Perfectly emulate **a classic Amiga computer**

Creative makers **show off unusual clocks**

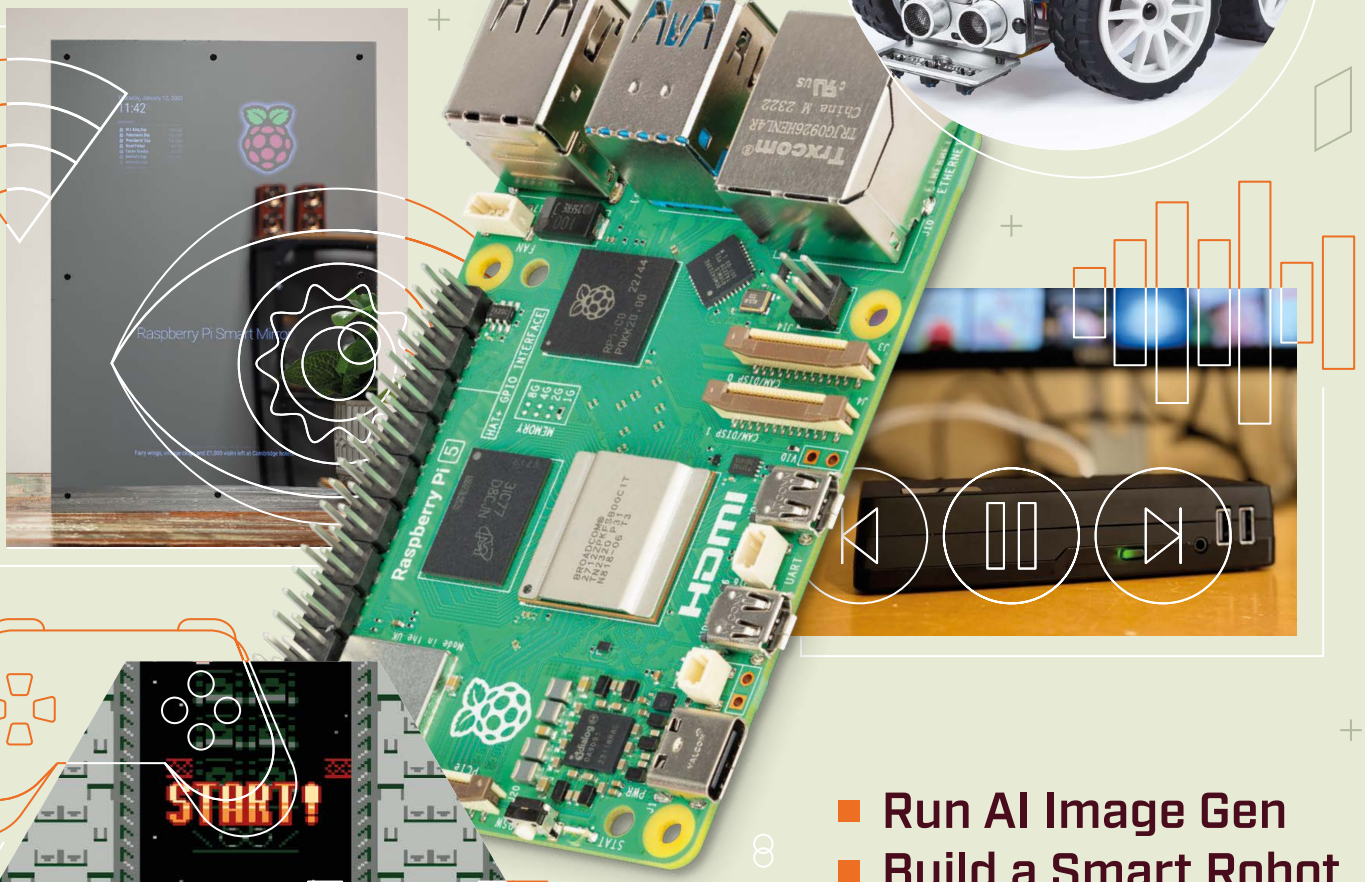
Building a **nostalgic, beige, Raspberry Pi 500+**

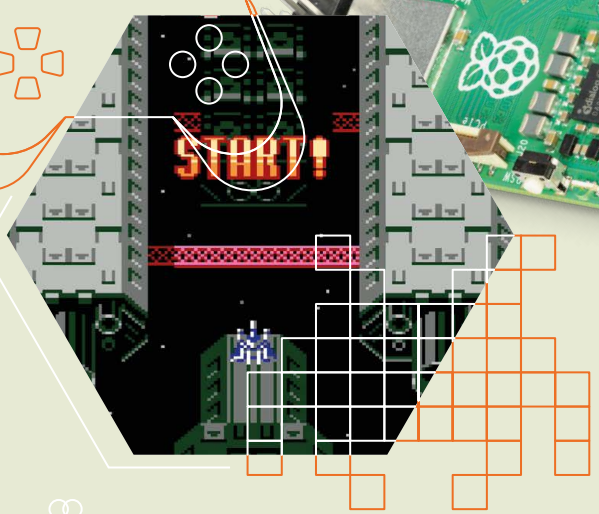
Raspberry Pi

Official Magazine
#164 | April 2026

MIGHTY PROJECTS

1 GB COMPUTER



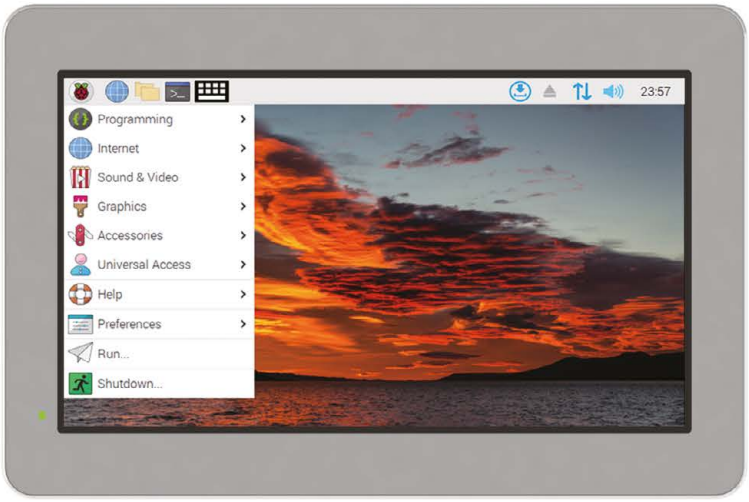


- Run AI Image Gen
- Build a Smart Robot
- Make a Magic Mirror

9 772977 440004 04
£7.99
ISSN 2977-4403 (Print)

Industrial Raspberry Pi

ComfilePi



UL LISTED

Powered by Raspberry Pi

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

COMFILE TECHNOLOGY

Welcome to Raspberry Pi Official Magazine

Editor
Lucy Hattersley
Lucy is editor of *Raspberry Pi Official Magazine* and loves nerdy environments. Honestly, CamJam even had a train set.

rpimag.co

This month I visited the fantastic CamJam (camjam.me) event in Cambridge, which is still going strong after 14 years thanks to the efforts of Michael, Tim, and Brian.

It was fantastic to reconnect with the maker community that gave Raspberry Pi its start in life. Philip Colligan from the Raspberry Pi Foundation and Eben Upton from Raspberry Pi Ltd both gave great speeches on the future of technology and education. Philip has written a paper called *Why kids still need to learn to code in the age of AI* (**rpimag.co/kidscodeai**). I'd encourage you to read it.

AI presents a challenge. We've got a rapidly changing technological environment and the ongoing shortage of SDRAM pushing up prices. Every cloud has a silver lining though and for me, this month, it was experimenting with the new Raspberry Pi 5 1GB. It's liberating to discover what you can do with a super-fast quad-core Arm CPU connected to a much smaller amount of RAM than usual. Like having an Alfa Romeo with a tiny petrol tank. It's super-fast; you just have to keep an eye on the petrol gauge and oil levels.

Raspberry Pi 5 1GB felt like a return to the first single-board Raspberry Pi computer I used to build a 'Wheely' robot, finish my MITx courses, and build early electronics projects.

This month's magazine felt like coming home. I hope you enjoy it.

Lucy Hattersley – Editor

PCBWay

FULL-SERVICE ELECTRONICS MANUFACTURER

PCB Fabrication / CNC | 3D Printing / PCB Assembly / OEM I EMS

CUSTOMIZED TECHNICAL PARTS & PCB MANUFACTURED

WHY CHOOSE PCBWAY?

PCBWay offers a wide range of services including PCB fabrication, PCB assembly and even CNC machining for over a decade, and has earned a distinguished reputation globally in the industry.

Hassle-free ordering

The digital quote-to-order platform puts you in the back seat. Upload a Gerber file and receive feedback soon.

On-time shipping

We maintain a 99% on-time delivery rate, working in three shifts to ensure your packages arrive fast.

Quality assurance

Our technicians have been working strictly to high standards. All the boards will go through the most stringent tests.

Customer support

The customer service teams work inshifts to provide 24-hour support. You can always contact a live customer service person.

Scan the QR code for more details!

CONTACT US:

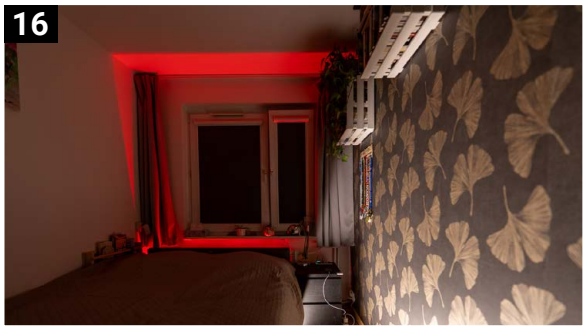
www.pcbway.com

service@pcbway.com

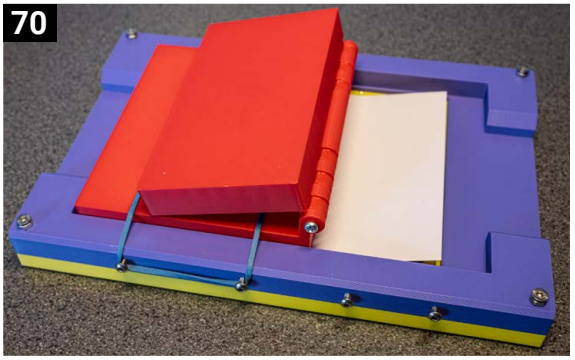
Contents

World of Raspberry Pi

Project Showcase

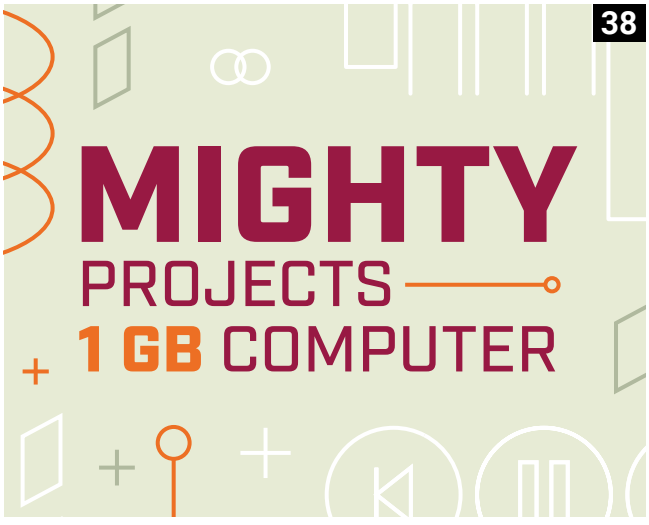


Raspberry Pi (ISSN No: 2977-4403, USPS No: 25-672) is published 12 times per year by Raspberry Pi Ltd, and distributed in the USA by Asendia USA, 701 Ashland Ave, Folcroft PA. Application to Mail at Periodicals Postage. Prices is pending at Philadelphia, PA and additional mailing offices. POSTMASTER: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



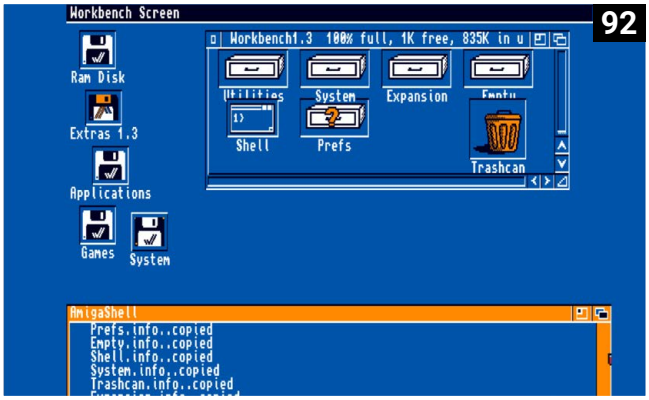
Top Projects

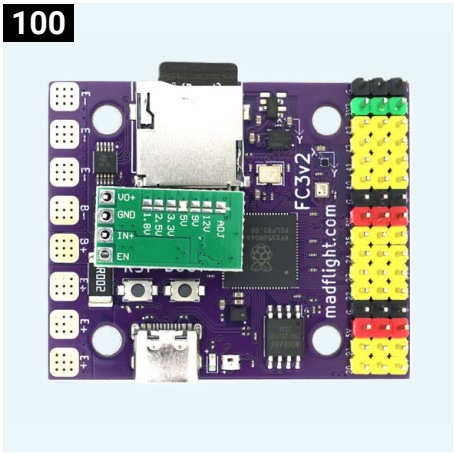
Feature



Tutorials

Feature





Reviews

Raspberry Pi Community

Competition

Raspberry Pi Smart Display Module

New product for displays coming soon. By **Lucy Hattersley**

▲ The module provides HDMI output to support a second independent video stream, along with an M.2 expansion slot for optional AI acceleration

Raspberry Pi has been working with Sharp Display Solutions Europe to develop a new product called Raspberry Pi Smart Display Module.

The Smart Display Module was revealed at Integrated Systems Europe (ISE) 2026 in Barcelona.

The new product is an adapter board for Raspberry Pi Compute Module 5 that is "designed to deliver high-quality, low-power display experiences for professional signage applications," according to Simon Burgess, enterprise business manager at Raspberry Pi.

"With the SDM MPi5 Kit, we are combining the flexibility and performance of Raspberry Pi with the modularity demanded by professional AV customers," said Erik Elbert, senior product manager at Sharp Display Solutions Europe. "This platform is not only about computing power today, but about creating an open, future-ready foundation that can support AI and new use cases directly within the display."

Conforming to the Intel Smart Display Module (SDM) specification (**rpimag.co/intel-sdm**), the Raspberry Pi Smart Display Module slots directly into any device that supports Intel's standard, drawing power from the display itself.

With the computer embedded inside the screen, installations are "clean, reliable, and easy to maintain," says Burgess, "making the Smart Display Module ideal for applications such as flight information systems, retail and corporate signage, and industrial displays."

Designed for users in the audio-visual and digital signage markets, Raspberry Pi Smart Display Module places the power, flexibility, and energy efficiency of Compute Module 5 into compatible display screens, with no external media player, cabling, or power source required.

The module also provides HDMI output to support a second independent video stream, along with an M.2 expansion slot for optional AI acceleration.

"We designed the Raspberry Pi Smart Display Module to be as straightforward to assemble as possible," says Burgess. "Customers can install it themselves without any specialist tools."

Enabling edge AI for digital signage

As organisations increasingly explore AI-powered digital signage, Raspberry Pi Smart Display Module offers an "efficient and practical solution". Designed to integrate easily with compatible AI accelerators, the module enables edge AI processing to take place directly inside the screen. "This allows users to run analytics and AI-driven applications locally, privately, and in real time, without reliance on cloud-based services."

▲ Raspberry Pi Smart Display Module fits into screens that meet the Intel SDM standard

Designed to deliver high-quality, low-power display experiences for professional signage applications

"Raspberry Pi technology is already used by thousands of businesses and powers hundreds of thousands of screens worldwide," notes Burgess. The introduction of Raspberry Pi Smart Display Module "further expands that ecosystem."

Embedding AI capability directly into displays will "enable businesses to innovate rapidly and adapt to changing requirements with an energy-efficient, easy-to-integrate modular solution."

Raspberry Pi Smart Display Module is expected to launch this summer with an expected price of $25 for the board without the Compute Module 5.

Beige is back

Take a fond look back to a time when a machine with 32kB of RAM could cost 10% of the price of a house. By **Andrew Gregory**

Maker

Toby Roberts

Maker-in-chief at Pi Towers, Toby has most recently been playing with AI running locally on Raspberry Pi.

@tobyrobertspi

Back when this writer was at school, before everyone carried a supercomputer in their pockets, computers were kept away from us children. The Acorn Archimedes, and before that, the BBC Micro, were stored in the smoky, coffee-drenched fog of the teachers' staffroom, which is why, I naturally assumed at the time, it acquired its beige, nicotine-hued tan. We'd play a text-based game called Granny's Garden for an hour once a fortnight, and somehow that was deemed enough to prepare us for the brave new world of digital technology that was hurtling towards us.

It is therefore hard for this writer to get misty-eyed about the BBC Micro, but it does have a place in computing history. For many, it was the first mass-market personal computer that they ever got their hands on. At the time of its launch in 1981, the pricier Model B was priced at £335; that's roughly equivalent to £1300 today. Despite that hefty price tag, the BBC Micro gained a massive following, thanks to its designers' underlying assumption that users would want to use it to learn to code. The BBC Micro kicked off a period in history that ended with the rise of gaming consoles, in which it was expected that kids would play games and learn to program on the same device. It inspired a generation of programmers, some of whom would grow up and go to work at Raspberry Pi, where they came up with another all-in-one keyboard and computer, aimed at users who want to get things done, but also want to tinker. That

01. The keycaps are high-profile injection-moulded caps chosen to imitate the original BBC Micro

02. If you weren't there, it's impossible to imagine how brown the 1980s in Britain were

> *The point of the 500+ is that the keys are swappable so you can mess about with it all you want*

computer is the Raspberry Pi 500+, and in the spirit of its illustrious forerunner the BBC Micro, Toby Roberts thought he'd give one a makeover.

But first, a confession: "The BBC Micro was everywhere at school, but I actually started with a ZX Spectrum," says Toby. "I used to spend days typing the code in from a magazine and then ages trying to work out which semicolon was supposed to be a colon. That was my entire BASIC programming life, because I didn't get much better than that. But it was in every school in the land, and there's a strain of BBC Micro DNA in Raspberry Pi, so I had to have a go at my own tribute."

Beige is beautiful

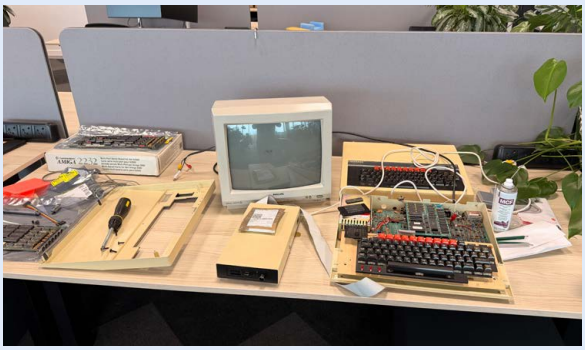
It was tricky at first to find the right shade of beige for the body, but after some perseverance Toby settled on RAL 1001 Beige enamel spray paint. "You can't just go straight for the colour – you need a half-decent plastic primer before you put the top coats on, and it's all about light coats – wait for it to dry and then build up to give yourself a nice clean finish. I think I did two coats of primer and two coats of the colour."

With the body embeigified, it was time to turn to the keycaps. One of the eye-catching things about the BBC Micro is the bright red row of function keys along the top row of the keyboard, which were meant to be programmable in BASIC. Games publishers took advantage of this to more closely integrate the gameplay with the machine – in the classic space trading game Elite, for example, you could press **F0** to launch your ship, **F4** to bring up the long-range star chart for inter-system travel, and **F5** to bring up the short-range chart to show where you are within the current star system. All important things. The keys Toby used came from niche parts supplier Signature Plastics LLC: "they have all kinds of nice keys you can buy, and of course the point of the 500+ is that the keys are swappable so you can mess about with it all you want."

▲ This exercise proved a great demonstration of Moore's Law: Raspberry Pi 500+ has a processor many times faster (2MHz vs 2.4GHz quad-core) and is cheaper than the BBC Micro even at its original 1981 price

8-bit nostalgia

1. Unlike the BBC Micro, Toby's tribute machine breaks out access to GPIO pins, so you can program in the physical world, not just on a screen.



2. There's a graveyard/shrine to the BBC Micro in the inner sanctum of Raspberry Pi Towers. This was the inspiration!

3. Next up for Toby is a ZX Spectrum build, or maybe a Commodore 64 – anything that involves recreating vintage hardware.

Quick FACTS

- The BBC Micro relied on external storage in the form of cassette tapes or floppy disks
- Vince Clarke, of Depeche Mode, Yazoo, and Erasure, is a big fan of the BBC Micro
- It wasn't just a British success story: the BBC Micro sold all around the world, propagating programming on a planetary scale
- Toby's got this model running Granny's Garden on a BBC Micro emulator…
- … hooked up to a massive CRT screen

StreamCrate

This versatile portable media server has been built around Raspberry Pi 4.

By **David Crookes**

Maker

Daveed Walzer

Daveed is a retired tour guide and somewhat reticent IT project manager. He's also solo parent to a six-year-old and owner of a janky (in a good way) VW camper-van.

rpimag.co/5tbmedia

For some years, Daveed Walzer had what he felt to be the perfect portable media server: a 2TB WD My Passport Wireless Pro. Having purchased it in 2017, he loved that it would wirelessly stream media to devices via a Plex server while working remotely in the French Alps and bouncing between ski stations.

"It was a tiny, grab-and-go, offline media server large enough to hold my entire media library," he recalls. "I used that thing in hotels, Airbnbs, friends' places, flights, tents, and more. I could connect to its hotspot and watch on my tablet or most TVs via a Chromecast puck. But, like a lot of long-lasting tech, as years passed, I started bristling at what I wished it could do."

Streaming ahead

Daveed said the 2TB drive capacity had become limiting. He preferred the open-source media server Jellyfin for true, local-only playback, but the device's firmware was Plex-only. It also lacked HDMI output, so every time he forgot his Chromecast, he couldn't use hotel TVs. He decided to build something to replace it based around a Raspberry Pi computer and asked AI for help.

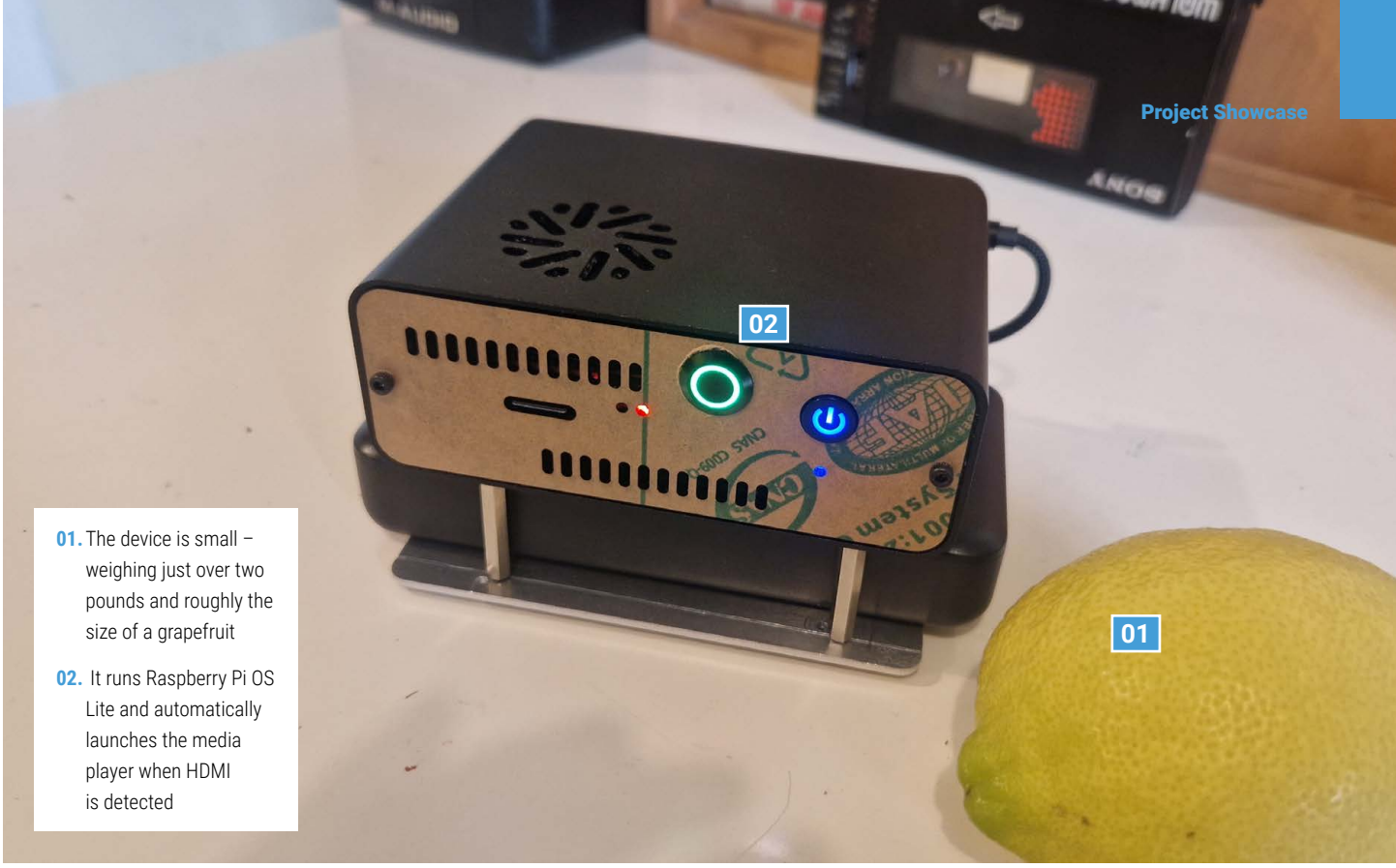
He'd recently tinkered with Raspberry Pi 3. "Over a week, I put together a slick multiroom audio system centred around Raspberry Pi running OwnTone," he says.

Encouraged, he asked AI if it was possible to recreate an updated version of WD Passport using Raspberry Pi. "It came back with 'yes', estimating a cost of $200," Daveed says. "But what really screamed 'let's do this' was the availability of a compact Geekworm Raspberry Pi 4 NAS case with a built-in 2.5″ SATA HDD bay." Add to that a 5TB 2.5-inch SATA HDD, 20,000mAh power bank, and an external USB Wi-Fi adapter, and he had StreamCrate. On paper at least.

Cracking the Kodi

Daveed's requirement was to store an entire mirror of his network-attached storage (NAS) master media library. He needed to stream from Jellyfin to whatever device or smart TV was around via Wi-Fi, or plug directly via HDMI and let Kodi on the Raspberry Pi play directly. When truly offline, he wanted streaming via a USB Wi-Fi adapter hotspot.

But there were some challenges. The case wouldn't fit a taller 5TB hard disk drive. Some Raspberry Pi limitations required workarounds too. The Kodi player



01. The device is small – weighing just over two pounds and roughly the size of a grapefruit

02. It runs Raspberry Pi OS Lite and automatically launches the media player when HDMI is detected

> *He asked AI if it was possible to recreate an updated version of WD Passport using Raspberry Pi*

was successfully set to auto-launch when Raspberry Pi detected an HDMI connected TV, but direct playback of some shows and movies proved problematic.

"For direct HDMI connection to TVs, Raspberry Pi is not just the server but also running the player. Raspberry Pi 4 cannot directly play some media encodings – with 10-bit depth and 4K H.264 being the key examples. I ran an analysis of my media collection and 20% of the content was unplayable via Kodi on Raspberry Pi," Daveed reveals.

"This ended up being the hardest part of the project: determining what needed to be converted and how to automate it so it all happened in the background. In the end, this was done via Tdarr transcoder, running in a Docker container on the NAS that hosts the master media library. It took days of tweaking to get working."

It certainly works well. "I have my entire library with me, playing on almost anything, anywhere," he says. "It's a wonderful feeling settling in for an early night in the camper-van, to finish with an episode of *Adventure Time* and the kiddo nodding off against me after a big day of big experiences."

Quick FACTS

- A 5TB HDD stores a synced copy of Daveed's media library
- Library auto sync is handled by rsync over Tailscale
- Streaming playback to local devices is via Jellyfin by default
- Direct playback to TVs is via HDMI using the Kodi player
- It offers ten hours of playback running off a 20,000mAh battery

▲ An Amazon Fire TV stick paired to Raspberry Pi enables full control of the interface. A button was also fitted to the device for easy shutdown

▼ The aluminium case was cut to fit the drive and a new custom baseplate was then added

Pico Light Alarm Clock

A Pico-based lamp did wonders for one couple, learns **Rosie Hattersley**

Maker
Pawel Skiba
Pawel is CTO of Raspberry Pi design partner RapidLab, based in Poland and a keen maker of Pico-based projects.

rpimag.co/lightclock

Isn't the change of seasons welcome? The birds are nesting, spring flowers are brightening up gardens, and trees beginning to bud. It's been a long winter for many reasons, made all the tougher for being so endlessly dark, cold and windy. No wonder some of us were keen to hibernate! Maker Pawel Skiba got an unexpected mid-winter wake-up call that prompted him to create a Pico-based Light Alarm Clock to make the transition to morning that little bit less of a jolt.

Sound the alarm

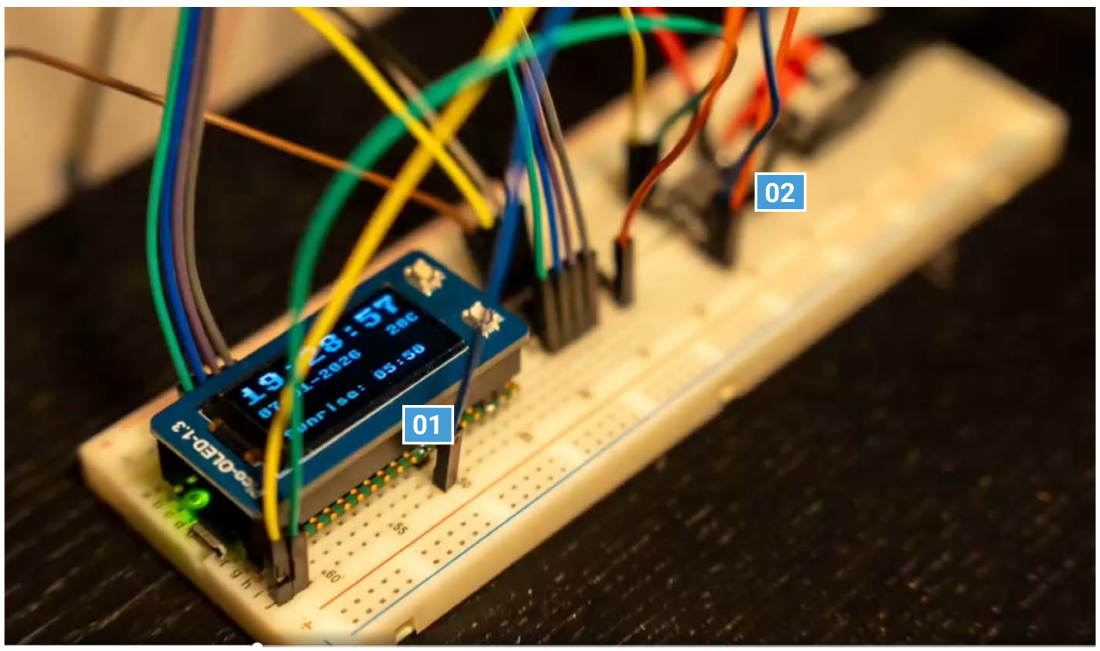
Pawel speaks for many of us when he bemoans the endless winter nights. "It's dark when we come back home after work and it's dark when the alarm clock again plays the same melody that irritates us more each morning." You'll probably also empathise with not wanting to leave a warm bed because it feels like it's middle of the night. Pawel was "passively" struggling with this fact but "finally decided to fight winter darkness when one day I heard my lovely wife saying: 'Enough! I'm about to buy a light-alarm clock and it will cost you 250 euros – plus tax'".

Thus spurred into action, Pawel soon established that it's scientifically proven that humans need light to wake up properly due to biological processes happening in our bodies when sunlight touches our skin. In essence, "there must be some light in the bedroom if I want to get rid of 'it's the middle of the night' and other heavy-head feelings".

Pawel's wife had already mentioned something that could address the lack of light issue. "Light alarm clocks simulate the sunrise process by slowly lighting up our bedrooms when we're still asleep." Starting with red light shining on the horizon, so-

▲ At 6am, the first red glow of dawn is simulated by red LEDs behind the bed

Pawel set about collecting the necessary components for a sophisticated lighting scenario to replace the dreaded alarm clock



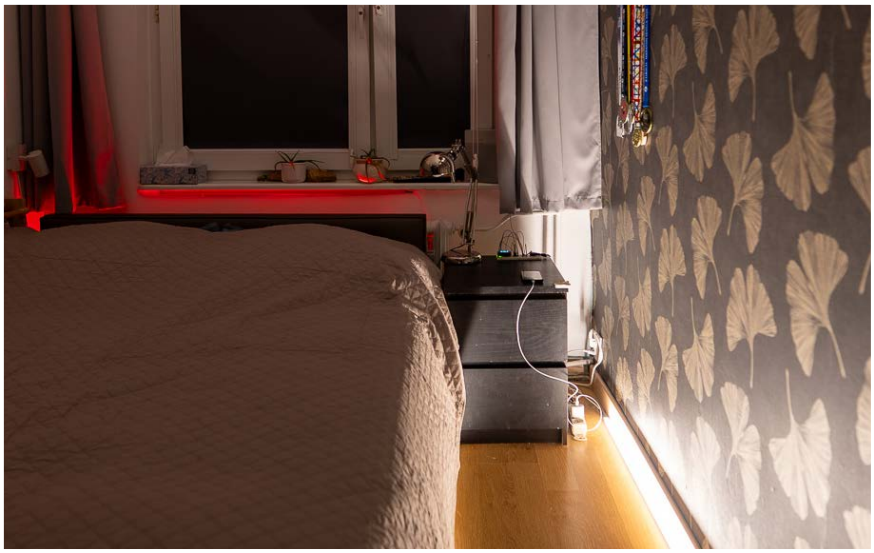
01. The Pico Light Alarm Clock features a mini display to show the current time

02. Switched via a Darlington array, different-coloured 12V LED strips simulate sunrise

called sunrise lamps graduate to lighter colours, suffusing the room with warm white and then a bright white light to help the owner wake up gradually, but fully. It took Pawel next to no time to realise that such light alarm clocks look like a regular alarm clock, but with the addition of "one or a few small but powerful LED diodes" to simulate a sunrise. His day job is chief technology officer for RapidLab, which specialises in Internet of Things research and design briefs for clients, so creating one of his own "was not rocket science".

Bathed in sunlight

In a bid to make his wife happy – and not press the Buy Now button at a certain online retailer – Pawel set about collecting the necessary components for a sophisticated lighting scenario to replace the dreaded alarm clock. Having looked into making his own sunrise lamp, he decided he wasn't that keen on them. Instead, Pawel created a setup for his own bedroom with much more light than a commercially available device is able to provide.



▼ The final light shines bright white LEDs directly into the sleeping couple's faces

▲ Subsequent lights – mainly warm white ones – suffuse the room for a gentle wake-up call

Pawel spent a few sleepless nights designing the system, starting out with a few powerful LEDs fitted with diffusers (so as not to damage anyone's eyes) to simulate dawn turning into day. In the end, he decided to have metre-long strips of light surrounding the bed, rather than individual LEDs. A strip of red lights behind the bed turns on 40 minutes before the couple needs to wake and is reflected on the wall behind them in a fair approximation of a dawn glow. Warm white LEDs run along one side of the bed, reflecting light upwards around 30 minutes before it's time to rise and shine, with further warm white LEDs turning on 10 and 20 minutes later at the base and head of the bed.

The whole setup is controlled by a Raspberry Pi Pico 2 W with a Waveshare 1.3-inch OLED module set up as a clock display. Pawel chose a Raspberry Pi-based setup for its low cost and reliability. He has used Pico in previous projects including the Locoloro interactive parrot that his company, RapidLab, created for an Ecuadorean restaurant in his home town (see issue 158, **rpimag.co/158**).

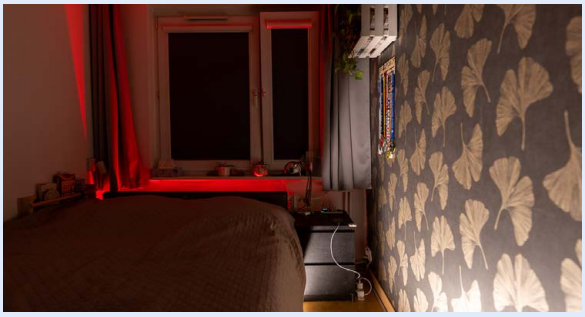
He coded everything in MicroPython to create "the sunrise magic". While the setup is not overly complex, the 5V needed to power Pico and the 12V required for the LED strips meant a 12V to 5V power step-down was needed. His prototype involved two separate power supplies but, as he observes, keeping the wiring simple made far more sense. "Too many wires are a bad omen for each wire," he reasons. Since Pico's 3.3V GPIO pins can't drive the LED strips directly, a ULN2003AN Darlington transistor array is used for switching. There was also the practical consideration of trying to build an elegant and sophisticated bedroom lighting system that didn't have tempting wires to attract the attention of curious children.

Pawel has been so pleased with how his Pico Light Alarm Clock turned out: "Nothing brings me more pleasure than building devices which physically work and solve daily challenges." This project is his favourite example to date.

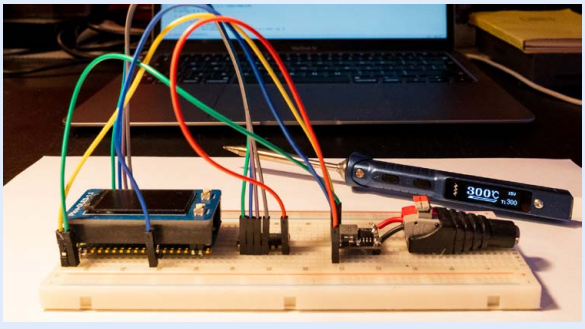
Quick FACTS

- Pawel had to act quickly before his wife splurged on a sunrise lamp
- He knew he could create a similar lamp for a quarter of the price
- Pico 2 W's wireless connectivity links it to local time
- Pawel's design has its origins in a sunrise simulator for snakes
- Both humans and reptiles are governed by sunlight hours

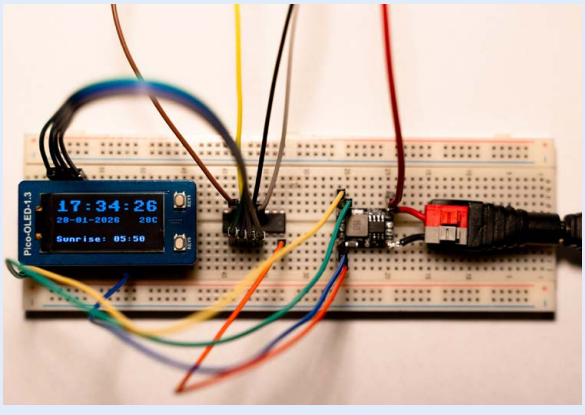
Let there be light



1. Pico 2 W's GPIO pins control the LED strips that gently and then decisively wake up sleepyheads. It retrieves the current time via its wireless connection.



2. A step-down converter accommodates the power differences between the 5V Pico and 12V LEDs, which switch on at intervals to simulate the arrival of dawn and then day.



3. As a Waveshare Pico OLED screen shows the time, there's no need to reach for your phone or a harsh overhead light.

Neural Network Camera

Creating neural networks for object identification, made easy with Raspberry Pi. **Rob Zwetsloot** identifies the work put into it

Maker
Connor
A senior high school student who loves using Raspberry Pi to make interesting contraptions, and is currently into machine learning projects.

rpimag.co/garagerobotics

Machine learning has obviously come a long way recently. Several years ago, a maker creating a working Pokédex (a digital compendium of monsters identified by sight) using computer vision spent a lot of time getting it to recognise only a small number of creatures from the famous game series. Now, makers like Connor are creating cameras that not only quickly create neural networks capable of object identification, but can also be used to test it.

> *Raspberry Pi 5, especially with the AI HAT+, is more than powerful enough to deploy a neural network*

"The project uses a Raspberry Pi, an AI accelerator, a battery backup, a 3.5-inch touch LCD display, and a Raspberry Pi Camera Module 3 to create an all-in-one means to assist in making neural networks," Connor explains. "Using the hardware and the software, the user can take photos and label them, and with the click of a button, the software will create the COCO (Common Objects in Context) formatted files to create a neural network. Then, it is just a simple step to use Google Colab to create an object-identifying neural network. Lastly, the camera can be used to test the object-identifying TFLite file. For example, a bird lover could use it to make a neural network to identify

01. The sleek white box is able to house a 5" Raspberry Pi Touch Display 2

02. A standard Raspberry Pi Camera Module 3 is used for the imaging

robins. A tourist could use it to make a neural network to identify different buildings in Italy."

Connor is currently updating the software to use Ultralytics YOLO (You Only Look Once) "since everything appears to be moving to YOLO-format for object detection".

Connor tells us the idea came about as he was thinking about tracking wildlife in his hometown of Miami. Iguanas can grow very large, and local peacocks can be very territorial, so a way for parents to keep track of what is near their children when they go outside can be very useful to keep them safe.

To the cloud

Raspberry Pi was the immediate platform of choice for this project from Connor: "The camera works seamlessly, and acquiring images is easy. On Raspberry Pi, a test and validation folder of images could be made, and these can then be converted to the COCO format. Raspberry Pi 5, especially with the AI HAT+, is more than powerful enough to deploy a neural network. The small form factor and low energy consumption allow for high portability."

It comes as a slight double-edge sword, though – the low power and portability, while essential to make this project work,

Quick FACTS

- The case is 3D printed…
- … which is why Connor originally used a 3.5" display
- Raspberry Pi 5 could theoretically do the processing
- Connor was also inspired by a 'data is power' ethos
- His app concept is something he wants to patent

▶ The 5-inch touchscreen aids in labelling up images

does mean the actual creation of the neural network is better suited to online services such as the previously mentioned Google Colab.

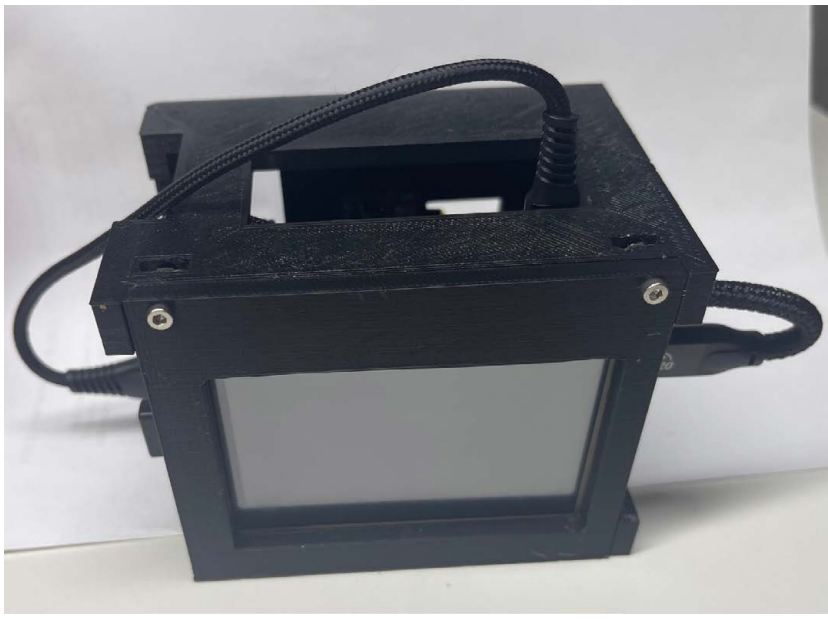
While Connor has been upgrading the camera overall – a recent addition of a 5-inch touchscreen has made it much more functional – a zoom function to help with labelling is high on the list of this evolving project, which may eventually include a connected smartphone app.

Evolving design

When we originally saw this project, the design was much more DIY, with what Connor described as a steampunk aesthetic.

"I moved away from the steampunk-style enclosure and redesigned it with a sleeker, more modern look," he says. "I also increased the touchscreen size using the 5-inch Raspberry Pi Touch Display 2. This change makes labelling much easier and more precise."

Connor tells us that he enjoyed making this project. "I feel that it has some real-world utility and, at the very least, is a great AI project."



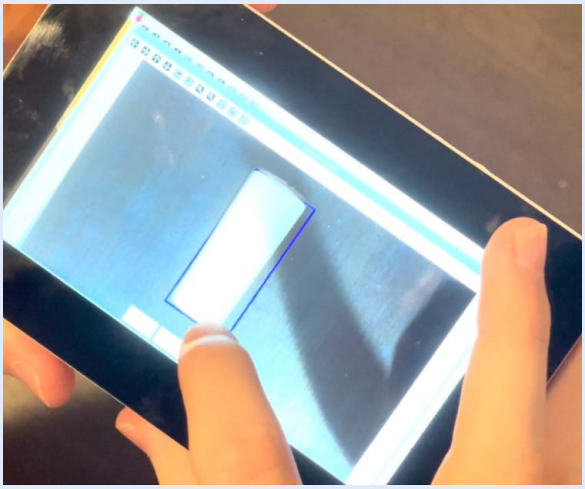
▶ An early prototype, described as 'steampunk' by Connor

▲ The 3.5-inch screen made it a bit tricky to label objects on the prototype

Teach the machine

1. Raspberry Pi displays a live preview from the camera on the LCD screen, on which the user presses 'Capture'.

2. Using a stylus, a box is created to label the object to be identified, and this is saved to a folder (either test or validation).



3. The Python program automates the conversion to COCO (soon YOLO) format, which, after a neural network using those files is created on Google Colab, can then be deployed in the same camera.

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**

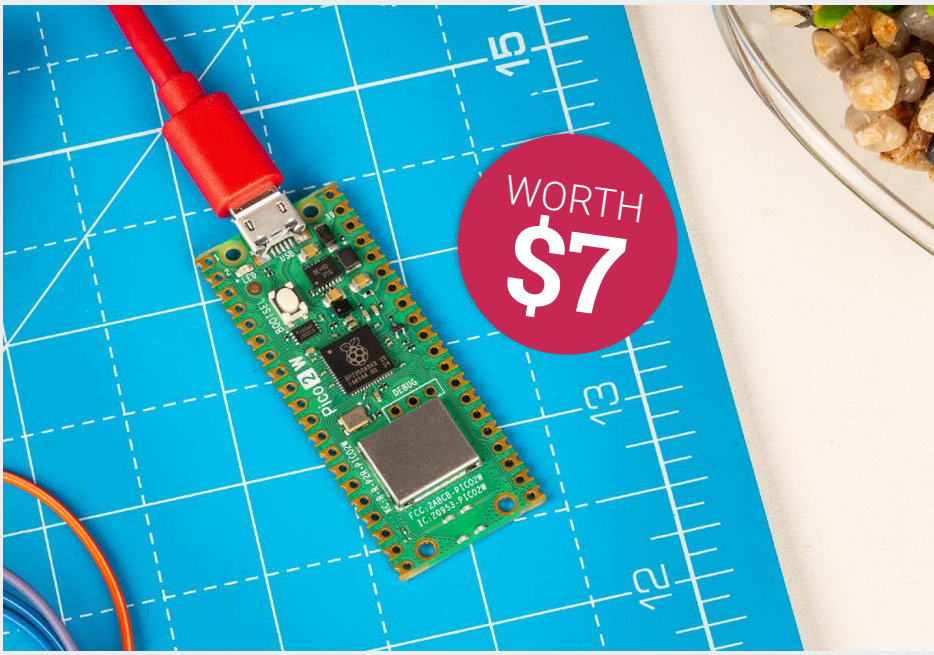
Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A choice of free Raspberry Pi Pico 2 W or Pico 2 is included with all subscriptions.

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

Cyberbrick

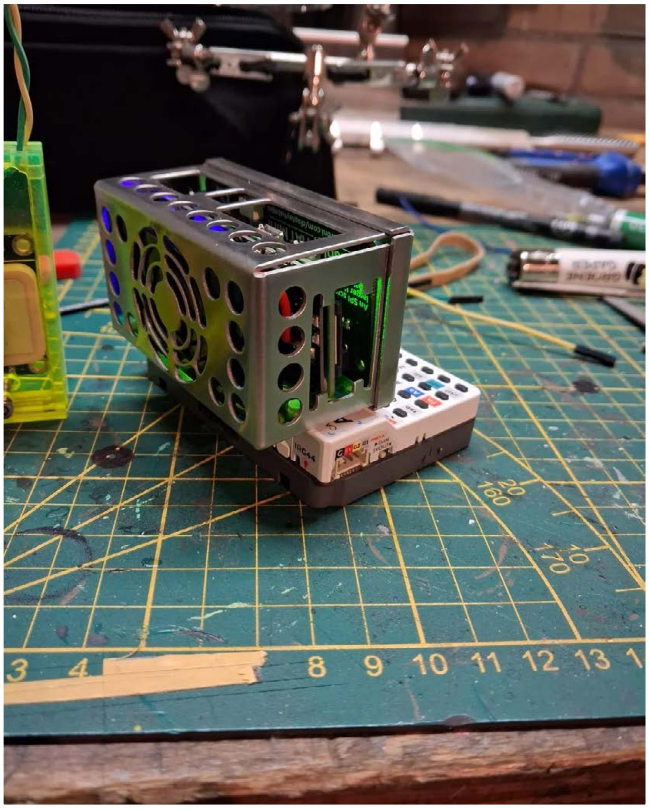
By **Arran Gunn**

rpimag.co/FANUC-terminal

We're used to seeing 3D-printed enclosures for computing on the go, but in this project Arran Gunn has come up with something slightly different: a stainless steel case. "I'm a metalworker and CAD designer", Arran says, "and I wanted a stainless steel case for my Raspberry Pi projects, so I've made multiple prototypes to make it possible without disturbing RF or losing range … I've been using the steel cases for over a year on different projects and I haven't run into any problems that can't be circumvented."

The Cyberbrick, as Arran is delightfully calling it, is comprised of a Raspberry Pi Zero 2 W, a PiSugar uninterruptible power supply, and a Pimoroni Inventor HAT Mini, with the adorable tiny keyboard coming from an M5Stack Cardputer. Arran has also added four fluorescent green sheets of acrylic, to give it a real 1970s/1980s sci-fi aesthetic.

▶ As well as being robust, the Cyberbrick weighs only 200-ish grams



Precise Indication of Sewage Storage

By **Seafoxc**

rpimag.co/ISS-sewage-indicator

The open internet has been a gift for citizen scientists. Websites collate tons of data for the eager amateur to sift through. So, with a data Swiss Army knife such as Python, or a more dedicated data analytics tool like R, anyone can track traffic jams to work out the most efficient time to set off for home, or keep an eye on pollution levels in our precious streams and rivers. You can track crime rates, and overlay the data on maps to give you an at-a-glance view of where the villains are.

Alternatively, you could do something more frivolous instead. Something like this. Maker Seafoxc has looked at the data coming out of the International Space Station, the crowning glory of co-operation and science, and somehow decided that the most interesting data coming out of it is the levels of waste stored in its sewage tank. That data inspired this puerile, silly, childish, and rather brilliant piece of data visualisation: the Precise Indication of Sewage Storage.

It uses a Raspberry Pi Zero 2 W, an OLED display, an LED to light up the – ahem – fluid, two pumps, and two MOSFETs. Much as we might wag our finger at how silly this is, it's a perfect form-meets-function design. You can see exactly what's being measured, and read the data far more intuitively than you would by reading a spreadsheet.

If you fancy making your own citizen science project with data from the ISS, it's all available here: **rpimag.co/ISSMimic**

Etch A Sketch CNC machine

By **Nikodem Bartnik**

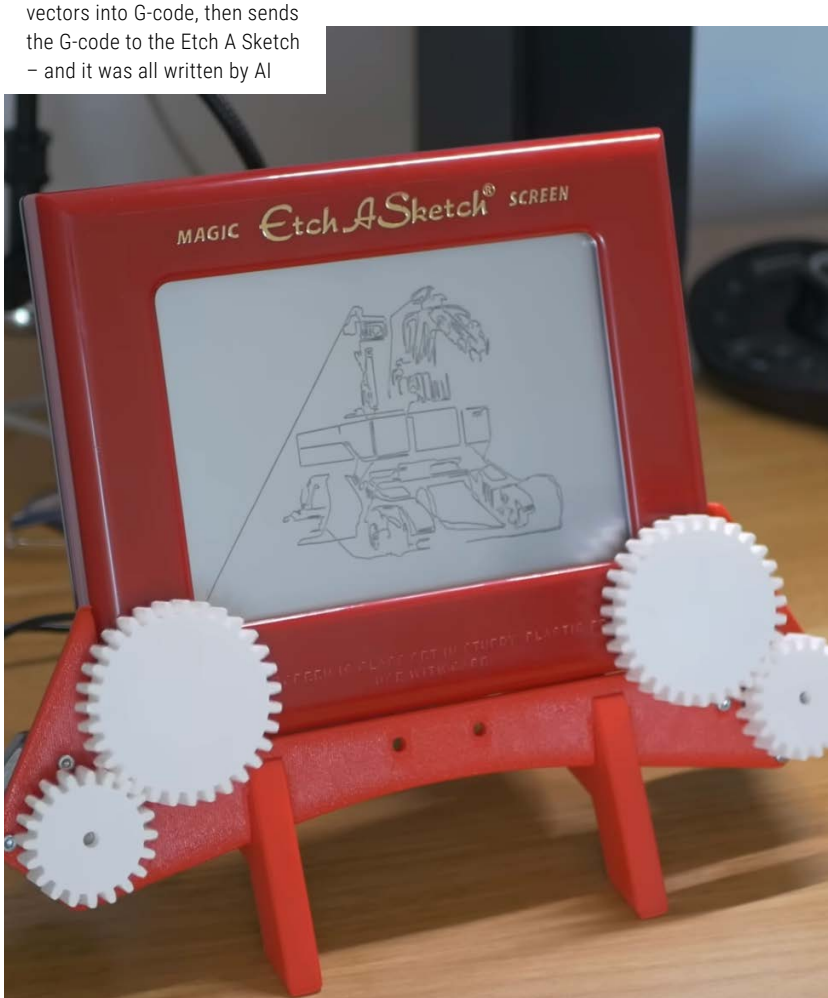
rpimag.co/Etch-a-sketch-dock

The humble Etch A Sketch, with its knobs that turn to trace a line on a screen in the X and Y axes, is ripe for computerisation. Nikodem Bartnik certainly thought so, as he set out to turn his into a CNC machine.

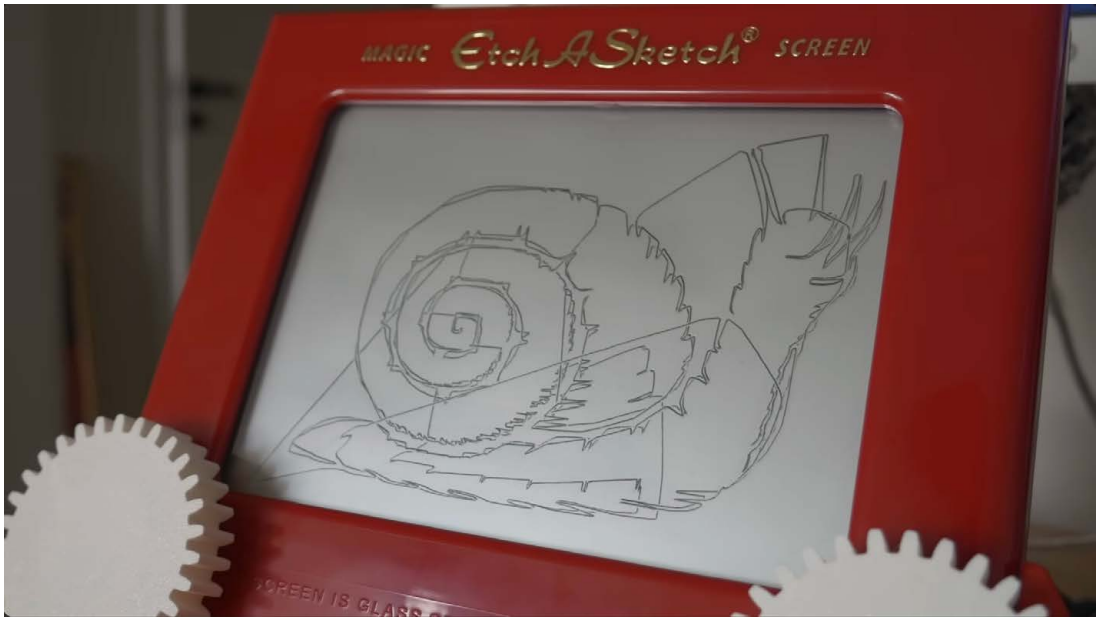
Nikodem's robot was originally going to attach to the Etch A Sketch, but after the first prototype he realised that this would make it unfeasibly heavy. As the way to erase the Etch A Sketch's screen is to pick it up and shake it, this wouldn't do. So instead he settled on a docking station design, which he modelled in Fusion 360 – the build comprises just three 3D printed parts for the frame, plus the gears that go over the Etch A Sketch's buttons, two stepper motors and drivers, and a Raspberry Pi Pico, all mounted on a homemade PCB milled on a CNC machine that Nikodem made out of a Dremel.

One thing that caught our eye about this build is that the software was written with AI; specifically Google Antigravity. It took a bit of debugging, and then a bit more when ChatGPT mangled the implementation of Bresenham's algorithm, which is a way of converting lines into discrete steps. Luckily, Nikodem had already used this algorithm successfully on another build, so he knew what to look for in order to fix it.

▼ The software controlling the Etch A Sketch logs the image, turns it into vectors, turns the vectors into G-code, then sends the G-code to the Etch A Sketch – and it was all written by AI




MAGIC Etch A Sketch


MAGIC Etch A Sketch® SCREEN

Vintage Radio Plex server

By **zef37**

rpimag.co/Plex-server

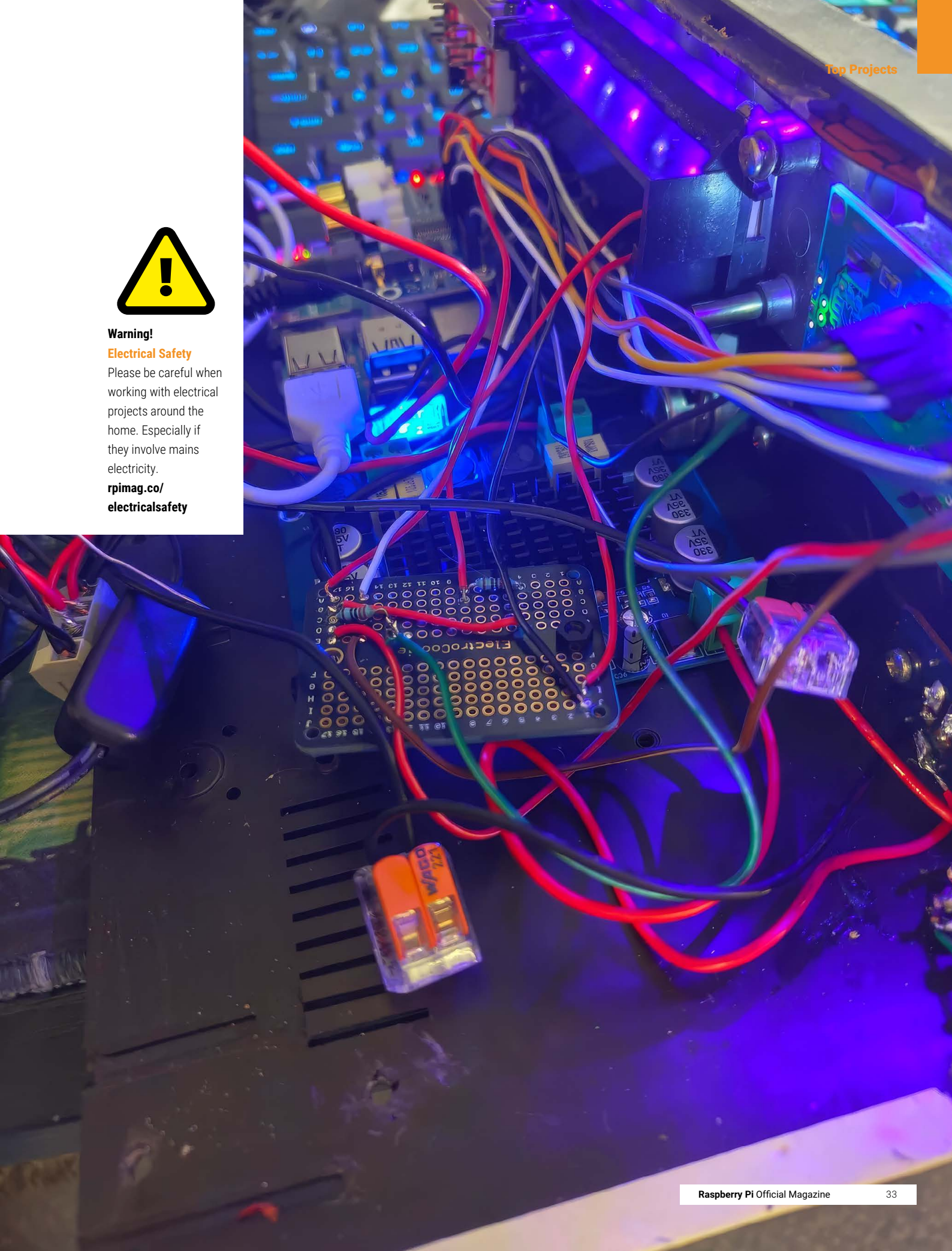
Not to sound like a bunch of old people, but we really do believe that lots of things were better in the olden days. Industrial design reached incredible heights with the Bauhaus, the Wiener Werkstätte, and the beautiful, form-follows-function designs of Dieter Rams for Braun in the 1950s and 1960s. We've lost a lot of the craft that goes into product design – most galling of all, we're losing tactile, clicky buttons, as designers force touchscreens on us. But if you can get hold of a piece of hardware from the 1980s or 1990s, and incorporate some of the original controls into a new project, you're 90% of the way towards creating something beautiful and unique.

In that spirit we applaud maker and Reddit user zef37, who has taken a vintage radio and turned it into a modern streaming/audio speaker with the aid of a Raspberry Pi 4.

Along with our single-board computer, there's a PCM5122 digital audio converter, TPA3116D2 DC12-26V 2×120W audio amplifier, a pair of three-inch 4 ohm 20W speakers, a 1602 16×2 LCD, and a really, really bright ultraviolet 5V LED strip for bezel lighting. These do seem like oddly specific parts to us, but if that's what you've got to hand, that's what you use – and the digital audio converter chip used is the same model found on the Raspberry Pi DAC Pro (and on similar products from Adafruit et al), so if you want to build something similar, the hardware is available to you.

The original internals were damaged beyond repair, but they did manage to keep the front panel knobs as potentiometers for the LCD backlight, volume, and Bluetooth/Plex server mode change, as well as the original rocker control for the power on/off switch.

▼ There's an antenna sticking up on this build, but it's just for show



Warning!

Electrical Safety

Please be careful when working with electrical projects around the home. Especially if they involve mains electricity.

rpimag.co/electricalsafety

3D print

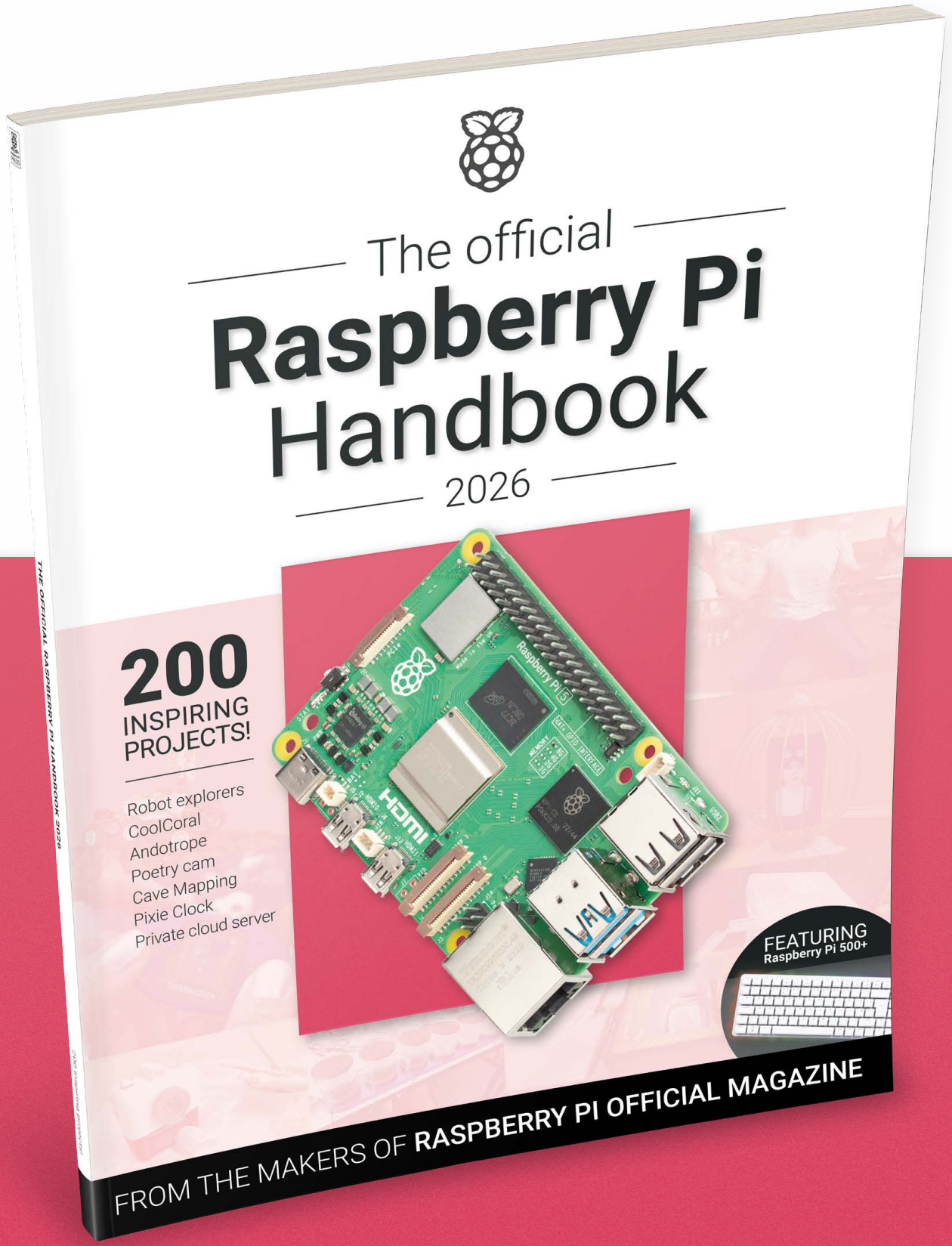
Sumer is icumen in – and with it, the scourge of hay fever

rpimag.co/Pollen

Sporopollenin is a natural polymer that makes up the tough outer walls of pollen grains. It's chemically inert and incredibly durable, so much so that pollen grains consistently show up in archaeological digs where everything else organic has decayed. Not only that, but it's present in the fossil record where other plant matter has decayed, meaning that our evidence for early plants comes not from leaves or seeds, but from pollen – tiny, invisible-to-the-naked-eye pollen. Sporopollenin is even resistant to UV light, protecting the fragile DNA of the plant that's stored within the pollen grain. Forensic scientists use pollen to date burials and solve crimes; art historians use pollen grain analysis to catch fakes. And yes, despite this fascinating, useful existence, pollen is also the cause of hay fever, which cancels out all the good pollen does and puts it firmly on the naughty list.

As the days get longer and thoughts turn inevitably to sitting indoors with the curtains drawn struggling to keep our airways open, we thought we'd print a few pollen grains to keep on our desk, much like the way that the Duke of Wellington kept a bust of Napoleon on his desk: to know his enemy. Our models are printed in matte PLA, with a layer height of 0.24mm.

We've got three examples of common pollens here: there are loads more available for download at the US National Institute of Allergy and Infections Disease, all of which were supplied to our American friends by the Bioimaging Research Hub at Cardiff University. We've gone for pollen from the hedge bindweed (Calystegia sepium), field fleawort (Senecio integrifolia), and field elm (Ulmus minor).


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

The official Raspberry Pi Handbook 2026

200 PAGES OF RASPBERRY PI

Get started guide covering every Raspberry Pi

Everything you need to know about the brand new Raspberry Pi 500+

Inspiring projects to give you your next project idea

Build a Raspberry Pi 5-powered media player

Explore the world around you with roving robots

Play retro horror games on Raspberry Pi 5

Buy online: rpimag.co/handbook2026

MIGHTY PROJECTS 1 GB COMPUTER

The low-cost, yet still very powerful Raspberry Pi 5 1GB model is ideal for a host of applications

By **Phil King**

With the same powerful BCM2712 system on chip (SoC) as the other Raspberry Pi 5 computers, the new 1GB variant offers a more affordable entry point for users who need extra processing grunt and/or features such as the Raspberry Pi connector for PCIe to add an NVMe SSD or AI HAT+. To this end, we've rounded up a range of project ideas that make good use of the 1GB model's performance, but don't require a large amount of RAM.

We follow that up with a handy guide on how to run the 1GB model in headless mode and keep an eye on memory usage, then a tutorial on running the Stable Diffusion deep learning model on it to create images from text prompts.

We've rounded up a range of project ideas that make good use of the 1GB model's performance

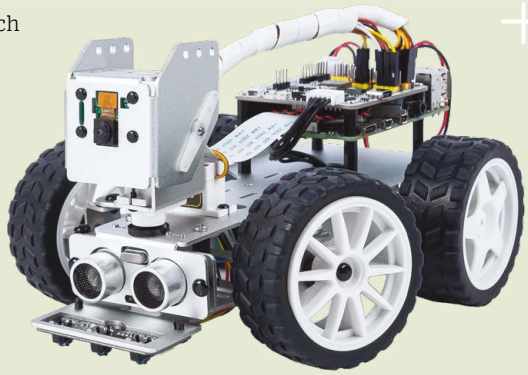
ROBOTICS

Raspberry Pi computers have long been favoured for robotics projects such as cars, arms, and even animals. So a Raspberry Pi 5 1GB is ideal for this purpose, just so long as you don't need your robot to use advanced computer vision.

There are countless kits available (e.g. the low-cost CamJam EduKit #3: **rpimag.co/edukit3**, and PiCar-X: **rpimag.co/picarx**). Or you can build your own robot vehicle from scratch, with components including a chassis, wheels, motors, and a dual H-bridge driver board to spin them forward and back. Note: you should never connect DC motors directly to the GPIO pins as they'll draw too much current.

Sensors such as ultrasonic or touch (and/or a camera) are required for a self-driving robot so it can detect obstacles and avoid them. We covered this in the second part of our MARS Rover tutorial in issue 148 (**rpimag.co/148**).

▼ The SunFounder PiCar-X is a quality four-wheeled robotic car kit



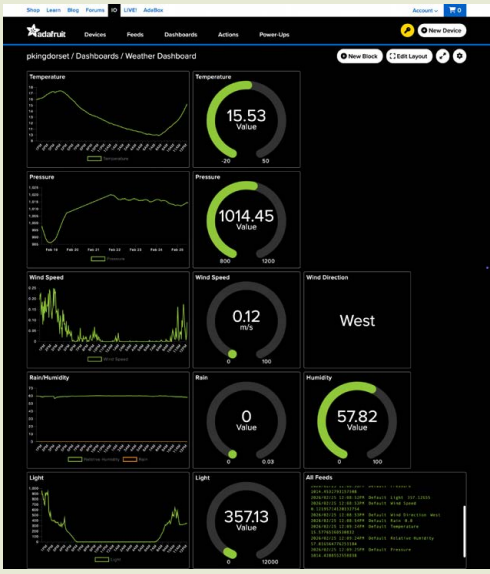
▲ An ultrasonic sensor enables your robot to detect and avoid obstacles

IOT & HOME AUTOMATION

IoT (Internet of Things) projects are an ideal use for Raspberry Pi 5 1GB. With its ability to connect to the internet via Wi-Fi, it can be used to collect data from one or more sensors attached to its GPIO pins and then send it wirelessly to the cloud (using a lightweight messaging protocol such as MQTT). You may then view the data on a web dashboard such as Adafruit IO – as we did with our weather station tutorial in issue 119 (**rpimag.co/119**).

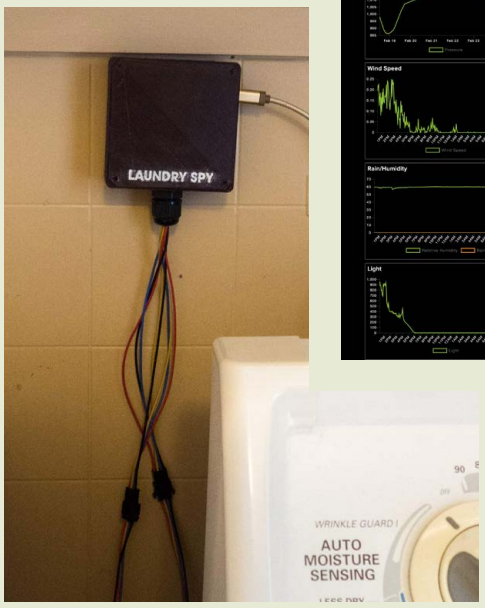
You can also use Raspberry Pi 5 1GB as an IoT hub to relay messages from a microcontroller and sensor(s) to create phone alert notifications, such as with the smart Laundry Spy (**rpimag.co/laundryspy**) which tells you when the washing machine has finished its cycle.

While at least 2GB of RAM is recommended to run the latest edition of Home Assistant (**home-assistant.io**) for home automation, you could still use Raspberry Pi 5 1GB as a node for an existing HA system – using the Raspberry Pi Remote GPIO integration – to read sensors and switch on connected devices.



▶ Weather station data is sent via MQTT to an Adafruit IO web dashboard

▼ A clever way to alert you to when the washing is ready



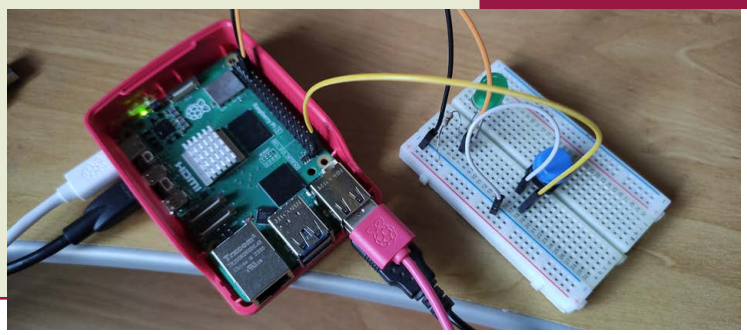
ELECTRONICS

Just like its siblings, Raspberry Pi 5 1GB is ideal for learning all about electronic circuits by connecting various components to its GPIO pins, typically via a solderless breadboard.

The standard beginner's project is to blink an LED on and off, then expand on this by switching it on/off with a push-button, as we covered in the Get Started with Raspberry Pi feature in issue 161 (**rpimag.co/161**).

For lots more electronic projects to try with Raspberry Pi, check out our book, *Simple Electronics with GPIO Zero* (**rpimag.co/gpiozerobook**). All the code is in Python, using the GPIO Zero library to make things simpler.

Raspberry Pi 5 1GB can emulate many classic computers and consoles



▶ Start with a simple circuit comprising an LED and a push-button to turn it on and off

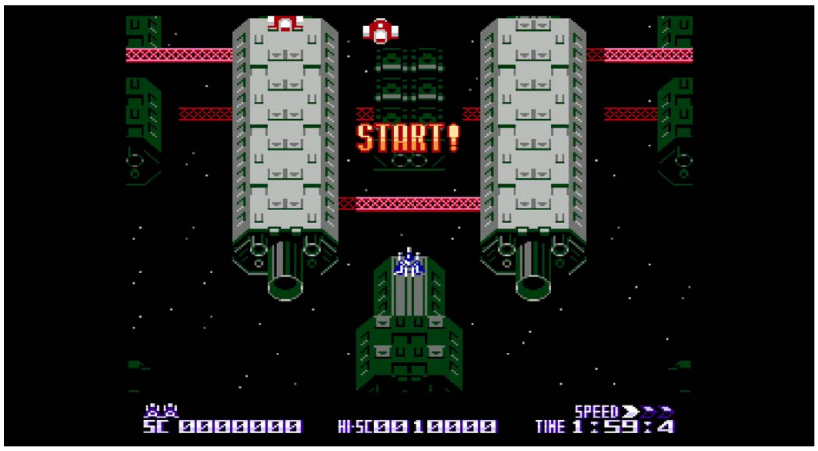
RETRO GAMING

As with most other models, Raspberry Pi 5 1GB can emulate many classic computers and consoles. Higher RAM is only really needed when trying to emulate more modern systems, so anything up to and including PlayStation 1 / Saturn / Dreamcast should work fine – such as NES, SNES, Mega Drive/Genesis, GBA, MAME, ZX Spectrum, C64, and Amiga.

The choice of OS is up to you: Recalbox (**recalbox.com**), Lakka (**lakka.tv**), and Batocera (**batocera.org**) should all work fine – as does RetroPie (**retropie.org.uk**), though you'll need to install it manually in Raspberry Pi OS as there's no ready-made OS image for Raspberry Pi 5.

Game ROMs can 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**.

▼ Blade Buster for NES is just one of the many retro games you can play on Raspberry Pi



PHOTOGRAPHY & VIDEO

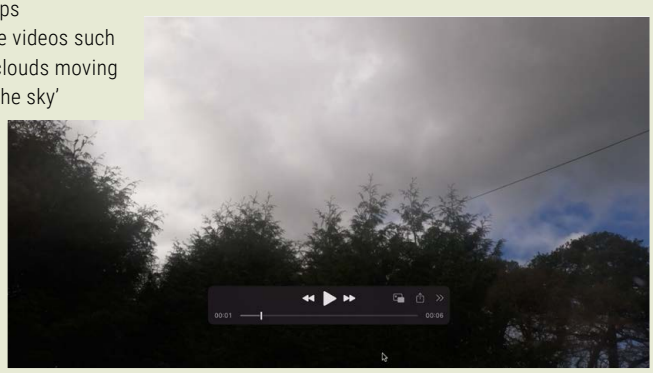
You can equip your Raspberry Pi 5 1GB with an official Camera Module (or High Quality Camera) by connecting it via a ribbon cable to one of the board's two dual-purpose camera/display ports – marked CAM/DISP 0 and 1.

When booted up, Raspberry Pi should detect the camera automatically. You can then start using it to capture stills or video. There are two main ways of doing this: by entering commands (rpicam-still and rpicam-vid) in the terminal/CLI, or using the Picamera2 library in your Python programs. Both methods support a wide range of advanced settings so you can fine-tune the results. See the camera software documentation for more details: **rpimag.co/camera-software**.

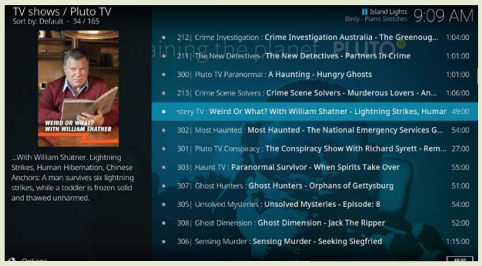
Popular camera-based projects include a wildlife camera trap (with a PIR sensor to detect movement), time-lapse videos (made from multiple stills), stop-motion animation, spy/security camera, and astrophotography (by attaching an HQ Camera to a telescope).

▲ Create stop-motion animations with a Camera Module and props

▶ Make time-lapse videos such as the classic 'clouds moving quickly across the sky'



▼ For a slick look, you can house your Raspberry Pi in a special case such as the Argon ONE V5



▲ Use Kodi add-ons to stream shows from free services such as Pluto TV

MEDIA CENTRE / NAS

Since Raspberry Pi 5 1GB has a connector for PCIe, you can use this with an M.2 HAT+ (or alternative) to connect an M.2 NVMe SSD (solid-state drive). Not only does this provide extra storage, but you can also boot Raspberry Pi OS from it instead of the standard microSD card.

As well as speeding up general performance with lightning-quick read/write speeds, the SSD is ideal for creating a media centre (to play/stream movies, TV shows, and music) and/or NAS (network-attached storage).

The easiest way to create a media centre is by using a Kodi-based OS such as LibreELEC (**libreelec.tv**) or OSMC (**osmc.tv**). For more details, check out our media player guides in issue 132 (**rpimag.co/132**) and 155 (**rpimag.co/155**).

Alternatively, you could set Raspberry Pi 5 up as a discrete NAS box so that files held on it can be accessed wirelessly by other devices on your network using the Samba sharing protocol. For setup details, see **rpimag.co/nasbox**.

▶ Create a smart 'magic mirror' with some two-way glass and a TV/monitor

MAGIC MIRROR

"Mirror, mirror on the wall... who is the smartest of them all?... Ah, it's you, because you're powered by a Raspberry Pi and can display all sorts of useful information such as news, weather, traffic, and my calendar."

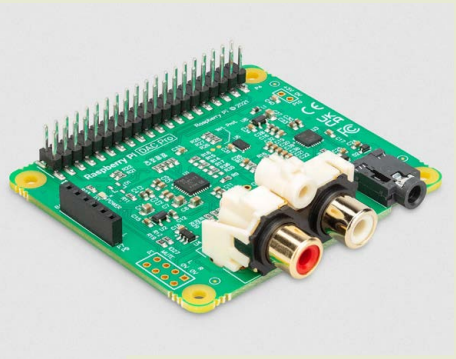
Building a magic mirror isn't as daunting as it sounds. You just need to source a suitably sized TV or monitor, cover it with some two-way mirror glass (buy it ready-made or make it by applying special film onto ordinary glass), and install it in a wooden frame – which you can make yourself if you're keen on carpentry.

Then it's just a case of installing the software, which you can find – along with all the documentation and an array of add-on modules – at **magicmirror.builders**. It's a good project for Raspberry Pi 5 1GB, although you will need to have it running the desktop version of Raspberry Pi OS to run the software.

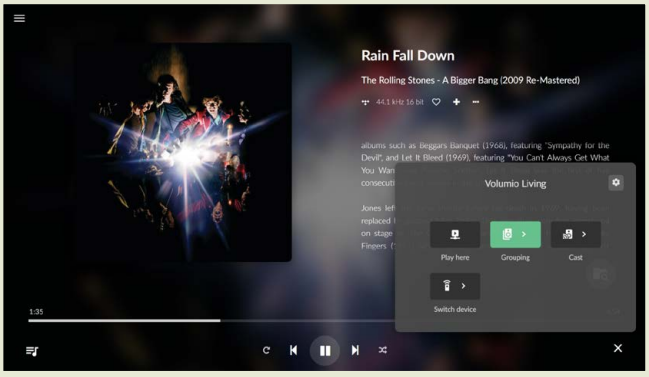
INTERNET RADIO / HI-FI

While Raspberry Pi 5 1GB doesn't have a built-in audio output, you can listen via Bluetooth headphones/speakers or through a TV connected by HDMI. Alternatively, for superior sound, there are several DAC HATs available to link it to your hi-fi equipment. With the Raspberry Pi DAC Pro, for instance, you can even enjoy high-definition 24-bit audio at 192kHz – far better than standard 16-bit CD quality.

Software-wise, there are numerous ways to enjoy music on your Raspberry Pi, including specialist operating systems such as Volumio (**volumio.com**), moOde (**rpimag.co/moode**), and PiCorePlayer (**picoreplayer.org**). Most should enable you to listen to locally stored files, popular streaming services, and internet radio stations – for aesthetic effect, house your Raspberry Pi in a vintage radio case. You can even cast playback to multiple smart speakers for multi-room audio.



◀ The DAC Pro delivers outstanding audio output from its RCA jacks and headphone port



▼ With Volumio, you can cast audio to smart speakers around your home

LED LIGHTING

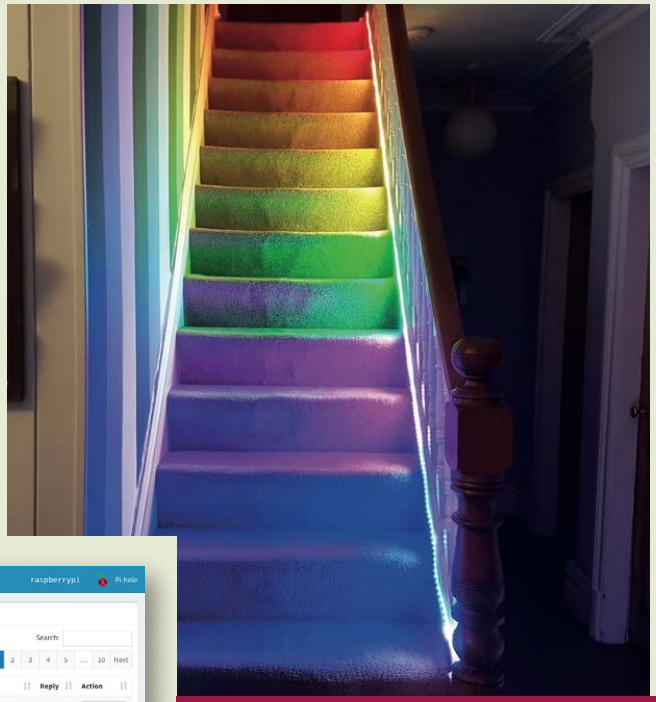
While you can use a Pico to control a strip of RGB LEDs, a Raspberry Pi 5 1GB gives you the extra processing power and versatility needed for more complex lighting projects.

For instance, you could create a custom light display to impress the neighbours, and even sync it up to music and/or have it controllable via a web interface.

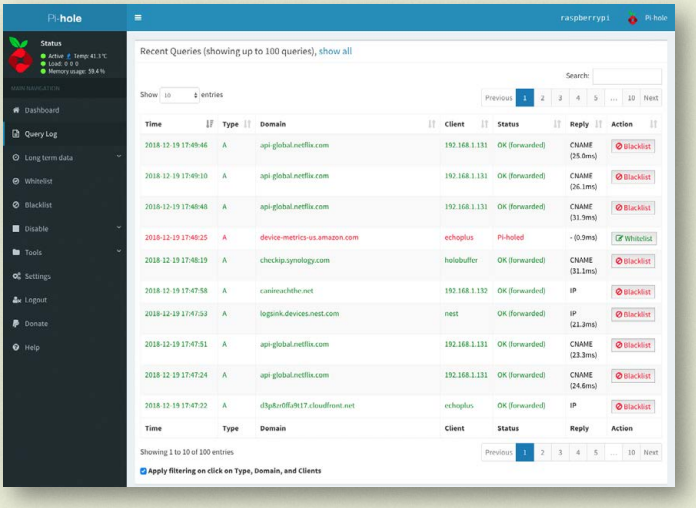
Multiple strings of lights can be turned on/off using a relay board connected to Raspberry Pi's GPIO pins, as in Devon Govett's setup: **rpimag.co/pichristmaslights**.

In addition, you can program complex colour-changing patterns for strips of addressable WS2812B LEDs (aka NeoPixels). This can also be used for other projects inside the home, such as mood lighting, tree lights, and stair lights.

▼ Lorraine Underwood's NeoPixel stair lights change colour with the temperature



Create a custom light display to impress the neighbours, and sync it up to music



▲ Access Pi-hole's options and statistics from its web dashboard on any device

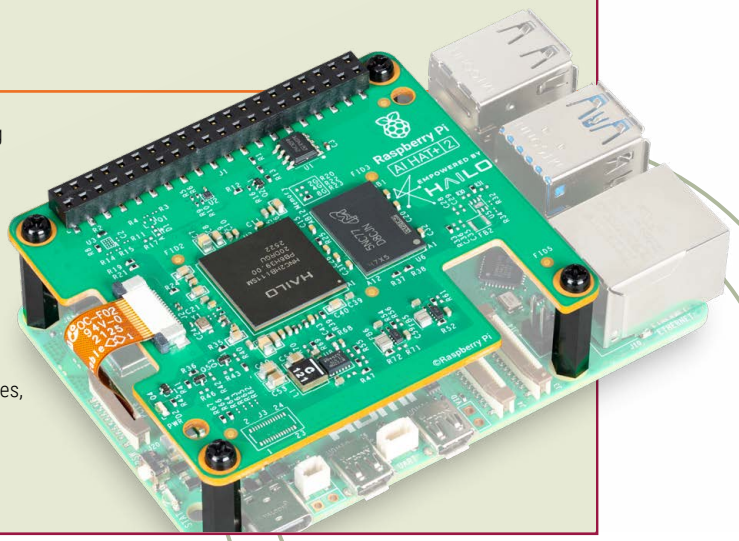
AD BLOCKER

Fed up with obtrusive adverts and pop-ups on websites? Pi-hole is your friend. Since it requires no video output, it can be run on a headless Raspberry Pi 5 1GB setup, and you can access its numerous options and statistics via a web dashboard. For details on how to set it up, see **rpimag.co/pihole**.

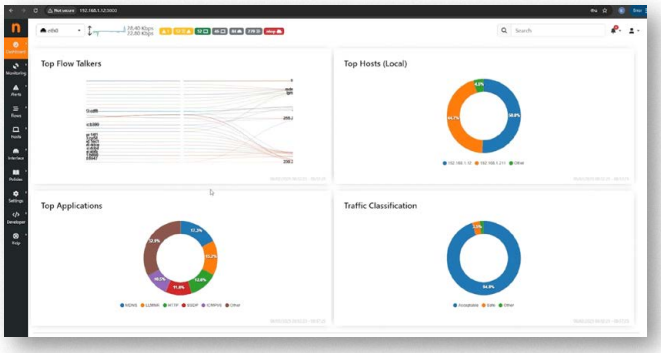
Pi-hole works as a DNS sinkhole. When a device on your network requests a website, the request goes to Pi-hole first. If the domain is on the blocklist, Pi-hole blocks it, otherwise it works as usual. You can customise the blacklist, as well as add sites to a whitelist. Since unwanted ads are blocked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

AI APPLICATIONS

While 1GB of RAM isn't really enough for running local LLMs (large language models), the PCIe connector on Raspberry Pi 5 1GB enables you to connect an AI HAT+ or AI HAT+ 2 to run computer vision models. Crucially, the AI HAT+ 2 has 8GB RAM of its own, so can also be used for on-device generative AI tasks such as vision-language models, voice-to-action agents, and more. For details of its best use cases, see **rpimag.co/aihat2best**.



▶ The AI HAT+ 2 can be used to run advanced AI applications



▲ The ntopng web dashboard shows graphs for various network aspects

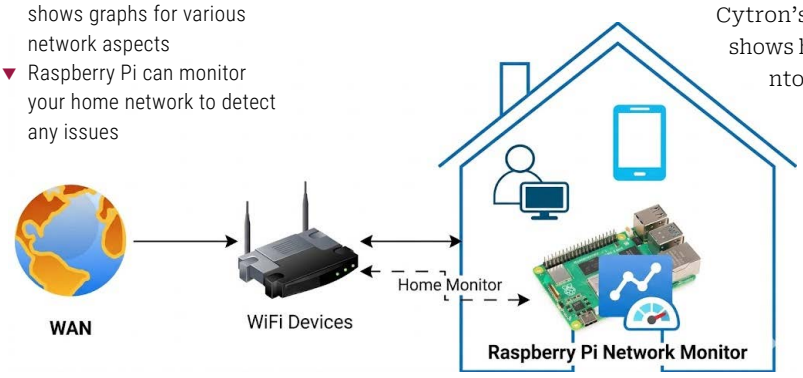
NETWORK MONITOR

To help you spot and diagnose any issues that arise with your home network, a Raspberry Pi 5 1GB can be set up as a versatile network monitor to continually survey performance, traffic, and security so you can fix any problems.

Cytron's step-by-step tutorial (**rpimag.co/ntopguide**) shows how to set up such a system using the open-source ntopng network-monitoring tool. This enables you to monitor bandwidth usage, identify connected devices, detect unusual network activity, and analyse network protocols.

SunFounder has also produced a handy guide to the top network monitoring tools and techniques: **rpimag.co/networkmonitor**.

▼ Raspberry Pi can monitor your home network to detect any issues



Make your RAM go further

Memory management in Raspberry Pi OS

Maker
Lucy Hattersley
Lucy wishes you could remember things as well as htop.

rpimag.co

This month we have been playing around with the new Raspberry Pi 5 1GB RAM. While the RAM shortage caused by the demands of AI infrastructure is annoying beyond belief, this has been a great chance for us to really get to grips with RAM.

Generating images in Stable Diffusion on a Raspberry Pi with 1GB gave us a good chance to fully explore the RAM setup in Raspberry Pi OS. And we thought we'd share some of our learnings here.

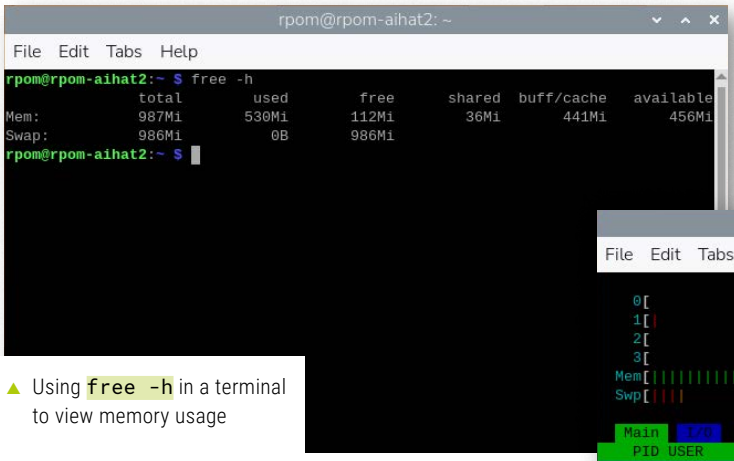
Check out your RAM

There are a few tools you can use to check out the RAM on your Raspberry Pi. Chief amongst these is the one called 'free'. Boot up your Raspberry Pi and open a terminal window. Now enter:

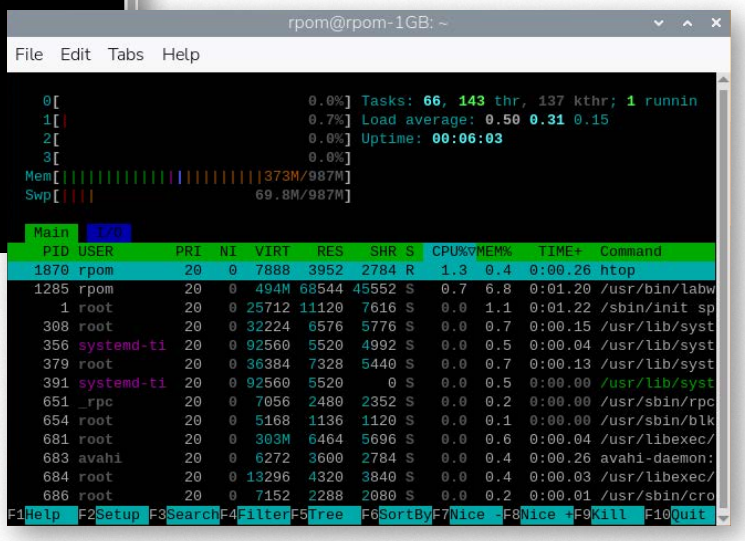
```
$ free -h
```

Here you'll see two rows, Mem and Swap:

- **Mem.** Physical RAM inside your computer
- **Swap.** A swapfile on the storage drive used as virtual RAM



▲ Using `free -h` in a terminal to view memory usage



▼ The htop interface running in Raspberry Pi OS

> *We're using just 266MB of RAM and have 720MB available*

And six columns:

- **total** – The total amount of memory (physical RAM or swap space).
- **used** – Memory currently in use by processes and the kernel.
- **free** – Completely unused memory sitting idle (normally quite low).
- **shared** – Memory used by temporary file systems (like /dev/shm).
- **buff/cache** – Memory used for disk buffers and file system cache. Linux uses spare memory to cache files from disk, which speeds things up. This memory can be quickly reclaimed if applications need it. This is why the free memory is typically quite low.
- **available** – An estimate of how much memory is available for starting new applications without swapping. This includes free memory plus reclaimable cache/buffers. This is more important than the 'free' column as an example of readily available memory.

Free gets its information from memproc, which you can view at **cat /proc/meminfo** if you want to get further under the hood.

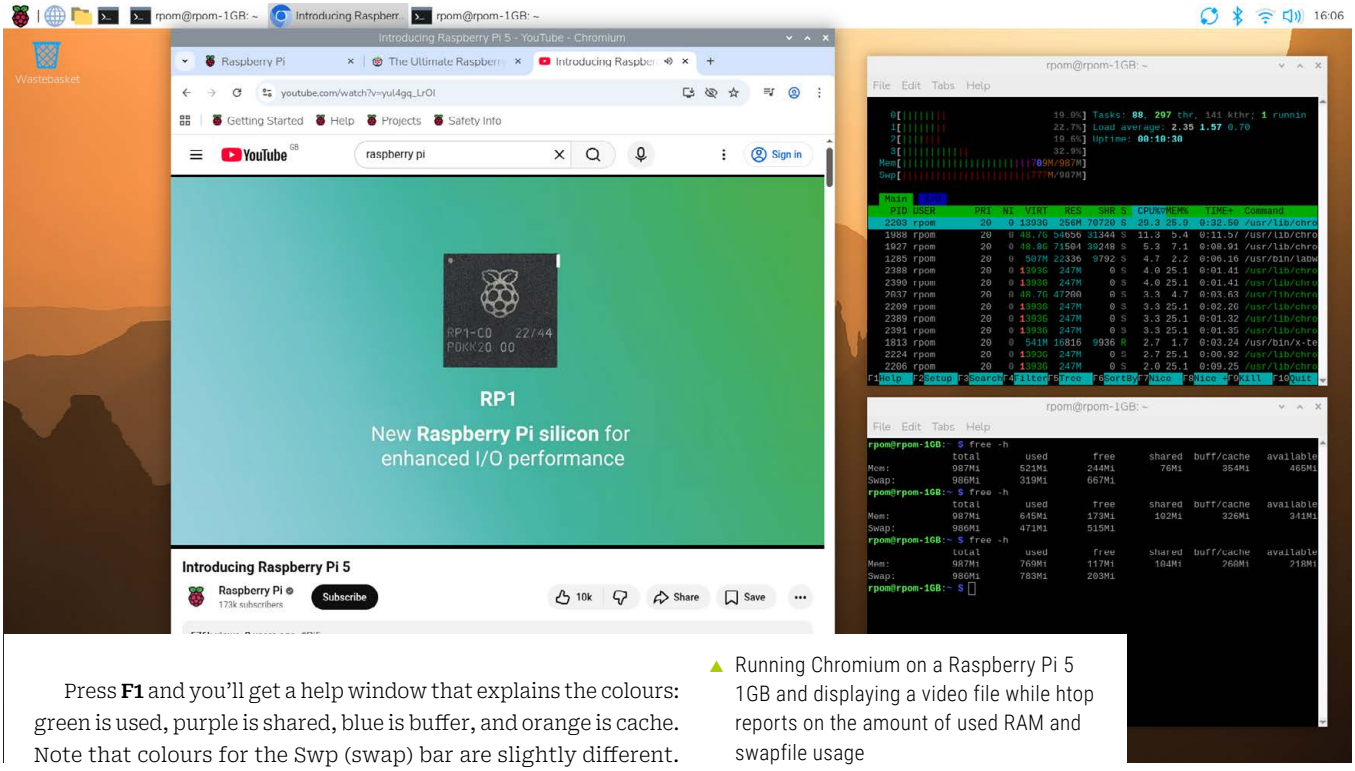
Take it to the htop

As we're not running many applications, we can tell that most of our RAM usage is going to be taken up with running Raspberry Pi OS in GUI mode. We'll switch to 'headless' mode shortly to see how much RAM we can free up, but first let's take a look at htop.

Top (table of processes) is a classic Unix utility that displays information on processes running in real-time on your system. An improved version is called htop (the 'h' stands for Hisham Muhammad who developed it) that has colour-coding and mouse support. Htop is installed by default with Raspberry Pi OS. In a terminal window, enter:

```
$ htop
```

...to start the program. At the top of the screen is a pseudo-graphical display of the four CPU cores (numbered 0 to 3) and Mem and Swp charts. You should see very little activity in the CPU, but half the Mem bar will display green vertical lines (with the remainder purple, blue, and orange).



▲ Running Chromium on a Raspberry Pi 5 1GB and displaying a video file while htop reports on the amount of used RAM and swapfile usage

Press **F1** and you'll get a help window that explains the colours: green is used, purple is shared, blue is buffer, and orange is cache. Note that colours for the Swp (swap) bar are slightly different. The Help screen also provides a selection of keyboard shortcuts. The two of which that are most useful are **P** to sort by CPU usage and **M** to sort by memory usage.

Press any key to return to htop and press **SHIFT+M** to sort by memory usage. Here we can see that most of the RAM is being taken up by the GUI interface:

- **labwc**. The Wayland compositor which is Raspberry Pi OS's GUI window manager.
- **Xwayland**. A tool to allow X11 display to Wayland.
- **wf-panel**. The taskbar applications.
- **pcmanfm -- desktop**. The file manager.

If we start to open more programs such as Chromium and Thonny IDE, we will quickly eat up our available RAM. Then we start to move into the swapfile, which uses our physical storage to swap programs in and out of RAM.

Swappiness

While a Raspberry Pi 5 1GB RAM can be used in desktop mode and all applications perform quickly, you'll quickly run into performance issues if you open multiple applications at once. This is because the desktop interface is taking up a third of your available RAM and – by default – when Raspberry Pi OS uses RAM, it turns to the swapfile. This value is known as 'swappiness' and can be investigated with the cat command:

```
$ cat /proc/sys/vm/swappiness
```

...which will return 60. This is a moderately aggressive value designed to quickly bring the swapfile into use. We can reduce this swappiness value to protect our microSD card from excessive write actions and prioritise RAM use for faster performance:

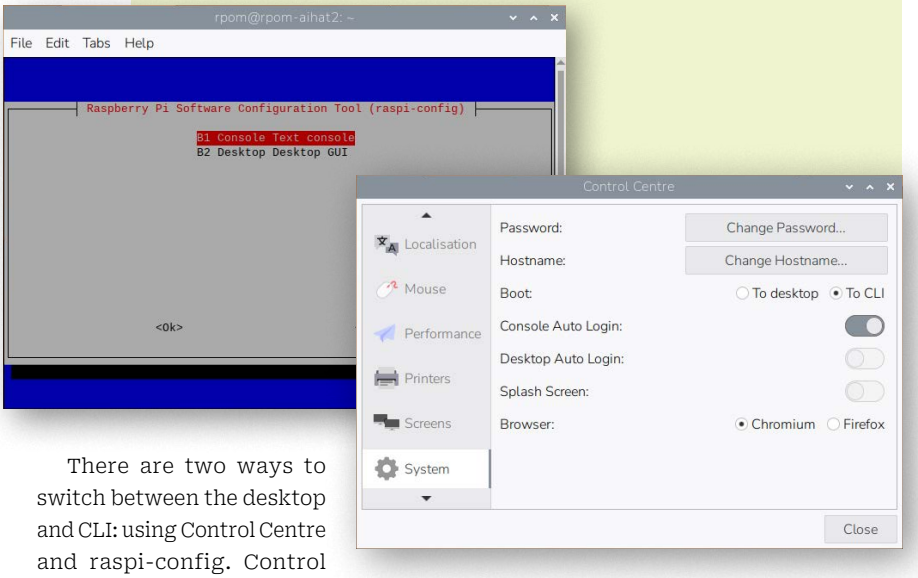
```
$ sudo sysctl vm.swappiness=10
$ cat /proc/sys/vm/swappiness
```

Note that this change is not permanent and changing the swappiness runs a risk of out of memory errors (OOM). For this reason, we don't recommend Raspberry Pi 5 1GB RAM for desktop computer use; it's better to move to a 2GB RAM model (or more) for this intended use case.

However, we really do recommend using Raspberry Pi 5 1GB RAM for electronics and maker projects. Especially if you turn the desktop off and start using the command-line interface.

Going headless

It is clear that we can reclaim a lot of the overhead on our 1GB of RAM by running the computer without the desktop interface. This is known as 'headless' mode, although officially it is referred to as booting to the command-line interface (CLI). Press **F10** or **CTRL+C** to exit htop.



◀ Choosing between the desktop and command-line interface in raspi-config and Control Centre

There are two ways to switch between the desktop and CLI: using Control Centre and raspi-config. Control Centre is a desktop app, so the inherent advantage of using raspi-config is that you can switch between booting to desktop and CLI mode from the command line. Here is how to switch to the CLI:

- **Control Centre:** Open Control Centre and choose System from the sidebar. Here you will see the Boot option with 'To desktop' and 'To CLI' as options. Choose 'To CLI' and then click Close. You will need to restart Raspberry Pi to switch from desktop to CLI.
- **Raspi-config.** Open a terminal and enter `sudo raspi-config`. Choose 1 System Options > S5 Boot > B1 Console Text console. Now choose <Finish> and then <Yes> at the 'Would you like to reboot now?' prompt.

Look mum, no desktop!

You will now boot directly into the command line and be faced with a black screen and flashing cursor. This may be familiar to some computer users who are just getting into their stride, age-wise. But for most of us, the lack of a desktop is unfamiliar territory. Fear not. You don't need the desktop. Let's take another look at free:

```
$ free -h
```

Now we see that we're using just 266MB of RAM and have 720MB available on our Raspberry Pi 5 1GB. This is a much more acceptable amount of free RAM for us to start running processes.

Start up htop with:

```
$ htop
```

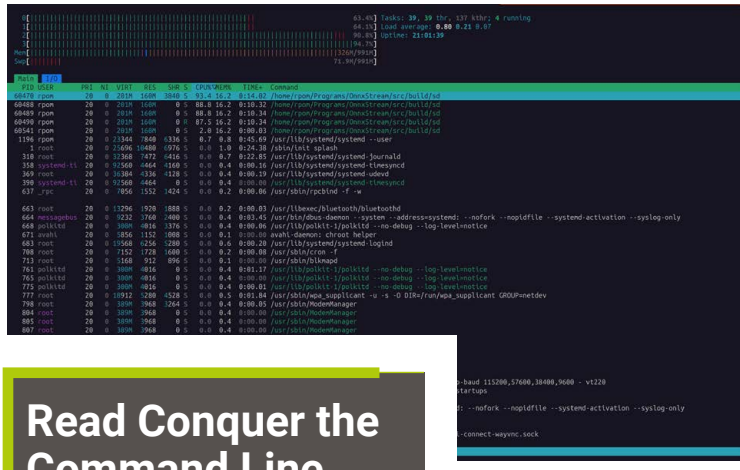
...and we can see much of our system is now free for us to experiment with.

Having just the single interface does lose much of the multitasking functionality inherent in the desktop. But don't worry because we can have up to six terminals running at once. Press **ALT+F2** to swap to our second terminal. Enter your username and password and you can now access the command line from another instance. Press **ALT+F1** and **ALT+F2** to jump between the two. You can repeat this for **ALT+F3**, **F4**, **F5**, and **F6**.

If you want to get back to your desktop interface temporarily, enter the command:

```
$ startx
```

This will boot up the desktop interface. You can return to the command-line interface by rebooting the system. Use Control Centre or `sudo raspi-config` to return to the desktop interface on an ongoing basis.



▲ Running htop from the command-line interface reveals much more available RAM on our Raspberry Pi 5 1GB RAM

Read Conquer the Command Line

If you want to get into working from the command line, take a look at Richard Smedley's book, *Conquer the Command Line*.

rpimag.co/commandlinebook

Part 01

Generate images with Stable Diffusion on a Raspberry Pi 5 1GB

Use OnnxStream to run an image diffuser in a limited RAM space

Maker
Lucy Hattersley
Lucy is in the pub with her "space horses", much to the amusement of all.

rpimag.co

Image generation is a powerful demonstration of what is possible on a Raspberry Pi 5 1GB RAM. Using OnnxStream, we can create images by streaming the weights from a Stable Diffusion model into our Raspberry Pi 5's four-core CPU.

This is great fun and puts heavy demands on our powerful CPU while keeping the RAM usage light. We should note that image generation is one of the more fascinating and controversial aspects of AI. Models such as Stable Diffusion are trained on several billion images.

Image generation is one of the more fascinating and controversial aspects of AI

Diffusers are one of the most modern image generation techniques. They are trained by adding Gaussian noise to images. They then run the process in reverse, gradually predicting and removing noise from an image in iterations known as steps. This iterative denoising process produces an image that is coherent to humans.

We will look at a library called OnnxStream by Vito Plantamura (**rpimag.co/onnxstream**). This decouples the inference engine from the weights. Models like Stable Diffusion contain over a billion weights; the inference engine uses these to make a decision on what each pixel should look like at each step.

Bookmark the GitHub

OnnxStream can be found on Vito Plantamura's GitHub page: **rpimag.co/onnxstream**

▲ Our best generated image of an astronaut riding a horse, generated using Stable Diffusion XL with 35 steps

Warning!
Random art
Image generators produce random artworks and although there are 'not safe for work' checks built-in, you may still get images that are NSFW. Children should be supervised.

Legal issues

On 4 Nov 2025, the UK High Court ruled that Stable Diffusion does not infringe Getty's copyright, since it "does not store or reproduce any Copyright Works (and has never done so)". This ruling is because the model was trained outside of the UK but is used in the UK. The issue of training models on copyrighted images inside the UK is still under legal consideration.
rpimag.co/sdvsgetty

Typically, they are all stored in RAM alongside the inference engine. With OnnxStream, just the inference engine is stored in RAM and the weights are streamed into RAM as they are needed.

This enables a version of Stable Diffusion to run on a Raspberry Pi Zero 2 W with 512MB RAM or our Raspberry Pi 5 with 1GB RAM.

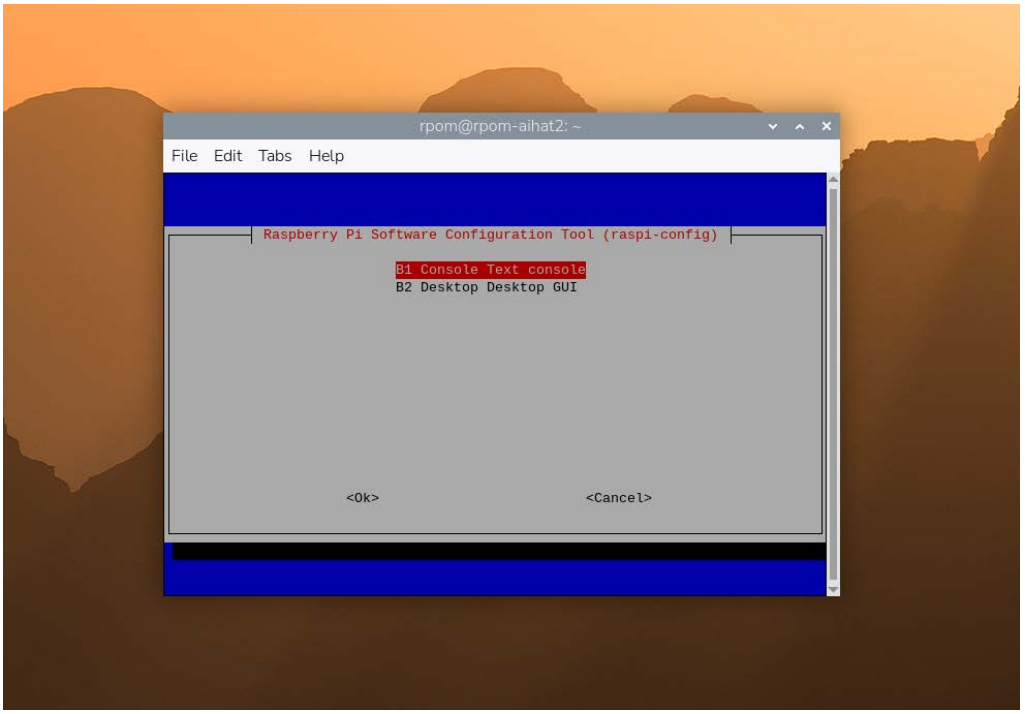
Image generation lends itself perfectly to projects that generate a prompt and display an image on a Raspberry Pi Display. You could grab weather data from an API and use it to generate the day's weather. Or a to-do list that shows your next task visually. We've also seen an interesting Raspberry Pi camera without a lens that grabs data from its current location, surrounding buildings, the time and weather, and then 'guesses' what the view is like.

Get Stable Diffusion model with OnnxStream

OnnxStream enables you to run Stable Diffusion v1.5 and even SDXL in small RAM spaces. You can even run it on a Raspberry Pi Zero 2 W with just 512MB RAM. The downside is that this increases the image generation time as the latter makes heavy use of our CPU resources.

It is an ideal test case for Raspberry Pi 5 with 1GB RAM, however, which blends four fast CPU cores with enough RAM to hold our inference engine and stream our weights.

We can even use Stable Diffusion XL 1.0 (SDXL) in a limited memory space. SDXL is a much larger model that generates 1024 × 1024 resolution images on Raspberry Pi. In this limited space, it is advised to run it in a command-line interface (CLI) rather than with the full graphical user interface (GUI). This frees up around 350MB of RAM, which on a device with limited RAM – such as the Raspberry Pi Zero 2 W with 512MB RAM or Raspberry Pi 5 with 1GB RAM – is a substantial amount. Running in headless mode will prevent our Raspberry Pi 5 with 1GB from using too much of the swapfile space.



▶ Using raspi-config to switch to headless mode

The downside to using OnnxStream over loading the Stable Diffusion full model is the time it takes to generate images. Each step takes around five minutes to run and a full 35-step image will take around three hours to produce.

Still, OnnxStream is a great way to play around with image generation in a memory-constrained environment. It also comes packed with features and makes downloading and using Stable Diffusion v1.5 extremely easy.

Switch to headless mode

Start with a fresh installation of Raspberry Pi OS Lite, or turn off the GUI. If you already are in the GUI, open a terminal window and enter:

```
$ sudo raspi-config
```

Select 1 System Options > S5 Boot > B1 Console Text Console. Choose Finish and select <YES> at the 'Would you like to reboot now?' prompt.

Build OnnxStream

We need to build from scratch, so make sure you have all the requirements. We're going to put the files in a folder called **Programs**. Create it with:

```
$ mkdir ~/Programs
$ cd Programs
```

Now, make sure we have the required program files:

```
$ sudo apt update
$ sudo apt install build-essential git cmake
python3
```

Raspberry Pi OS should already have the latest versions of most of these. It will need to install cmake and its dependencies.

Now clone the GitHub repo and build the files from source:

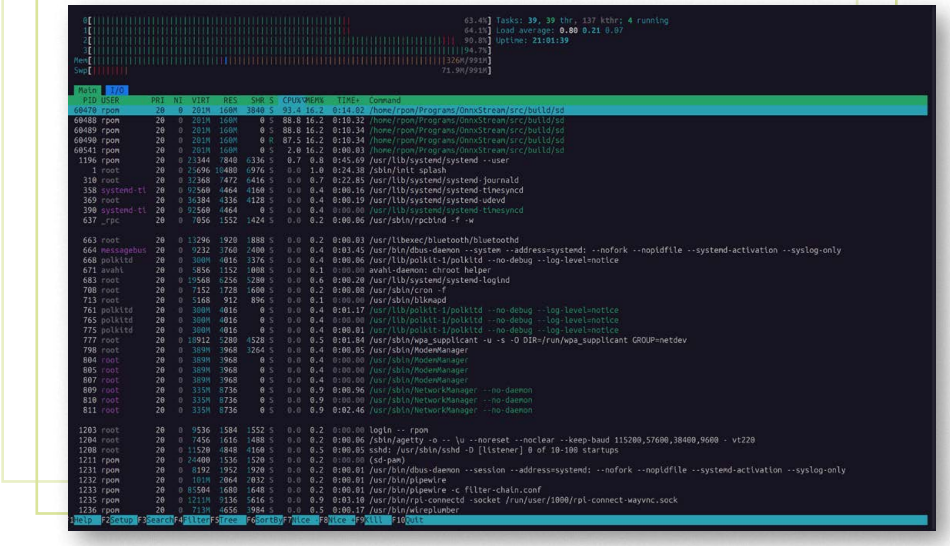
```
$ git clone https://github.com/vitoplantamura/
OnnxStream.git
$ cd OnnxStream/src
$ mkdir build
$ cd build
$ cmake ..
$ cmake --build . --config Release
```

The command to run OnnxStream's version of Stable Diffusion is `sd`. You will find it in **~/Programs/OnnxStream/src/build**. You should still be in this folder, but you will need to navigate to it on future uses:

```
$ cd ~/Programs/OnnxStream/src/build
```

We can now activate sd by using the `./sd` command. First, let's see what commands are available with:

```
$ ./sd --help
```



▲ Checking CPU and RAM usage with htop

This lists all commands. The ones we're interested in are:

```
--xl            Runs Stable Diffusion XL 1.0
                instead of Stable Diffusion 1.5.
--turbo         Runs Stable Diffusion Turbo 1.0
                instead of Stable Diffusion 1.5.
--models-path   Sets the folder containing
                the Stable Diffusion models.
--output        Sets the output image file.
--prompt        Sets the positive prompt.
--neg-prompt    Sets the negative prompt.
--steps         Sets the number of diffusion
                steps. Default is 10.
--ram           Loads the entire UNET model into
                RAM for faster inference. Sets
                --not-tiled.
--download      A[uto] / F[orce] / N[ever]
                (re)download current model.
--rpi           A[utodetect] / F[orce] /
                D[isable] to configure the
                models to run on a Raspberry Pi.
--rpi-lowmem    Configures the models to run
                on a Raspberry Pi Zero 2 W.
```

Check in with htop

Switch to another terminal instance with **CTRL+ALT+F2**. Enter your login and admin password. Now enter:

```
$ htop
```

And press **SHIFT+M** to sort by memory usage. You'll see the `sd` command taking up all of your CPU processing power and a chunk of your physical memory.

Press **CTRL+ALT+F1** to return to your default terminal space. Note that **F1**...**F6** spaces are available.

Set sd instead of ./sd

We place `./` in front of `sd` to inform the OS to use the file in the current folder (indicated with a single `.`). This prevents the OS from heading off into the usual PATH environment variables to look for the sd command there.

If we add an alias to the sd file to our **.bashrc** file, we can access sd from anywhere:

```
$ sudo nano ~/.bashrc
```

Add this line underneath '# some more ls aliases':

```
alias sd='~/Programs/OnnxStream/src/build/sd'
```

Exit nano with **CTRL+O** and **CTRL+X** and reload **.bashrc**:

```
$ source ~/.bashrc
```

OnnxStream on Hugging Face

OnnxStream downloads its weights from Hugging Face. You can see the model on Hugging Face Hub.
rpimag.co/sd-onnx

Our first command will be to instruct it to download the weights for the standard model, which is Stable Diffusion v1.5. First, let's create a **Models** folder:

```
$ mkdir ~/Models
$ cd ~/Models
$ sd download
```

OnnxStream will now download the weights for the default model: Stable Diffusion v1.5. When it is finished, it will automatically run the default prompt "a photo of an astronaut riding a horse on Mars" with ten steps. Each step will take around a minute to run.

When it has finished, it will save the file as **./result.png**. Because we are running in headless mode, you can either swap back to the GUI to view it or use SFTP from another computer running a GUI to grab the file.

Take a note of the user@hostname value in the command line (ours is rpom@rpom-1GB) and use this to grab the file from another computer:

```
$ sftp <user>@<hostname>
$ cd Models
$ get result.png
$ exit
```

Open the **result.png** file and take a look. You should have a pretty neat-looking astronaut riding a horse.

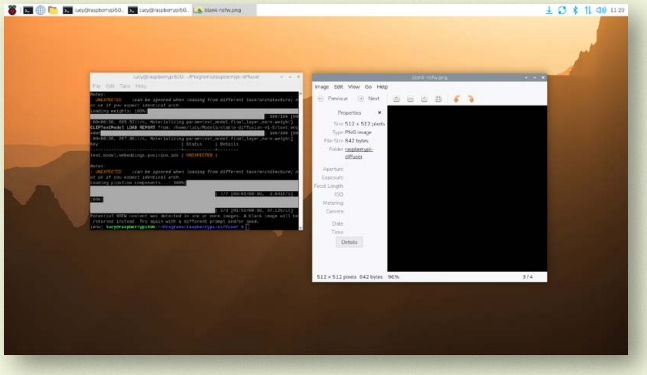
Prompt your model

Now we have our first image, we can take a look at the model. We can check its file size with the `du` command:

```
$ cd ~/Models
$ du -bsh *
```

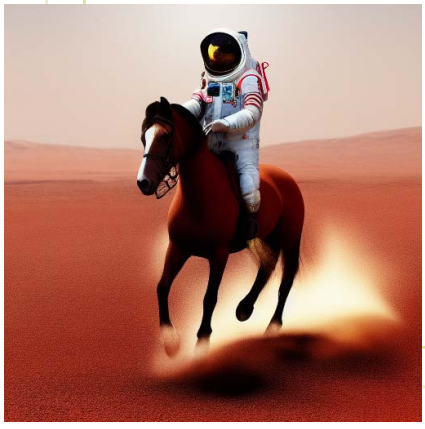
Blank image?

Sometimes you will get a warning that reads: "Potential NSFW content was detected in one or more images. A black image will be returned instead. Try again with a different prompt and/or seed."

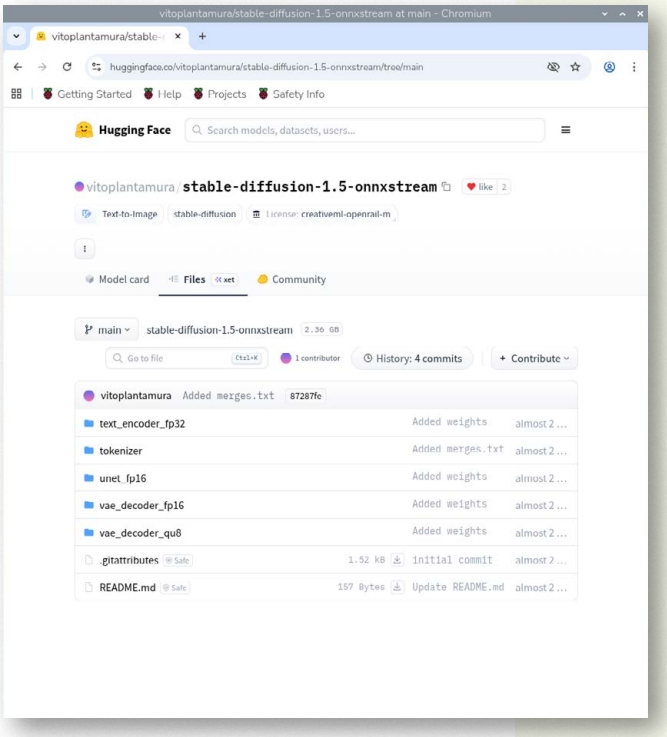


◀ Images deemed NSFW (Not Safe For Work) will be outputted as a blank PNG

OnnxStream is a great way to play around with image generation in a memory constrained environment



▲ The default Stable Diffusion OnnxStream prompt of an astronaut riding a horse



▶ The Hugging Face website

This will show our **result.png** file and the **stable-diffusion-1.5-onnxstream** directory, clocking in at 2.2GB. This is a lot more space-efficient than other models. Let's use our `sd` command to create more images.

We move to the home folder because we can run sd from anywhere and use the options to direct sd to the **Models** folder and store our images in the **Pictures** folder. Replace <user> with the name of your user folder. You can get this with `echo $USER`.

```
$ sd --prompt "a photo of a black cat with
green eyes" --neg-prompt "blurry" --models-path
"/home/<user>/Models" --output "/home/<user>/
Pictures/black-cat.png" --steps 3 --rpi-lowmem
```

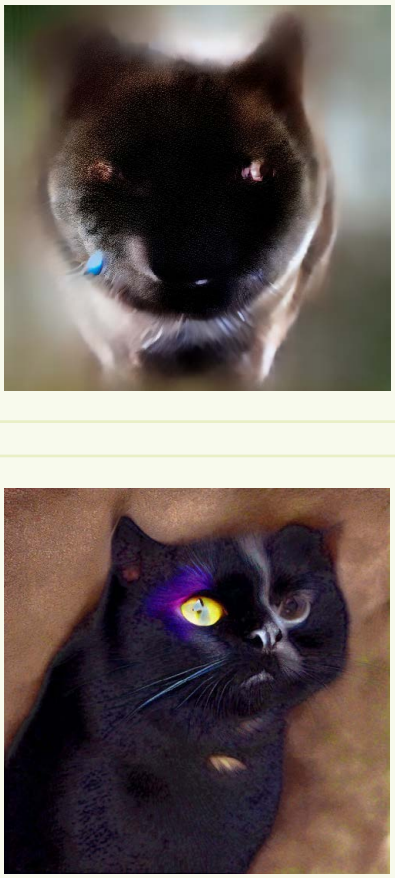
This produces an image of a black cat in **Pictures**. It should take around three minutes. Let's break that command down:

- `sd` run the stable diffusion model
- `--prompt` what we want the image to contain
- `--neg-prompt` what we want the image to avoid
- `--models-path` where we downloaded the stable-diffusion-1.5-onnxstream model
- `--output` where to save the image file name
- `--steps` how many steps the diffusion process should take
- `--rpi-lowmem` optimises the model for Raspberry Pi Zero 2 W (which works on our Raspberry Pi 5 1GB model)

Experiment with different prompts and steps. Around 30 to 40 steps generally produces good results. If you have a Raspberry Pi with 4GB RAM or more, you can replace `--rpi-lowmem` with `--rpi`.

The 35 steps

We find 35 steps produces good-quality images and takes around one hour to produce an image on a Raspberry Pi 5.



▲ An image of a "black cat with green eyes" generated with Stable Diffusion 1.5 using 5, 10, and 35 steps. This demonstrates how each step denoises the image further to remove randomness and add clarity

▲ Images of our astronaut riding a horse generated using ten steps with Stable Diffusion XL. This demonstrates innate higher quality than the SD model, although some elements are still fuzzy – this could be improved upon by using more steps

▶ An image of a black cat with green eyes generated using Stable Diffusion XL with 35 steps

Using Stable Diffusion XL

Our OnnxStream installation can also run Stable Diffusion XL. SDXL is a much more recent, and powerful, image generation model. It comes with a longer runtime, but the images have a higher resolution (1024 × 1024) and tend to be higher-quality.

You can download Stable Diffusion XL in OnnxStream using the `--xl` flag:

```
$ cd ~/Models
$ sd --download --xl
```

> *SDXL is a much more recent, and powerful, image generation model*

The model will download the weights and then generate the default image of an astronaut riding a horse on Mars with ten steps. This took around 40 minutes on our Raspberry Pi.

Once OnnxStream has finished downloading and testing the model, you can run it on Raspberry Pi by adding the `--xl` flag to your command:

```
$ sd --xl --prompt "a photo of a black cat
with green eyes" --neg-prompt "blurry"
--models-path "/home/<user>/Models" --output
"/home/<user>/Pictures/black-cat-sdxl.png"
--steps 35 --rpi-lowmem
```

Expect it to take around three hours to run completely. But the result should be a pleasingly convincing cat image generated on a Raspberry Pi with just 1GB RAM.

▶ A default test image of an astronaut on a horse. This is used as a test because it combines two things not normally seen together

▶ Each run will produce a different result as it removes noise from an image in each step. But models produce similar images

▶ You can produce an image of most things by adjusting the positive and negative prompts. Image quality is determined by step count. Here is a cat produced with just five steps

Raspberry Pi's GPIO header allows you to connect electronic components and control them with code you've written yourself. Python is the most popular programming language for controlling electronics on a Raspberry Pi, particularly the functions in the GPIO Zero library. With this book, you'll learn how to use GPIO Zero as you build a series of simple electronics projects.

- ***Simple electronics projects including:***
  - *Program some LED lights*
  - *Add a push button to your project*
  - *Build a motion-sensing alarm*
  - *Create your own distance rangefinder*
  - *Make a laser-powered tripwire*
  - *Build a Raspberry Pi robot*

BUY ONLINE: **rpimag.co/gpiozerobook**

Unusual tools: wrap your prototype

In the days before surface-mount components and breadboards, the wrapping tool was the king of the lab

Maker

Dr Andrew Lewis

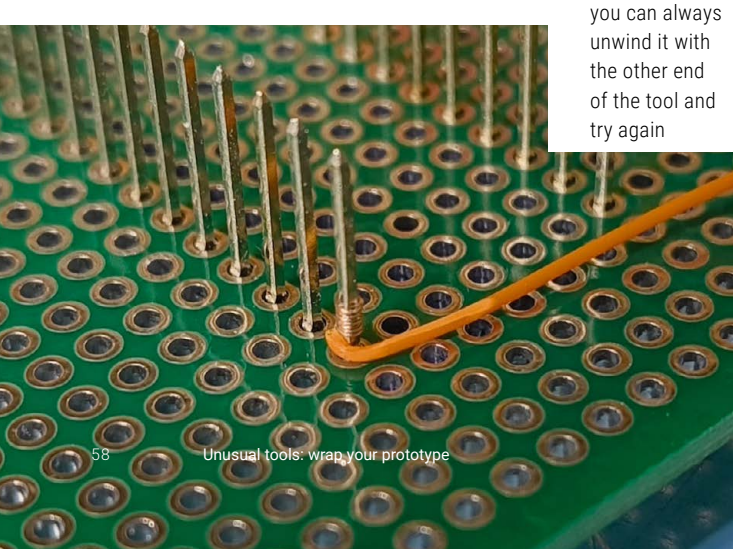
An artist and maker with 30+ years of experience. Maker of novelty items, writer, and former research scientist.

lewis-workshop.com

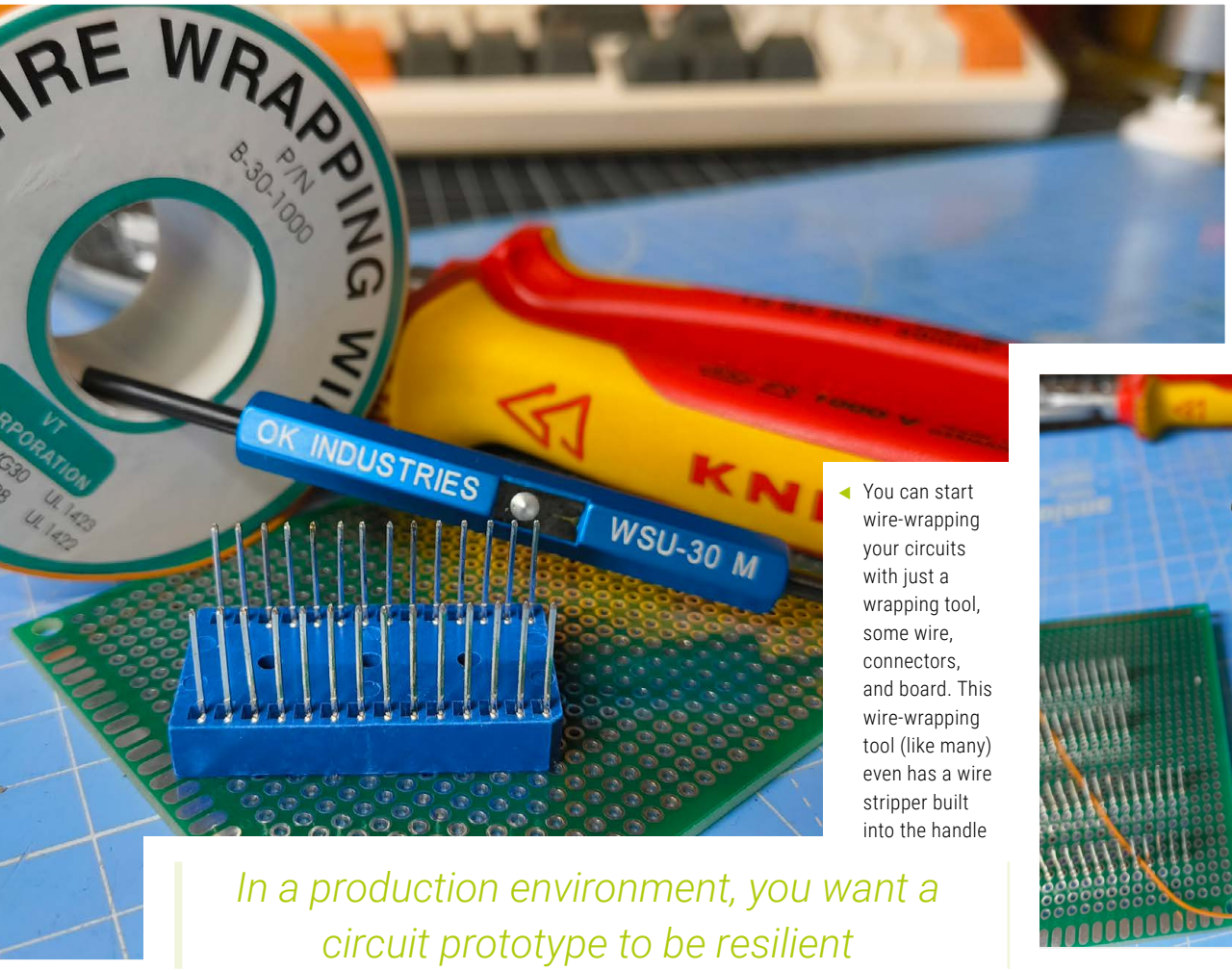
Wire-wrapped circuits were once the gold standard for prototyping, although it's uncommon to see them in the wild these days. You could go to any electronics supplier and buy special versions of sockets and connectors with wire-wrap posts on them. Wire wrapping is pretty simple, but it's also an incredibly powerful process that helped shape the technology landscape of the 1970s, 1980s, and even the 1990s.

DuPont wires are not the only way

You take a protoboard with pre-drilled holes in it, add your components through the board, either directly or by placing them into inline sockets. Then you would twist wire around the protruding lugs on the reverse of the protoboard using a special tool. For the home hobbyist, a manual screwdriver-like wrapping tool that you spin around in your fingers would be the most common choice. It's small, inexpensive, and easy to use. More professional users would probably opt for a semi-automatic tool that rotates automatically when you squeeze a trigger. Real high-rollers would have an electric tool that rotated for a set number of turns when you pushed the trigger. This would be the sort of tool you'd probably find on the desks of Sinclair Research Ltd or Commodore International in the 1980s – where multiple technicians would spend their time wire-wrapping revisions on the boards that eventually became the home computers we all remember.

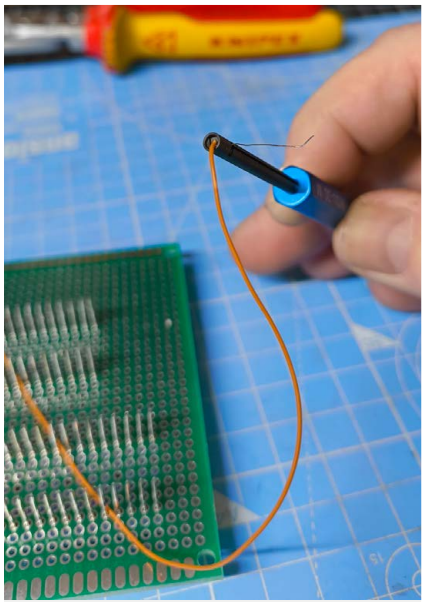


▼ A nicely wrapped wire has a little bit of insulation at the end to aid with strain relief. If you make a mistake, you can always unwind it with the other end of the tool and try again



▼ Feeding the very thin wire into the end of the wire-wrapping tool can be tricky if you don't know where to look. The notch is to the side of the main hole, near the groove in the shaft

◀ You can start wire-wrapping your circuits with just a wrapping tool, some wire, connectors, and board. This wire-wrapping tool (like many) even has a wire stripper built into the handle



In a production environment, you want a circuit prototype to be resilient

Despite the apparent simplicity, wire wrapping has a few advantages over breadboarding or dead-bug style soldering. Wire-wrapped circuits are very sturdy when you compare them to breadboarded alternatives. In a production environment, you want a circuit prototype to be resilient enough to carry between desks and survive general handling. You also want your circuits to be easy to rework. If you make a mistake or change the design, you don't want to wait for a new board to etch, desolder components, or cut tracks with a razor blade to change the design. With wire wrapping, you just turn the tool around and use it to unwrap the wire.

How to become a successful wrapper

When wrapping wires yourself, there are a few tips and tricks that you should know. Wrapping tools are designed to use a specific type of 30AWG wire. You need to strip about 2.5cm of insulation away from the wire, and insert it into the notch in the wrapping tool. Once the tool is seated properly on the post, twist it clockwise until the wire is fully wrapped. Ideally, the first turn of the wire should still have the insulation on, as this helps with strain relief and makes the prototype more resilient. It's very important not to nick the metal wire when removing the insulation, as this will weaken it and probably cause it to snap.

That's a wrap

The wire-wrapping tool is still very much a tool for everyone. While not suitable for every scenario, a wire-wrapping tool is useful to have when messing around with microcontroller boards and simple circuits. You can still buy sockets and connectors with wire-wrapping posts on, although they're a lot scarcer than they used to be.

Part 04

Object design with Python in FreeCAD: paths and fonts

Build your understanding of how shapes are represented in FreeCAD

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

robmiles.com

In this episode, we are going to delve into the inner workings of FreeCAD and also make some nice luggage tags. We will discover how to create curved lines and use them to add elements to the faces of objects we want to extrude, including character designs. You can find all the sample programs here: **rpimag.co/pythonfreecad**.

Figure 1 shows luggage tags that were produced by the program we will be creating. The text elements are generated from TrueType fonts. Let's see how we can make them.

Going straight

Up until now, all the objects that we have created have been made up of straight lines. Our luggage tag needs a curved edge for the cord hole. Let's discover how to use a curve to express this part of the shape, starting with a shape made of lines and adding a curve to it.

```
v1 = App.Vector(0, 0, 0)
v2 = App.Vector(width, 0, 0)
v3 = App.Vector(width, depth, 0)
v4 = App.Vector(0, depth, 0)
```

▲ **Figure 1:** The letters are embedded in the tag surface

We have seen the code here before. Each statement creates a vector specifying the position of a vertex which is one of the four corners of a rectangle. The values in `width` and `depth` set the size of the rectangle. We can create lines between the four vertices as follows:

```
e1 = Part.makeLine(v1, v2)
e2 = Part.makeLine(v2, v3)
e3 = Part.makeLine(v3, v4)
e4 = Part.makeLine(v4, v1)
```

The code above creates variables `e1` to `e4`, defining the four edges of the face we are going to create. An edge is described in terms of a start vertex and an end vertex. Each edge starts at the end of the previous one, and the end of the final edge, `e4`, takes us back to the start of the shape. We can use these edges to create a wire which contains the path around the shape:

```
wire = Part.Wire([e1, e2, e3, e4])
```

The `Part.Wire` function accepts an ordered list of edges and creates a path. If the path ends at the same place as it started (which ours does), the path is closed and can be used to create the face of an object. A complex shape may contain many edges.

Figure 2 shows the path that we have created. It also shows the vertices that define the corners and each of the edges. We can use the value of `wire` to create a face that can be extruded to make a solid object.

▼ **Figure 2:** The drawing operation is performed in an anticlockwise direction. This will become important later

We will discover how to create curved lines and use them to add elements to the faces of objects we want to extrude

```
panel_face = Part.Face([wire])
panel = panel_face.extrude(App.Vector(0, 0,
height))
```

These statements make the panel object by creating a face from the wire and then extruding it, resulting in a rectangular panel. The height (thickness) of the panel is specified by the value in `height`.

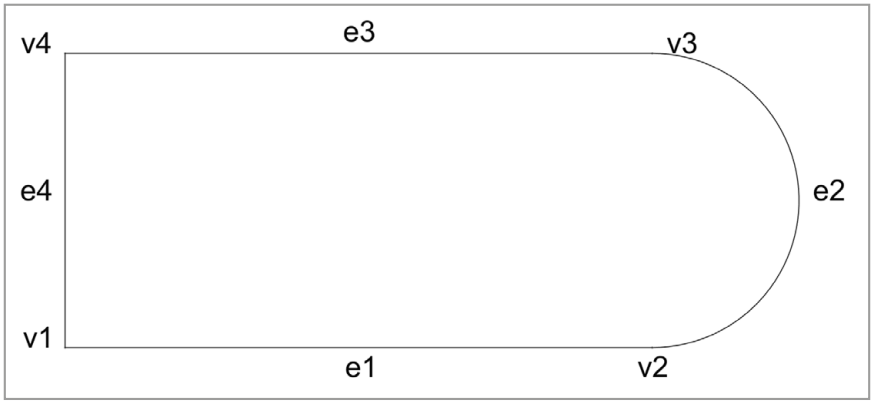


Figure 3: The curve replaces the edge between v2 and v3

Throwing a curve

We want to replace one of the edges in the face with the curved edge of the luggage tag.

Figure 3 shows where we want to add a curve. There are several ways of describing a curve in FreeCAD. We are going to specify the curve as three vertices: start, middle, and end. The curve needs to pass through the vertex on the far right of the tag shown in **Figure 3**. We need to work out the position of this vertex. We want to make the curve part of a circle that has a diameter which is the depth of the tag.

```
curve_radius = depth/2.0
```

The statement above calculates the radius of the curve we want, which is half of the depth of the tag. Remember that the radius of a circle is half the diameter. Now that we have the curve radius, we can use it to work out the position of the curve middle.

```
curve_middle = App.Vector(width+curve_
radius,curve_radius,0)
```

We calculate the x position of `curve_middle` by adding `curve_radius` to the width of the tag, while the value of `curve_radius` (which is half of `depth`) provides the y value. If you find this confusing, try drawing it out on paper. That's how the author worked it out. Now that we have a vertex defining the middle of the curve, we can use it to create the lines that define the new tag shape.

```
e1 = Part.makeLine(v1, v2)
e2 = Part.Arc(v2, curve_middle, v3).toShape()
e3 = Part.makeLine(v3, v4)
e4 = Part.makeLine(v4, v1)
```

The block of code above makes a tag shape with the curved section. The edge `e2` is now a curve. The `Part.Arc` function accepts vectors defining the start, middle, and end of the curve and returns an arc which is converted into an edge by the `toShape` method.

QUICK TIP

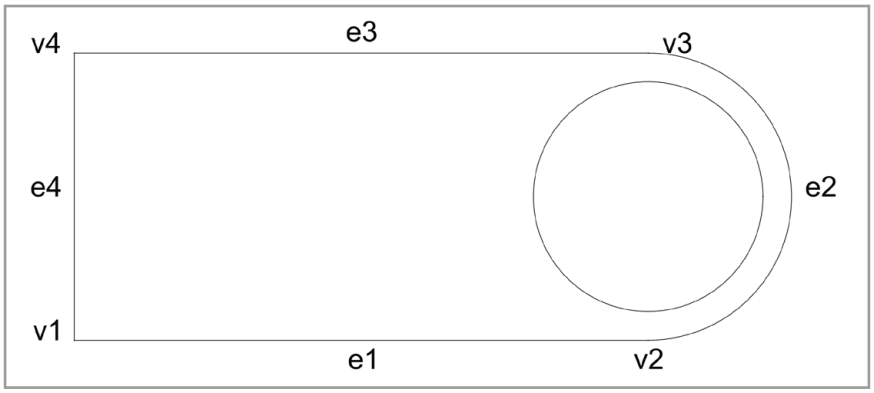
We could convert the other lines into arcs to make more complex curved shapes. We can also move `curve_middle` to change the shape of the curve.

Now that we have our edges, we can use them to create a wire which describes the path around the outer edge of the tag in the same way as we did before:

```
tag_wire = Part.Wire([e1, e2, e3, e4])
```

The statement above creates a wire called `tag_wire` that describes the path around the outside of the tag. We can use this wire to create a face.

```
panel_face = Part.Face(tag_wire)
```



▶ **Figure 4:** The centre of the hole is on the line between v2 and v3

The previous statement feeds `tag_wire` into the `Part.Face` function to create a face that could be extruded to make a tag. However, before we do any extrusion, we need to add a hole to the tag design.

Hole-hearted

Now that we have the wire for our tag face, we need to make a hole in it for the cord that will tie it to our suitcase. **Figure 4** shows what we are trying to make. We have already seen that we can use the FreeCAD cut operation to subtract one object from another, so we could cut a hole in the tag by making a cylinder with the required size and position and then cutting this from the tag. However, there is another way of creating a hole, and this is to add another wire to the definition of the tag face. This wire will describe a circular path to be followed to remove a circle from the tag. Before we create the wire, we need to decide the position of the circle and its size. The circle must fit inside the tag with a margin around it. We can make the size of the margin a variable:

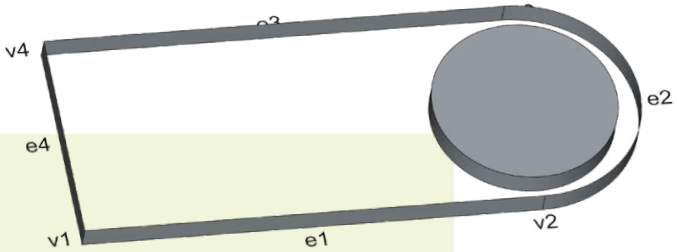
```
hole_margin = 5.0
```

The statement above creates a variable called `hole_margin` which is set to 5.0, which means that the hole will have a 5mm margin around the edge. Now we can calculate the radius of the hole we want to make:

```
hole_radius = curve_radius - hole_margin
```

The statement above calculates the radius of the hole we want to make. We calculate it by subtracting the margin from the radius of the outer curve. Next, we need to work out where the hole needs to be. The centre of the hole should be halfway up the right-hand edge of the tag.

▼ **Figure 5:** If you view the design from directly above, it looks fine, but as soon as you use a 3D view, you can see something is wrong




```
hole_pos = App.Vector(width,curve_radius,0)
```

The code above creates a vertex with the required position. Now we can use these values to create a circle for our hole.

```
tag_hole = Part.makeCircle(hole_radius,
hole_pos)
```

The `makeCircle` function returns a circle with the specified radius (`hole_radius`). The circle is located at the specified position (`hole_pos`). We can convert this into a wire:

```
hole_wire = Part.Wire(tag_hole)
```

The statement above creates a wire called `hole_wire` which contains the path defined by the circle. We now have two wires: `tag_wire` and `hole_wire`. These contain paths which define the outer tag shape and the hole to be cut into it. We can feed these wires into the `Face` function to make a face with a hole in it:

```
panel_face = Part.Face([tag_wire,hole_wire])
```

The statement above calls the `Face` function to create the panel face for our tag design. The `Face` function is provided with a list of the wires it must work with. There is only one problem with this code. It doesn't work.

Figure 5 shows a 3D view of the tag we have just made. Instead of a solid tag with a hole in it, we have a strange perimeter and a solid hole. This is what happens if we use the `Face` function to put one solid object inside another. We need to tell the `Face` function that the circle is not solid but should be subtracted from the tag. We do this by using the winding order of a shape. Note that this is the word 'winding' in the context of winding up a clock by turning a key, not making a baby go burp. The winding order of a shape is the order in which it is drawn.

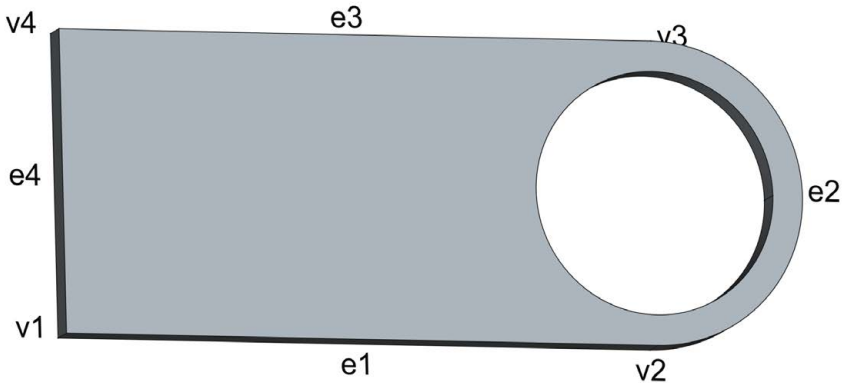


Figure 6: The outline and the cutting shape now have different winding directions

If you look at the edges in **Figure 5**, you will see that the drawing action (going from `e1` to `e4`) follows a counterclockwise direction. The winding order of the tag is counterclockwise, or CCW for short. FreeCAD adopts the convention that anything drawn CCW is to be added to the drawing. Unfortunately, the default winding order of a circle produced by the `makeCircle` function is also CCW, so when the two are combined they create an invalid model. If we place the circle outside the tag, it would be added correctly to the model. What we need to do is reverse the winding order of the circle so that it is marked to be removed from the tag. A wire object provides a reverse method:

```
hole_wire.reverse()
```

The statement above reverses the winding order of the `hole_wire` and so it will now be subtracted from the tag and we get the shape that we want.

> *We need to reverse the winding order of the circle*

Figure 6 shows the tag with the hole cut in the correct position. You might be wondering why we have spent all this time finding a way of making holes in things which is more complicated than the cutting operations that we used earlier. This is because if we want to make a very large number of holes in an object (perhaps a 3D printed cheese grater), it is much more efficient to create a face with the holes in and extrude it than it is to perform many cut operations. Knowing how wires are used to represent paths is also useful when working with font designs in FreeCAD. With that in mind, let's add some text to our tag.

Paths and fonts

FreeCAD can use fonts on your computer to create faces which can be extruded and fused with the tag. The text can be added on top of the tag, or we can create a separate object which can be printed in a different colour. First, we need to find a font to work with. Different computers store their fonts in different places. We can use a little helper function to find the fonts on our system:

```
def find_font():
    candidates = [
        # Windows
        r"C:\Windows\Fonts\arialbd.ttf",
        # Linux
        "/usr/share/fonts/truetype/liberation/
        LiberationSans-Bold.ttf",
        # macOS
        "/System/Library/Fonts/Supplemental/
        Arial Bold.ttf",
        "/Library/Fonts/Arial Bold.ttf",
        # Flatpak
        "/usr/share/fonts/liberation-fonts/
        LiberationSans-Bold.ttf"]
    for p in candidates:
        if os.path.exists(p):
            return p
    raise FileNotFoundError("Font not found")
```

The `find_font` function above contains a list of font locations for different machines. It works through each location and returns the first font file it finds.

```
font = find_font()
```

The previous statement sets the value of font to a path to a font file. But before we can use that font, we must set some formatting values.

```
text_margin = 1.0
text_height = 1.0
text_size = depth - (2*text_margin)
```

The code above sets `text_margin` (the space above and below the text on the tag) to 1mm. It then sets `text_height` (the height of the extruded text) to 1mm, and finally `text_size` (the depth of the tag minus the top and bottom margins) is calculated. Now we can make our text shape. The Draft library contains the function we are going to use to do this, so let's import this now.

```
import Draft
```

Now that we can use the Draft library in our program, the `makeShapeString` function in it converts a string of text into a shape which can be extruded.

```
shape_string =
Draft.makeShapeString(String="ROB",
      FontFile=font, Size=text_size,
      Tracking=0)
```

The statement above calls `makeShapeString` to make a shape string called `shape_string`. The function renders the text passed into the `String` argument. It uses the font file specified by `FontFile` and the size of the shape is set by the `Size` argument. The `Tracking` argument allows you to add extra spacing between the characters. Now that we have our `shape_string`, all we have to do to make some 3D text is to extrude the shape in the string:

```
text = shape_string.Shape.extrude(App.
Vector(0,0,text_height))
```

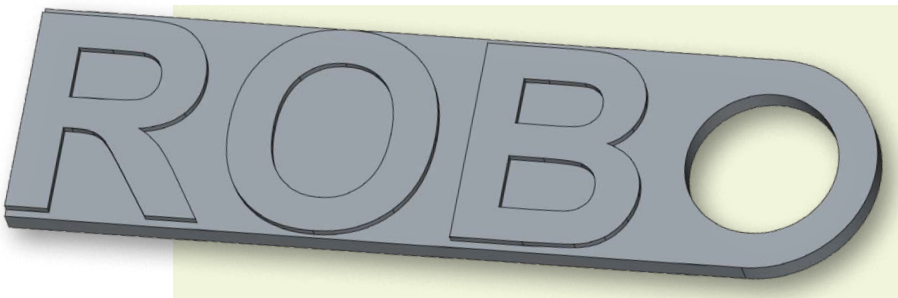
▲ **Figure 8:** The 31 tags took around three hours to print

This code calls the `extrude` method on the `Shape` inside `shape_string`. The `extrude` method has one parameter, which is a vector giving the direction of the extrusion. Once we have the string, we can add it to the luggage tag.

Figure 7 shows a completed luggage tag with the text laid on top of the tag background. It turns out to be quite simple to make tags with coloured text embedded in them. We can cut the text out from the tag and then add it as an object to be printed.

Batch printing

One of the wonderful things about writing programs to make 3D objects is the ease with which you can produce large numbers of different objects. The tags in **Figure 8** were produced by a Python program that generated multiple tags as a single printable object which was printed with a blue layer and a yellow layer. You can find the program in the sample code in the GitHub repo at **rpimag.co/pythonfreecad**.



◀ **Figure 7:** The sample program contains extra code which sets the length of the tag to match the length of the text string

Fabric weather dashboard

Create your own cloud... and the silver lining is that it tells you the weather!

Maker

Nicola King

Nicola is a freelance writer and sub-editor. In this issue, she's making some home décor that tells her the local weather, eats up some of her fabric stash, and provides a comfy headrest. Three wins!

@holtonhandmade

▼ Our paper template, ready to be pinned to the fabric. We chose a cloud, but use your imagination

This month's craft/tech mash-up concentrates on making something that will hopefully prove useful when you want to know what the weather is up to. The older this author gets, the more she is becoming preoccupied with the weather and, given recent downpours in this neck of the woods, when the rain will finally be stopping. (Living in the UK, you'd think she'd just be used to it!)

We'll be sewing a stuffed cloud (other appropriate shapes might be a rainbow, lightning symbol, a huge rain drop, or a sun – we'll leave that up to you), which we'll then link up to a battery and an e-paper screen with a built-in Pico W which will inform us of local weather whenever we want an update.

Sew up a storm

So, to begin, we first need a template. Feel free to draw your own free-hand if you wish, but we chose to download a simple (and free) cloud shape which we enlarged to the size we required. We then printed it out on several pieces of paper, sticking them together with some tape. At its widest point the cloud is 57cm wide and 31cm tall – a decent size, even allowing for the seam allowance when we sew the item together. Once you have your paper template, pin it to your fabric and cut out two pieces of material using fabric scissors.

A word on the fabric here – we used a white (use dark grey if you're feeling sombre) bouclé-type material which we think looks particularly cloud-like. It was also fairly inexpensive at £6.99 for half a metre (**rpimag.co/boucle**). If you have a fabric stash, raid it and see what you can find, or take a look in second-hand stores to find cheap fabric pieces or clothing items that could be usefully upcycled.

A stuffed cloud which will inform us of local weather

▲ This cloud cushion can tell you the latest weather conditions on its e-paper display

YOU'LL NEED:

- Pencil
- Paper
- Paper scissors
- Sticky tape
- Ruler/tape measure
- Fabric for body of item
- Pins
- Stuffing material
- Sewing machine or hand-sewing needle
- Matching thread
- Fabric scissors
- Cutting mat (optional)
- Hook and loop tape
- 2×AA battery pack
- Badger 2040 W (or other e-paper display with wireless connectivity)

Technology and the weather

Over the centuries, weather forecasting has undergone something of a revolution. Observing nature and animal behaviour, sky colours, and cloud formations were once the only way of guessing really what the weather might bring. Pine cones (opening scales and releasing seeds in warmer weather, and closing them when it's damp) are a piece of weather lore as old as the hills, for example. Other than watching for any changes in the environment, our ancestors had no way of knowing what was around the corner weather-wise.

Today, we are moving towards forecasting that is heavily reliant on AI, machine learning, and complex supercomputers running advanced mathematical equations and models. For example, the Met Office's new machine (**rpimag.co/metsupercomputer**), unveiled in May 2025, is the first cloud-based supercomputer dedicated to weather and climate science. In a time of accelerating climate change, this is clearly crucial in order that accurate predictions of how the weather is going to affect us long-term can be made.

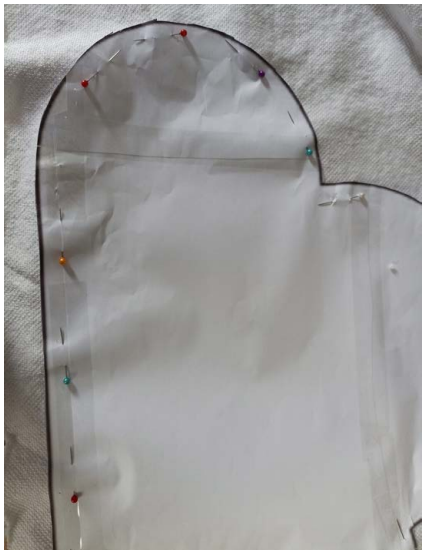
In addition, next-generation satellites continue to push the boundaries. The first images from the Meteosat Third Generation-Sounder (MTG-S) satellite have recently been received – the long-term hope is for even more accurate weather data for Europe and Northern Africa.

QUICK TIP

Once you've completed your stuffed creation, throw it on the sofa and use it as a cushion. The battery and screen are removable when you want to get comfy!

QUICK TIP

If using a sewing machine, check that you have the correct machine foot to feed the fabric through, based on the type of fabric you are using.



▶ Pin the template to your fabric and ensure you use plenty of pins for stability

Once you've cut out your two fabric pieces, think about how you want to attach the battery and screen to your creation. We plumped for a pocket, so we cut a rectangle that was large enough to hold the battery. We then hand-sewed the top edge over for neatness and then attached the pocket to the front piece of our cloud, by turning in each of the three sides by about 0.5cm and top-stitching it in place. We also added a piece of hook-and-loop tape to the front of the pocket so that the other half could be attached to the back of the screen, thus enabling the easy adhesion of screen to pocket. Once the pocket was attached, we pinned the two cloud pieces together (right side to right side) and decided how we wanted to sew them – by hand or by machine? We chose to hand-sew ours as the material was a little 'slippery' when we put it on the machine to try and sew it. We used a simple backstitch (**rpimag.co/backstitch**) and sewed all the way around, leaving an opening around 10cm long on the base.

Next, you need to turn the cloud inside out, so that the right sides are on the outside, and you can then stuff it with whatever you choose. We had a 250g bag of toy stuffing to hand and, using the whole bag, the cloud was beautifully filled out. All that remains is to hand-sew the small seam at the base (that we used to turn the cloud the right side out). You should now have one plump cloud, ready to receive its weather-reading tech.

Come rain or shine – further inspiration

Now, using a Raspberry Pi to help predict the weather is obviously nothing new – if you do a quick search, you'll find plenty of original and clever ideas. Below are just a few inspiring projects that will hopefully make you want to sift through your tech stash and make something similar, or come up with your own variations of…

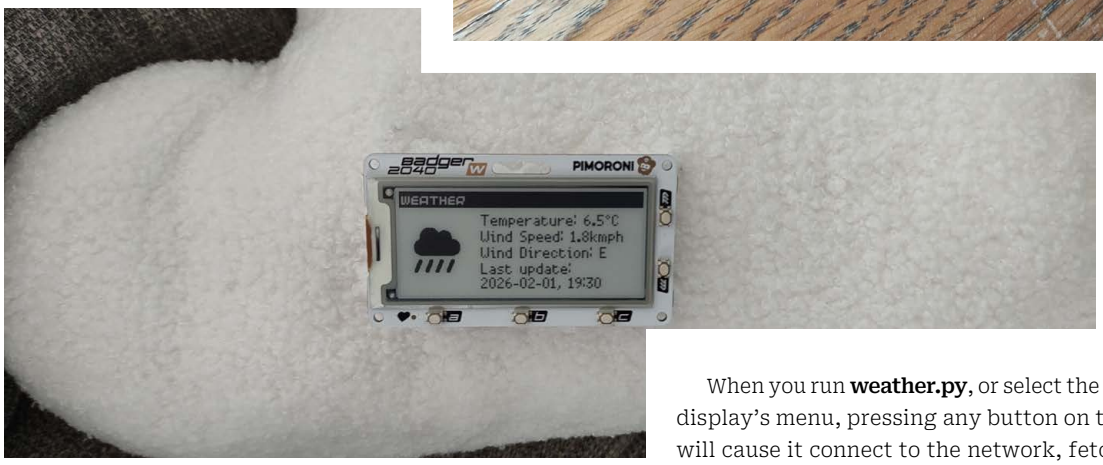
- Admittedly, this one is a family affair – a weather station that this author's husband made a few years ago (**rpimag.co/hsweatherstation**). A Pimoroni Weather HAT and Sensors Kit enabled him to build a weather station that is still working well on our garage to this day!
- This next weather-worthy design is a weather satellite receiving station (**rpimag.co/weathersat**). This is somewhat more involved, as you need to build an antenna, but could be a project to work towards.
- We also like these LED matrix weather lamps (**rpimag.co/weatherlamp**) that display weather data. Practical… but pretty too!

▶ Do leave a gap in your sewing so you can turn it right sides out

QUICK TIP

In terms of stuffing material, you can use old clothing that's been cut into small pieces, an old cushion filler, etc. Whatever you use, just ensure it's clean.

▲ A 2×AA battery pack powers our Badger 2040 W via an input on the rear



◀ With the battery in the pocket, the e-paper display sits on the outside and can be attached with hook-and-loop tape

QUICK TIP

If adding a pocket to the front, make sure it is not too big, or it will sag. You need a snug fit to hold the battery pack in place.

The technology bit

To show the current weather conditions, we opted for a Pimoroni Badger 2040 W e-paper display with a built-in Raspberry Pi Pico W on the rear. This has the wireless connectivity we need to fetch the weather data – in this case from **Open-Meteo.com**, an open-source weather API that offers free access for non-commercial use and doesn't even require an API key.

You could use a different setup, perhaps with a larger screen and a Pico W / 2 W or Raspberry Pi computer, but the Badger 2040 W makes it easy with its built-in weather example (see **rpimag.co/badgeros**). All you need to do is connect it to a computer and use a Python IDE like Thonny to alter a couple of code files to fill in your own details. In **weather.py**, change the LAT and LNG values to your own location's latitude and longitude (easily found by right-clicking on it in Google Maps). Then, in **WIFI_CONFIG.py**, add your wireless network's SSID (name) and password, along with your country's two-letter code (e.g. GB or US) – find yours at **rpimag.co/isocodes**.

▼ Suitably cloud-like fabric that's as soft and squishy as it looks

When you run **weather.py**, or select the Weather app from the display's menu, pressing any button on the front of the board will cause it connect to the network, fetch the latest weather data, and show it on screen.

QUICK TIP

Use plenty of pins when pinning the template to fabric as it will make it easier to cut an accurate shape. Too few pins, and scissors can lose their way!

Naturally, you'll need to power your weather display. The Badger 2040 W has a handy input in the rear to add a battery pack – we used a 2×AA one which fits neatly into the pocket we made. You can attach the display to the front of the pocket with hook-and-loop tape (as we did), or a few stitches through the corner holes.

So, take this idea and run with it, changing and adapting it to fit your own crafty ideas. Maybe you'd prefer to insert the weather tech into a picture frame that you decorate with some weather-related artwork for the kitchen wall, or you could sew it into the pocket of a rucksack so that you have it with you wherever you go... we forecast some fabricating fun.

3D printed contact print holder for cyanotype prints

Create unique artworks from film negatives (and other things) using cyanotype paper

Maker

Rob Miles

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

robmiles.com

Cyanotype printing provides a cheap, easy way to make images from photographic negatives. In this article, we'll see how to make good-looking prints using a 3D printed frame.

Figure 1 shows a variety of pictures printed using the cyanotype process. They were created by putting a photographic negative on top of light-sensitive paper and leaving it in sunlight for a while. The images were taken on black and white film in two sizes: 120 roll film and 5 × 4in large format sheets. Cyanotype paper turns deep blue when exposed to light, which means that

> *The resulting print is the same size as the negative*

when used in conjunction with a negative, you end up with a positive image on the paper.

This kind of image is called a 'contact print' because the negative and the printing medium are in direct contact with each other. The resulting print is the same size as the negative. You can contact-print 35mm negatives. The results are a bit small, although you can print strips of negatives which look really good.

Figure 1: The image on the top right was made by exposing a sheet of paper to light with some blocks on top of it

QUICK TIP

The word 'blueprint' comes from the time when copies of designs and plans were made on cyanotype paper.

Paper please

You can make your own cyanotype paper by adding potassium ferricyanide crystals to ferric ammonium citrate and painting the resulting mixture onto blank paper that you store in the dark before use. This is not particularly hazardous (although you must take sensible precautions) and is useful if you want to make very large prints, but it is easy to obtain sheets of cyanotype paper from your favourite online supplier. Look for the size 'A7', which works well for 4 × 5in large-format negatives.

Cyanotype paper is quite safe to use and is processed using water. However, the author wouldn't give a sheet to anyone he thought might lick it.

Warning!

Chemicals

Always wear nitrile gloves and, ideally, a mask when handling powdered chemicals to avoid breathing in dust. Work in a well-ventilated area, and wash immediately with soap and water if skin contact occurs.

You've been framed

The author wanted a frame large enough to make contact prints of large format (4 × 5in) negatives. Unable to find any at a sensible price, he designed his own. The only parts you'll need to add are a sheet of 3mm thick glass around 130mm by 100mm, some M3 bolts and some elastic bands. The design (**Figure 2**) was created as a Python script running inside the FreeCAD program. If you want to make a frame of a different size, you can modify the source code which is available with the resources for this article.

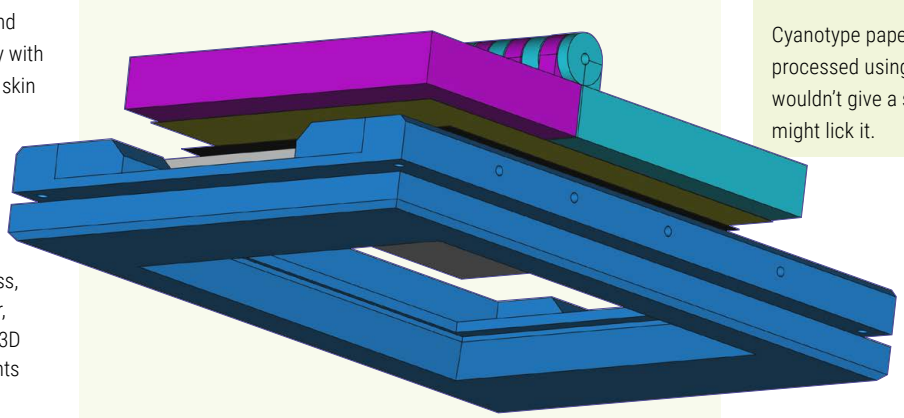
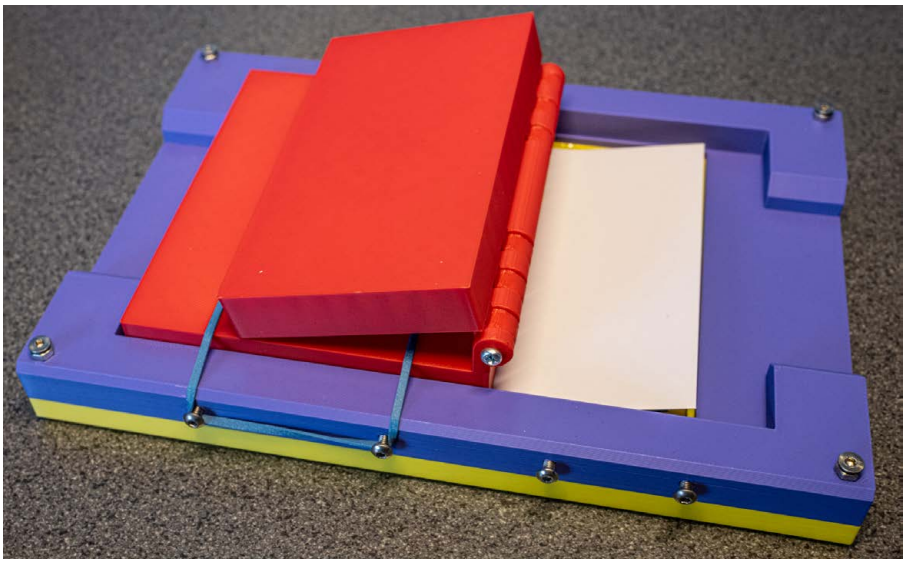


Figure 2: The design also shows the glass, film, and paper, as well as the 3D printed elements



▲ **Figure 3:** The back and front are screwed together

▶ **Figure 4:** You can see the negative behind the glass

Making prints

To make a print, you put your negative up against the glass with the paper behind it. Make sure that the film emulsion (the slightly matt finish side) is directly up against the paper. Then you put in the back of the frame and use elastic bands to hold it in place. **Figure 3** shows paper being loaded into the frame. The back is in two parts so that you can check on a print during exposure. There are cut-outs in the sides of the frame so that you can put lengths of film into the back as well as individual negatives. When you put the back into the frame, you should push it all the way to the end containing the largest part. That way, it is easier to open the smaller end to check the print.

To expose the print, you just leave the frame in the brightest sunlight you can find, as shown in **Figure 4**. Check on the exposure every ten minutes or so and take the paper out when you can clearly see an image on it. Then rinse the paper under running water for a few minutes. This removes unused chemicals and reveals the final picture. After washing, you can hang the print up to dry or place it between two sheets of blotting paper if you want a completely flat print.

Exposure can be tricky. Remember that if there is not enough blue in your print, it is under-exposed (i.e. it needs more time in the sun). Look at the picture of white blocks on the top right of **Figure 1**; this was made by putting the paper in the light with some blocks on top of it. The paper went blue where the light hit it and was white where there was no light. So, lots of light means lots of blue. Lots of white means not enough light.

Unique and analogue

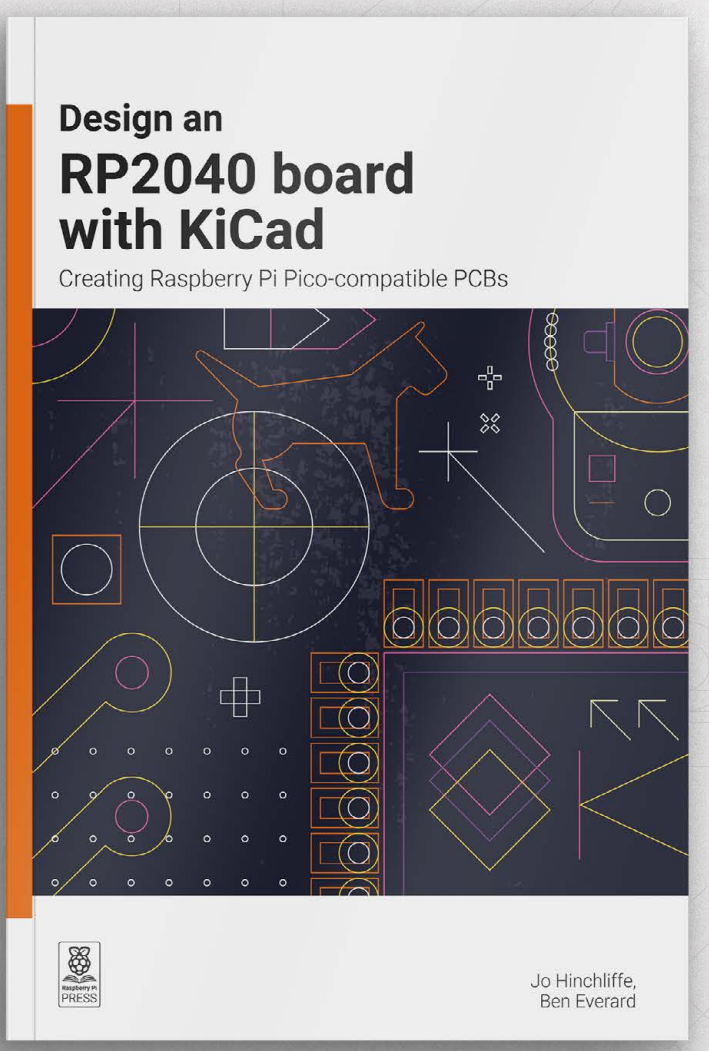
Prints made directly from negatives are special in they are completely analogue. Most pictures that we see today have been digitally processed at some point, whereas cyanotype images are made by chemistry and physics. The dynamic range (the ratio between the lightest and darkest elements in a picture) of cyanotype prints is not as wide as modern photographic technology, but you can use this to your advantage to make things stand out in your pictures.

The field is ripe for experimentation. You should remember that there are no right answers, only things which look good to you. You can find the frame design in the GitHub repository: **rpimag.co/CyanotypeFrame**.

Make your own sunshine

You can expose your prints with an ultraviolet (UV) lamp in place of sunlight. Ultraviolet lights are sold quite cheaply for hardening nail varnish. They work well, although they may only expose for 60 or 90 seconds and you will have to make repeated exposures to get a good result. Move the lamp slightly between each exposure to get an even coverage of the image.

If you are going to do a lot of prints using an ultraviolet light, make sure that it doesn't overheat – the lights can get quite warm with repeated use. You should also avoid exposing yourself to too much ultraviolet light, as you might get a tan you don't want. You could build a light-tight box in which you expose your prints.



KiCad is an amazing piece of free and open source software that allows anyone, with some time and effort, to make high-quality PCB designs.

- ***Create a schematic for a microcontroller board using Raspberry Pi's RP2040***
- *Select the right components*
- ***Customise the hardware for your needs***
- *Lay out and route the PCB design*
- ***Prepare your board for manufacture and assembly***
- *Write software to get your design working*

Buy online: rpimag.co/kicad2040

Part 08

EDSAC (Electronic Delay Storage Automatic Calculator)

In the running to create the first stored-program computer

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

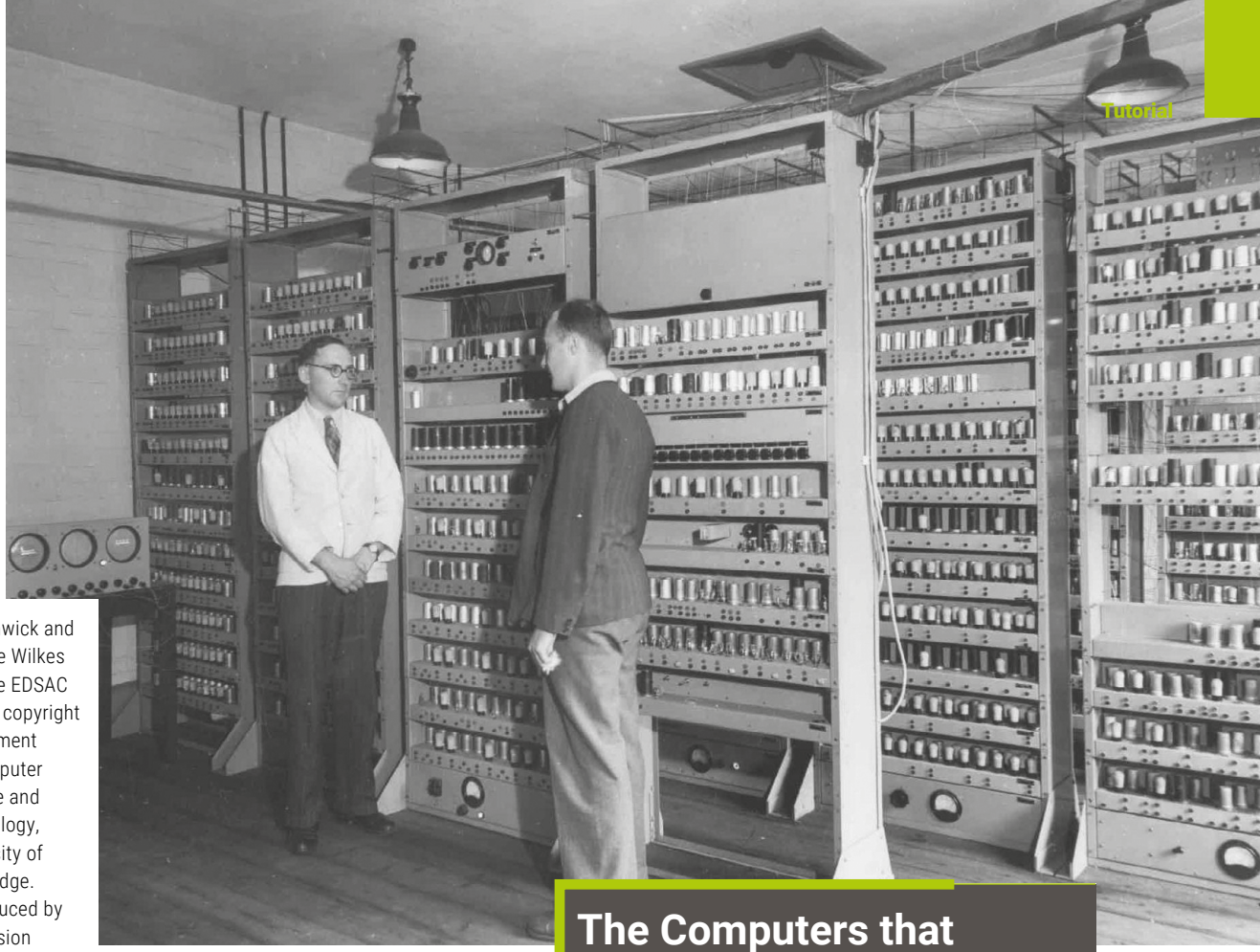
Click-click-click. This was the inauspicious, mundane sound that marked the true beginnings of stored-program computing, as a thin thread of tape wound its way through a reader on 6 May 1949. Two minutes and 35 seconds later, the teleprinter spluttered into action, producing a table of squares from zero to 99.

And it all happened in Cambridge, England, 3500 miles and an ocean away from Philadelphia where the EDVAC was still being assembled.

By all rights, the EDVAC should have been the first stored-program computer. Presper Eckert and John Mauchly, the ENIAC's main creators, fully understood their first computer's limitations and started planning the EDVAC, its natural successor, in mid-1944. That was more than a year before the ENIAC became operational.

With mathematician John von Neumann helping to create a logical outline for this next-generation computer, the US Army giving its financial backing, and a team that was uniquely armed with experience gained from creating the world's first general-purpose electronic computer, the Moore School was surely in the perfect position.

But no. That honour would ultimately fall to Britain, with the Manchester Baby (see part 7 of this series) acting as proof-of-concept and EDSAC following a matter of months later. There are numerous reasons for this, and many people to earn credit, but the giant among them is the pragmatic genius of Maurice Wilkes.



Bill Renwick and Maurice Wilkes with the EDSAC **Image:** copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission

While Eckert and Mauchly were bogged down by patent disputes and their desire to set up a commercial computer company – leading them to leave the army-backed EDVAC project – Wilkes was a one-man decision-making machine. He also had the riches of the University of Cambridge to draw upon. And he was eminently practical.

Unlike his Cambridge contemporary Alan Turing, who would lead the ACE project, Wilkes's interests had always leaned to applied rather than pure mathematics. After graduating with first-class honours in 1934, he gained his PhD in 1937 through work at the University of Cambridge's famous Cavendish Laboratory, where he studied radio wave propagation.

In his tribute to Maurice Wilkes, published in the Biographical Memoirs of Fellows of the Royal Society,[1] Martin Campbell-Kelly writes: "The work involved excursions in a motor car, towing a caravan loaded with portable electrical measuring equipment, taking readings in the field, and undertaking a mathematical analysis afterwards." Campbell-Kelly adds that in "every way Wilkes was in his element".

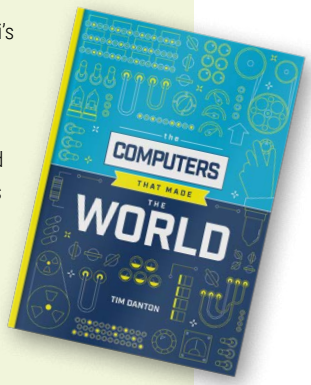
> *The work involved excursions in a motor car, towing a caravan loaded with portable electrical measuring equipment, taking readings in the field, and undertaking a mathematical analysis afterwards*

1 Martin Campbell-Kelly, 'Sir Maurice Vincent Wilkes: 26 June 1913 – 29 November 2010', in Biographical Memoirs of Fellows of the Royal Society, Vol 60 (2014), **rpimag.co/mwmemoir**, pp433-454

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



The mathematical work involved solving differential equations, so when Wilkes discovered that John Lennard-Jones – a professor of theoretical chemistry at Cambridge – had commissioned a model of Vannevar Bush's differential analyser, he asked permission to use it. Despite using Meccano parts for many of the components, this was no toy but a working tool; Wilkes rapidly became its master, publishing several papers using its results. The Meccano-based analyser played another role, too.

"This was a model intended for propaganda purposes so that [Lennard-Jones] could get the money to have a full-scale differential analyser built according to Bush's plans," Wilkes told an audience in 1979,[2] "but instead of just saying he wanted a differential analyser and a room to put it in, he said he wanted a computing laboratory." The university agreed, although they insisted on calling it the Mathematical Laboratory.

Lennard-Jones was appointed part-time director of the Laboratory in October 1937 with Wilkes its sole member of full-time staff, only for World War II to intervene shortly before the full-size differential analyser was installed. For the next six years, the Laboratory would be seconded to the war effort[3] while Wilkes devoted his time to radar research and installations.

For example, he worked out that by measuring the amplitude of radar echoes, you could measure the size of the vessel being tracked. He passed this information to operators of coastal radar systems who were then able to work out which types of vessel were being tracked. Before this, the equipment had even confused seagulls for boats.

2 'The Birth and Growth of the Digital Computer, lecture by Professor Maurice Wilkes, 1979, Computer History Museum, **rpimag.co/mwlecture**

3 According to a PhD dissertation submitted by Mary Goretti Croarken in 1985, it was used by the Ministry of Supply. 'The Centralization of Scientific Computation in Britain 1925-1955', p83

◀ Maurice Wilkes and EDSAC **Image:** copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission

Wilkes returned to Cambridge in the summer of 1945 to discover that Lennard-Jones wished him to take on the running of the nascent Mathematical Laboratory. Wilkes readily agreed, and soon outlined what he saw as the top priorities for the lab: to teach science undergraduates practical computational techniques and to provide a computing service. At this point, such a service primarily meant access to desk calculators and the differential analyser.

In one of the many strokes of good fortune that littered the EDSAC's creation, Wilkes had struck up a friendship with Douglas Hartree before the war. Wilkes had visited him in Manchester to see the university's full differential analyser in action. "He would go to any amount of trouble to help people," wrote Wilkes in his memoir.[4] "He also took me along to see the people at Metropolitan-Vickers, who had been responsible for the building of his machine and who were expecting to be responsible for the Cambridge one."

Unlike Wilkes, Hartree was central to British computing efforts during the war, and had seen the ENIAC in America long before its existence was made public. It was Hartree who fuelled Wilkes's interest in digital computing on his return to Cambridge, with talk not only of the ENIAC but also the work going on at Harvard.

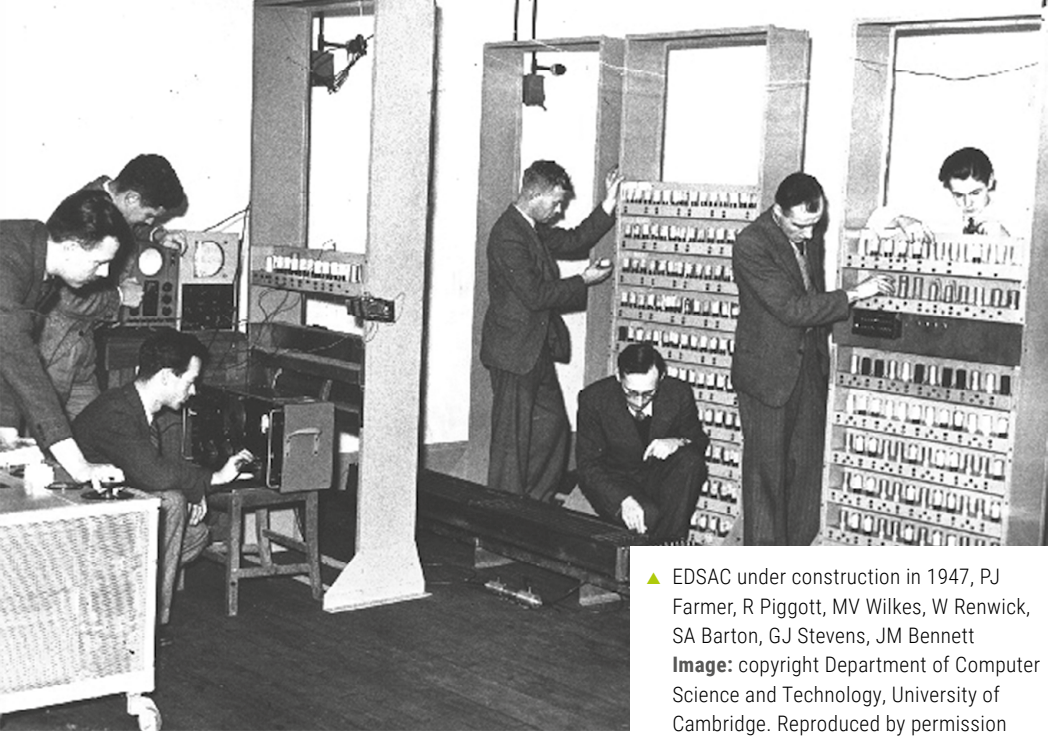
Then, in May 1946, Wilkes was visited by Leslie Comrie, who had – with remarkable foresight – created a commercial computing company in 1936. This was naturally based on mechanical methods, but Comrie immediately saw the importance of the ENIAC. And most importantly, he lent Wilkes his copy of the 'Draft report on the EDVAC' for one night. Wilkes stayed in the office late into the evening so he could read it before Comrie left the following morning.

Time for another piece of good fortune. "An event which shaped the whole of my subsequent career occurred in the summer of 1946," Wilkes later recalled,[5] "when I received a telegram from Dean Pender of the Moore School of Electrical Engineering in Philadelphia, inviting me to a course on digital computers." Wilkes immediately accepted, but could not travel to America quite as quickly as he wished due to the British government retaining control of all shipping.

"Those days, it was common for cargo ships to have accommodation for twelve passengers," said Wilkes at EDSAC 99, an event that took place at the University of Cambridge to mark

4 Maurice Wilkes, *Memoirs of a Computer Pioneer* (MIT Press, 1985, ISBN 978-0262231220), p29

5 'The Birth and Growth of the Digital Computer, lecture by Professor Maurice Wilkes, 1979, Computer History Museum, **rpimag.co/mwlecture**



▲ EDSAC under construction in 1947, PJ Farmer, R Piggott, MV Wilkes, W Renwick, SA Barton, GJ Stevens, JM Bennett **Image:** copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission

the 50th anniversary of the first program.[6] "I travelled on such a ship, but there were 36 passengers so we were a little crowded." These were all men, and the shipping company wanted to make sure no women were among them. "It is the only time in my life in which I had a medical examination directly to determine my sex."

The delays with transportation, plus problems with the ship's engines, meant Wilkes only arrived for the final two weeks of the eight-week course. Fortunately, the first six weeks covered areas he already knew well due to his extensive experience with differential analysers and knowledge of the EDVAC report. He devoured the lectures by John Mauchly and Presper Eckert, who were extremely generous in sharing their knowledge with the attendees. Mauchly even gave Wilkes a personal tour of the ENIAC and went through the construction details with him.

Through Hartree's contacts and invitations, Wilkes also had the chance to speak to other luminaries of the computing world. This included Howard Aiken at Harvard, who demonstrated the Harvard Mark I in operation and gave him a glimpse of the Mark II, then under construction. He also visited Samuel Caldwell, who had helped Vannevar Bush develop the original differential analyser at MIT, and picked the brains of Herman Goldstine about the EDVAC's design over a dinner in Philadelphia.

"I felt at the end [of the trip] that I knew all that was to be known about this new field. And I gradually found myself gripped by the idea that, come what may, we were going to build a computer in Cambridge," Wilkes later wrote. In his final week in Philadelphia, and during the course of the return journey on the Queen Mary, he sketched out his early plans for what would become the EDSAC.

Wilkes knew that creating a means for storing data, the memory, would be the biggest challenge. Based on Eckert's suggestion, he didn't hesitate to choose mercury delay lines. These had become a vital component in World War II, where they were used to store sound waves from radar. In the EDSAC, each 'blip' fed into the line represented a one in binary, so each delay-line could store a number – a series of zeroes (no sound) and ones – that could be read at the other end. "They were cleaned up, amplified, resynchronised, and put back in at the beginning," said Wilkes at the EDSAC 99 event. "So these pulses went round and round and round and in that way the pattern of the pulses was preserved and stored."[7]

At this point, Wilkes had another stroke of luck, meeting Tommy Gold, who provided a design for a mercury delay line that he had helped to develop during the war.[8] Wilkes first built a single tank to Gold's specifications, although he skimped on the diameter so that less mercury was required. After all, he had to build 32 tanks for the final product.

Each tank, or delay line, was about 1.5 metres long, and could store 576 binary digits (bits).[9] The 576 bits in each tank was divided into 16 36-bit words. Each 36-bit word had one bit used to separate the words, and one bit that indicated the sign of the number (negative or positive). Across 32 tanks, that meant EDSAC could store 512 numbers, each with ten decimal digits.

The great thing for Wilkes is that he was in charge. While the EDVAC team had two masters to answer to, in the Moore School and the US Army, Wilkes had a clear remit, endless internal resources to draw upon, and no chain of command. "I didn't have to ask anybody 'could I build a computer, please?'. I didn't have to put it in any proposal. I didn't have to arrange any budget. I was in charge and I could go ahead," he said.[10]

6 Maurice Wilkes, 6 May 1999, EDSAC 99 conference, EDSAC part 1, **rpimag.co/edsac99part1**

7 Wilkes provided a rather more technical description in a 1948 article called 'A Discussion on Computing Machines', published by the Royal Society of London. "The delay unit itself consists of a tube filled with mercury. The ends are closed by means of two similar X-cut quartz crystals. Electric pulses applied to one of the crystals gives rise to ultrasonic pulses which travel through the mercury with the velocity of sound. When they reach the far end of the tube, they are reconverted by the second crystal into electric pulses."

8 Maurice Wilkes, 'The thinking behind EDSAC', in Resurrection, Autumn 1990, p7

9 An Ultrasonic Memory Unit for the EDSAC, Maurice Wilkes, Computer Conservation Society, **rpimag.co/edsacmemory**

10 'The Birth and Growth of the Digital Computer', lecture by Professor Maurice Wilkes, 1979, Computer History Museum, **youtu.be/MZGZfsr1KfY**

▶ Wilkes with mercury delay lines
Image: copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission

Wilkes went into more detail in an article for the Computer Conservation Society in 1990.[11] "It was not a project to build a computer only. It was a project to build a computer, to learn how to use it, and then to solve some problems." And he understood what sort of problems needed to be solved thanks to his regular dealings with students.

Nor was he concerned about creating the perfect computer. "[We] just barged ahead on the EDSAC and the rule was that if you had got something that would work, you didn't spend another hour on making it simpler or cheaper, you went ahead with it."

He also made one key pragmatic decision early on: to aim for a pulse rate of 500Hz (500 pulses per second) for the vacuum tubes rather than the 1000Hz that the EDVAC and BINAC used. Wilkes knew that this would mean his computer would be slower than rivals, but as the EDSAC would be so much quicker than what it replaced, he felt it was the best decision. It is certainly one of the reasons for the EDSAC appearing first.

Wilkes's practical side also came to the fore. Right from the start, he knew he would have to build parts himself and so the Mathematical Laboratory included its own instrument workshop. The EDSAC could not have been completed without the skills of instrument maker Gordon Stevens, for instance.

"We had to make things like tape readers," said Stevens at an event in 1999 marking the EDSAC's 50th anniversary,[12] "and modify various telegraph equipment so that you could get information into the machine. In EDSAC, all of the stuff that went into the machine was on telegraph tape."

While Wilkes was keen to dismiss the notion that EDSAC was built from "war surplus junk", he took every opportunity he could to use army surplus equipment, including the crucial valves. Much had to be built to specification, too, such as the two giant 'batteries' that would store the mercury delay lines and keep them at the required temperature. That job fell to the university's central workshop, part of its Engineering Department, while Wilkes outsourced construction of the chassis to a local firm.

By January 1947, the trio of Wilkes, Gold, and the lab's principal assistant, Philip Farmer, had managed to get the first mercury delay line working. From this point on, the team slowly grew and with it, despite continuing hardware shortages, momentum. Electronics engineer Bill Renwick arrived in March 1947, Richard Kimpton joined straight from school that summer, before Vic Clayden and Herbert Norris rounded out the team.

A further boost came in July 1947 when Raymond Thompson and Oliver Standingford from the famous Lyons coffee house visited. This ultimately led to Lyons offering Wilkes a £3000 cash injection and the services of an assistant, with the idea that this extra pair of hands would learn on the job and aid Lyons with the building of their own copy of the EDSAC. Wilkes readily accepted.

Although Wilkes is rightly lauded as the creator of the EDSAC, he also drew upon the skills and knowledge of two University of Cambridge students. First, he delegated some key design work to research student John Bennett. "I was responsible for designing, constructing and testing the main control unit," Bennett wrote in 1999.[13]

"This unit sequenced the machine through the cycle of extracting from the store and decoding instructions (orders, we called them), extracting operands, initiating individual arithmetical and logical processes, and proceeding to the next instruction. I also designed, constructed, and tested the bootstrap facility."

Bennett was joined by fellow research student David Wheeler in September 1948. And he made an immediate impact. Wheeler was responsible for the programming system, and proved instrumental not only in the success of EDSAC but also in the creation of programming concepts that would become hugely influential. He was the one who created the 'initial orders' for

11 Maurice Wilkes, 'The thinking behind EDSAC', in Resurrection, Autumn 1990, p4

12 'Behind the green door', University of Cambridge Collections, EDSAC 99, **rpimag.co/edsacgreendoor**

13 EDSAC 1 and after – a compilation of personal reminiscences, Dr David Hartley, University of Cambridge Computer Laboratory, April 1999, **rpimag.co/edsac1andafter**

> *Its whole reason for being was to serve as a useful tool for the university, and Wilkes was keen for it to be as user-friendly as possible*

EDSAC, which loaded the instructions from the tape. He also formalised the idea of a closed subroutine, although it should be emphasised that John Mauchly deserves credit for the idea of subroutines (even if he did not call them as such).

Wilkes also relied on Bill Renwick who had joined in 1947 as a research assistant but became the de facto chief engineer according to Gordon Stevens, the EDSAC's principal instrument maker. He would die, far too young, in 1971, and was evidently much missed at the 50th anniversary in 1999. "He was a fine engineer and a fine man," said Stevens.[14]

Indeed, much of the credit for the EDSAC actually working should go to Bill Renwick. "I was what would nowadays be called the chief architect," said Wilkes.[15] "I also designed many of the early chassis and put them through their preliminary debugging. I then handed them over to Renwick who, as chief engineer, had the task of putting all these chassis together and making them work." Over time, Renwick would take over more of the design side too.

By the autumn of 1948, all the components were created: now, like Frankenstein's monster, everything had to be combined into one. Again, this job fell to Renwick. In February, they could read instructions from the input tape into the memory. Next step: attaching a teleprinter so that output could be read. Next, came the slow and agonising process of problem-solving, a portent of all the debugging of computers to come.

Finally, on 6 May 1949, came that magical click-click-click that began this article, followed by the chirruping of the teleprinter with the table of results. The EDSAC worked. The team had created the world's first stored-program, general-purpose electronic computer. It says much about David Wheeler that his first response was to write a program to find prime numbers.

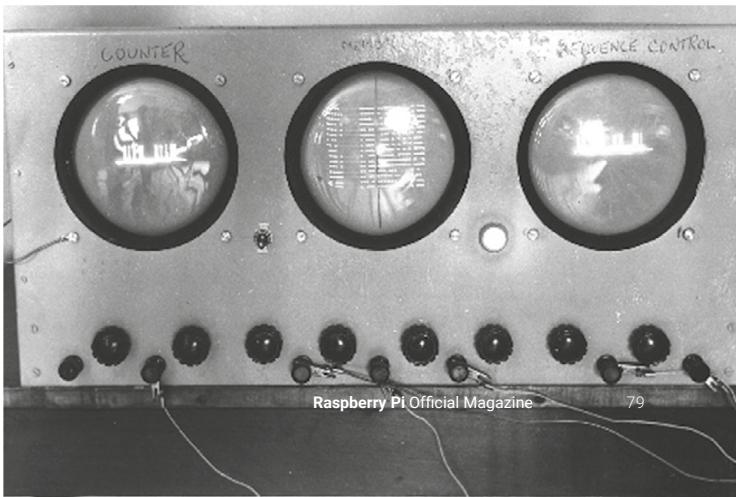
While there are good reasons to celebrate the 6th of May, it was by no means the end of the EDSAC's development. Its whole reason for being was to serve as a useful tool for the university, and Wilkes was keen for it to be as user-friendly as possible. For example, creating a subroutine to convert decimal numbers into binary so that students would not need to reinvent the wheel each time.

After a few weeks, Wilkes felt the EDSAC was ready for its first public demonstration: at a conference held in Cambridge on the apt topic of high-speed automatic calculating machines. Along with many from academia, attendees included representatives from Ferranti, Elliot and Lyons, a handful of UK army officers, and several interested parties from abroad – including France, Germany, Sweden, and the Netherlands.

The attendees were suitably impressed, but you will struggle to find much mention of it in the papers of the time. That's despite Wilkes and Renwick giving a personal demonstration of the EDSAC to a reporter from The Daily Telegraph, one of the UK's most prominent newspapers, which only found room for the computer's arrival on page 15 of its edition on Friday 17 June 1949.

"NEW 'BRAIN' STORES ORDERS" was the story's hardly compelling headline. "It has a 3500-valve 'brain' weighing about a ton," the unnamed reporter wrote, before alluding to some concerns (even more relevant today) that these electronic brains could take over the world. "Statements that have given almost human qualities to the machine, and alarmed some people, are misleading. Its marvel is in being able to read, subtract, multiply, and divide automatically at speed, completing in weeks what would take humans years."

▼ 9-inch cathode ray tubes (CRTs) used as monitors to display the contents of the computer's memory and other registers
Image: copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission



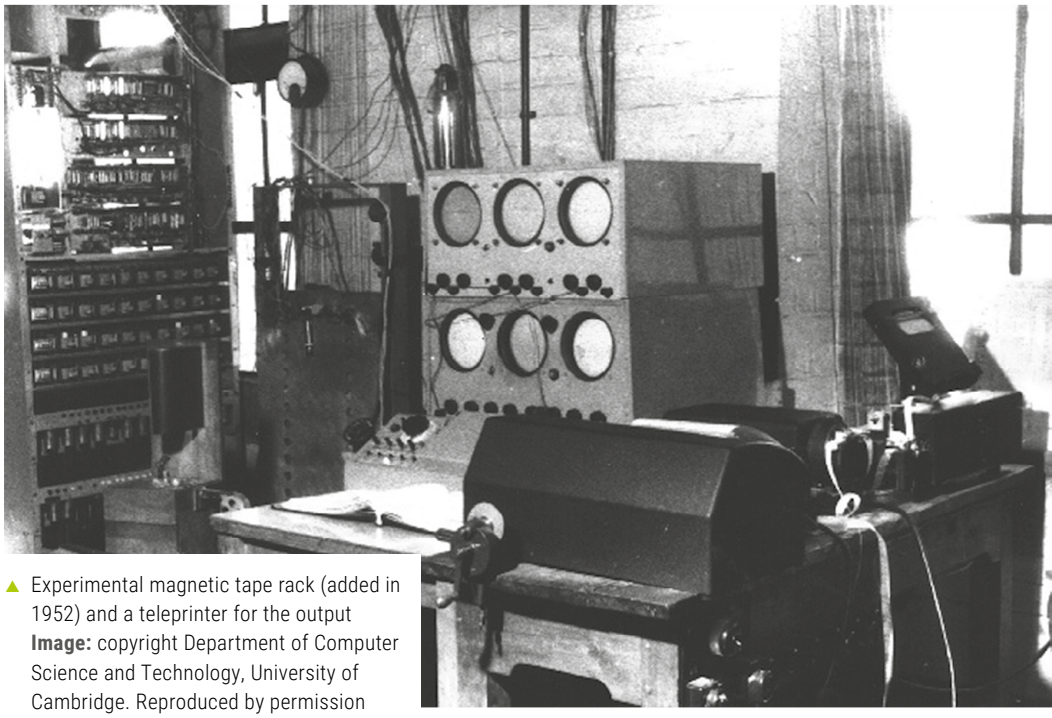
14 Behind the green door', University of Cambridge Collections, EDSAC 99, **rpimag.co/edsacgreendoor**

15 Maurice Wilkes, 6 May 1999, EDSAC 99 conference, EDSAC part 1, **rpimag.co/edsac99part1**

Wilkes directly addressed this in his interview with The Daily Telegraph. "I liken it to a calculating machine operated by a moron who cannot think but can be trusted to do what he is told," he said. "If you tell the machine to do something silly, it goes on doing it."

He went on to explain the types of calculations he anticipated the EDSAC would solve, including nuclear physics, aerodynamics, and astrophysics – such as working out why stars are a certain size and brightness.

Aside from the PR exercise, Wilkes had more immediate, practical concerns. "By June 1949, people had begun to realise that it was not so easy to get a program right as had at one time appeared," he wrote in his 1985 book, *Memoirs of a Computer Pioneer*.[16] "It was on one of my journeys between the EDSAC room and the punching equipment [one floor below] that, hesitating at the angle of the stairs, the realisation came over me with full force that a good part of the remainder of my life was going to be spent in finding errors in my own programs."



▲ Experimental magnetic tape rack (added in 1952) and a teleprinter for the output **Image:** copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission

Always keen to share their knowledge, Wilkes, Wheeler, and Stan Gill would compile their notes on programming the EDSAC – including a chapter on debugging – that was shared informally as a typewritten manuscript but would become the world's first programming book when published in 1951. Officially called 'The Preparation of Programs for an Electronic Digital Computer,' it was affectionately shortened to WWG to reflect the three authors' surnames.

By this time, the EDSAC was operating 24 hours a day, seven days a week, but like all the early computers it was far from reliable. A team of engineers was on hand to fix it during the day but "when they left in the evening, we programmers would continue running our programs until the next breakdown happened, when we would switch off and go home," reported D H Shinn, then a research student.[17]

That normally worked well, giving them a chance to get some rest. "However, there was an occasion, probably in March 1950, when EDSAC refused to break down," said Shinn. "It continued working throughout the night. When the engineers arrived in the morning... I was the only programmer left; the others had gone home to have their breakfasts; I soon went home to have my breakfast, and a good sleep."

The Cambridge team continued to improve the reliability and feature set of the EDSAC over its life, it remained a temperamental beast, delivering electric shocks to one user in the mid-1950s. "The EDSAC 'console' was an old wooden table; when using the machine, one could sit with one's knees under it," remembered Jenifer Haselgrove (née Wheildon Brown, later Leech), a research student between 1953 and 1956.[18]

"One hot summer night I was working late, wearing shorts. Sleepily I reached out to the left to put a data tape in the tape reader, and was woken sharply by an electric shock. A little investigation revealed an unprotected rheostat, with mains voltage on it, under the table. My bare knee had been pressing against it."

So EDSAC wasn't perfect. Its 500-word memory also limited the problems it could tackle. However, as David Wheeler eloquently put it in a 1987 interview, "a machine selects the problems that they can do".[19] In EDSAC's case, he said, it selected "differential

16 Maurice Wilkes, *Memoirs of a Computer Pioneer*, p145

17 EDSAC 1 and after – a compilation of personal reminiscences', Dr David Hartley, University of Cambridge Computer Laboratory, April 1999, **rpimag.co/edsac1andafter**

18 As before

19 An interview with David J Wheeler by William Aspray, Charles Babbage Institute, 14 May 2007, **rpimag.co/wheelerinterview**

equations, Fourier transforms, and other things which were not quite so space-intensive."

This also ties in with a different computing philosophy at Cambridge compared to its American counterparts. Wilkes always intended EDSAC to be a tool for use by students at the university, and didn't want it to be tied up for weeks at a time on a single problem. He also wanted to create a generation of computer-literate graduates, and this legacy can be seen echoing through the years.

Cambridge, unlike its great rival the University of Oxford, provided defining moments of computing in the following decades. Many electronics and early computer companies – including Sinclair Research and Acorn, creators of the ZX Spectrum and BBC Micro respectively – based themselves in Cambridge, at least in part to benefit from the students who took Computer Science courses.

> *Perhaps that is how the EDSAC should be remembered: as a scrappy computer. It wasn't the most beautiful, it wasn't the most reliable, but it worked*

Some individual graduates are particularly noteworthy. Steve Furber and Sophie Wilson co-created the ARM architecture that almost every smartphone processor in the world relies on. Then there's Eben Upton, the CEO and co-creator of the Raspberry Pi, the best-selling computer in history[20]. Would this innovation have happened without Wilkes's foresight and drive? It somehow seems unlikely.

The EDSAC also had a direct descendent: what came to be known as EDSAC 2, with the original often referred to as EDSAC 1. Wilkes began thinking about it in earnest once the first machine's teething problems had been largely solved, and he was keenly aware that its successor needed to be both faster and more reliable.

Wilkes secured £25,000 of funding from the Nuffield Foundation in June 1951, by which time he had some clear ideas in his mind. One of which was a switch from serial operations to a parallel system. At that point, he kept his options open in terms of memory. Mercury delay lines remained the proven technology, but it was slow compared to the rest of the computer, so would be a bottleneck.

He hoped that a superior alternative would come to light during the development process, and this proved absolutely correct. After seeing the MIT Whirlwind project on a visit to America in August 1952, he realised that EDSAC 2 must use the same ferrite core memory. The only problem was that he needed to raise a further £10,000 to pay for it, but fortunately the Nuffield Foundation understood his arguments and put up the money. Wilkes then commissioned Mullard's factory, in Blackburn, Lancashire, which created ferrites, to build the cores to his specification.

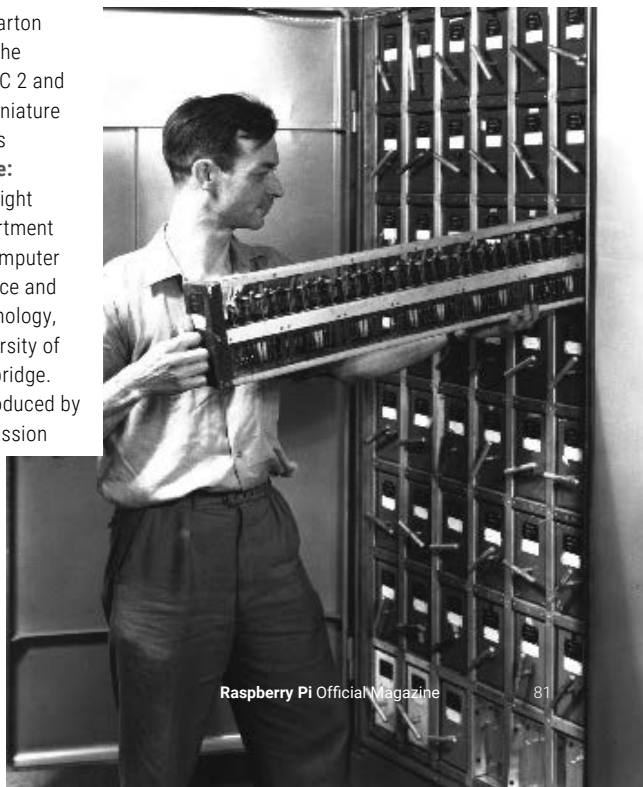
David Wheeler, who had left in 1951 to lecture on programming methods at the University of Illinois, returned to Cambridge in September 1953 and proved an immediate help. He even created a 'control matrix', using the cores and his exceptional programming skills, to make using the EDSAC 1 easier. With the control matrix attached, the upgraded computer became known as EDSAC 1.5.

By early 1958, EDSAC 2 was ready to take over computing duties within Cambridge. Keen to reclaim the space, and in a decision Wilkes later regretted, the team disassembled EDSAC 1 that summer and sold most of it for scrap.[21]

And perhaps that is how the EDSAC should be remembered: as a scrappy computer. It wasn't the most beautiful, it wasn't the most reliable, but it worked. And its influence is still felt at Cambridge University to this very day.

20 Les Pounder, 'Raspberry Pi celebrates 12 years as sales break 61 million units', Tom's Hardware, 29 February 2024, **rpimag.co/61million**

21 Fortunately several parts remain intact, and are safely stored at the Science Museum in London. A partial replica is also on show at the UK's National Museum of Computing, found on the Bletchley Park estate, Buckinghamshire.



▶ Sid Barton with the EDSAC 2 and its miniature valves **Image:** copyright Department of Computer Science and Technology, University of Cambridge. Reproduced by permission

Part 09

Conquer the command line: remote Raspberry Pi

Learn how to access your Raspberry Pi from remote PCs and devices with Secure Shell (SSH)

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

about.me/RichardSmedley

It's great that Raspberry Pi is so portable, but sometimes you may want to use it without taking it with you. Here, Raspberry Pi OS is a real strength, as Unix-like operating systems have been used this way for over 50 years.

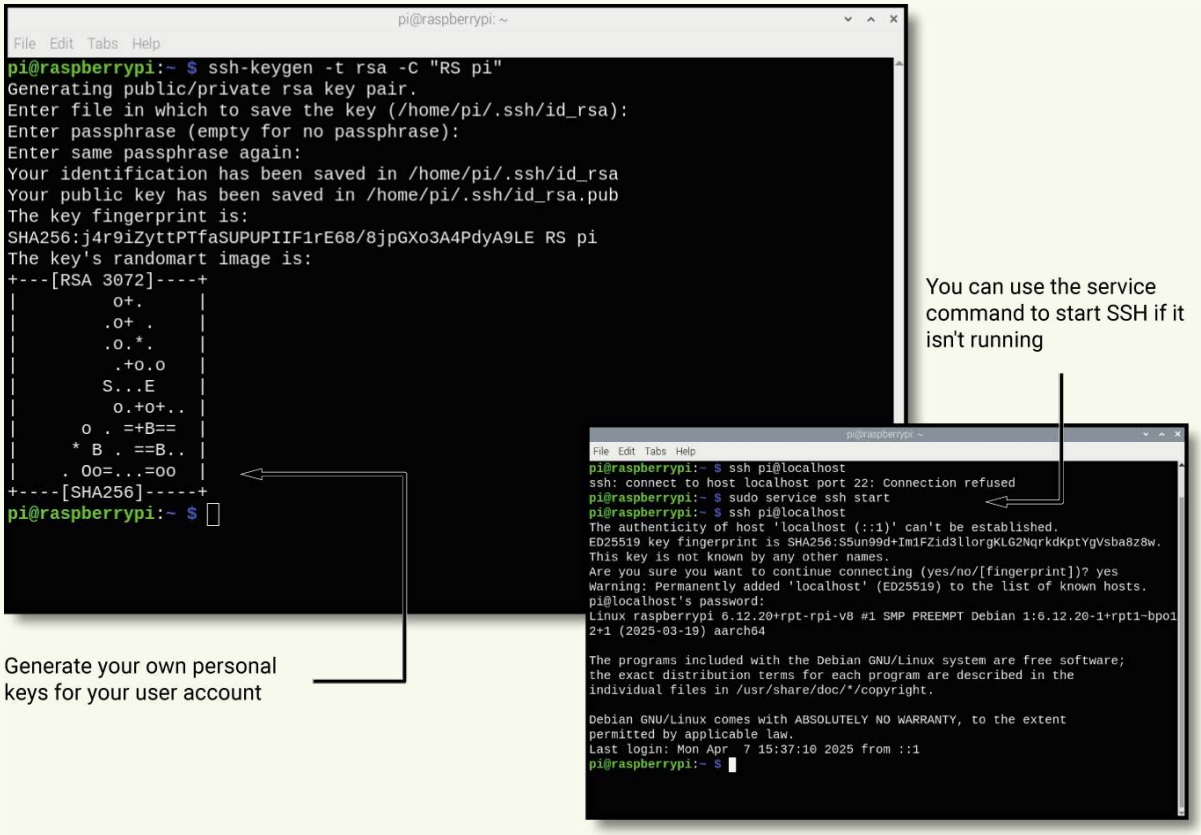
Over time, as the internet has given the opportunity for malicious users to connect to computers, old standards like Telnet and rlogin have been replaced by Secure Shell (SSH), based on public-key cryptography. Once set up, secure connections are simple, and open to scripted, automatic connection for your projects. As a reminder, please set a secure password on your Raspberry Pi before accessing SSH.

Try connecting to your Raspberry Pi from another computer on your network

If the SSH server is not enabled by default on your version of Raspberry Pi OS, run `sudo raspi-config` and go to the Interface Options settings to enable SSH. Now you can try connecting to your Raspberry Pi from another computer on your network.

Connecting with SSH

From a Mac or GNU/Linux computer, use `ssh` from a terminal to connect to Raspberry Pi. You can use the command-line OpenSSH client on Windows 10 and 11 PCs; for earlier PCs, install an SSH client like PuTTY (**putty.org**), which also works with SCP, Telnet, and rlogin, and can connect to a serial port. Android users have a variety of terminal emulators on the Play Store they can use too.



You can connect to your Raspberry Pi with the command `ssh [username]@[computer name]` – for example, `ssh pi@raspberrypi` – and enter your password (replace `pi` with your username and `raspberrypi` with your hostname). The first time you connect, you'll be prompted to confirm the authenticity of your Raspberry Pi. In some cases, you may need to replace the hostname with your Raspberry Pi's IP address. You can find the IP address assigned to Raspberry Pi with `ifconfig` (note the 'inet addr' for the eth0 or wlan0 interface) or `hostname -I`.

You should now be at the command-line interface of your Raspberry Pi. If you got any sort of error, check from Raspberry Pi that SSH is really up and running by entering `ssh localhost` on Raspberry Pi itself. If that works, SSH is up and running on Raspberry Pi, so take a closer look at network settings at both ends to diagnose the problem.

Hello, World

Now that you can access Raspberry Pi on the local network, it's time to share with the world. This is very easy to set up — and extremely secure — thanks to Raspberry Pi Connect, which allows you to not only connect via screen-share with any running Raspberry Pi, but also use a remote command-line interface just like SSH. Set up an account over at **rpimag.co/connect** and then on your Raspberry Pi desktop, click the Raspberry Pi Connect icon in the top right — it looks like two small squares connected via a circle.

From there, you can sign into your Raspberry Pi Connect account from your Raspberry Pi, adding it to the devices you can access from the web. When this Raspberry Pi is on, you'll find it listed on your Connect account with a 'Connect via' button for

Interrupted service

While you can restart most services with `sudo service ssh restart`, replacing `restart` with `reload` permits configuration changes to be registered with less disruption, which is key for some projects.

it. Click on that and select Remote Shell to access the command line from anywhere else in the world.

Bye bye FTP

File Transfer Protocol (FTP) was not designed for security: data, and even passwords, are transmitted unencrypted. The Secure Copy Program (SCP), which runs over SSH, is best for transferring files. The syntax of the command mimics the command-line `cp` program:

```
$ scp pi@raspberrypi:/home/pi/testfile1 .
```

Here we're transferring a file from Raspberry Pi, across a local network, to the current location (the dot shortcut). Note that you can use wildcards for groups of similarly named files, and can recursively copy directories and their contents with the `-r` switch after `scp`.

A secure key

You can use a personal key to skip the password prompt when using SSH to connect to your Raspberry Pi from another computer. The first step is to create a private/public key pair.

On the remote computer (the one you will use to connect to your Raspberry Pi), open a terminal or command prompt and run `ssh-keygen -t rsa -C "comment"`, where `"comment"` is anything you want to identify the key with: name, email, or machine and project, for example. When asked where to save it, accept the default.

You'll be asked for a passphrase to secure the key — if you press **ENTER**, you'll get a key with no passphrase, which is not ideal. That means that anyone who possesses the private portion of the key pair can log in without a password. If you already have a private/public key pair that you use to log into other computers, you can use that instead of creating a new one.

If you accepted the defaults, your personal keypair will now be on your computer, in your home directory, under the **.ssh** subdirectory. You will need to copy the contents of the public key (**id_rsa.pub**) to a file called **~/.ssh/authorized_keys** on your Raspberry Pi.

If you're in a terminal in your home directory on the computer, you can copy the file over to your Raspberry Pi with this command on GNU/Linux or macOS:

```
$ ssh-copy-id pi@raspberrypi
```

Replace `pi` with the username you use on your Raspberry Pi, and `raspberrypi` with the hostname of your Raspberry Pi.

On a Windows PC, it's perhaps easiest if you open the **.ssh\id_rsa.pub** file in Notepad or another plain text editor and copy its contents. From a terminal program (PowerShell, Windows Terminal, or Command Prompt, use SSH with your password to log into your Raspberry Pi (`ssh pi@raspberrypi`), run `nano ~/.ssh/authorized_keys`, go to the end of the file and paste the key onto a new line with **CTRL+V**, then save the file.

Once that's done, the next time you use SSH to connect to your Raspberry Pi, you'll find that you're asked for your passphrase rather than your Raspberry Pi password. Most operating systems let you avoid entering the passphrase every time you connect. Most systems have some kind of SSH authentication agent. This includes macOS, recent versions of Windows, and most modern GNU/Linux distributions.

On Windows, you may need to configure the agent to start automatically. Right-click on the Start menu and choose Windows PowerShell (Administrator), Terminal (Admin), or Command Prompt (Admin), whichever option is available to you. Next, run these two commands and close the PowerShell, Terminal, or Command Prompt:

```
sc.exe config ssh-agent start=auto
sc.exe start ssh-agent
```

With the SSH agent running, the first time you connect to your Raspberry Pi using your key, you'll be prompted for that passphrase, but it won't prompt you on subsequent connections (on Windows, you will first need to run the command `ssh-add` to add the key to the agent). The agent stores credentials securely and unlocks them when you have logged into your computer. Some operating systems, such as Ubuntu, may prompt you to choose whether the key should be automatically unlocked each time you log in.

Lost keys?

The private key half of your key pair (for example, **.ssh/id_rsa**) should be kept secret – but safe, too. Keep a backup of the private key on a memory card in a secure location.

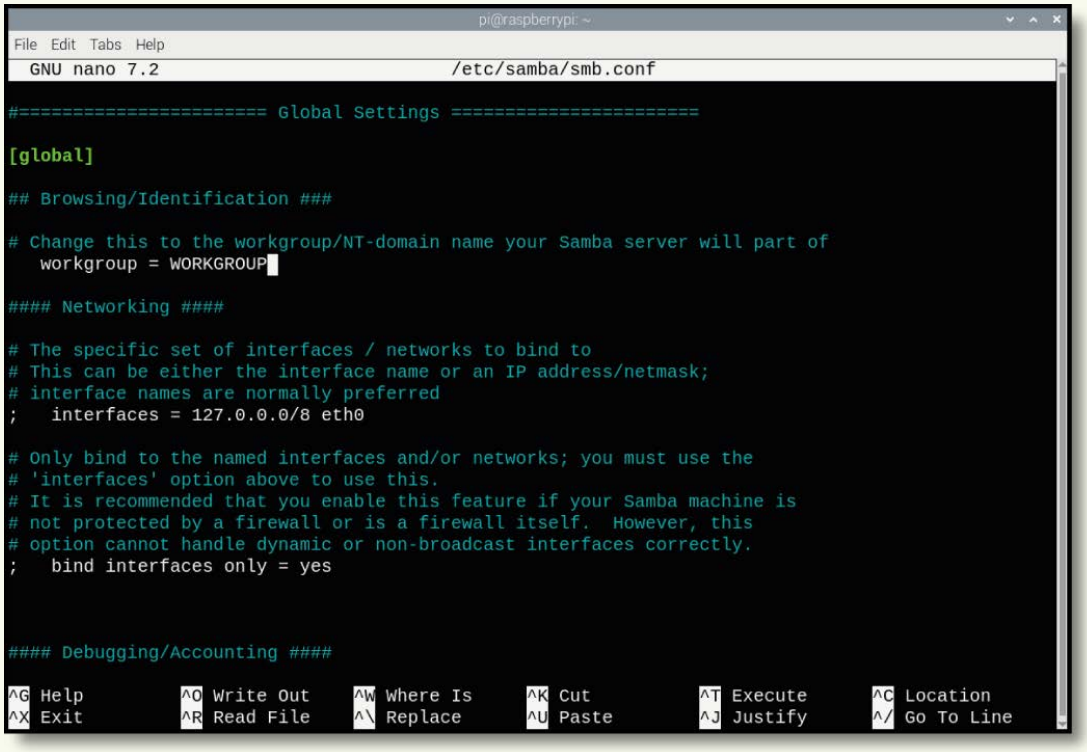


Figure 1: There's a lot of configuration in Samba, but simply adding your WORKGROUP name to the default settings should get you up and running

Samba steps

Samba is extremely well-documented, with separate man pages for everything from **smb.conf** to `smbpasswd`, and excellent online books at **samba.org** – look for **smb.conf** examples.

Once you confirm SSH works without passwords, you can edit the file **/etc/ssh/sshd_config** to include the line `PasswordAuthentication no`. This will prevent your Raspberry Pi from accepting password-based logins, and will only permit you to log in with a key.

Shared drive

You may be using a service like Dropbox to share files between computers. There is no need to do this on a local network, as the Samba networking protocol on Raspberry Pi lets Windows PCs access it as a shared drive. Samba is already installed in recent versions of Raspberry Pi OS, or you can install it using:

```
$ sudo apt install samba samba-common-bin
```

Edit **/etc/samba/smb.conf** with a WORKGROUP value (**Figure 1**). For Windows XP and earlier, try `workgroup = WORKGROUP` and/or `HOME` (For Windows 7 and above). Ensure that Samba knows `pi` is a network user:

```
$ sudo smbpasswd -a pi
```

Then restart with:

```
$ sudo service samba restart
```

Raspberry Pi should now show up in Windows Explorer under Network. You can fine-tune **smb.conf** to control what's shared (including printers), and set access permissions.

Part 05

Measure CPU usage with an RGB LED

Learn how to use an RGB LED and get it to show CPU load

Maker
Phil King
A long-time Raspberry Pi user and tinkerer, Phil is a freelance writer and editor with a focus on technology.

philkingeditor.com

YOU'LL NEED

- 1× solderless breadboard
- 1× RGB LED
- 3× 100Ω resistors
- 4× pin-to-socket jumper wires

We lit up a standard LED in part 2 (issue 159), using the `LED` class. GPIO Zero also offers an `RGBLED` class for controlling – guess what – an RGB LED! In this chapter, we'll make use of this to light up our LED in different shades by altering the red, green, and blue values. Then we'll code up a little program that tracks the Raspberry Pi's CPU usage percentage and adjusts the LED between green and red accordingly to show how much processing power it's using.

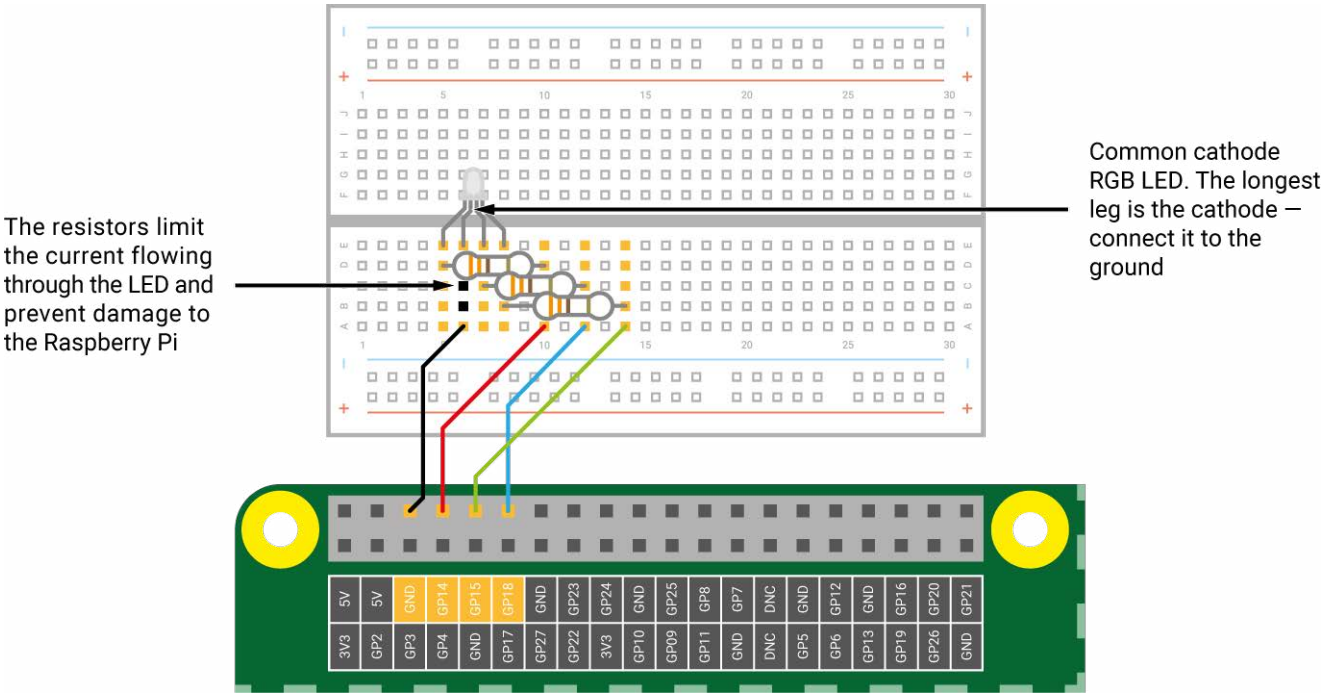
Select your RGB LED

Light-emitting diodes (LEDs) are cool. Literally. Unlike a normal incandescent bulb, which has a hot filament, LEDs produce light solely by the movement of electrons in a semiconductor material. An RGB LED has three single-colour LEDs combined in one package. By varying the brightness of each component, you can produce a range of colours, just like mixing paint. There are two main types of RGB LEDs: common anode and common cathode. We're going to use common cathode for this project.

Connect the RGB LED

As usual, it's best to turn the Raspberry Pi off while connecting our circuit on the breadboard. LEDs need to be connected the correct way round. For a common cathode RGB LED, you have a single ground wire – the longest leg – and three anodes, one for each colour. To drive these from a Raspberry Pi, connect each anode to a GPIO pin via a current-limiting resistor. When one or more of these pins is set to HIGH (3.3V), the LED will light up the corresponding colour. Connect everything as shown in **Figure 1**.

DOWNLOAD THE FULL CODE:
rpimag.co/gpiobookgit



▲ **Figure 1:** Wiring up the RGB LED

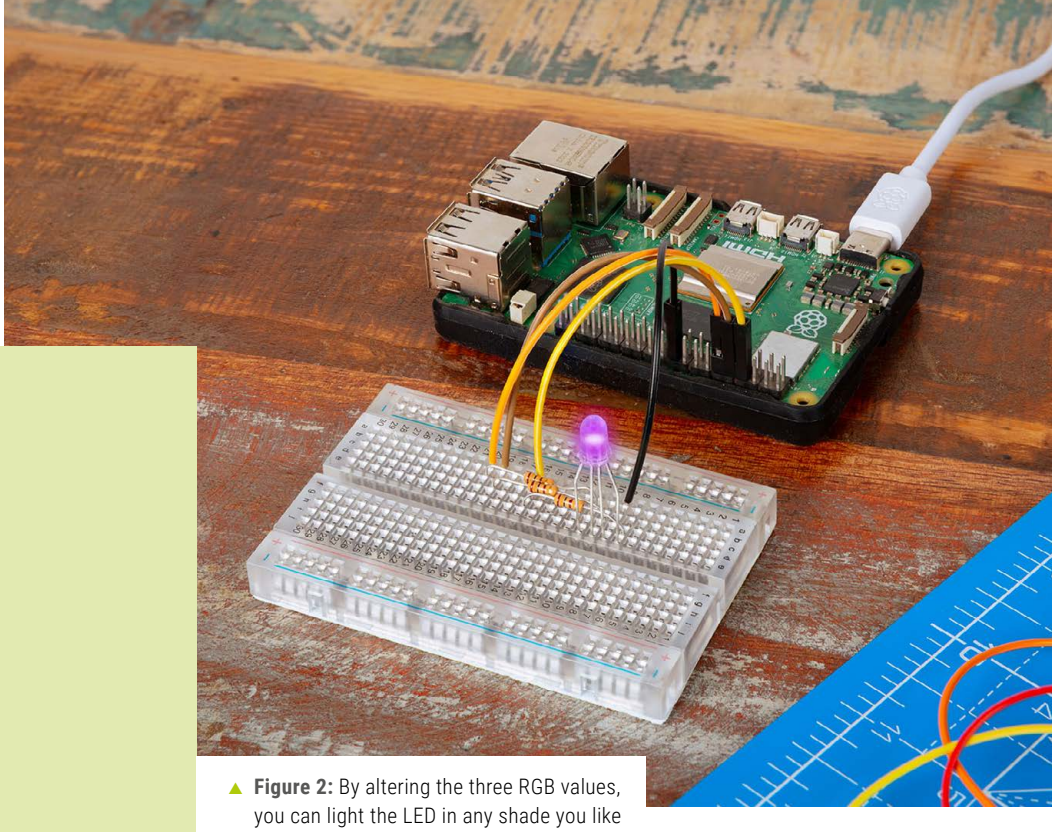
An RGB LED has three single-colour LEDs combined in one package

If your RGB LED is common anode, connect the LED's common pin to the 3V3 GPIO pin instead of GND, but leave the other pins wired as shown. You must, however, add `active_high=False` to the call to `RGBLED()` in the code.

Here, we wire the cathode (long leg) to a GND pin, while the other legs are wired via resistors to GPIO 14, 15, and 18 (your red, green, and blue legs may be different than shown here). The resistors limit the amount of current flowing, to avoid damaging your LED or Raspberry Pi; we've used 100Ω, but you could use a slightly higher ohmage, such as 330Ω.

Test the LED

With the `RGBLED` class in GPIO Zero, it's easy to alter the colour of the LED by assigning values of between 0 and 1 to red, green, and blue (see **Figure 2**). On the Raspberry Pi, create a new file, enter the following code (overleaf), and save it as **rgb_led.py**.



▲ **Figure 2:** By altering the three RGB values, you can light the LED in any shade you like

```
from gpiozero import RGBLED
from time import sleep

led = RGBLED(14, 15, 18)

led.red = 1 # full red
sleep(1)
led.red = 0.5 # half red
sleep(1)
led.color = (0, 1, 0) # full green
sleep(1)
led.color = (1, 0, 1) # magenta
sleep(1)
led.color = (1, 1, 0) # yellow
sleep(1)
led.color = (0, 1, 1) # cyan
sleep(1)
led.color = (1, 1, 1) # white
sleep(1)
led.color = (0, 0, 0) # off
sleep(1)

# slowly increase intensity of blue
for n in range(100):
    led.blue = n/100
    sleep(0.1)
```

At the top, we import the `RGBLED` class from GPIO Zero, along with the `sleep` function from the time library. We then set the variable `led` to the `RGBLED` class on GPIO pins 14, 15, and 18, for red, green, and blue. We then make `led.red` equal to 1 to turn the LED a full red colour. After a second, we then change the value to 0.5 to reduce its brightness.

We then go through a sequence of colours using `led.color`, assigning it a tuple of red, green, and blue values to mix the shades. So, `(1, 0, 1)` shows full red and blue to make magenta. You can vary each value between 0 and 1 to create an almost infinite range of shades. Finally, we use a `for` loop to slowly increase the intensity of blue. Run the program, and it will cycle through its sequence of colours, and then it will exit on its own.

Brightness is an illusion

Truth be told, we're not actually changing the LED's brightness at all. We're using a feature called *pulse-width modulation* or PWM. A digital output can only ever be on or off: 0 or 1. Turning a digital output on and off is known as a *pulse* and by altering how quickly the pin turns on and off, you can change, or *modulate*, the *width* of these pulses – hence 'pulse-width modulation'. By default, GPIO Zero sends 100 pulses per second (100Hz).

The PWM duty cycle controls the pin's output: a 0 percent duty cycle leaves the pin switched off for all 100 pulses per second, and effectively turns the pin off; a 100 percent duty cycle leaves the pin switched on for all 100 pulses per second, and is functionally equivalent to just turning the pin on as a fixed digital output; a 50 percent duty cycle has the pin on for half the pulses and off for half the pulses. Although 100Hz is not fast from a computer's perspective, it is fast enough to trick the human eye and create an illusion that the brightness is changing.

If you're using a common anode RGB LED, you must set `active_high` to `False`, which inverts the logic used to illuminate the LED, where the pin is driven HIGH (a 3.3V signal) to turn the LED off and LOW (0V) to turn it on, which is why we asked you to connect the common anode pin to the 3V3 GPIO pin earlier. You'll need to change:

```
led = RGBLED(14, 15, 18)
```

to:

```
led = RGBLED(14, 15, 18, active_high=False)
```

Add a new library

We now want to get our RGB LED to change colour between green and red, to show the CPU usage of the Raspberry Pi to which it's connected, so we can track how much of its processing power we're using at any time. For this, we'll need the psutil Python library. If it's not already installed, you can install it with this command:

```
$ sudo apt install python3-psutil
```

This will let us look up the CPU usage of the Raspberry Pi as a percentage number, which can then be used in our code to vary the LED's colour.

Python virtual environments

If you want to install a newer version of psutil (or any other Python library), you must set up a virtual environment. This lets you install optional libraries without affecting your main Python environment. If you're happy with the version that's available through `apt`, you don't need to follow these instructions. To set up a virtual environment, use the following command to create a virtual environment in the **env** folder in your home directory:

```
$ python -m venv --system-site-packages ~/env
```

Next, you can run the following command from any directory to start using the virtual environment (you'll need to run this for each new Terminal window, SSH session, or console login):

```
$ source ~/env/bin/activate
```

You should then see a prompt like the following:

```
(env) username@hostname:~ $
```

To leave the virtual environment, run the following command from any directory:

```
$ deactivate
```

You can find instructions for other configurations, such as per-project environments, at **rpimag.co/venv**. If you don't feel like running `source ~/env/bin/activate` every time you want to use the environment, you could add that line to your **~/.profile**. After you've activated your virtual environment, you can install psutil or any other Python module. Make sure you see the `(env)` prompt (if not, run the `source` command shown earlier to load the virtual environment). Run the following command:

```
$ pip install psutil --upgrade
```

You should periodically update the pip command itself with `pip install pip --upgrade`.

Measure CPU usage

Create a new file, enter the following code, and save it as **cpu_usage.py**. As with the previous example, if you're using a common anode RGB LED, you must set `active_high` to `False`.

```
from gpiozero import RGBLED
import psutil, time

led = RGBLED(14, 15, 18)

while True:
    cpu = psutil.cpu_percent()
    r = cpu / 100.0
    g = (100 - cpu)/100.0
    b = 0
    led.color = (r, g, b)
    time.sleep(0.1)
```

At the top, we import the modules we need, including `psutil`. We then set the `led` variable to the `RGBLED` class on GPIO 14, 15, and 18, for red, green, and blue. In a never-ending `while True:` loop, we set the `cpu` variable to the percentage of CPU usage via `psutil`, then assign the red and green LED values accordingly, and light the LED.

Try running the code. The LED should light up: its colour will indicate how hard your Raspberry Pi's CPU is working. Green means less busy, turning redder as the CPU becomes more heavily loaded. Start up some other applications to test it. If you have an original Model B, you'll probably find that just running Chromium is enough to turn the LED red. If you have a Raspberry Pi 5, you may need to start lots of things running to have any impact! You can exit the program with **CTRL+C**.

Customise your project

The example code only uses the red and green components of the LED: the blue value is always set to zero. You could swap things around and create a different colour gradient (e.g. blue to red) or put together a fancy function that maps a percentage value onto all three colours. Have fun with the colours and maybe even have it look at other resources to monitor – GPIO Zero includes several internal pseudo devices you can use in your code, including ones that represent times of day, CPU temperature, and disk usage. See **rpimag.co/gpiozinternal** for details.


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

BOOK OF MAKING 2026

STEP INTO THE WORLD OF MAKING!

- DISCOVER YOUR NEW FAVOURITE HOBBY
- LEARN THE DIGITAL SKILLS THAT MAKE THE WORLD GO ROUND
- TRY YOUR HAND AT 3D PRINTING, ELECTRONICS, PROGRAMMING, AND MORE
- DISCOVER PROJECTS THAT YOU CAN DO IN AN HOUR, AFTERNOON, OR WEEKEND

rpimag.co/BookOfMaking2026

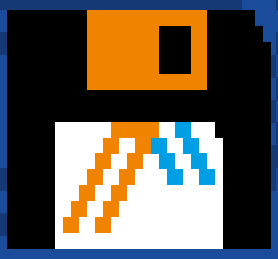
Part 01

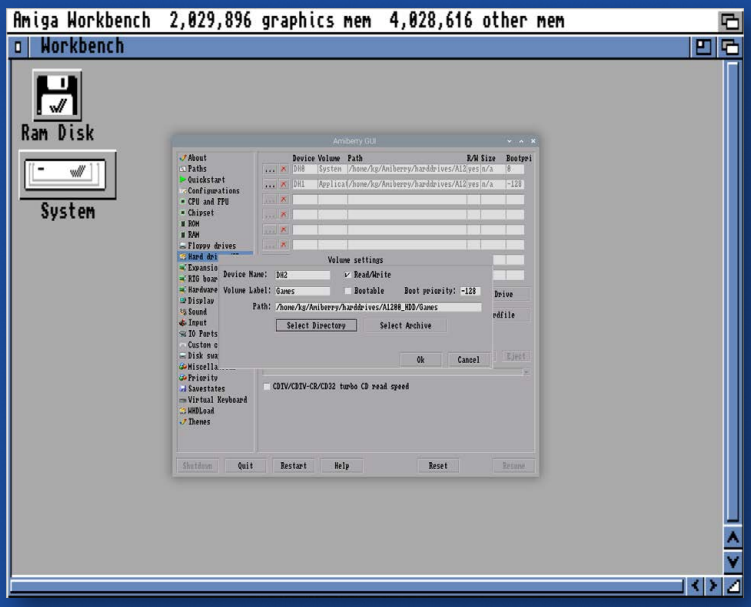
EMULATE A RETRO AMIGA DESKTOP

With legal firmware and OS images and plenty of applications in active development, the Amiga is an ideal retro desktop ideal to emulate for work as well as play

Maker
K.G. Orphanides
K.G. wasn't an Amiga kid, but always had a soft spot for mod trackers and night-long games of Scorched Tanks on a friend's A4000T.

rpimag.co/owlbear





You can add entire directories as virtual Amiga hard disks, making it easy to copy files in and out of your emulated OS

We'll be using Blitter Studios' Amiberry, a fork of the WinUAE emulator

Warning!
Copyrighted material
This feature discusses the use of legally distributed software and system ROM images for a variety of vintage Amiga systems. However, please note that many other titles are distributed online without their rights holders' permission, and are not legal to use under UK law.
rpimag.co/legalroms

This month we're going to emulate the Commodore Amiga. This classic 16/32-bit home computer of the 1980s and '90s is still widely and enthusiastically supported, through original hardware, new hardware and FGPA clones, and emulation. We'll be using Blitter Studios' Amiberry (**amiberry.com**), a fork of the WinUAE emulator, optimised for Raspberry Pi's aarch64 architecture, although it'll also run on x86 and amd64, and the Mac version supports Apple Silicon aarch64.

Install Amiberry

To make it easy to keep updated to the latest version, we're going to install a flatpak of Amiberry. If you'd prefer a DEB file, you can obtain these via Blitter Studios' GitHub (**rpimag.co/amiberrygit**). We'll start by opening a terminal and installing flatpak:

```
$ sudo apt install flatpak
$ sudo apt install plasma-discover-backend-flatpak
$ flatpak remote-add --if-not-exists flathub https://dl.flathub.org/repo/flathub.flatpakrepo
$ sudo reboot
```

Rebooting ensures that paths to all the UDG data directories that flatpak required to smoothly install and run software from the command line are correctly set. Once Raspberry Pi restarts, open a terminal window and type:

```
$ flatpak install amiberry
$ flatpak run com.blitterstudio.amiberry
```

You should also be able to find Amiberry's shortcut in the Games folder in Raspberry Pi OS's main menu, but we'll be using it for more than just games.

Obtain Amiga operating system images

Amiberry by default launches the open-source AROS (**rpimag.co/aros**) firmware ROM image and waits for media to be mounted, in the form of either a floppy or hard disk image. We want to use Amiga Workbench which, along with the Amiga's system firmware ROMs, is managed by Italian software firm Cloanto, and available to buy as 'Amiga Forever'. Unlike C64 Forever, this is a commercial product, so you're going to have to pay for it. Amiga Forever 11 – available from **amigaforever.com** – costs $20 for a Value edition, which only includes version 1.3 of the Amiga Kickstart ROM and Workbench environments (required by some older Amiga software) plus 25 games and 25 demoscene productions.

If you can afford it, we recommend spending $39.95 on the Plus edition, which includes the AmigaOS 3.1 ROM and system files as well as version 1.3 and a large set of other historic ROM images and OS installation disks for various Amiga computers, 50 games, and 100 demoscene works. We used Amiga Forever 11 when writing this tutorial, so we could run AmigaOS 3.1. Amiga Forever is distributed as an ISO file, which you can mount and browse to extract the files and software that you wish to use.

Particularly useful for our purposes are various files in the Amiga Forever disc's **Amiga Files/Shared/** folder, including firmware ROM images, ADF floppy and HDF hard drive images, and a pair of fully configured directories populated to run Workbench in **Amiga Files/Shared/dir/** – we'll copy these to our hard drive and mount them as hard disks in Amiberry later. **Games** and **Demoscene** directories can be found under **Amiga Files/Titles/**. Although some games are cracked, Cloanto has received explicit permission from their developers to distribute them.

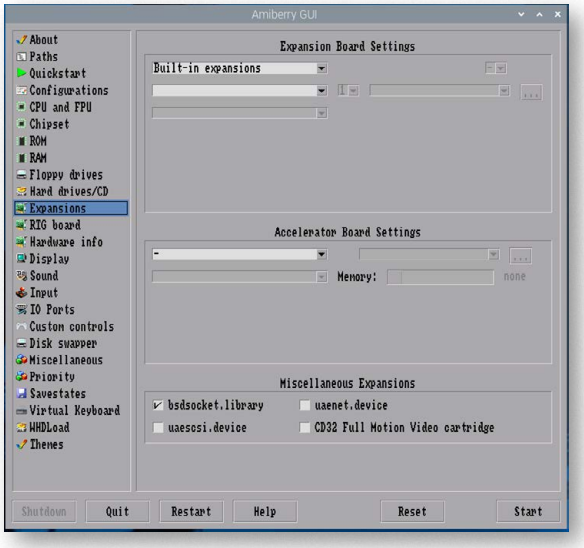
To mount the current version of the ISO, you'll need to extract the zip file it's generally distributed as. Open a terminal in the ISO file's directory, and type:

```
$ sudo mount -o loop amiga-forever-dvd.iso
/mnt
```

You'll now be able to access the disc image's contents in the **/mnt** directory. We copied everything we wanted to a folder in our home directory named **Emulation/Amiga**, particularly the **/dir** directory and its **System** and **Work** subdirectories. Both your emulated and host operating system will be able to access these directories, although files copied to them will not be visible to your virtual Amiga until it's next restarted.

We'll place some files where Amiberry expects to find them. System firmware ROMs should go in the **Amiberry/roms/** directory the emulator creates in your home directory. We also copied our ADF files into floppies and our HDF files into **/harddrives**. While this isn't required – you can load these files from anywhere – it makes life a little easier.

With that, you'll want to configure your virtual Amiga. The Amiga was in active development until 1994, with the 32-bit A1200 with an integrated keyboard, desktop PC style A4000, and A4000T – a tower build of the same hardware – being the last models, on sale until 1996, with a wide range of CPU, graphics and memory expansion units available to this day. Unofficial successors to the A4000T would use PowerPC.



To enable internet access easily, go to Amiberry's Expansions tab and tick bsdsocket.library. You'll need to install a browser such as AWeb if you want to browse the web

Configure an emulated Amiga 1200

We've opted to emulate the A1200, running Cloanto's latest Workbench 3.1. We'll select all the hardware and software specifications we want, make our emulated Amiga as powerful as possible, and save it as a system configuration. To do this:

Open Amiberry from the Raspberry Pi menu or from the terminal.

Click on the Quickstart tab. If the Start in Quickstart mode box is ticked, untick it. From the Amiga model pull-down, select Amiga 1200, then in the Config pull-down, select 4MB Fast RAM expanded configuration.

Go to the CPU and FPU tab. Ensure that 68020 is set and CPU Speed is set to A500/A1200 cycle exact emulation for more accurate timing in older software, particularly games, and 4× CPU frequency. If you want to run 3D rendering applications such as the Persistence of Vision Raytracer (POV-Ray), then you might want to add an FPU co-processor here if you're working with tasks such as ray tracing.

> We've opted to emulate the A1200

In the ROM tab, select the **amiga-os-310-a1200.rom** file we previously copied over to Amiberry's **roms** directory.

In RAM, you can safely put the Chip slider at 2MB and every other memory slider at maximum, including Processor and Fast RAM.

We'll use the Hard drives/CD tab to mount a couple of directories that we'll configure to behave as Amiga hard drives. Click Add Directory/Archive, then Select Directory, and browse to the **~/Emulation/Amiga/dir/System** directory we copied over earlier. Click Ok to select it as your first hard disk, DF0, then click Ok again. Repeat this process to add **~/Emulation/Amiga/dir/Work** as DF1. You can also mount HDF files here, as well as CD and tape images.

Under RTG board, you can add a virtual graphics card – UAE (Zorro III) is a good choice. Some hardware configurations will allow you to give it up to 128MB VRAM. If you do this, you'll want to install the Picasso96 graphics driver. Find instructions at **rpimag.co/picasso96**.

The Display tab configures how your emulated Amiga will look on Raspberry Pi. It works well and correctly maintains your aspect ratio if you set your Screen Mode to Full-window and leave all other settings at default.

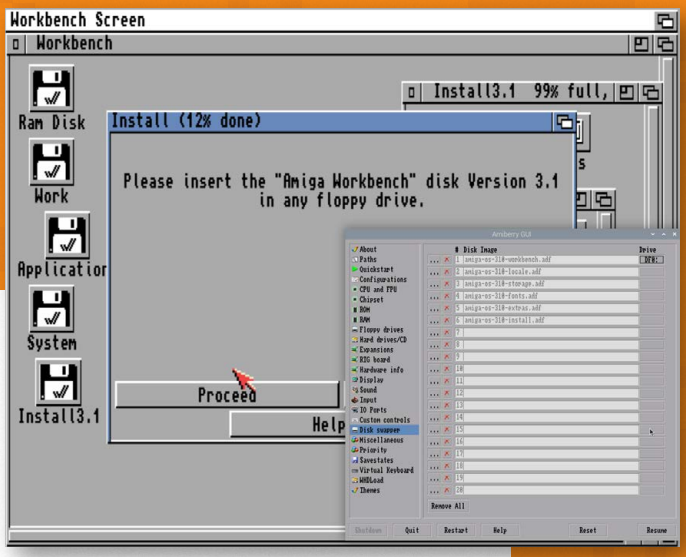
If you have a joystick or controller, make sure it's plugged in and go to the Input tab. Port 0 should be set to System Mouse and Port 1 should autodetect your controller – our 8BitDo SN30 Pro was detected as an Xbox 360 Controller. To confirm functionality, open a terminal and type the following to install and run a joystick test suite:

```
$ sudo apt install jstest-gtk
$ jstest-gtk
```

You'll probably want to tweak your emulated Amiga's settings as you go, but for now, let's save them. Go to the Configurations tab, type **Amiga 1200 (Upgraded)** into the Name field, then press Save.

Press Start to try your new config. Press **F12** to freeze emulation and open the Amiberry GUI (handy for complicated floppy disk swaps). To release the mouse so you can interact with other programs while Amiga emulation is running, click the middle mouse button.

It's worth taking the time to browse all of Amiberry's tabs, as these include a huge range of useful features from, under Miscellaneous, the ability to share your host and emulated client's clipboard, release mouse control on **ALT+TAB**, to – in I/O Ports – support for USB and traditional MIDI devices. The Expansions tab is especially worth your attention. Here, you can add dedicated graphics boards, sound cards, various PCMCIA cards and controllers, and even pass your internet connection through to your emulated Amiga, by ticking bsdsocket.library under the Miscellaneous expansions.



You'll have to swap multiple disk images when installing Workbench 3.1

Install Workbench 3.1

Cloanto provides a number of hard disk image files and directories, complete with useful installed applications. You can just copy the **~/Emulation/Amiga/dir/** folder from the Amiga Forever disc image and use that. But if we want to have a few different emulated Amigas, it can be useful to install our own hard drive folders.

Make sure the A1200 configuration we saved earlier is loaded, then, in the Floppy Drives tab, select **amiga-os-310-install.adf**, which we copied to Amiberry's floppies directory earlier. You can also add all of the 3.10 installation floppy images (workbench, locale, storage, fonts, extras and install) to the Disk swapper tab to make it faster to switch between them during installation.

In the Hard Drives tab, make sure none are currently listed, and click the Add Directory/Archive button. Set Device Name as DH0, click Select Directory. It'll open in Amiberry's **harddrives** directory. Click Create Folder, and create a directory called **A1200**. Go into that and create subdirectories named **System**, **Applications**, and **Games**. You might also want to add a **Work** or **Files** hard disk (directory) to save files you create. Finally, enter the **System** folder you just made, and click Ok to select it as your DH0 hard disk volume. Click Ok again to confirm this. Use the Add Directory feature to mount your **Applications** directory as DH1 and **Games** as DH2. Untick the Bootable box for both of these.

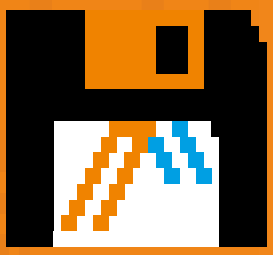
Now click Start to boot your emulated A1200. Workbench will load, with the disk mounted as Install 3.1. Double-click to open it, then double-click the **Install** directory, and then your chosen installation language.

Click Proceed, then Install Release 3.1. You can choose to allow it to auto-configure everything, or give you varying degrees of choice about how your installation is handled.

If you select Intermediate User, you'll get some degree of interactivity here, but won't have to approve every action. Click Proceed With Install, stick with the default options of a real installation and no logging of installation actions, then click Proceed. It'll ask if you wish to install release 3.1 on the System: partition. Click Yes. Select the languages you wish to install support for, then click Proceed.

You probably don't need to connect a printer, so leave all printer selections unticked and click Proceed. Select the keyboard map that best matches your Raspberry Pi 500+'s (British in our case), then click Proceed. You'll be prompted to insert the Amiga Workbench 3.1 disk. Press **F12** to open Amiberry's menu. If you previously put all your floppies into the Disk swapper, just click the blank button in the Drive: column next to your workbench disk to switch it to the floppy drive (DF0:), then click Resume. You can also manually select a new floppy image to load in the Floppy drives tab.

A shortcut for the Workbench 3.1 disk should appear in your Amiga's GUI and you can click Proceed to install its contents to your system drive folder. You'll successively be asked to swap in the other disks from the set: Locale, Extras, Fonts, Storage, and then the Install disk again. Finally, you'll be prompted to eject the disk (press the DF0: button on the disk swapping tab or the Eject button in the floppy drives tab) and click Proceed to reboot.



Or install Workbench 1.3

The process for installing Workbench 1.3 is similar. This OS with GUI dates from the Amiga 500 era, but it will run on an A1200 if you want to try it out without speccing up a new emulated machine. Full compatibility with some vintage software may require slower A500 or A600 hardware emulation, though.

Create and mount new hard drive directories, as described above, and mount the **amiga-os-134-workbench.adf** and **amiga-os-134-extras.adf** disk images using Amiberry's Floppy drives interface.

Boot the computer. When the GUI opens, double-click on the Workbench1.3 disk directory and double-click on the Shell. Type:

```
copy DF0: DH0: all
```

All the files from the disk will be slowly copied to your **System** drive (directory). When that's done, you can reset the Amiga. It'll reboot significantly faster. You can also run the same copy all command from the Extras disk we mounted with the following command:

```
copy DF1: DH0: all
```

Workbench 1.3's Extras include extra fonts, tools, and AmigaBASIC, so you can even start writing your own Amiga programs.

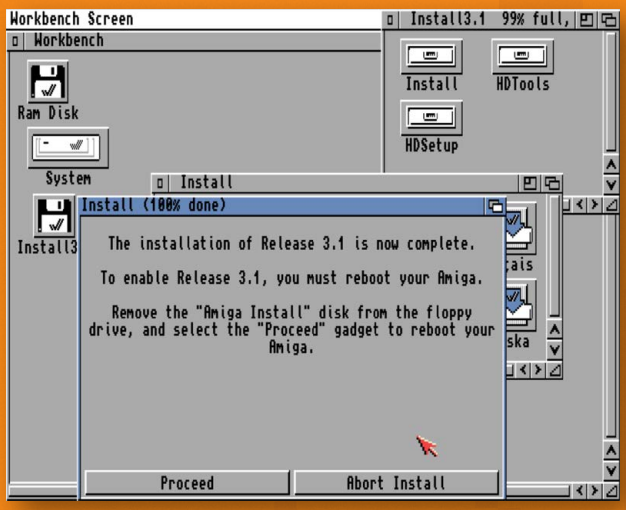
> Workbench 1.3's Extras include extra fonts, tools, and AmigaBASIC

Removing the retro

While we're emulating vintage hardware and using Cloanto's legacy OS images, the licence and source code for versions of AmigaOS beyond 3.1 (amigaos.net) are actively owned and developed by Hyperion Entertainment.

AmigaOS is available for both older Motorola 68000 (M68k) based Amigas and more recent PowerPC models, both of which Amiberry can emulate. AmigaOS is currently at version 3.2 for M68k-based systems and 4.1 for PPCs, with the latest updates to each released in October 2015, and costs €40.

Even if you stick with Workbench 3.1, there are some programs that'll need a more powerful processor than the A1200's stock Motorola 68020 CPU. You'll want to emulate at least a 68060 to run the latest Amiga version of Milkytracker, for example (rpimag.co/milkytracker). Fortunately, hardware upgrades are easy when you're emulating.



Workbench 3.1 is a little drab-looking, but you can add emulated graphics cards and extra icon sets like NewIcons to add some colour

Where to get Amiga software

Because the Amiga is still supported as a living platform, albeit a highly niche one, you'll find plenty of up-to-date software to run on it. Aminet (**aminet.net**) is one of the world's oldest online software repositories, with new programs added almost every day for a variety of different Amiga hardware specs and operating system versions. It's most easily browsed via FTP using a client such as FileZilla, pointed at **de3.aminet.net**. From here, you can explore and download a range of free software in compressed LHA format.

Fan site Lemon Amiga is packed with reviews, articles, and a comprehensive index of software for the platform. It also maintains links to active software developers and publishers (at **rpimag.co/lemonsoft** and **rpimag.co/lemongames**) that still put out releases for the Amiga.

A number of notable Amiga programs have been open-sourced, or at least had their source code made publicly available over the years. You can find a list of some notable examples at **rpimag.co/amigasources**. Although not all of these are guaranteed to compile, they're an interesting educational reference, particularly if you want to get into writing your own Amiga programs.

Amiga PD (**amigapd.com**) is a charityware site that compiles public domain games, both old and new. For a more curated public domain game pack, check out the Retro32_Public_Domain_Download_Pack (Games) at **rpimag.co/amigapdgames**.

We've been relying on downloadable open-source and public domain releases in this tutorial, but you can, of course, buy Amiga software. We have a selection of original software from the 1990s. and most of the original floppy disks still work. You'll also find Amiga specialist retailers such as **amiga-shop.net**, and instructions on how to mount them.

Meanwhile, if you are working with floppy disks, an adapter card such as a Greaseweazle (**rpimag.co/rpi5floppycd**) allows you to image and even mount Amiga floppy disks from an IBM PC 3.5in disk drive. You can also use it to create images of Amiga floppies you already own (many Amiga software licences in the UK give you the right to back them up).

You may wish to install the software on your generously proportioned virtual Amiga hard disk, rather than continuously swapping disks. See **rpimag.co/greaseweazle** for detailed info about using Amiga floppies with a Greaseweazle.

To access Workbench's menus, click the left mouse button. You'll find them at the top of the screen.

Amiberry Shortcuts

F12: Open settings and pause emulation

Ctrl + F12: Switch between full-screen and windowed modes

Pause: Pause emulation

Pause + End: Switch between warp/normal emulation speed

PrintScreen + End: Take a screenshot

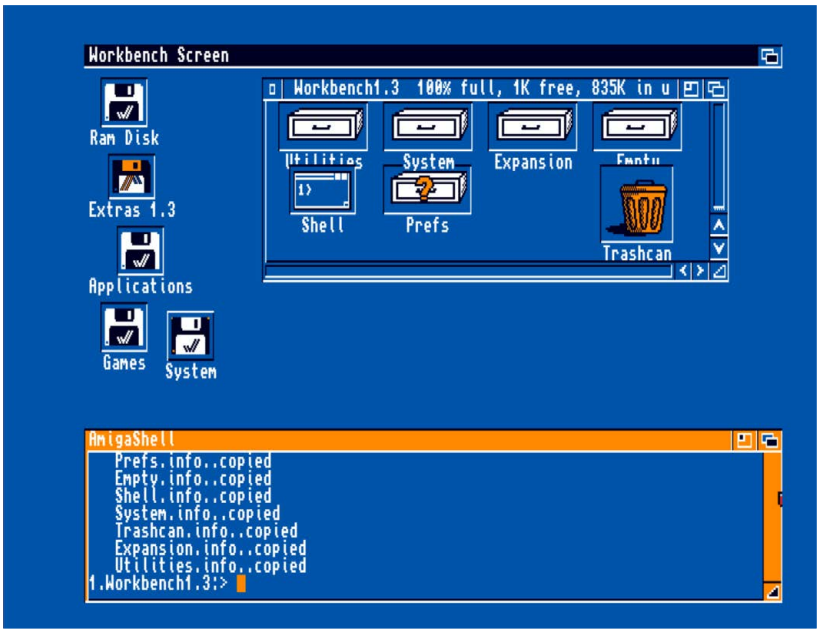
Middle mouse button: Releases mouse focus without pausing emulation

Ctrl + F11: Quit emulation

End + F1: Swap the floppy disk in DF0

You can find and reconfigure all of these shortcuts via the Miscellaneous tab of Amiberry's GUI.

We prefer the lush blue default colour scheme of Workbench 1.3



Installing software

AmigaOS, AmigaDOS, and Workbench look like they're not a million miles away from Linux and X or DOS and Windows, but their interaction paradigms and commands can be radically different, as they're a product of parallel evolution, rather than being influenced by either of those operating systems. Providing full instructions on working with them is beyond the scope of this article, but we'll take you through downloading, extracting and running an Amiga application: the free Final Word 2 text editor.

Download it from **rpimag.co/finalword**. We'll need to be able to extract it, so we'll have to download and install the LHA decompression tool. We'll also install the Amiga software installer. These are freeware programs hosted on Aminet. Download them directly from **rpimag.co/lhatool** and **rpimag.co/lhainstaller** and put them into your Applications hard disk. Other programs may have dependencies – Aminet generally lists these.

If the files that you copy to your **Applications** directory aren't visible in the GUI, select the affected window, right-click and hold, then from the top bar highlight Window > Show > All files and then let go of the right mouse button to select it. If your icons are a jumbled mess, select Window > Clean up. Then select Window > Snapshot > All to keep them in place next time you reboot. You'll need to do this for each file drawer.

Boot your emulated Amiga by loading the AmigaOS 3.1 configuration we saved earlier from the Configurations tab and then pressing Start. Double-click the System drawer to open it. Then double-click the System drawer you'll find inside that, and finally double-click on the Shell command. This opens the AmigaDOS shell.

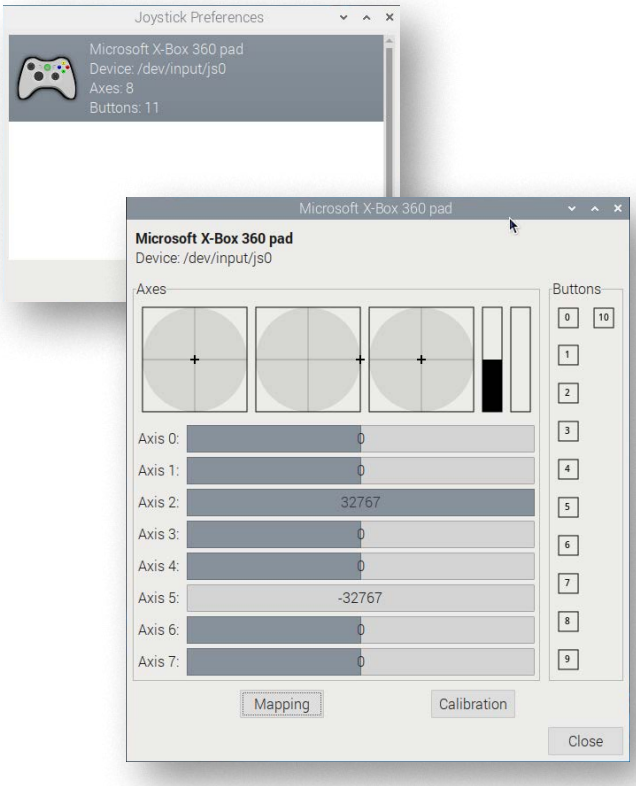
We'll use the `info` command to list our mounted drives, `cd` to change to one and navigate it, `list` the files, install LHA and use it to extract our software. Command names are not case sensitive. You can browse through previously issued commands with the up and down arrow keys.

```
info
cd DH1:
list
lha.run Ram:
copy Ram:lha_68020 C:lha
lha x Installer-43_3.lha
cd Installer-43_3
copy Installer43_3/Installer C:
cd //
lha x Finalword2.lha
cd finalword
run FinalWord
```

For additional instructions on setting up other aspects of Amiga's Workbench GUI, see the Green Amiga Alien Guide at **rpimag.co/greenguide** – and for a rundown of some tools to enhance Workbench, Everything Amiga's article at **rpimag.co/perfectwb**.

For a summary of more AmigaDOS 3.1 shell commands, see **rpimag.co/amigadosshell**, while for version 1.3, we like this reference sheet, which compares Amiga commands against those from other operating systems: **rpimag.co/amiga13shell**.

If you're using Hyperion's AmigaOS 4.0 or above, you'll find its command reference at **rpimag.co/amigaosshell**.



We'll install jstest-gtk on Raspberry Pi to help make sure our joysticks work

ONLY THE **BEST**

RP2350 boards

By **Phil King**

Raspberry Pi's RP2350 microcontroller chip is the brains of Raspberry Pi Pico 2, along with a host of third-party boards. It's been a while since we looked at some of these RP2350-based devices, back in issue 148 (**rpimag.co/148**), and many more have appeared since, so we reckon it's a good time to revisit the topic.

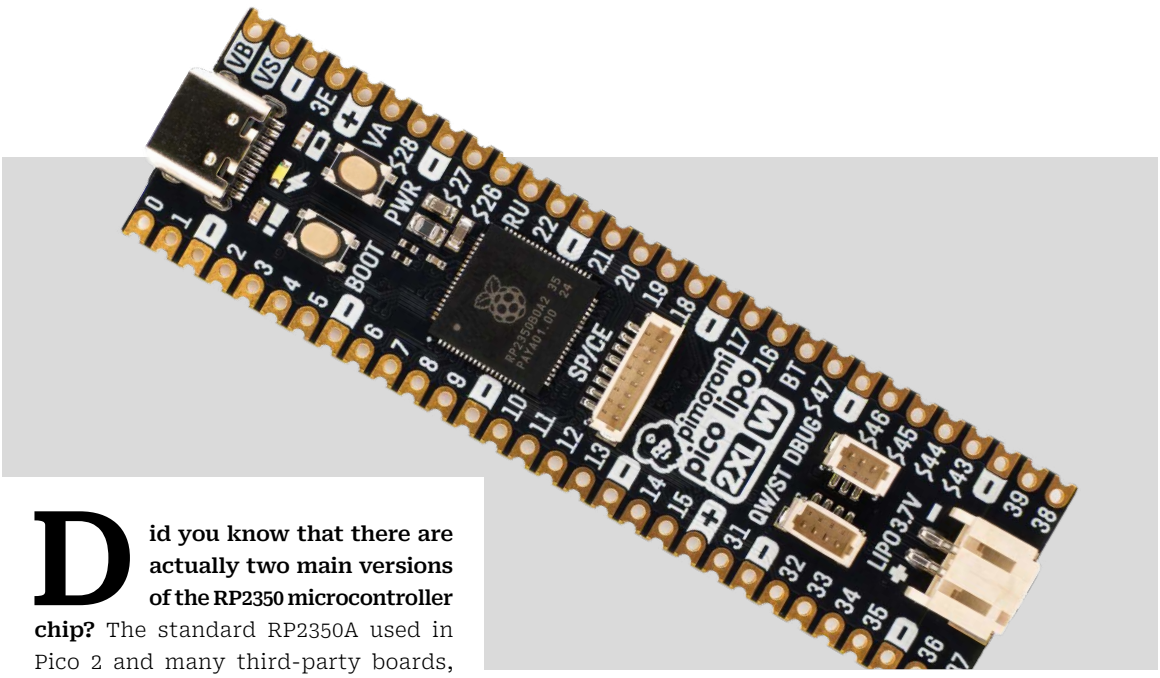
The RP2350 features dual Arm Cortex-M33 processors running at 150MHz, 520KB of on-chip SRAM, and twelve PIO state machines. So, compared with the RP2040 microcontroller from the original Pico, it offers a major performance boost for handling more complex computational tasks.

No wonder it's been used in such a wide range of third-party boards and devices – you can check out the full product catalogue at **rpimag.co/rp2350catalogue**.

We'll be taking a look at a few of the most interesting ones here, equipped with a variety of special features such as battery power input/charging, extra GPIO pins and connectors, motor/servo controllers, an Ethernet port, IMU, and even a mini LCD touchscreen.

Pico LiPo 2 XL W

Pimoroni | £21 / $23 | pimoroni.com



Did you know that there are actually two main versions of the RP2350 microcontroller chip? The standard RP2350A used in Pico 2 and many third-party boards, and the RP2350B which adds 20 more pin connections including 16 extra GPIO (and five more ADC channels).

The Pico LiPo 2 XL W makes full use of the RP2350B with an elongated board to break out those 20 extra pins. Helpfully, the rest of the pinout is the standard Pico one and all the (unpopulated) pins are labelled on the top of the board, so it's easy to find the ones you need. Fittingly, memory and storage have also been super-sized, with 8MB RAM and 16MB of flash.

A USB-C connector is used for power and programming. You can also power the board from a LiPo or Li-ion battery (not supplied) via a two-pin JST connector; there's on-board battery management and charging circuitry.

Other features include connectors for Qwiic / STEMMA QT, debug, and SP/CE (SPI/serial), along with boot/user and power buttons. An RM2 radio module offers Wi-Fi and Bluetooth connectivity.

▲ Longer than a Pico 2, it adds 20 extra pins for when you need more!

Verdict

Extra GPIOs, battery input/charging, and a lot more.

Inventor 2350 W

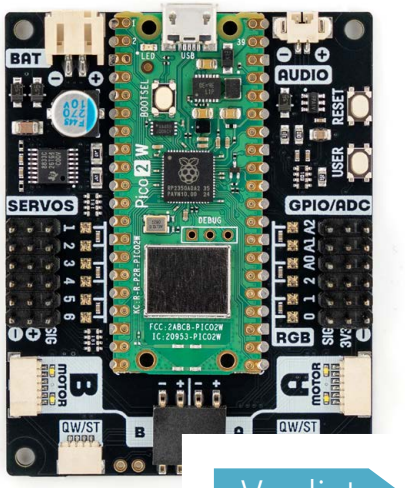
Pimoroni | £35 / $38 | pimoroni.com

With myriad connections and extra features, this versatile board makes it easier for beginners to get started with coding and electronics. It's powered by a standard Pico 2 W (with built-in wireless connectivity) soldered to the top of the board, surrounded by ports, pins, NeoPixels, and buttons – in an identical layout to that of the excellent Inventor 2040 W it supersedes.

For robotics, there are headers for up to six servos, along with an on-board motor driver connected to two JST-SH ports; with six pins, the latter are for encoder-equipped motors, but you can still wire standard ones up to a four-pin connector.

In addition, there are six GPIO headers (including three analogue inputs), two Qwiic/STEMMA QT ports (for sensors), an I2C / Breakout Garden unpopulated header, a JST-PH input for optional battery power, a two-pin connector for 1A 1.5W audio output, plus handy User and Reset buttons. Phew!

Software-wise, it's easy to use thanks to comprehensive libraries and code examples for MicroPython and C/C++.



▲ A plethora of labelled connections make it easy to find what you need

Verdict

A feature-packed board, ideal for electronics and robotics.

RP2350 1.43" AMOLED Round Display Dev Board

Waveshare / The Pi Hut | £23 / $31 | waveshare.com / thepihut.com

Verdict

A very slick RP2350-based touch display with a built-in IMU.

▶ There are so many project possibilities for this mini (48 × 48mm) device

Most Pico-style boards lack any form of display, although you can buy add-ons to enable video output. Waveshare, however, produces a range of RP2350-based devices with a built-in display – in various shapes and sizes.

This particular model is circular and comes with an optional metal case with cut-outs for the boot and reset buttons, USB-C port, microSD card slot (for extra storage), I2C and UART four-pin headers, and GPIO breakouts on the rear.

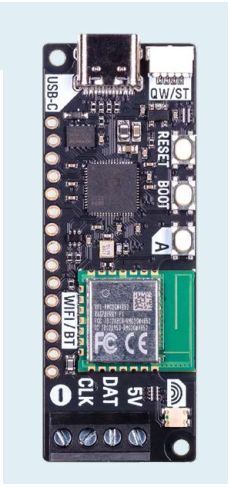
With a 466 × 466 pixel resolution, the colour AMOLED screen looks vibrant and has capacitive touch. Combined with the built-in six-axis IMU and an RTC with battery header, this could make it useful for a wearable or pocket project – perhaps a fitness tracker?

While the MicroPython firmware is limited to a handful of code examples, the C/C++ firmware is far more comprehensive, including the use of LVGL (Light and Versatile Graphics Library) to render text, images, and an example GUI complete with animated graphs and pop-up tiny keyboard.

Plasma 2350 W

Pimoroni | £17 / $18 | pimoroni.com

▶ Light up your world with a Plasma 2350 W and LED strip



Verdict

An easy and fun way to control a string of addressable LEDs.

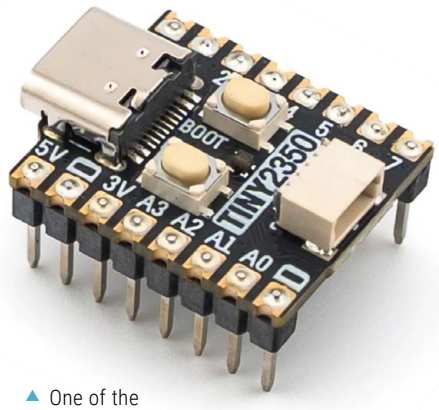
We reviewed the original Plasma 2350 back in issue 146 and now there's a version with built-in wireless connectivity, which opens up extra possibilities for controlling a connected strip of WS2812/NeoPixel or APA102/DotStar RGB LEDs – the optional starter kit includes a 10m string of 66 frosted LED stars.

It's dead easy to get started. Just connect your LED string's three or four wires to the board's screw terminals, then use its USB-C port to access the MicroPython firmware, library, and numerous code examples on a computer. Impressive effects include falling snowflakes, alternating/random blinkies, sparkles, fire, pulsing, and a sweeping rainbow. More can be found via some community resources, including an impressive demo with 77 effects.

To give the board access to your wireless network, you'll need to add your Wi-Fi details to a **secrets.py** file. You can then try out web-based examples such as setting the colour of the LEDs via the CheerLights IoT system on social media, or the lighting effect according to the weather from Open-Meteo.

Tiny 2350

Pimoroni | £8 / $9 | pimoroni.com



▲ One of the smallest RP2350-based boards around

If you need a really small RP2350-based board for a project where space is very limited, the Tiny 2350 is ideal. Measuring a mere 22.9 × 18mm, it really is tiny, around the same size as a standard UK postage stamp.

Along with a boot select button, the board crams in a handy reset button, RGB LED, and 4MB of QPSI flash storage, although there's no wireless connectivity. A USB-C port is used for power and connecting to a computer for firmware installation and coding in MicroPython, CircuitPython, or C/C++.

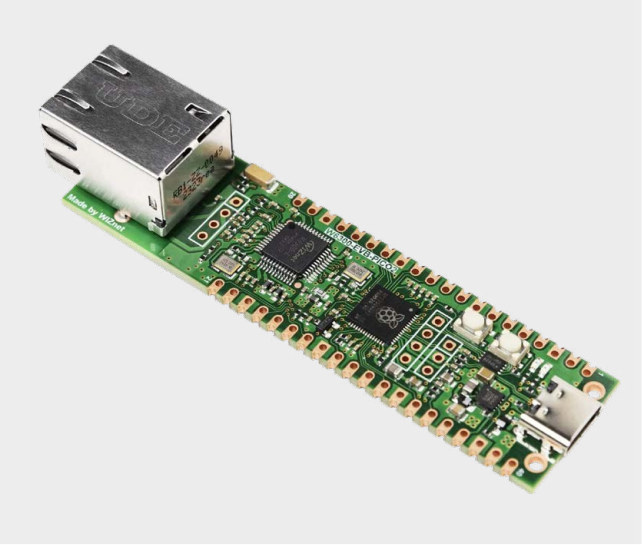
The Tiny 2350 is available with or without pre-soldered pin headers. One obvious downside is that the number of pins is reduced compared to a standard Pico 2. There are 16 in total, including 12 GPIOs (plus another couple on the Qwiic/STEMMA QT port). They include four 12-bit ADC channels, though, and encompass two channels each of the I2C, SPI, and UART protocols.

Verdict

A teeny-tiny RP2350 board that's ideal for smaller projects.

W6300-EVB-Pico2

WIZnet / The Pi Hut | £11 / $15 | wiznet.io / thepihut.com



While many RP2350 boards boast built-in wireless connectivity, sometimes you may need a wired Ethernet connection for improved reliability, security, and speed. Possible projects include a low-power HTTP web server, network monitoring, IoT data logging, and industrial devices.

WIZnet produces an Ethernet HAT to use with standard Pico boards. Alternatively, you can use one of the firm's range of Ethernet-equipped RP2040 and RP2350 devices. This RP2350 one is based around WIZnet's own W6300 chip: a 10/100 Ethernet controller with a hardwired TCP/IP stack that supports both IPv4 and IPv6 – as do the W6100 models, while the W5500-based boards are limited to IPv4.

All of them feature an identical pinout to Pico/Pico 2, with 40 pins and 26 multipurpose GPIOs, so it's all very familiar. The documentation includes C/C++ firmware/code examples for Ethernet, FreeRTOS, AWS, Azure, and chip performance. There's no MicroPython firmware available for the RP2350 models yet (unlike the RP2040 boards); hopefully that will come soon.

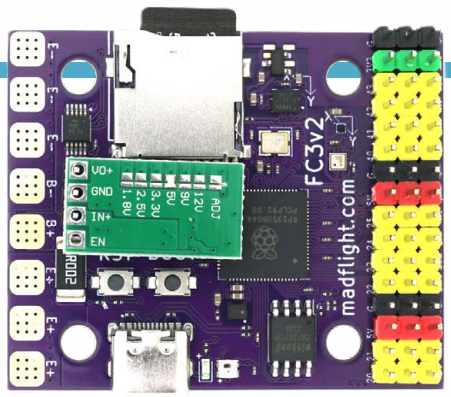
Verdict

Perfect for projects requiring a wired network connection.

- Want a Pico 2-like board with an Ethernet port? You've got it!

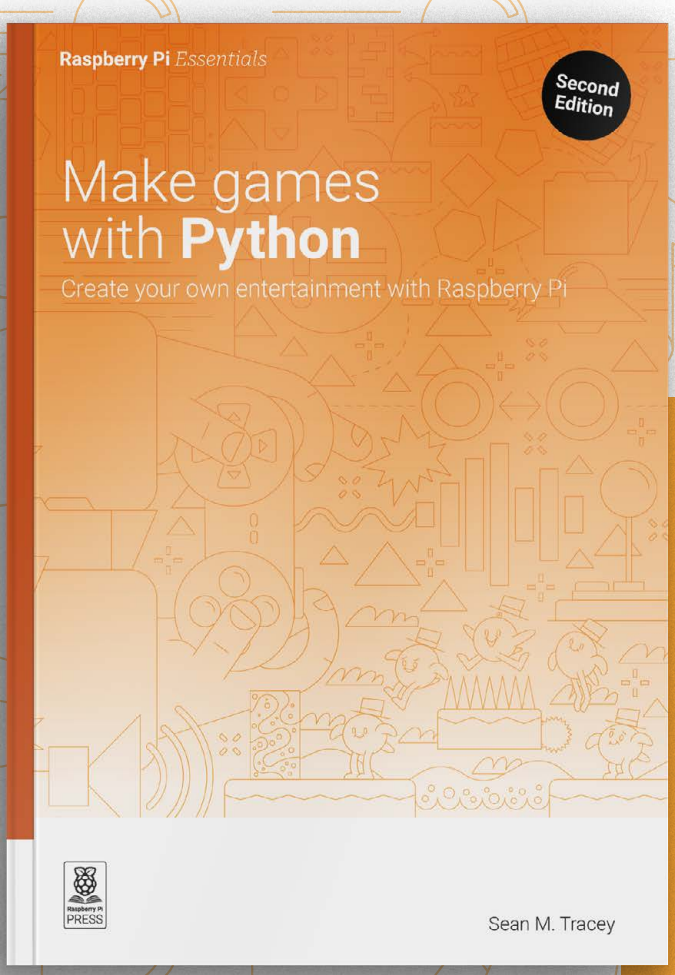
FC3V2 – RP2350B FLIGHT CONTROLLER

Madflight | £22 / $29 | madflight.com



Want to build an RP2350-powered drone? This flight controller makes it a lot easier with an on-board six-axis IMU, magnetometer, barometer, and battery monitor. It also has extra power/ground pins and standard 30.5 × 30.5mm mounting holes to fit a standard drone frame.

- An RP2350B-based flight controller with built-in IMU and more



While millions of us enjoy nothing more than spending hours racking up high scores on our favourite video games, too few are exposed to an even more gratifying way to spend time — making them.

This book teaches Python and Pygame development, helping you to understand the games you play and create almost anything your imagination can come up with.

- ***As you work your way up to creating your own shoot-'em-up game, you'll learn how to:***
  - *Create shapes and paths*
  - *Move sprites and detect collisions*
  - *Handle keyboard, mouse, and gamepad input*
  - *Add sound and music*
  - *Simulate physics and forces*

BUY ONLINE: **rpimag.co/makegamesbook**

Badgeware

A trio of smart interactive badges powered by RP2350. By **Phil King**

Pimoroni **rpimag.co/badgeware** From £50 / $55

SPECS

DISPLAY:
2.7-inch, 264 × 176 e-paper (Badger); 2.8-inch, 320 × 240 colour IPS LCD (Tufty); 39 × 26 (minus islands) white LED matrix (Blinky)

FEATURES:
RP2350A microcontroller, 16MB QSPI flash storage, 8MB PSRAM, 802.11 b/g/n wireless LAN, Bluetooth, 5× user buttons, Home/Boot and Reset/Sleep buttons, polycarbonate case with LED lighting

POWER:
1000mAh LiPo, rechargeable via USB-C port

DIMENSIONS:
84 × 76 × 20mm

▲ The Tufty is our favourite, featuring a vivid colour LCD screen and more example apps

▶ The Badger model with the app launcher menu shown on its e-paper display

Pimoroni's new range of RP2350-based Badgeware wearable devices (lanyard included) offer a major upgrade over the originals. There are three models to choose from: the Badger (with e-paper screen), Tufty (colour LCD), and Blinky (LED matrix). The first two follow on from their namesake RP2040-powered predecessors, while the Blinky is an interesting addition to the line.

One of the first things you'll notice is that these badges are taller than their predecessors, to accommodate a larger display (whose exact size varies slightly between models). The five user buttons are positioned similarly along the bottom and right-hand edges, but are bigger and chunkier than before, so easier and more comfortable to press.

The Badgeware devices feature a slick and sturdy design with an integral translucent 'jewel-coloured' polycarbonate case clipped to the rear – it even has four white LEDs to light it up. Inside, there's a 1000mAh LiPo battery that's easy to recharge via a USB-C port – the latter is also used to connect the badge to a computer for programming.

Also on the rear – along with tiny Home/Boot and Reset/Sleep buttons –

The Blinky's LED matrix display works well for scrolling text or, as shown here, an animated clock

you'll find an SWD (Serial Wire Debug) connector along with an I2C one for connecting Qwiic/STEMMA QT breakouts – or the optional STEM Kit which features a mini joypad, multi-sensor stick, and two JST-SH cables.

Take your pick

Of the three models, the Tufty catches the eye first with its vibrant colour LCD screen – 2.8-inch with 320 × 240 pixels. As with the other two badges, powering it up reveals a launcher menu of app icons to navigate with the user buttons. Along with mass storage mode, there are 13 pre-installed examples (all written in MicroPython) to play with, including a few simple games, a clock, pomodoro timer, 'Sketchy Sketch' drawing, impressive graphical demos, and (naturally) an ID badge.

The Badger model features a 2.7-inch (264 × 176) monochrome e-paper display that should prove a lot easier on the battery, although it takes about a second per screen refresh. There are only four pre-installed apps: the ID badge, clock, hydration meter, and a fun 3D mini adventure called The Compendium.

The Blinky features a bright white LED matrix (872 pixels) that extends around the user buttons. Pre-installed apps comprise an ID badge (in the form of scrolling text in a choice of fonts), clock, graphical demos, snake, and a 3D game (as on the Tufty) where you steer a spacecraft through gaps in walls.

> *You can adapt an existing app or create a new one from scratch using the comprehensive API*

Make it your own

Using a USB cable inserted into the badge's USB-C port, there are two ways to connect it to a computer. Holding the Home/Boot button and tapping Reset puts it into mass storage mode if you ever need to re-flash the MicroPython firmware (a special Pimoroni flavour).

Otherwise, tapping just Reset twice opens a folder with the device name on the computer desktop so you can view the file system and apps. This enables you alter the **secrets.py** file to connect to your wireless network, as well as add your own (50 × 50 PNG) avatar and social media details for the ID badge app.

By opening an IDE such as Thonny, you can program the device in MicroPython, just as you would with a standard Pico. You can adapt an existing app or create a new one from scratch using the comprehensive API, which includes functions for images, text, shapes, vectors, fonts, algorithms, and more – see **badgewa.re/docs** for the detailed documentation.

Each badge has a polycarbonate case that covers the rear components, including a built-in battery

Verdict

Whichever model you choose, it's a sturdy smart badge with built-in battery, powerful RP2350, wireless connectivity, and an excellent API for coding custom apps.

9/10

Kiwi+ USB

An upgraded Kiwi KVM brings USB device sharing between machines. **Rob Zwetsloot** makes the switch

Cytrence **rpimag.co/kiwiplus** £89/$119

SPECS

I/O
USB-C (host connection), USB-C (input connection), USB-A (shared connection), HDMI, 6× GPIO

CONNECTIVITY
1080p video, human input devices, virtual serial connection, UART, ATX, USB devices

DIMENSIONS
46 × 46 × 15mm

Verdict

The Kiwi KVM with some nice upgrades that can really streamline development, with a fairly reasonable price point.

9/10

Kiwi's KVM is not a regular KVM, in that you're not just sharing a keyboard, video, and mouse between two systems. It also allows you to capture the video from your other computer and display it on a host PC, complete with the ability to send mouse and keyboard commands back to the target. Like a local remote VNC, but with slightly better response times.

We were surprised how well the original version worked, especially compared to expensive game capture cards that don't even allow for that kind of interactivity between systems. Cytrence has since put out several new versions of Kiwi under a '+' label – one with built-in shared storage, another that creates a local network between both machines, and a third with the simpler concept of a shared USB port.

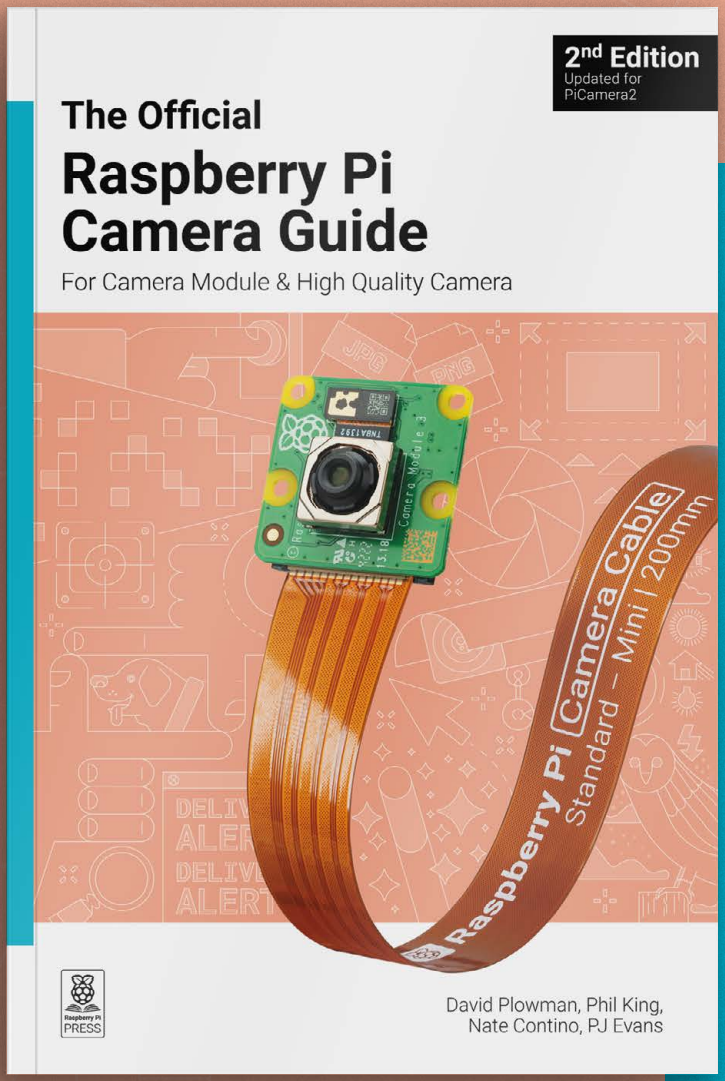
Plus it

The Kiwi+ USB retains the same great and very responsive features of the original, along with some of the dev options of the Kiwi Pro, and this new ability to switch a USB port between the host and target machines. Like before, it's controlled via a dropdown menu on the capture window,

▲ Kiwi+ has an extra port for USB-A devices to connect to

> *Switch a USB port between host and target*

allowing you to change which machine is using the USB port. It only works on one system at a time due to the limitations of, well, technology. So if you wanted some dedicated storage on a flash drive for both systems to use, you're better off choosing the Kiwi+ Drive option. For everything else, and for testing software and hardware, it's perfect. Switching between systems isn't really slow and is about as fast as moving a USB device from one port to the neighbouring port, so it is also quite convenient. You are paying for that extra convenience, though, with the '+' range coming in at $40 more than its predecessor – although that still makes it cheaper than a lot of its competition and it has the extra dev features too.



Add the power of HDR photography, Full HD video, and AI image recognition to your Raspberry Pi projects with Camera Modules.

- ***Getting started***
- *Capturing photos and videos*
- ***Control the camera with precision***
- *Add artifical intelligence with the AI Kit*
- ***Time-lapse photography***
- *Selfies and stop-motion video*
- ***Build a bird box camera***
- *Live-stream video and stills*
- ***...and much more!***

BUY ONLINE: **rpimag.co/cameraguide**

APPLY TO POWERED BY RASPBERRY PI

Our Powered by Raspberry Pi logo shows your customers that your product is powered by our high-quality Raspberry Pi computers and microcontrollers. All Powered by Raspberry Pi products are eligible to appear in our online gallery.
rpimag.co/poweredbypiapply

Powered by Pi accreditation roster is growing and growing. **Rosie Hattersley** welcomes some of the latest entrants

At a presentation to hobbyists and makers at CamJam in Cambridge at the very end of February to mark 14 years of Raspberry Pi, co-founder Eben Upton detailed the origins of the company and the runaway success of the delightful single-board computer that started it all. Eben noted that the team always expected the brand recognition side to focus on Pi ('π') and thought themselves clever to have used the mathematical symbol in the name. As it turned out, it was the fruity part that came to dominate visuals denoting the world's most successful computer.

The Raspberry Pi graphic is now synonymous with our beloved computer, which inspires ardent loyalty as well as many hundreds of compatible products and accessories. Distinguishing a top-quality product is made very much easier once you know to look out for the Powered by Pi logo. It denotes a product accredited by Raspberry Pi and one that has gone through rigorous testing, with nearly 400 products now bearing the scheme's badge.

You can find detailed information about the compliance regulations and testing procedures for every Raspberry Pi product at **pip.raspberrypi.com**. To browse other products with the Powered by Raspberry Pi badge, view the catalogue at **rpimag.co/poweredbyraspberrypi**.

Sharp IGZO MPi5 Kit

Japan | rpimag.co/sharp-display

Sharp and NEC have had a partnership for many years using Raspberry Pi Compute Module in their digital signage offerings, proving a versatile solution for industrial products. The IGZO MPi5 Kit makes good use of the superior processing power of the 64-bit Arm Cortex-A76 2.4GHz quad-core chip in Raspberry Pi Compute Module 5, running a special version of Raspberry Pi OS for digital signage. The hardware includes a slew of USB and Ethernet connections and an elegant, cable-free design so it can be slotted into a display seamlessly with no need for any externally mounted hardware. The unmatched price-performance ratio, reliability, and proven success story in industrial applications, along with its low energy consumption, make the CM5-based Sharp IGZO MPi5 Kit an excellent choice for digital signage applications.

HiFiBerry DAC 8X

Switzerland | hifiberry.com

Modul 9 is a Swiss company that specialises in high-quality audio and embedded devices and was one of the first audio brands to design hardware specifically to take advantage of Raspberry Pi's DSP capabilities. The HiFiBerry range is its affordable but well-engineered range of digital audio processors and amps. The DAC 8X is the first eight-channel digital audio converter for Raspberry Pi 5, supporting both CD quality (16-bit) and 24-bit high-resolution audio playback. However, the HiFiBerry range offers Raspberry Pi audiophiles a wide array of product features, including ADCs (analogue-to-digital converters) and various inputs and connectors for microphone and line-in uses. All products in the HiFiBerry DAC line-up work with any Raspberry Pi computer with a 40-pin header. All except the entry-level model have built-in volume control and come with a customer satisfaction guarantee. As you will see from the gallery, HiFiBerry has inspired many a project build: **rpimag.co/hifi-gallery**.

PinToMind

Norway | pintomind.com

Drawing attention to menus, providing dynamic information to visitors, and simply advertising your small business and what it sells are instantly more eye-catching if you use digital signage. We can't vouch for the quality of your products or your visual design skills, of course, but there are plenty of Raspberry Pi-compatible software services that provide templates and guides to help you come up with something that looks great and doesn't require expensive installation. Norway's PinToMind offers a range of options, including a free one if you're hosting a festival and just need a simple, low- or battery-powered screen setup. Details for using this ingenious digital signage software with Raspberry Pi are at **rpimag.co/rpi-player-setup**. Both preinstalled and install-it-yourself options are offered, so you can quickly and easily create digital content using PinToMind's templates and share them on any sort of screen, including via a mobile app.

AtumX Snowflake

India | atumx.in

AtumX's beginner-friendly electronics kit is intended for wannabe coders to take their first steps into the world of building projects using Scratch or Python. The Snowflake is an RP2040-based microcontroller board that can work with 15+ sensors to check the current temperature, light levels, and whether there are any obstacles ahead. Designed for Code Clubs and teaching environments, Snowflake connects to a laptop (or Raspberry Pi) via its USB cable so there's no need to first negotiate a Wi-Fi setup before getting down to the fun stuff such as making blocks move, lights flash, and buzzers buzz. The kit links to India's education curriculum, and takes STEM students from first steps to gaining the confidence to tackle simple robot design.

Raspberry Pi Plastics ActionCAM

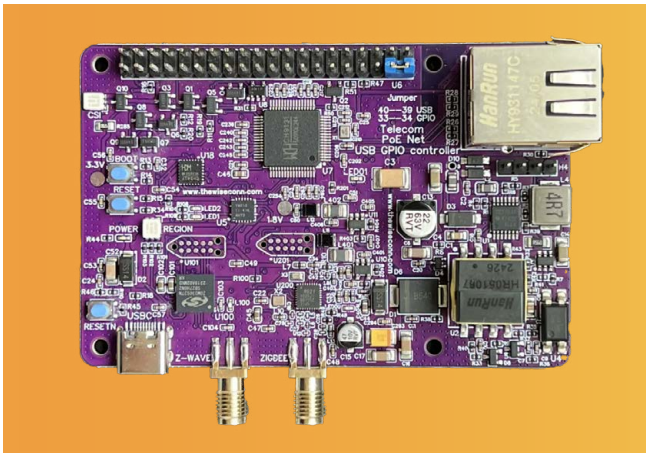
Wales | raspberrypiplastics.com/actioncam

Designed and manufactured in Pencoed, where a certain fruit computer is made, the ActionCAM is a rugged, splashproof, outdoor camera that can be used to stealthily capture wildlife shots or be used as a wearable video cam. Time-lapse photography, stills, and motion capture modes are all supported, as is a slow-mo mode in which 720p footage is shot at 60 frames per second. The ActionCAM's battery offers around three hours of use before needing to be recharged, so it's eminently useful if you want to record a hike or other outdoor activity. It comes with a 16GB microSD card and can be bought with or without the Raspberry Pi Zero W needed to control it. The Zero W of course also provides wireless connectivity, meaning the ActionCAM's shutter can be triggered remotely and different camera modes selected. The camera in question is a Raspberry Pi Camera Module v2. Spring is here at last, so it's a great option for getting out and about exploring nature.

Wiseconn PoE Z-Wave 800 & ZigBee 3.0 Hub

USA | thewiseconn.com/products

Smart home devices may use a variety of different connectivity standards, which can be confusing. To make it easier, Wiseconn's hub supports both Z-Wave and ZigBee wireless network standards, along with Bluetooth and Matter (via Wi-Fi). It's billed as a 'next-gen smart city controller' with energy efficiency a huge selling point versus some other connected device options. Whether you want to automate a few light bulbs and switches or light up a whole neighbourhood (it can connect up to 4000 devices), Wiseconn claims its Raspberry Pi 4- and 5-compatible controller's power over Ethernet (PoE) and dual high-gain external antenna connectors, as well as low latency, make it a strong choice. It's fully compatible with Home Assistant and HomeSeer home automation platforms, too.



10 amazing:

non-traditional clock projects

Weird ways to tell the time with a Raspberry Pi

We love it when folks are creative with a Raspberry Pi. While there are definitely many Raspberry Pi devices out there plugged into factories and industry doing important jobs, it's nice to see when people do something artistic – or just plain funny – with a Raspberry Pi. A great example of that is some of the ways people have taken the ancient concept of telling the time, and given it a twist. Here are just some of our favourites.

01. Counter-rotating clock

Shifted perspective

rpimag.co/rotatingclock

The project that sparked the idea for this list, where the hour hand remains stationary but the entire face rotates instead. Editor Lucy does not enjoy it.

02. Falling clock

Manufactured error

rpimag.co/fallingclock

When Burke McCabe decided to fix his clock, which apparently made a screeching noise, he added a camera so the clock would know when it was being looked at. After which, it would launch itself off the wall.

03. Colour word clock

Say the time

rpimag.co/colwordclock

Word clocks use a word-search-style mixture of words and letters to tell you the time, any time, in the 12-hour style. There's word clocks available in other languages too.

04. Smart Nixie tubes

Retro tech

rpimag.co/smartnixie

A striking bit of a retro technology updated with a Raspberry Pi 3 to accurately tell the time. The perfect prop for your steampunk movie.

05. Moon and Tide Clock

Alternate time-telling

rpimag.co/tideclock

Less weird and more 'cool and classy', this vintage carriage clock uses e-ink displays to show the current moon phase, as well as the tide timings. It looks really lovely.

06. Robot Arm Clock

Automated manual progression

rpimag.co/robotarmclock

When your clock breaks, how do you fix it? Well, Hendrik Ohrens programmed a robot arm to replace a hand on the clock so that it will manually move the minute hand. Obviously!

07. Pico solar system display

The cosmic ballet

rpimag.co/picosolar

An early but memorable Pico project simulates the positions of the planets in our humble Solar System depending on what time it's set to. You could set it to the future (or past) if you wish to miss work.

08. Flip clock

Got you babe

rpimag.co/flipclock

Flip clocks are retro chic, and memorably featured in *Groundhog Day*, and with modern 3D printing and Raspberry Pi know-how, it's dead easy to make one yourself. Every day.

09. Self-snoozing alarm

Get more rest

rpimag.co/selfsnooze

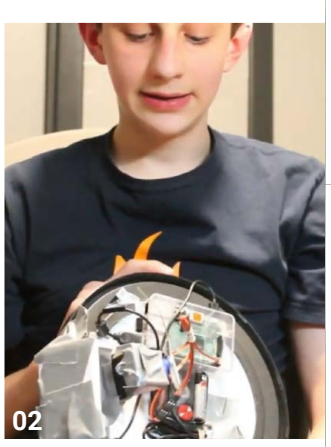
A bit like that box that closes itself, this lovely retro-styled alarm clock art piece will go off – only for the robot arm to snooze itself for you.

10. Dual Spiral Marble Clock

Time rolls on

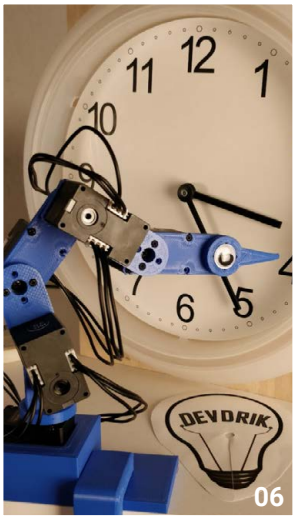
rpimag.co/spiralclock

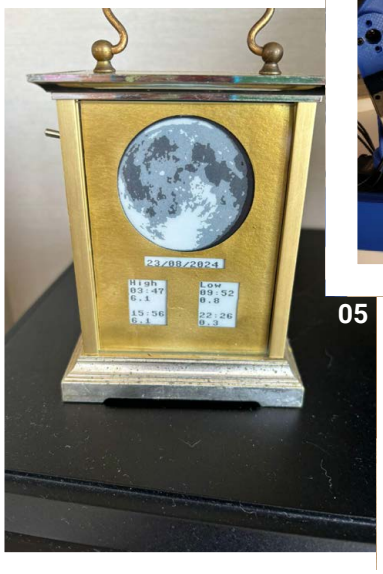
Very simply, a spiral slowly turns, changing the position of a marble so that it accurately lines up to the hour on one spiral, and the minute on the other spiral. Why? Why not!


02


HATWENTY
FIFVTEEN
LF*PASTO
NINEIGHT
ONETHREE
TWELEVEN
FOURFIVE
SIXSEVEN
GurgleApps.com
03


4 S I 5
04

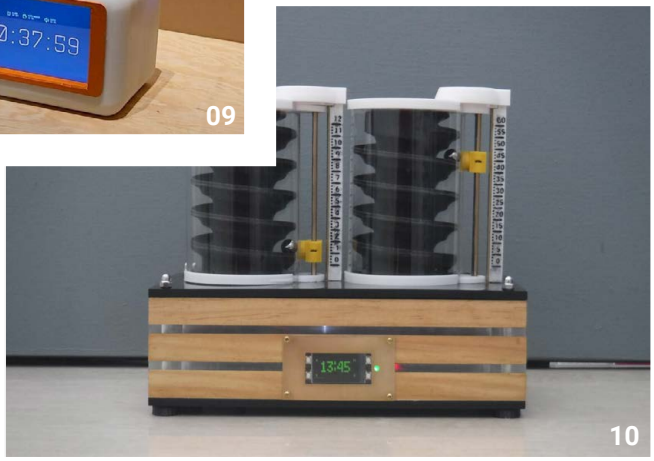

DEVDRIK
06

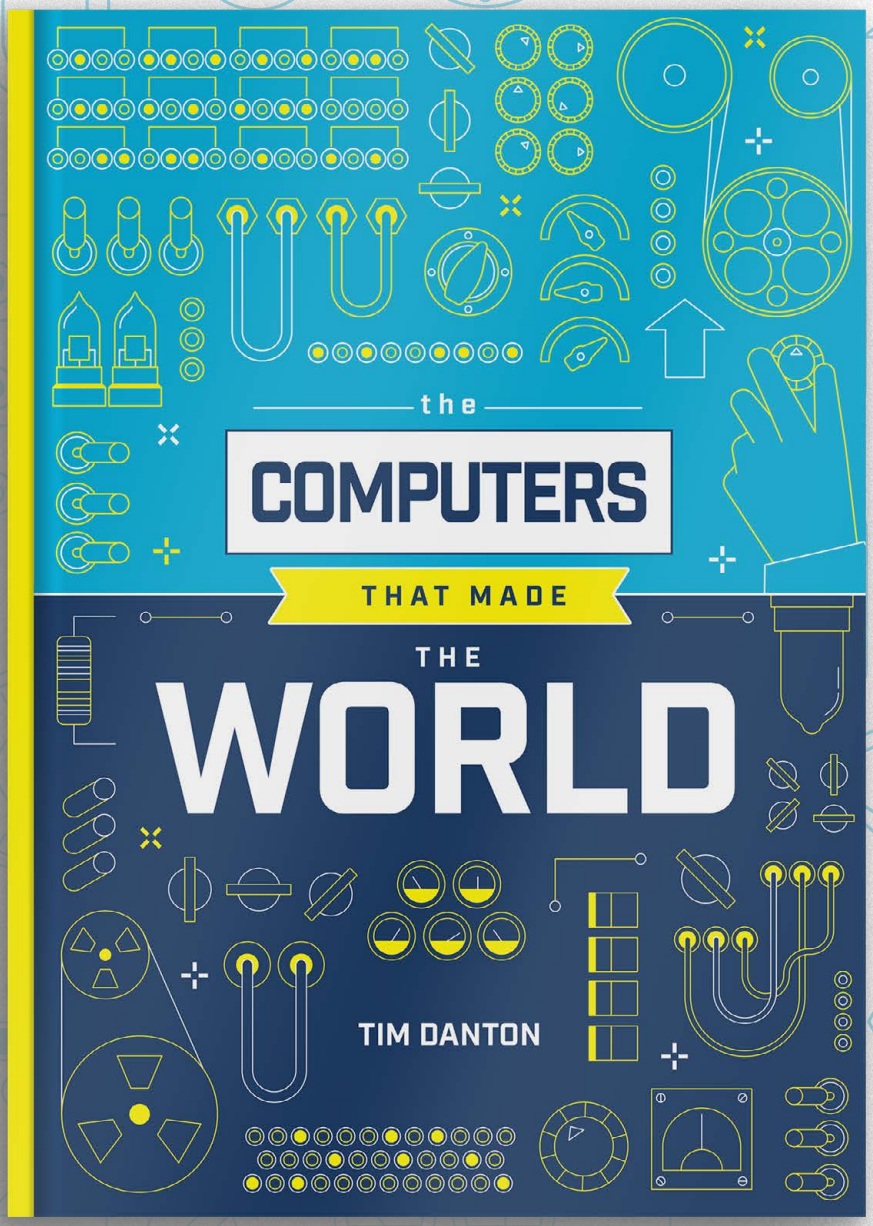

23/08/2024
05


pico display
A
B
X
Y
RGB
27 JUL
2022
WEDNESDAY
19:39
pimoroni.com/picodisplay
07


20:37:59
09


18 32
08


13:45
10


the
COMPUTERS
THAT MADE
THE
WORLD
TIM DANTON

The Computers that Made the World

How twelve pioneering computers built between 1939 and 1950 helped shape modern technology

In 1940, a *computer* was someone who ploughed through gruelling calculations each day. A decade later, a computer was a buzzing machine that filled a room. This book tells the story of how our world was reshaped by a dozen such computers — and the geniuses that brought them into being, from Alan Turing to John von Neumann. But this isn't just a story about how these computers came to be, or the people behind them: it's a story about how a new world order, built on technology, sprang into being.

This world tour through the modern history of computing begins in 1939 with the first electronic digital computer, the Atanasoff-Berry computer (ABC). From there, it moves on to the Berlin-born Zuse Z3 and Bell Labs' Complex Number Calculator, before we enter the World War II era with Colossus, Harvard Mark I, and then ENIAC, the first general-purpose digital computer. You'll also discover the fascinating stories behind the Manchester Baby, EDSAC, EDVAC, UNIVAC, Princeton IAS, and Alan Turing's Pilot ACE and the birth of artificial intelligence.

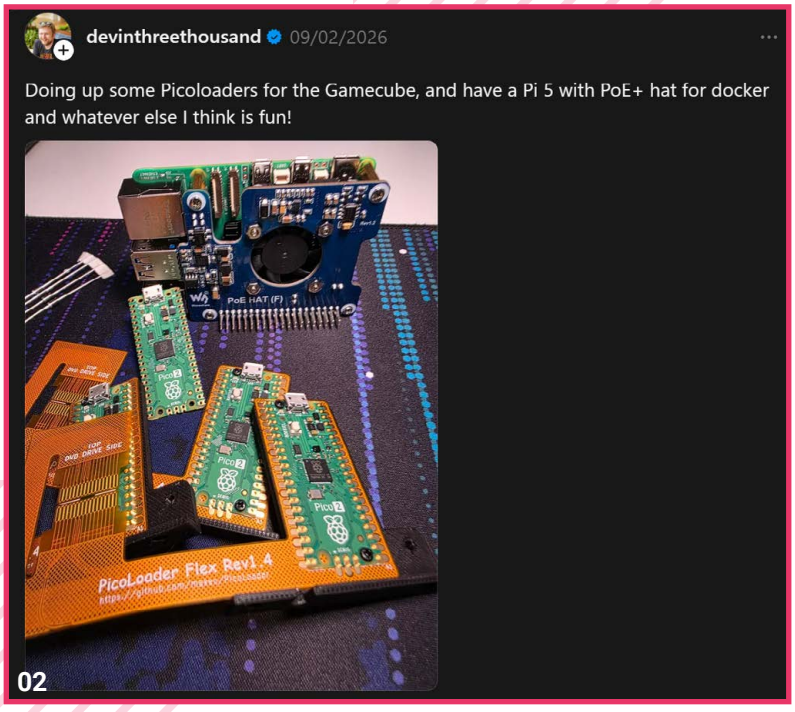
In *The Computers that Made the World*, you'll not only learn about the computers that shaped the world we live in, but what happened behind the scenes.

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. Stock tickers are an easy little web API project, and look good on e-ink
02. The GameCube had special miniDVD-size discs that were its own proprietary format
03. It's really a lovely way to have a workstation
04. Stable voltages can be essential for healthy systems
05. We do love when a screen can power *and* display Raspberry Pi desktop
06. Hacky Racers is making its way back!
07. We have more from the party over the page
08. Is this the world's smallest weather station? Probably not, but it is tiny

04

PenguinTutor (Stewart Watkiss)
@penguintutor@fosstodon.org

@rpimag I've been creating some new videos to accompany my electronics series.

This week I've created a video on voltage regulators, perfect for supplying a stable voltage for powering a Pico.

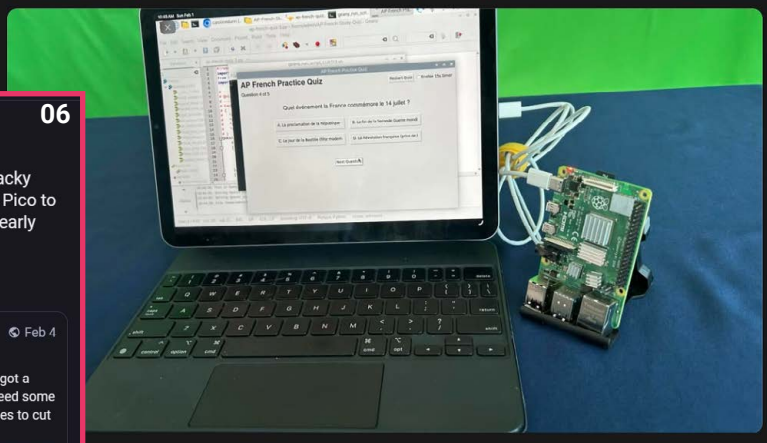
youtube.com/watch?v=-uSdWWigv6I

YouTube
What's Insid
LM7812) #E

05

thecarolinedunn 09/02/2026

Using my iPad as a power source and monitor for my RPi4 youtu.be/rvCaN...



06

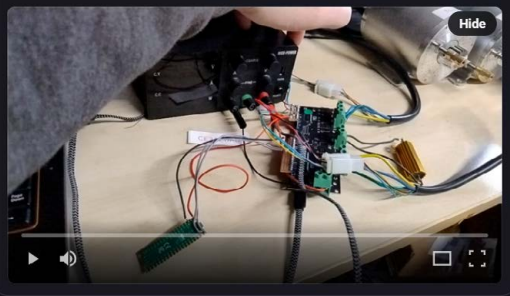
East Essex Hackspace
@eehackspace@chaos.social

@rpimag East Essex Hackspace is current rebuilding its Hacky Racer with twin motors, an ODrive to control them and a Pi Pico to do mixing of throttle, steering and brakes. We're still in the early stages...

chaos.social/@eehackspace/1160...

East Essex Hackspace
@eehackspace@chaos.social
Feb 4

Progress on our Hacky Racer is slow but yesterday things got done. We've got a MakerBase ODrive v3.6-56V to run our dual motor setup. This is going to need some mixing of the two based off throttle and steering and also the brake switches to cut drive when braking. So we've connected a Pi Pico.



07

Dr Footleg (he/him)
@drfootleg@fosstodon.org

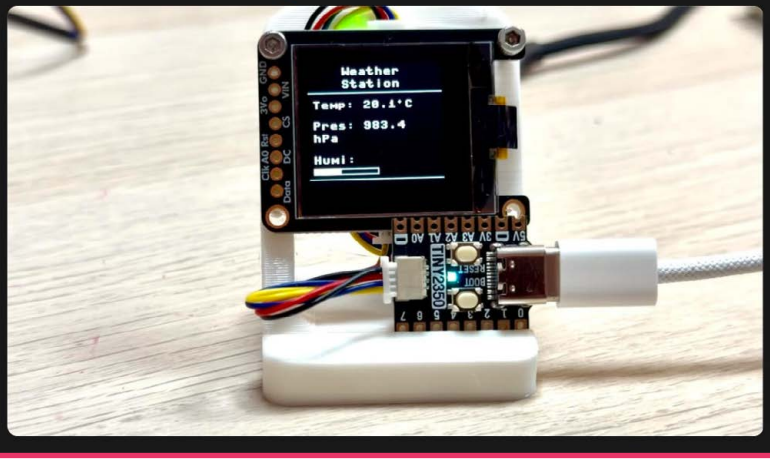
@rpimag Nat and I demonstrated the cloud chamber project at the #RaspberryPi birthday party in Cambridge on Saturday. We had lots of great conversations over it with visitors to Makespace. #MakerMonday

08

kevinmcaleer › MakerMonday 24/02/2026

Happy Maker Monday - this week I created an environment sensor with a display, using the @pimoroni Tiny 2350 and an @adafruit i2c display.

Write up on kevsrobots.com/blog... (with video too)



Raspberry Pi Birthday CamJam

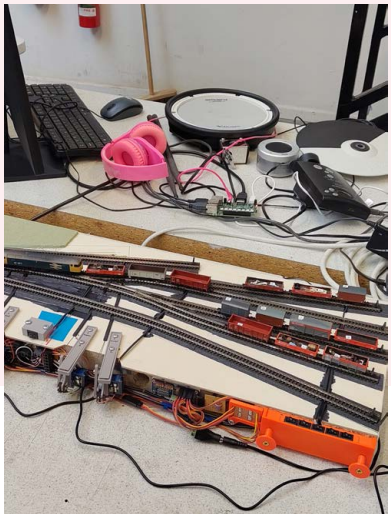
Raspberry Pi turns 14, and there's no acne in sight

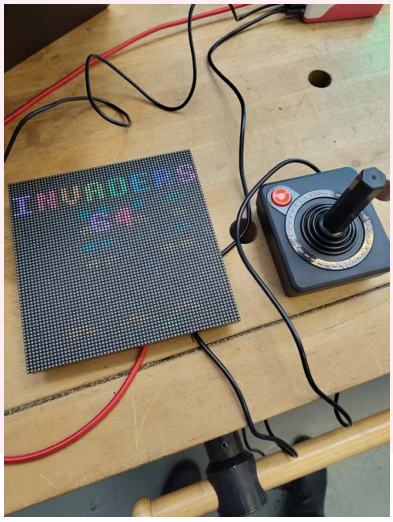
Many Raspberry Jams and maker events love to celebrate the birthday of Raspberry Pi – which is technically 29 February but when your birthday is a leap day, you get to choose a nearby day anyway.

One of the oldest Raspberry Jams is CamJam, and they had a special birthday event at Makespace Cambridge where folks were showing off their projects, chatting Raspberry Pi, and there was even a talk by Eben himself.

Here's to many more years of Raspberry Pi.

▼ Train sets are breaking containment from the Maker Monday pages





▲ Raspberry Pi A600 is a nod to the original computer that inspired the naming of the full computer types

◀ With semi-limited pixels comes games that are a reminder of days gone by

Crowdfund this

Crowdfunding campaigns to keep an eye on

Open Stack

This is an IoT HAT for Raspberry Pi that can also run on its own once disconnected from Raspberry Pi. It supports 4G, GNSS satellite connection, and Bluetooth without needing to be connected to Raspberry Pi either, making it an interesting hybrid product. It's already hit its funding goal at time of writing too.

▶ **rpimag.co/openstack**

Flipo

After the release of Raspberry Pi Pico, we often saw the question: "should I use a Pico or ESP32?" Fret no longer, as with the Flipo you can use both. Just flip from one side to the other! This Kickstarter isn't live yet, but we're interested to see how it comes along!

▶ **rpimag.co/flipo**

Your Letters

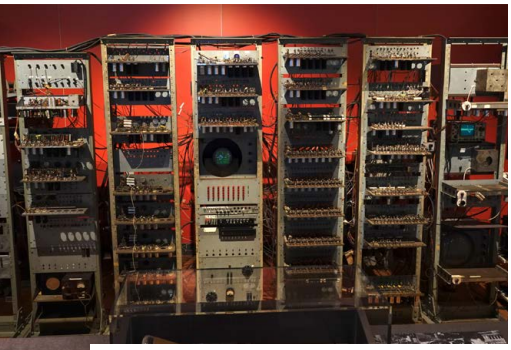
Magic

The glowing orb/magic sphere in issue 164 is brilliant! Quite literally, as it glows! I've used potentiometers before, for gain and volume controls in analogue circuits, and I've just about got my head around using a potentiometer with an analogue-to-digital converter (ADC) to take the varying resistance of a potentiometer and turn it into a digital signal that MicroPython can make sense of. But I've never used a potentiometer as the axis of a moving part, and taken the position of that potentiometer to control the brightness of a glowing orb. Hats off to whoever made that; it must have been a real eureka moment when they realised that they could do it.

William, via email

▶ Rotate the two halves of the sphere to control the brightness of the internal LEDs

It's very clever, isn't it? What's almost as impressive as the original idea is the quality of the execution: it's absolutely flawless. We want one!



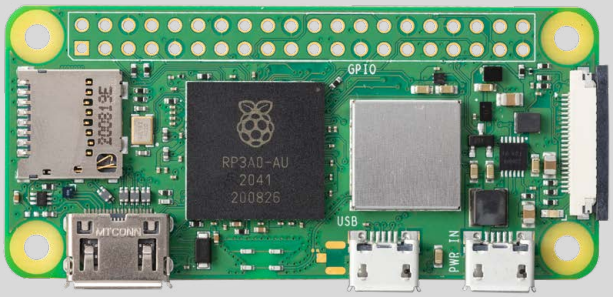
▲ While you're in Manchester, why not visit the Britons Protection public house, and have a gander at its tiled artwork depicting the Peterloo Massacre?

Manchester, Baby!

You've reached the Manchester Baby in your *Computers That Made the World* series, so I just have to point out that you can see a replica of the Baby, built in 1998 by volunteers with the help of some of the original designers, at the Science and Industry Museum in Manchester. If you visit on the right day, you can even see it working.

Emily, via email

We have access to so much good stuff, for free, on these islands that it's easy to take for granted. Italy may have been the home of the Renaissance, the USA is the birthplace of the Atomic Age, but we have legitimate claim to be the birthplace of the computer age. It's been a privilege to publish extracts from *The Computers that Made Britain* – not only is it a good read, it's an insight into how much individuals can still make advances in a field that's dominated by big companies. Our advice: start by building something small, then work up to something bigger.



▲ Not every project needs all-new components – you'll find your next build much easier if you reuse bits from your last one

Reuse, recycle

Something Rob wrote in last issue struck a chord with me, about the recycling of mechanical parts in the rides at Disney World: if you've got A Thing, it dramatically increases your odds of developing Another Thing. When you develop a product, you're not just building the widget itself, you're building tools, a skill set, a supply chain, an inventory of parts that presumably gets cheaper per unit the more units you build. It makes sense for companies but it's also true for individuals – I made a greenhouse monitor last year, and I've still got a load of wires, sensors etc. that I used because it just made sense to buy two of everything. All I need is an extra Raspberry Pi Zero W and a battery and I'd be able to knock another one up in less than half the time it took me to build the first one. The rides at Disney World may have nothing to do with the world of Raspberry Pi, but manufacturing is manufacturing whatever the scale, and we can learn the same lessons.

Paul, via email

Forgive the apparent tangent, but we were thinking something similar about the Po Valley in Italy: thanks to geography, it was a good place for humans to grow wheat and raise cattle; then it became a good place to make cheese. Fast-forward a few iterations of human activity and it's now one of the best places in the world to build supercars and helicopters. We build on what's gone before; that includes the components we have left over, but most importantly it's the knowledge of what we learned when building the project before last, even if it failed.

Contact us!

- @rpimagazine
- @rpimag
- rpimag.co/facebook
- magazine@raspberrypi.com
- forums.raspberrypi.com

Community Events Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

01. Breaking Breadboards Brighton

- **Wednesday 22 April**
- **Freedom Works Brighton, Brighton, UK**
- **rpimag.co/bbb164**

The Breaking Breadboards team are excited to announce their April 22nd Event at Freedom Works in Brighton. This is the perfect opportunity for tech hardware enthusiasts in Brighton and beyond to connect, and share their passion for all things tinkering!

02. Cornwall Tech Jam

- **Saturday 25 April**
- **FibreHub, Pool, UK**
- **rpimag.co/ctj164**

Each month at Cornwall Tech Jam, join a welcoming, hands-on space for young people to explore the fascinating world of technology and coding. Whether they're just starting or have some experience, children and teens can dive into exciting coding challenges, build their own projects, and work with tech gear that's not always accessible at home.

03. Raspberry Pint

- **Tuesday 28 April**
- **Duke Of Sussex Waterloo, London, UK**
- **rpimag.co/hrp164**

A hybrid event taking place online as well as at the venue, it features presentations about building personal or professional projects with Raspberry Pi, Arduino, ESP32, micro:bit, etc. The organisers also welcome presentations about skills and techniques such as website design, PCB design, software development, 3D printing, soldering, etc.

04. Hobby-X

- **Thursday 30 April to Sunday 3 May**
- **Kyalami Grand Prix Circuit, Midrand, South Africa**
- **rpimag.co/hobbyx26**

Visit Raspberry Pi Approved Reseller PiShop at Hobby-X to get hands-on with some of the latest Raspberry Pi products. Each year, a wide spectrum of materials, equipment, supplies, and ideas are showcased to the general public, encouraging South Africans to take up a creative craft or hobby.

FULL CALENDAR
Get a full list of upcoming community events here:
rpimag.co/events

05. International Drone Show 2026

Official Raspberry Pi Event

Wednesday June 3 to Thursday 4 June
HCA Airport, Odense, Denmark
rpimag.co/ids26

The Raspberry Pi team is delighted to be exhibiting at the International Drone Show in Odense, Denmark, in June. There, you'll be able to meet the team and see a range of Raspberry Pi technology. You'll see how individuals and businesses in the UK and around the world use Raspberry Pi to support their projects and applications, and discover how Raspberry Pi can help you with your own solutions.

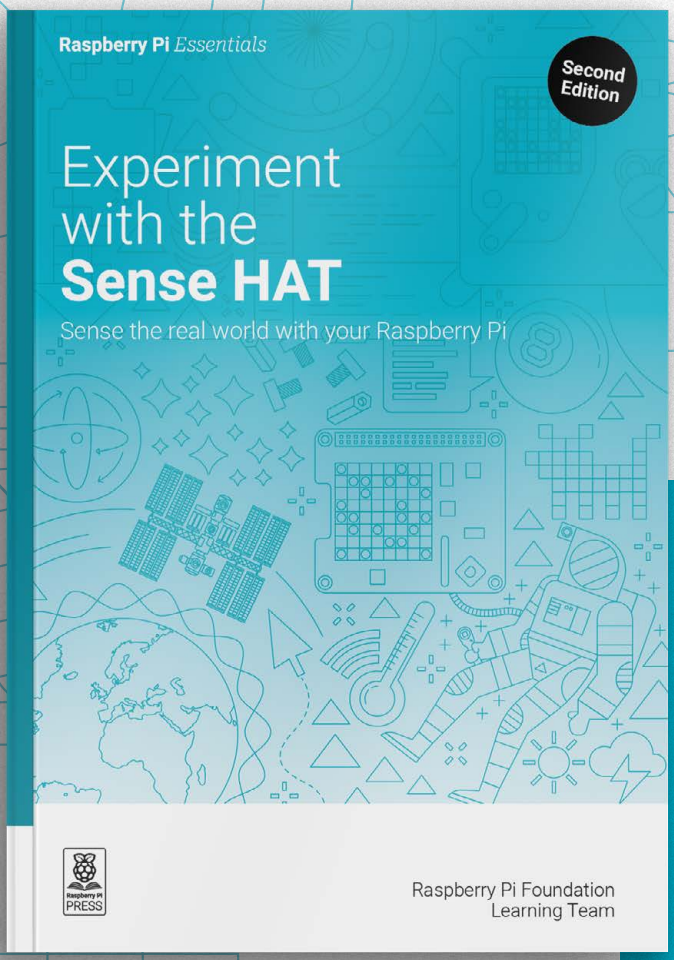
Win 1 of 5 256GB Raspberry Pi Flash Drives

The new Raspberry Pi Flash Drive is no ordinary flash stick – as well as including high-capacity storage, it has smart reading and writing features that make it work faster and better overall compared to other portable storage out there.

Head here to enter:
rpimag.co/win
Learn more:
rpimag.co/flashdrive

Terms & Conditions

Competition opens on **25 March 2026** and closes on **30 April 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.



The Sense HAT is an incredibly versatile and flexible bit of kit with plenty of obvious uses, along with a huge number of less obvious ones, that you'll love to make and share. Updated for the latest Raspberry Pi devices and hardware, this book has everything you need to get started.

- ***Getting started with Sense HAT***
- ***Learn by building:***
  - *A digital twist on the Magic 8 Ball*
  - *Your own interactive pixel pet*
  - *A sparkly light show*
  - *An environmental data logger*
  - *Flappy Astronaut, a low-res, high-fun video game*

BUY ONLINE: **rpimag.co/sensehatbook**

REACH A GLOBAL COMMUNITY

Advertise in **Raspberry Pi Official Magazine**

Our readers are passionate about technology and the Raspberry Pi ecosystem. From DIY enthusiasts to professional engineers.

Your advertisement can sit alongside our cutting-edge tutorials, features, and reviews. And we have flexible advertising options for a range of different businesses.

Take the next step –
advertise with us today!

Email Charlie Milligan at: **charlotte.milligan@raspberrypi.com**

Official Magazine
#165

run a raspberry pi cloud server

Get your data out of tech giants' hands with Raspberry Pi

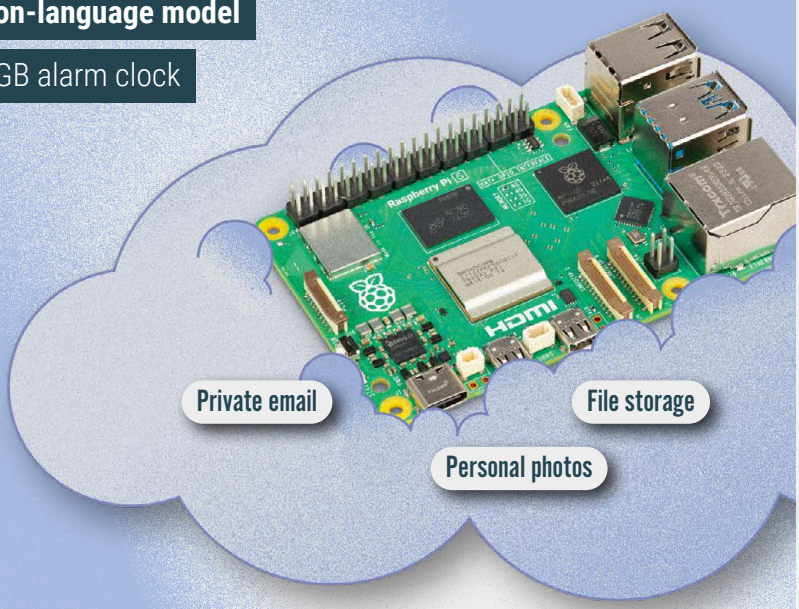
On sale 30 Apr

PLUS!

The best music projects

Run a vision-language model

Build an RGB alarm clock



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, Nicola King, Phil King, Andrew Lewis, Rob Miles, KG Orphanides, 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

ISSN: 2977-4403 (Print)
ISSN: 2977-4411 (Digital)

Cheap is a feature

Proper allocation of one's fun vouchers is an art form, says **Andrew Gregory**

I had a bit of a shock this month. It was my own fault really: after talking to Toby Roberts, Raspberry Pi's in-house maker of awesome things, about his BBC Micro tribute, I thought I'd install an emulator on my Raspberry Pi 500+ and enjoy some retro computing. Not a BBC Micro – although I can remember using one at school, I didn't get deep enough into programming to have any sort of nostalgia for that machine. No, I wanted to recreate another old system – the Sega Mega Drive, courtesy of the Sega Genesis Collection of legal ROMs available via Steam. There's something about pretending to be a super-fast blue hedgehog that brings a nostalgic tear to my eye, so why not install RetroPie and see if after all these years I've still got the skill to go one-on-one with Dr Robotnik?

> *Linux reminded me that it's there with a big, beautiful black screen of failure*

The RetroPie website clearly recommends downloading the OS image to an SD card and installing it as your operating system. I thought I knew better, and tried instead to pick and choose the packages I would need without reinstalling Raspberry Pi OS. I've been using Raspberry Pi OS for so long now that I almost forgot that it's Linux. Happily, Linux reminded me that it's there with a big, beautiful black screen of failure.

Everything is fixable

The first time I managed to install over the Linux bootloader (for this is what I reckon I must have done), I was terrified: it was on a laptop that took me three months to save up for, and I'd rendered it useless, with no alternate device on which I could search Google to find out what I'd done and how to fix it. This time though, I had at my disposal Raspberry Pi 5, Raspberry Pi 4, and a phone, so I could keep working, and work out what I needed to do to fix the broken computer with no – or at least, minimal – stress.

There's at least one lesson here: follow the instructions. The developers of software X, Y, or Z know best, and I should just do what they say. What on earth was I going to achieve by installing a load of separate packages, each with their own dependencies stretching back into the mists of versions past?

But the other lesson is that the more affordable the machine you're working on, the more you can afford to take risks, and taking risks is how we learn. A broken Linux install on a Raspberry Pi 1GB that cost £40 is a learning opportunity; a broken bootloader on a £1500 MacBook is the stuff of nightmares. I've always thought that affordability is an essential feature, but really it's the one essential feature that makes everything else possible. That's true whether you're learning hardware or software, by the way – if you're not sure whether an idea you've had is going to work, you're best off prototyping it as cheaply as you can, and much as I love my 500+, there are better options for experimentation. And perhaps most importantly, with the money you save on computing hardware, you can afford a joystick with big clicky buttons that makes you feel like a fighter pilot.

Andrew Gregory – Author
Andrew has recently discovered the benefits to his posture of sitting up straight and using a proper keyboard like a grown-up.

rpimag.co

HiPi.io

HighPi 5S

The new case from the HiPi team

Reliable case for complex projects:

- Built for Raspberry Pi 5
- Rapid tool-free assembly and disassembly
- Large internal volume for HATs even with the active cooler installed
- Camera cable slot to connect to external cameras
- Power button and external status LED
- Passive & active cooling options
- Secure microSD cover
- VESA mount support

Customization options for volume orders:

- Output ports molded to your exact specs
- Logo printing for a branded finish

Available at these great Pi stores:

PiShop.us PiShop.ca ThePiHut Pi-Shop.ch Welectron.

Contact your favorite Pi store if it's not listed here

PiKVM

Remote control **redefined**

Manage your servers or PCs remotely!

Powered by Raspberry Pi

PiKVM V4 Mini

Small, cost-effective, and powerful!

- Power consumption in idle mode: just 2.67 Watts!
- Transfer your mouse and keyboard actions
- Access to all configuration settings like UEFI/BIOS
- Capture video signal up to 1920x1200@60 Hz
- Take full control of a remote PC's power

PiKVM V4 Plus

The most feature-rich edition

- More connectivity
- Extra storage via internal USB3.0
- Upgraded powering options
- More physical security features
- Extra HDMI output
- Advanced cooling solution

A cost-effective solution for data-centers, IT departments or remote machines!

Available at the main Raspberry Pi resellers

core ELECTRONICS · PiShop.ca · RPishop.cz · Welectron. · elektorstore · PIMORONI

Pi-Shop.ch · The PiHut · PiShop.us · CloudFree · KUBII · Cytron Marketplace

ROBU.IN Your ideas, Our Parts · ANTOP COMPUTERS & ELECTRONICS · netsket mall · DIREKTRONIK · RaspberryPi.dk

HiPi.io

No reseller in your country?
Check shop.hipi.io (import fees might apply).

List of official resellers by country: