATHERPROOF ENCLOSURE

Pimoroni | £14 / $15 | pimoroni.com

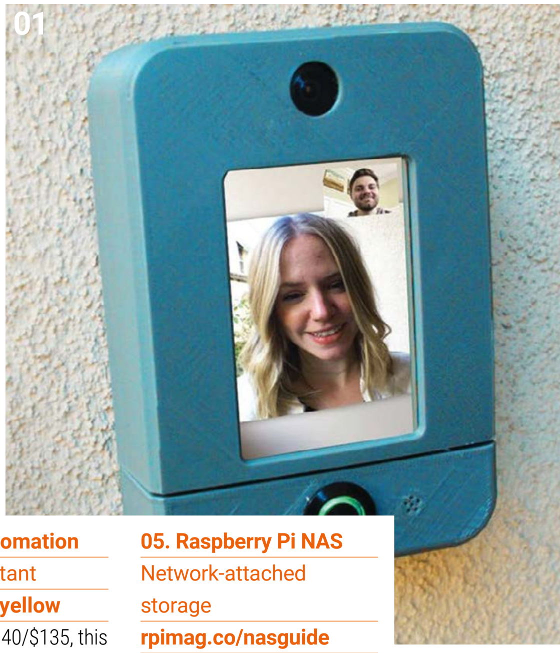
To place electronic sensors outdoors, you'll need a way to keep them dry and, for accurate temperature readings, out of direct sunlight. A piece of PVC pipe may suffice, but a proper weatherproof enclosure is better. This one is a 'Stevenson screen' type that maintains good airflow.

▼ Protect your outdoor sensors with an enclosure

10 amazing:

local network projects

Power up your home network with one, two, or ten Raspberry Pi projects



We live in an era where we can do a lot from home in terms of both work and play, along with wireless devices that can make use of convenient home networking too. Raspberry Pi can make your home network just that much better – we use it a lot ourselves – and here are just some ways to do that.

01. Smart doorbell

Who's there?

rpimag.co/rpidoorbell

Not only does this replace your off-the-shelf smart doorbell with cheaper and more customisable hardware, you can also use it like an intercom. Take that, Nest.

02. Raspberry Pi router

Manage your network

rpimag.co/rpirouter

The router from your ISP can be hit or miss, but with Raspberry Pi you can make your own router with all the custom settings you might be missing out on elsewhere.

03. Multiroom audio

Sonos who?

rpimag.co/mraudio

If you're a big music fan, and/or like throwing house parties, it can be fantastic to have the same music playing throughout the house, synced up properly too! You can do that here.

04. Home automation

Prebuilt assistant

rpimag.co/hayellow

Costing from £140/$135, this kit is powered by a Raspberry Pi Compute Module to make home automation that much easier. It runs the powerful Home Assistant by default to connect to all your devices.

05. Raspberry Pi NAS

Network-attached storage

rpimag.co/nasguide

This simple Raspberry Pi project can be one of the most useful, especially if you regularly move between several devices like we do but need access to the same files everywhere.

06. Pi-hole

Secure your network

pi-hole.net

Web ads are a common vector for cyberattacks, and with browsers removing ad blocker add-ons, the best way to solve this is to block them entirely from your network.

07. Raspberry Pi Zero NAS

Lightweight storage

rpimag.co/naszero

You could simply swap out a Raspberry Pi Zero with a full-size Raspberry Pi, or make a very tiny NAS the size of a plug you put into a socket, making maximum use of the reduced dimensions.

08. PiFi

Extend your Wi-Fi

pifi.org

Whether you need to create an impromptu wireless network extension (useful in the summer) or add some better Wi-Fi to an old hotel room (useful for holidays), PiFi is easy to set up and even has inbuilt VPN options.

09. Network monitor

Log your packets

rpimag.co/netmonitor

If you're having network issues, you'll wish you had been logging all your network traffic in advance to figure out where the problem might be. We know from experience.

10. DIY home automation

Fully customisable assistant

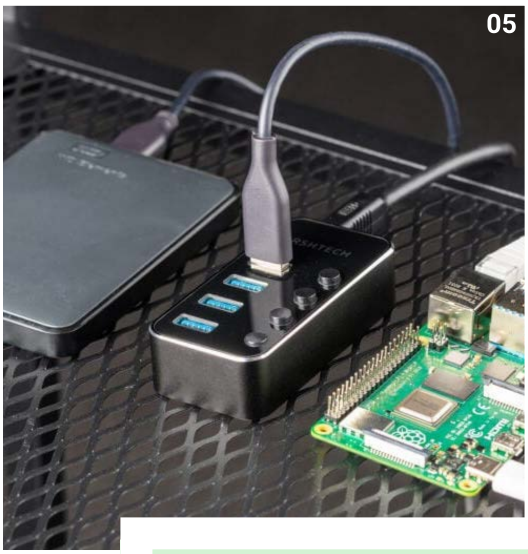
rpimag.co/handbook2026

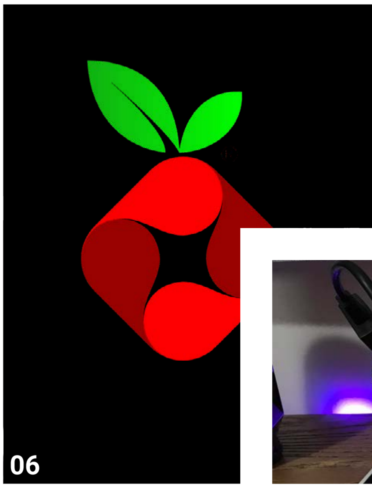
If you'd rather do the hard work yourself, we have a great guide on how to make your own custom home automation device that uses Home Assistant to do the exact same great things.


02

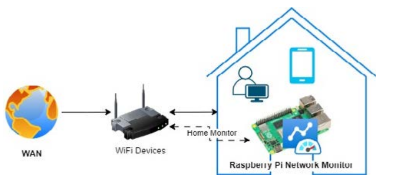

03


04


05


06


07



09

08


10

Maker Monday

Amazing projects direct from social media!

Every Monday, we ask the question: have you made something with a Raspberry Pi over the weekend? Every Monday, our followers send us amazing photos and videos of the things they've made.

Follow along to #MakerMonday each week over on our various social media platforms!

01. An extra cool function to an already very cool project

02. A very neat and tidy package

03. You too can improve your old projects whenever you like

04. It's been a little while for us making any Zero-based camera projects, too

05. Lidar is a powerful tool, when you get it to work

06. Why get social media replies on your phone when you can get them physically?

07. A melding of two products to make something your own

08. RP2350 projects absolutely count. An impressive emulator!

DerTobi
@DerTobi@mastodon.social

@rpimag I found a good use for the second core of the RP2350: the Open DMG Display now supports USB video output.

embedded-ideas.de/posts/251116…

Translate

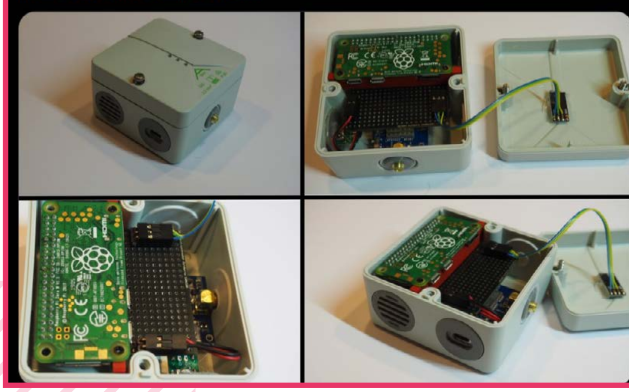
embedded-ideas.de · Nov 16

USB video output for the Open DMG Display

01 USB video output has been added to the Open DMG Display. It is now possible to cap…

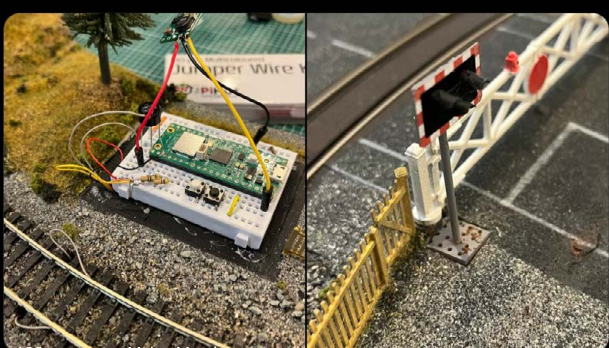
Ulli Sundermann **02**
@ullisun58

Hi #MakerMonday my weekends are always dedicated to my Lawn Mower Robot projects. This weekend I completed the GPS-RTK basestation. LC29H(BS) and a 👆 Pi Zero W V1.1 is included. Is very cheap and works efficient. The python Code is available on github.com/ullisun/GPS_RTK



Pater Practicus **03**
@PaterPracticus

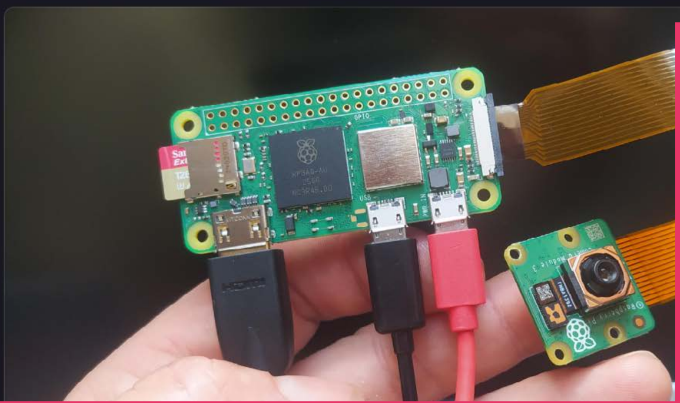
Revisiting and updating an older project on the model railway. I've designed new 3D printed light cowls for the level-crossing signs and tidied up the power supply for the @Raspberry_Pi Pico controller, so it runs off the AC track voltage. #MakerMonday



04

concretedog
@concretedog · Nov 17

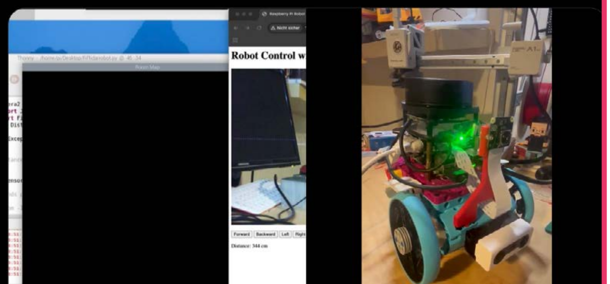
@rpimag this is for an upcoming article but I've been playing pi zero 2w and camera module v3 writing some scripts so it boots and automatically shoots some video. Been ages since I mucked around with this stuff... using the shiny RPI monitor which also rocks! #MessyDesk



05

Roland Schulz 🕹️⚡🤖🎮👾
@r_schulz_maker

Hi #MakerMonday, I got some @Raspberry_Pi projects in process. The Lidar upgrade on my LegoBot is installed and and works perfectly. The laser scan is clearly visible but the GUI Room Map is still black😩. This means additional night shifts to fix the problem.
#micropython



06

Maddie D. Reese
@maddiedreese

Yes, my first ever Raspberry Pi project! Connected a website and a printer to a Pi to allow anyone around the world to send me a message. I've received over 500 so far!

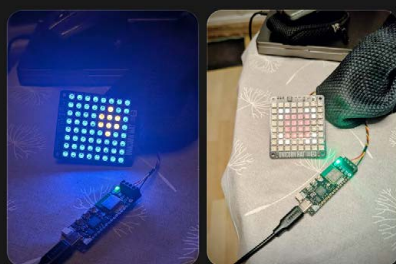
Maddie D. Reese @maddiedreese · Nov 2
Printer went a little viral on /r/sideproject on Reddit!

Here's a sampling of the 500+ messages I've gotten from Italy, Germany, Ukraine, Mexico, Thailand, Bulgaria, Switzerland, Brazil, Kurdistan, and many more!...
Show more

07

briancorteil › **MakerMomday** 17/11/2025

I spent most of the weekend converting the original @pimoroni Unicorn hat Raspberry Pi Python library for MicroPython using their Plasma library running on their Plasma boards or their Pirate MicroPython. Most examples run without modification :-)



08

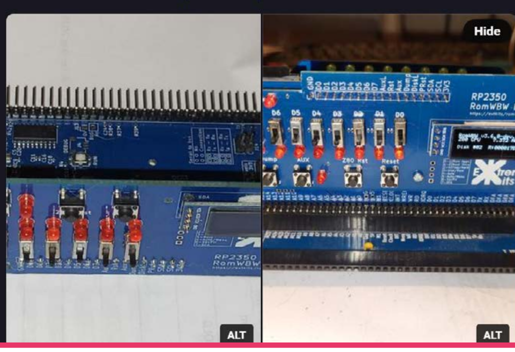
Extreme Electronics
@Extelec@mstdn.social

@rpimag If RP2350b counts ?

Z80 emulation running RomWBW as "bios" with CPM via SD card. Front panel interface to the os and an io interface to an RC2014 bus.

Made as a kit for ExtremeKits.

Details at extkits.co.uk/product/rp2350-r...



Raspberry Pi smart telly

We got an email from ten-year-old Anthony, who was excited to show us that he got the Kodi media player working on Raspberry Pi hooked up to his TV, an excellent project for Raspberry Pi newbies.

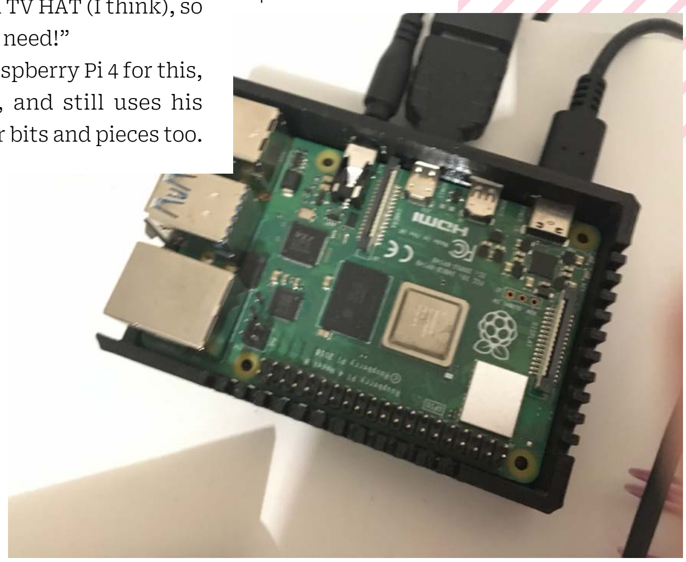
"Today I have made a smart TV powered by a Raspberry Pi to replace my normal (and terrible) smart TV system," he says. "My existing TV can't even run YouTube without it crashing in the first ten minutes, so I knew the main goals I wanted to achieve...

1. Make it view photos and videos
2. Make it run YouTube, BBC iPlayer and/or Netflix
3. Watch TV shows offline

"I think I knew exactly what type of software I would use that can be capable of all of this! And that is Kodi. Plus, Kodi can also play games with an extension and can be your TV with a TV HAT (I think), so it's more than what I need!"

Anthony used a Raspberry Pi 4 for this, an excellent choice, and still uses his Raspberry Pi for other bits and pieces too.

▼ The simple setup needed to get a powerful smart TV



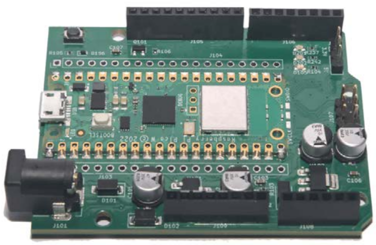
Media Centre

Want to learn more about making media centres with Raspberry Pi? We have a great digital guide on building your own media centre (**rpimag.co/mediaplayer**) and you can also check out issue 155 (**rpimag.co/155**) for a similar guide.

Crowdfund this

Great crowdfunding projects this month

Pico@ardu



Fancy using an Arduino shield on your Raspberry Pi Pico? This project's aim is to facilitate exactly that. Simple add a Raspberry Pi Pico, then pop the shield on top. It converts any voltages to be safe between the two as well.

▶ **rpimag.co/picoardu**

Breadboard-2350

Billed as a 'truly breadboard friendly' RP2350 development board, it has 30 pins along one side instead of the usual 40 over two, with the idea being that you can use the entire length of a half-size breadboard instead of having a Pico bridge the middle. There's extra I/O in the design, too.

▶ **rpimag.co/bboard2350**

Your Letters

Aliens!

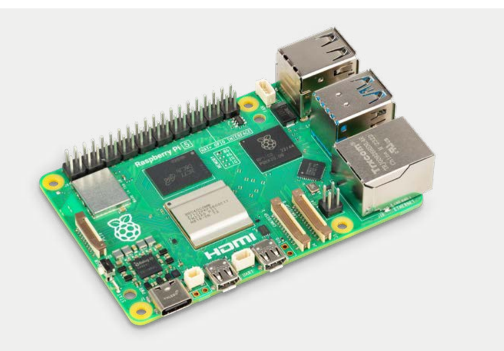
I've been struggling to get my head wrapped around the DreamHAT+ Radar board. I remember reading your review, and thought it sounded promising, and I've had emails from various online hardware purveyors telling me to buy one, but not really thought what I could do with one that would justify the purchase price.

Rob Smith's *Aliens* tracker changed that. Flippin' heck, that's a quality bit of kit! The ability to find aliens crawling around in your walls is a godsend all year round, not just at Halloween. I feel like I did when I first realised you could get GPS modules for Raspberry Pi: this is proper technology that should be reserved for grown-ups, but somehow is now available for me to play with. What a world! Only one question remains: which is better: *Alien*, or *Aliens*?

Pat, via email

Our knee-jerk reaction to your question would be to say that *Alien* is the better film. But we find ourselves quoting from *Aliens* far more than we do the original film, in particular the line about taking off and nuking the site from orbit, which gets more and more applicable each day. We even have soft spot for *Alien 3*, with its prison governor played by Barnsley's Brian Glover. But to return to the world of Raspberry Pi for a moment: the DreamHAT+ Radar is indeed a serious bit of kit. One of the brilliant side effects of Moore's Law is that all sorts of cool sensors are getting cheaper all the time, enabling manufacturers to turn them into maker-friendly boards like the DreamHAT+ Radar. Get hold of one, play with it, have fun, and be grateful that we live in a world where you can do so.

Rob Smith's M314 Alien MotionTracker



Could a Raspberry Pi 5 help to keep you warm?

Keep warm

I saw a story last week about a household in Essex that got its heating bill reduced by playing host to a Raspberry Pi cluster. I don't understand the physics, but the excess heat from whatever the computers were doing went towards heating the house, and it saved them money. Where do I sign up? And do the computers need to be doing anything useful or can I just be playing retro games?

Edin, via email

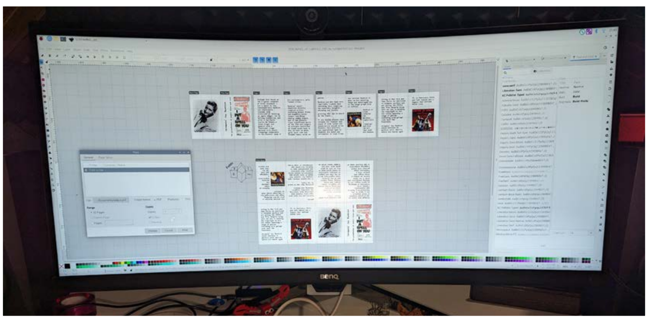
We saw that article too (on the BBC News site). We can see how it works at small scale, in a household that needs a lot of heating (I think I remember reading that their heating bill was in the region of £300 a month, which is loads more than we're paying). In a house where you're out most of the time, or you just don't want it too warm, a cluster of Raspberry Pi 5 computers working 24-7 might produce more heat than we can use. But we don't understand the physics of it either.

Zine publishing

I read with interest your tutorial on desktop publishing with free software. After an early career spent moving tiny pieces of metal around to apply ink on to paper (typesetting, as we used to call it), I retrained in web design and never thought about print again... but it's still here! Good on you for showing how everyday tools like a Raspberry Pi can lay out magazines, posters and whatnot. We get told that print is dead, but for some applications, you can't beat it.

Neil, via email

We certainly like to think so! Print has many advantages over glowing pixels: it doesn't suffer from screen glare, it doesn't run out of batteries, and once it's in your hands, you know it's not going to disappear if a server somewhere goes down or a link breaks. There's also something cool about self-made print: the word 'amateur' comes from the Latin word for love. If you're an amateur, you're doing something for the love of it, and that comes across in every zine project we've ever seen. Contrast this with websites, where very often it looks the designer hates everything about human eyeballs. We're delighted that free software has so many options for publishing projects, and we're also delighted that you can choose how you want to read *Raspberry Pi Official Magazine* – whether digitally or in print.



▲ You can use Inkscape to create and publish a zine

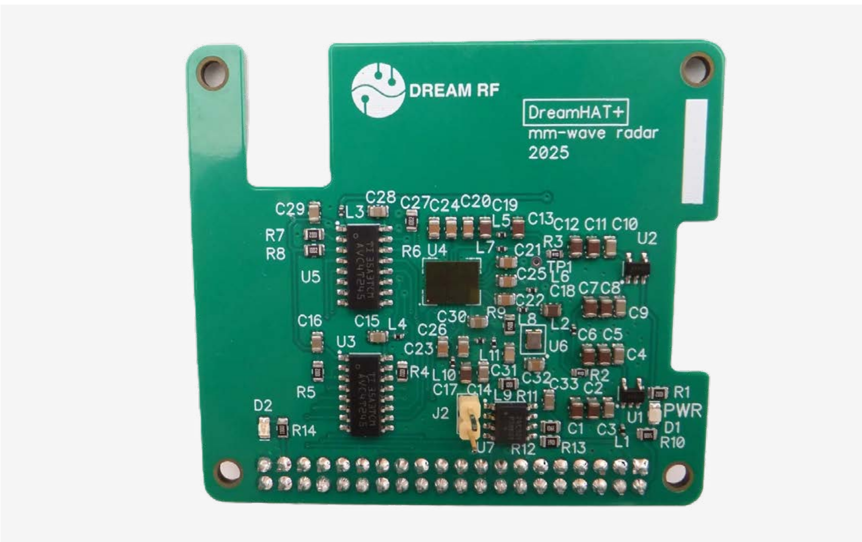
Contact us!

- @rpimagazine
- @rpimag
- rpimag.co/facebook
- magazine@raspberrypi.com
- forums.raspberrypi.com


PROJECTS FOR MAKERS & HACKERS
BUILD A CUSTOM KEYBOARD WITH RASPBERRY PI PICO
CUSTOM CONTROLS WITH A 3D PRINTED BIG BUTTON
BOOK OF MAKING
2026
CONTROL PROGRAMMABLE LEDS FOR DAZZLING LIGHTING
EVERYTHING YOU NEED TO KNOW ABOUT BATTERY POWER
FROM THE MAKERS OF RASPBERRY PI OFFICIAL MAGAZINE

Win 1 of 3 DreamHAT+

DreamHAT+ is a Hardware Attached on Top (HAT) kit that gives Raspberry Pi a 60GHz millimetre-wave radar for you to play with. That means adding spatial sensing to robots, scanning a room for moving objects, or even building a working motion sensor straight out of *Aliens*, as done by Rob Smith – we featured this in a project showcase last issue.



Head here to enter:

rpimag.co/win

Learn more:

rpimag.co/products

Terms & Conditions

Competition opens on **17 December 2025** and closes on **29 January 2026**. Prize is offered to participants worldwide aged 13 or over, except employees of Raspberry Pi Ltd, the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from Raspberry Pi Official magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram, Facebook, Twitter (X) or any other companies used to promote the service.

Official Magazine
#162

Next Month

ROBOT ARMS

Get to grips with robotics

On sale 22 January

PLUS!

Build a Pico timer

Astronomy projects

Sony IMX500 Brain Builder

Editorial

Editor
Lucy Hattersley
lucy@raspberrypi.com

Features Editor
Andrew Gregory
andrew.gregory@raspberrypi.com

Features Editor
Rob Zwetsloot
rob@raspberrypi.com

Sub Editor
Phil King

Advertising
Charlotte Milligan
charlotte.milligan@raspberrypi.com
+44 (0)7725 368887

Design

Head of Design
Jack Willis

Designers
Sara Parodi, Natalie Turner

Illustrator
Sam Alder

Brand Manager
Brian O Halloran

Contributors

David Crookes, Tim Danton,
Rosemary Hattersley, Jo Hinchliffe, Nicola King,
Phil King, Andrew Lewis, Rob Miles,
Richard Smedley

Publishing

Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com

Director of Communications
Helen Lynn

CEO
Eben Upton

Distribution

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

Subscriptions

Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
rpimag.co/subscribe
rpipress@subscriptionhelpline.co.uk

Raspberry Pi PRESS
ISSN: 2977-4403 (Print)
ISSN: 2977-4411 (Digital)

New year, new projects

What ideas and experiments will 2026 bring?

I've probably thought this at the beginning of every year for the past 15 or so but… 2026 truly does sound like a random future year that a 1970s science-fiction writer would use, doesn't it? Aliens, spaceships, moon bases, perfectly balanced meals in pill form, these are the kinds of things we should have now. Although we do have ridiculously powerful pocket computers, electric cars, watch phones, and tiny computers you can pop in other things to power them up. See, I was able to shoehorn Raspberry Pi in there somehow.

> *It really has just become a normal part of (my) life to make something smart with a Raspberry Pi*

It really has just become a normal part of (my) life to make something smart with a Raspberry Pi. Not just in the superfluous way like a smart safe you can open with a good whack, but better TVs, better cooking equipment, better mirrors even. With the AI capabilities of Raspberry Pi (the good machine learning and analysis stuff, not the make an image of a six-fingered dog-man you can marry stuff), this is only going to get better.

Considered upgrades

There's a joke that often does the rounds on my social media circles: 'Tech enthusiasts: my entire house is smart – Tech workers: the only piece of technology in my house is a printer and I keep a gun next to it in case it makes a noise I don't recognise'. Obviously hyperbolic, but I do generally agree. Not everything needs to be connected, mainly because that connection can easily go wrong.

I do want to experiment more in this new year though, while making sure there are plenty of fail-safes so that I don't lose use of the item I'm trying to upgrade, of course. Pico is great for stuff like this, as it's also a little more limited in what it can do – I'm not going to be able to use it to put a graphical web browser on a shelving light display.

Computer changes

You may already know this, but a lot of the magazine is produced on non-Raspberry Pi computers for various reasons. However, it doesn't stop us from doing what we can on a Raspberry Pi. I recently grabbed myself a Raspberry Pi 500+ and it is very, very nice. I love typing on it. As a lot of this job does involve that, I've been working on ways to rearrange my workstation and make it so it becomes my primary machine for work. A few less distractions there as well, which is always a good thing.

One day it would be great to be able to do all the design work on a Raspberry Pi. I bet that day is coming sooner rather than later.

Rob Zwetsloot – Author

Rob is entering the final year of his 30s, so often looks forward to spending weeks sitting perfectly still.

rpimag.co