Hear the world with **a camera kit & Azure AI language module**

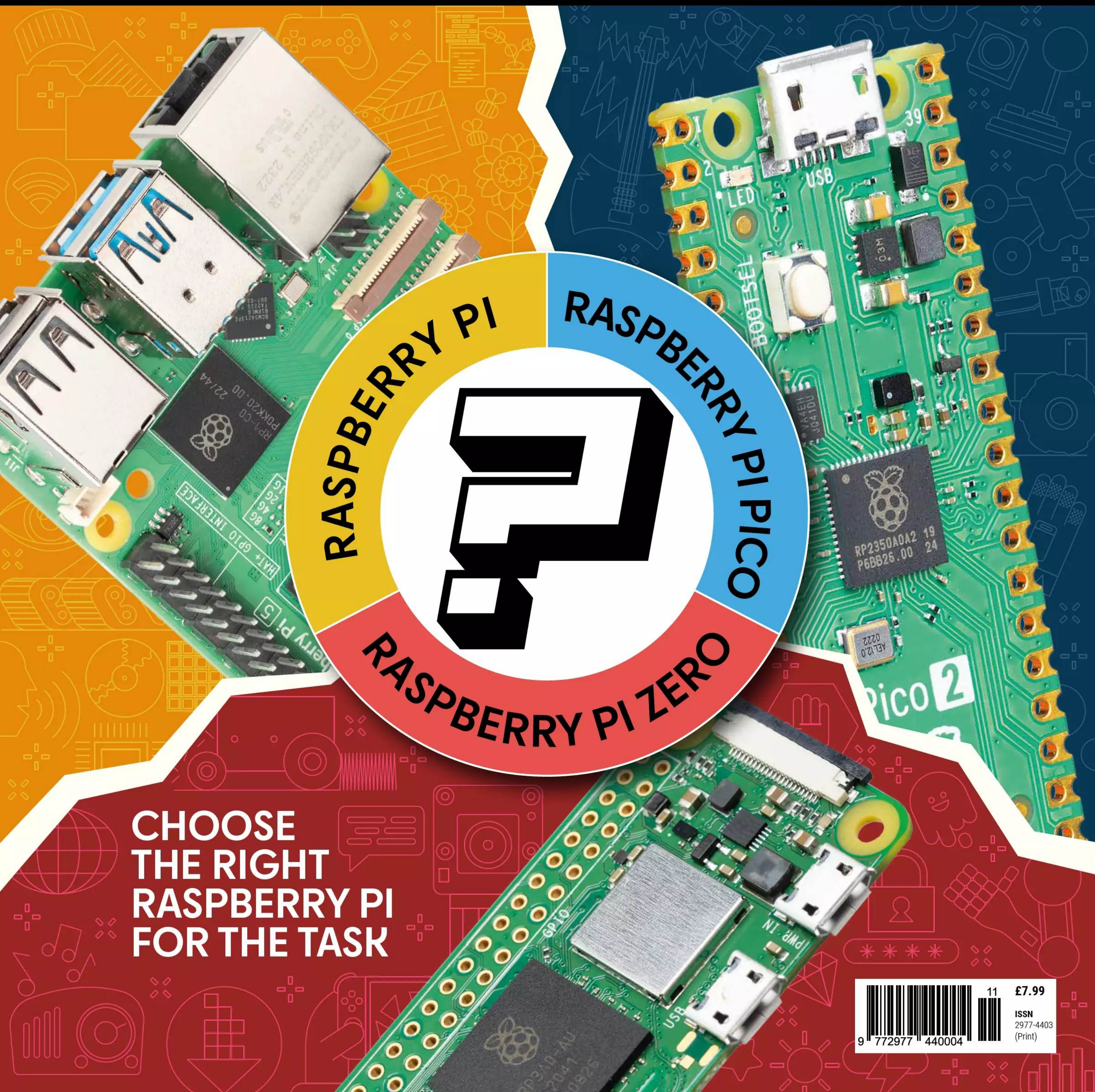
Get ready for **Christmas lights with NeoPixels**

Behind the scenes **with Raspberry Pi OS Trixie**

Raspberry Pi

Official Magazine
#159 | November 2025

HEAD TO HEAD

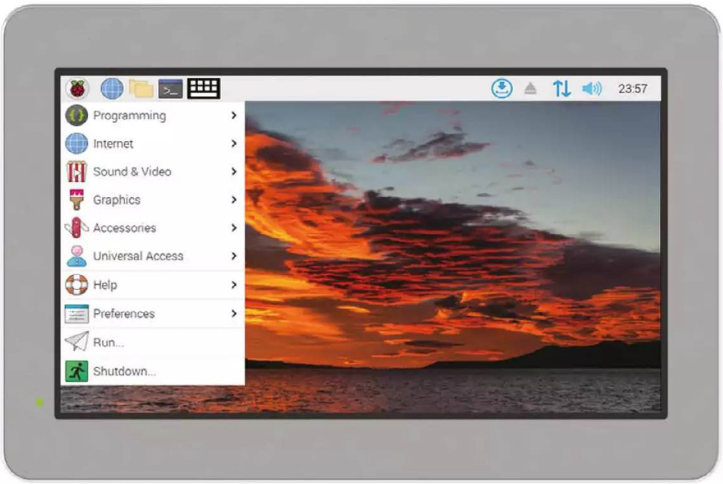


CHOOSE THE RIGHT RASPBERRY PI FOR THE TASK

£7.99
ISSN 2977-4403 (Print)
11
9 772977 440004

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
Andy Bryant just dropped a huge box of chocolate bars on the team and Lucy's main to-do this month is to go for more runs.

rpimag.co

Raspberry Pi started out with just 10,000 boards and an aim to encourage computer science take-up amongst students. Starting from humble beginnings, the little computer become a true British success story. Raspberry Pi now has a whole range of devices to choose from: there's a board for everyone.

Choosing the right Raspberry Pi for your project can feel like being a kid in a sweet shop. This month, Ben's Christmas lights project is based around Raspberry Pi Pico, while KG's x86_64 emulation build uses Raspberry Pi 5. If you want to get started with GPIO, a Raspberry Pi Zero 2 WH is ideal.

I've been using Raspberry Pi 500+ with 16GB RAM to test out AI coding and Rob's been using one to power up Trixie, the latest version of Raspberry Pi OS.

Most Raspberry Pi computers are so versatile they can turn a hand to almost anything, but you should know each one's strengths. This month Phil put together a detailed head-to-head that pits Raspberry Pi Model B against Zero and Pico. This guide will help you analyse each model and pick the right one for your project.

There's a lot to do this month. It'll be Christmas before you know it, so let's get our projects ready.

Lucy Hattersley – Editor

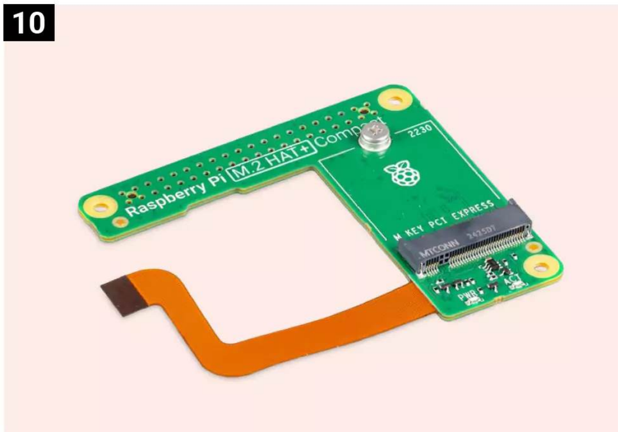
STATEMENT OF OWNERSHIP, MANAGEMENT, AND CIRCULATION (required by Act of August 12, 1970: Section 3685, Title 39, United States Code). 1. Raspberry Pi. 2. (ISSN 2977-4403). **3. Filing Date**: September 10, 2025. **4. Issue Frequency**: Monthly. **5. Number of Issues Published Annually**: 12. **6. Annual Subscription Price**: $80. **7. Complete Mailing Address of Known Office of Publication**: Publishers Service Associates 2406 Reach Road Williamsport, PA 17701. **8. Complete Mailing Address of Headquarters or General Business Office of Publisher**: Raspberry Pi 194 Science Park Cambridge, CB4 0AB UK. **9. Full Names and Complete Mailing Addresses of Publisher, Editor, and Managing Editor. Publisher**: Raspberry Pi 194 Science Park Cambridge, CB4 0AB UK; **Editor**: Lucy Hattersley Raspberry Pi 194 Science Park Cambridge, CB4 0AB UK; **Managing Editor**: Lucy Hattersley Raspberry Pi 194 Science Park Cambridge, CB4 0AB UK. **10. Owner**: Raspberry Pi Ltd 194 Science Park Cambridge, CB4 0AB UK. **11. Known Bondholders, Mortgagees, and Other Security Holders Owning or Holding 1 Percent or More of Total Amount of Bonds, Mortgages, or Other Securities**: Raspberry Pi Holdings PLC 194 Science Park Cambridge, CB4 0AB UK. **12. Tax Status**: Has Not Changed During Preceding 12 Months. **13. Publication Title**: Raspberry Pi. **14. Issue Date for Circulation Data Below**: September 25, 2025. **15. Extent and Nature of Circulation: A. Total Number of Copies (Net press run)**. Average No. Copies Each Issue During Preceding 12 Months: 2312. No. Copies of Single Issue Published Nearest to Filing Date: 2608. **B. Paid Circulation. 1. Mailed Outside-County Paid Subscriptions**. Average No. Copies Each Issue During Preceding 12 Months: 1320. No. Copies of Single Issue Published Nearest to Filing Date: 1509. **2. Mailed In-County Paid Subscriptions**. Average No. Copies Each Issue During Preceding 12 Months: 0. No. Copies of Single Issue Published Nearest to Filing Date: 0. **3. Paid Distribution Outside the Mails Including Sales Through Dealers and Carriers, Street Vendors, Counter Sales, and Other Paid Distribution Outside USPS**. Average No. Copies Each Issue During Preceding 12 Months: 366. No. Copies of Single Issue Published Nearest to Filing Date: 394. **4. Paid Distribution by Other Classes of Mail Through the USPS**. Average No. Copies Each Issue During Preceding 12 Months: 0. No. Copies of Single Issue Published Nearest to Filing Date: 0. **C. Total Paid Distribution**. Average No. Copies Each Issue During Preceding 12 Months: 1686. No. Copies of Single Issue Published Nearest to Filing Date: 1903. **D. Free or Nominal Rate Distribution. 1. Free or Nominal Rate Outside-County Copies**. Average No. Copies Each Issue During Preceding 12 Months: 6. No. Copies of Single Issue Published Nearest to Filing Date: 6. **2. Free or Nominal Rate In-County Copies**. Average No. Copies Each Issue During Preceding 12 Months: 0. No. Copies of Single Issue Published Nearest to Filing Date: 0. **3. Free or Nominal Rate Copies Mailed at Other Classes Through the USPS**. Average No. Copies Each Issue During Preceding 12 Months: 0. No. Copies of Single Issue Published Nearest to Filing Date: 0. **4. Free or Nominal Rate Distribution Outside the Mail**. Average No. Copies Each Issue During Preceding 12 Months: 0. No. Copies of Single Issue Published Nearest to Filing Date: 0. **E. Total Free or Nominal Rate Distribution**. Average No. Copies Each Issue During Preceding 12 Months: 6. No. Copies of Single Issue Published Nearest to Filing Date: 6. **F. Total Distribution**. Average No. Copies Each Issue During Preceding 12 Months: 1692. No. Copies of Single Issue Published Nearest to Filing Date: 1909. **G. Copies not Distributed**. Average No. Copies Each Issue During Preceding 12 Months: 626. No. Copies of Single Issue Published Nearest to Filing Date: 705. **H. Total**. Average No. Copies Each Issue During Preceding 12 Months: 2318. No. Copies of Single Issue Published Nearest to Filing Date: 2614. **I. Percent Paid**. Average No. Copies Each Issue During Preceding 12 Months: 99%. No. Copies of Single Issue Published Nearest to Filing Date: 99%. **Publication of Statement of Ownership**: will be printed in the October 23, 2025 issue of this publication. **Signature and Title of Editor, Publisher, Business Manager, or Owner**: Brian Jepson, Publishing Director. I certify that all information furnished on this form is true and complete. I understand that anyone who furnishes false or misleading information on this form or who omits material or information requested on the form may be subject to criminal sanctions (including fines and imprisonment) and/or civil sanctions (including civil penalties).

Contents

World of Raspberry Pi

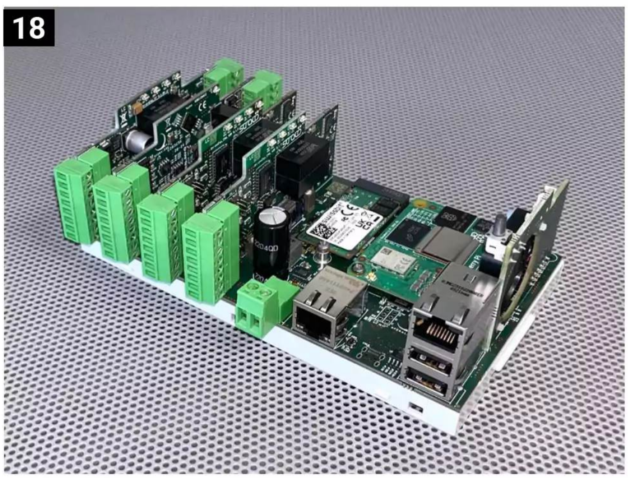
Project Showcase

10



16

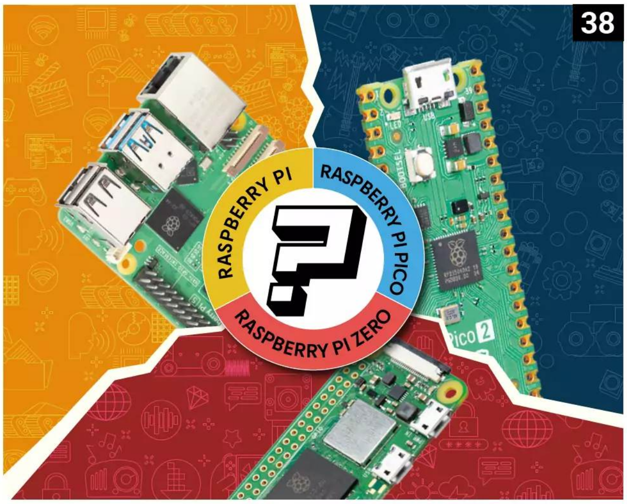
18



Raspberry Pi is published monthly by Raspberry Pi Ltd., 194 Cambridge Science Park, Milton Road, Cambridge, England, CB4 0AB. Publishers Service Associates, 2406 Reach Road, Williamsport, PA 17701 is the mailing agent for copies distributed in the US. Periodicals Postage paid at Williamsport PA. Send address changes to Raspberry Pi c/o Publishers Service Associates, 2406 Reach Road, Williamsport, PA 17701.

Top Projects

Feature



Tutorials

Feature


DigiKey
DigiKey
DigiKey

Reviews

Raspberry Pi Community

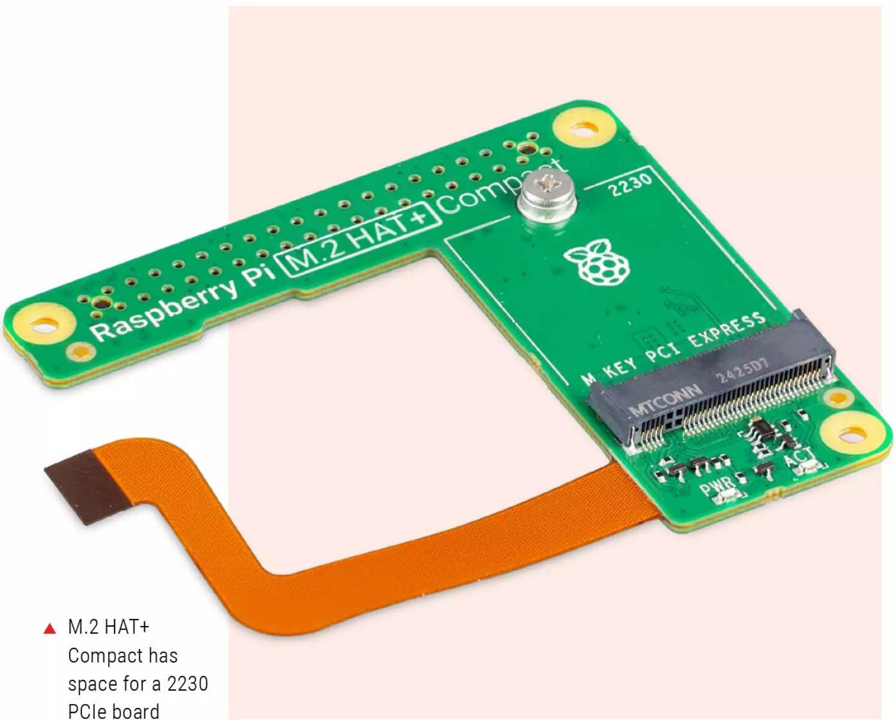
Competition

Win 1 of 10 M.2 HAT+ Compact

Disclaimer: Some of the tools and techniques shown in Raspberry Pi Official Magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. Children should be supervised. Raspberry Pi Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in Raspberry Pi Official Magazine. Laws and regulations covering many of the topics in Raspberry Pi Official Magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in Raspberry Pi Official Magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

M.2 HAT+ Compact

Smaller M.2 HAT+ that fits in Official Case available now. By **Lucy Hattersley**



▲ M.2 HAT+ Compact has space for a 2230 PCIe board

A new, smaller M.2 HAT+ Compact designed to fit in the official case was announced by Raspberry Pi on 22 October 2025.

Writing on the blog, Eben Upton, Raspberry Pi CEO and co-founder said: "The Compact variant replaces the FPC cable and socket with a single flex-rigid PCB; this is the first time we've used this technology, and we're rather pleased with how it's turned out."

Express connection

Raspberry Pi 5 was the first Raspberry Pi product to feature a PCI Express (Peripheral Component Interconnect) interface. To save space, this was exposed on a small FPC (Flexible Printed Circuit) connector. "Soon after launch, we saw a variety of third-party peripheral boards, as well as adapters to allow users to connect their own M.2 form-factor PCI Express cards," said Upton.

▲ M.2 HAT+ Compact mounted on Raspberry Pi 5

A firm personal favourite since I got my prototype unit earlier in the year

"When we launch a flagship product, it can be hard to know exactly which new feature will catch people's imaginations," explained Upton. The original M.2 HAT+, launched in May 2024, forms the basis for AI Kit and SSD Kit products.

The original M.2 HAT+ is "a great solution when using your Raspberry Pi 5 uncased or in a larger enclosure," said Upton, "but while you can use it with our Active Cooler (**rpimag.co/activecooler**), it doesn't fit into the official case (**rpimag.co/case**) with the fan installed."

To help M.2 HAT+ Compact fit into the official case alongside the fan insert, it replaces the FPC cable and socket with a single flex-rigid PCB. "This is the first time we've used this technology," said Upton, "and we're rather pleased with how it's turned out."

"The new M.2 HAT+ has been a firm personal favourite since I got my prototype unit earlier in the year. A cased Raspberry Pi 5, with even faster I/O performance, and without compromising aesthetics: we hope you like it."

You can get the M.2 HAT+ Compact for $15 from Raspberry Pi resellers, see the product page for more information: **rpimag.co/m2hatplus.**

▲ The compact board design enables you to house a Raspberry Pi 5, M.2 HAT+ Compact, and SSD inside the official case with the fan fitted

Azure Hear The World Project

A keen photographer's aid for visually impaired users gets Raspberry Pi to describe what the camera sees, learns **Rosie Hattersley**

Maker

Marco Gerber

Software engineer Marco is an expert in cloud architecture and a Python coder, but a relative newcomer to Raspberry Pi.

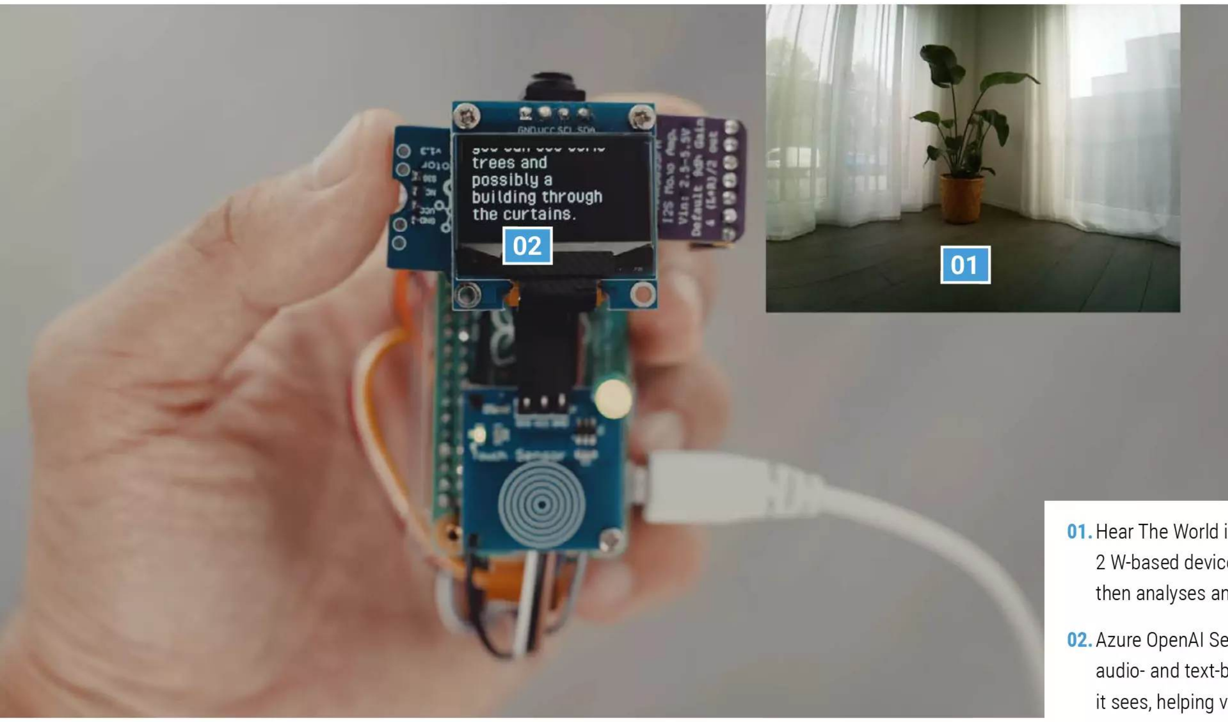
rpimag.co/heartheworldgit

Swiss software engineer and cloud architecture specialist Marco Gerber enjoys getting hands-on with tech in his free time as well as at work. His expertise in machine learning led him to recognise the potential for a handheld AI device that could describe the surrounding environment and that could be particularly useful as an accessibility tool for people with hearing or visual impairments. His idea was to develop a "point and shoot" device that would provide an audio description of the user's surroundings. The result was the Raspberry Pi Zero 2 W-based Hear The World device, which offers both text and spoken descriptions alongside haptic feedback.

His idea was to develop a "point and shoot" device that would provide an audio description of the user's surroundings

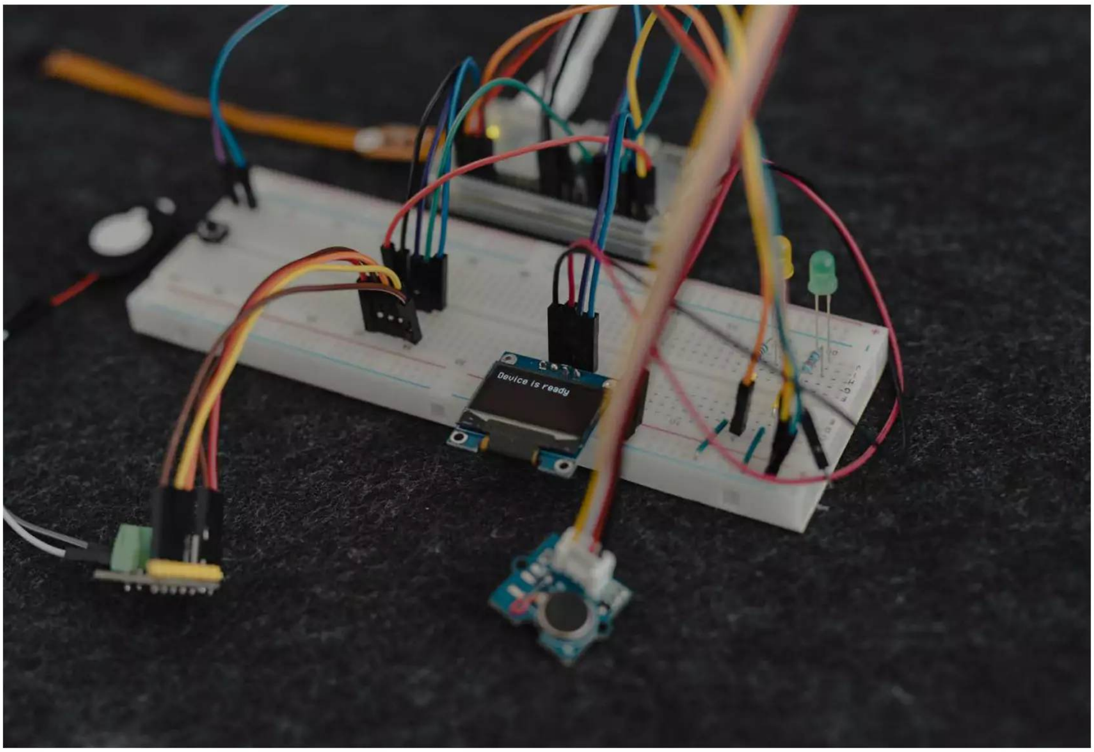
Instant inspiration

Zurich-based Marco is a Microsoft MVP with a focus on AI and Azure, so cloud-based projects are second nature. He enjoys trying out new technologies, "Raspberry Pi and various sensors being some of my favourite ways to combine hands-on tinkering with learning". His other passions are travel and photography. With a keen visual eye, Marco found himself experimenting with early multimodal AI models capable of analysing images. This led on to interpreting individual images from a webcam, sequences of frames and, eventually, a form of video analysis in which the AI processes information about its surroundings. "Later, I also explored audio models, creating small pipelines



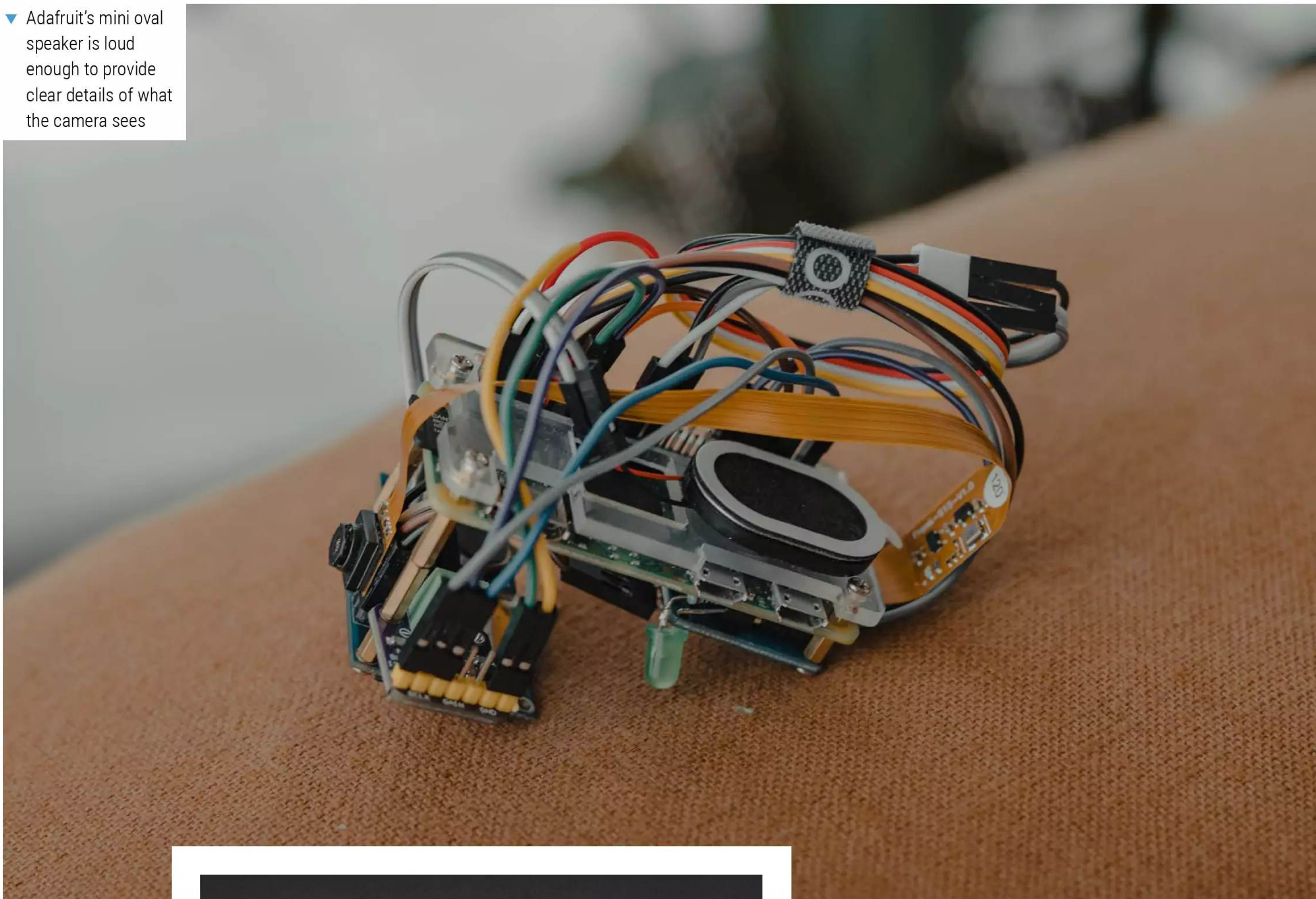
01. Hear The World is a Raspberry Pi Zero 2 W-based device that photographs a scene, then analyses and describes it

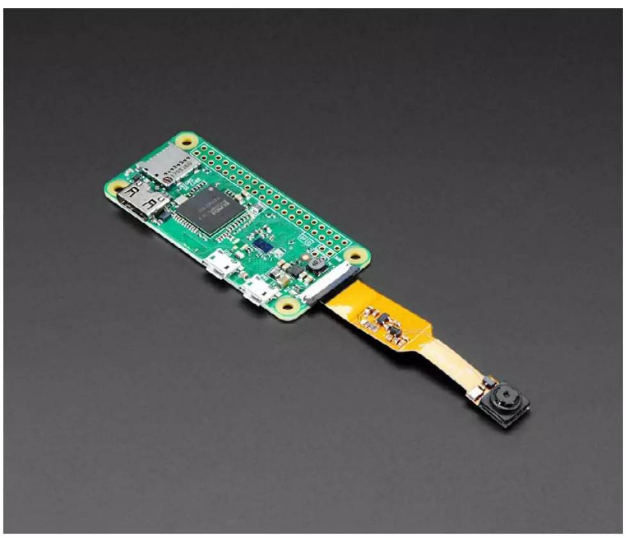
02. Azure OpenAI Service (GPT-4) provides audio- and text-based descriptions of what it sees, helping visually impaired users



◀ Marco's Azure AI image recognition device began with a breadboard and testing each component in turn

Adafruit's mini oval speaker is loud enough to provide clear details of what the camera sees





Attaching the camera to Raspberry Pi Zero 2 W

where an image could be transformed into text, and that text into audio – essentially image-to-text-to-audio."

Having got this far, it struck Marco that Raspberry Pi could serve as the perfect platform for an image to text to audio project. "With its camera port and compact form factor, it opened the possibility of building a handheld device that could describe the surrounding environment – something particularly useful as an accessibility tool for people with hearing or visual impairments," he says. Having a tangible use case in mind motivated him to push the idea further.

Marco was already an enthusiast of both Python and Raspberry Pi thanks to his AI work and "the countless great libraries" that he could use in his project. The compact dimensions of Raspberry Pi Zero 2 W seemed ideal for his portable project. However, most aspects of the project were new territory for him, "so it was very much a process of learning by doing".

Getting the speaker working with Raspberry Pi required a DAC (digital-to-analogue converter), but also an amplifier to boost the resulting file enough to be audible. "Each component had its own quirks – different libraries, different formats, and unique requirements – which added complexity but also made the project exciting."

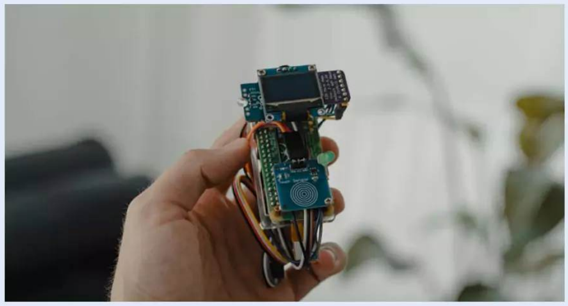
Conclusion

Overall, Marco is pleased with his foray into assistive technology, completed as and when over a couple of months in his free time. "It could be adapted to transcribe spoken words and display them for hearing-impaired people, or even serve as an interpreter to translate sign language into spoken language," he suggests.

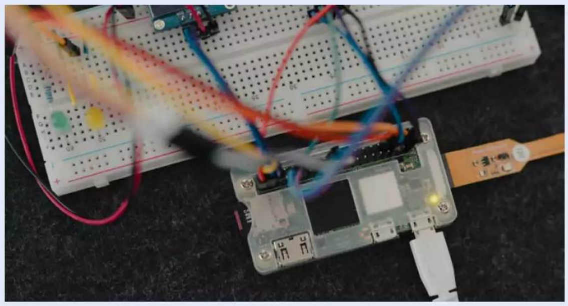
Quick FACTS

- Marco says he is "not very skilled" at manufacturing or 3D printing
- Raspberry Pi's modular setup meant he could progress without them
- His project could be adapted to display spoken words on a screen
- He believes his project could work well for transcribing sign language
- It could even speak signed language aloud

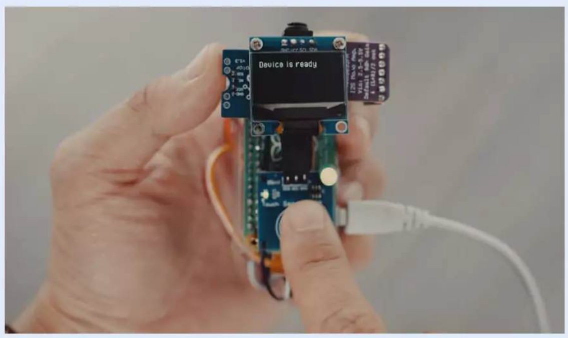
Survey the scene



1. Pressing a large button on The Hear The World device results in a wide-angle photo being taken by the Adafruit Zero Spy Camera and analysed by Azure GPT 4.0.



2. Raspberry Pi Zero 2 W or a standard Raspberry Pi can be used to control everything and process the data. An LED screen attached via GPIO provides a scrolling description of what the camera has photographed.

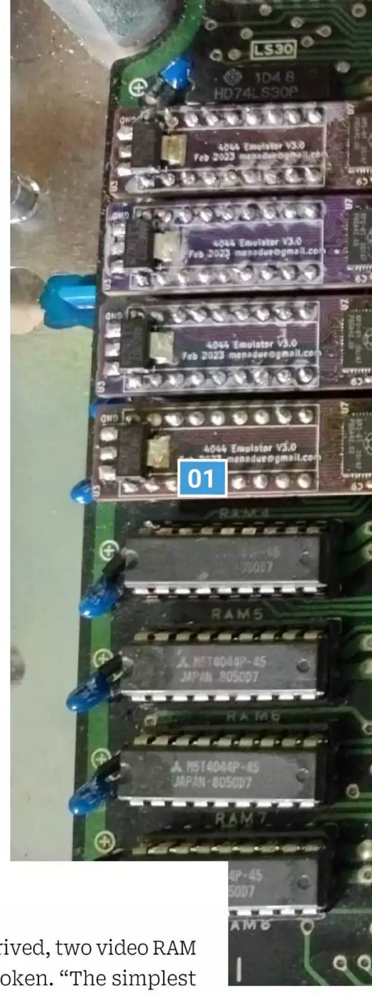


3. Azure AI Speech generates a description that is played through the prototype gadget's tiny speaker. A subtle vibration provides an alert that the device is about to make an announcement.

Casio FX9000P RAM replacement

A broken vintage computer has been brought back to life thanks to RP2040.

By **David Crookes**



Maker

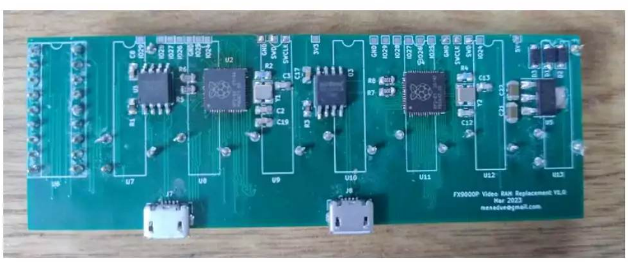
Andrew Menadue

Works with embedded firmware by day and resurrects and recreates vintage calculators and computers for fun.

rpimag.co/menadue

When most people think of Casio, calculators and watches usually spring to mind. But, over nearly 80 years, this Japanese electronics company has developed many other devices including computers such as the Casio FX9000P. Powered by a Z80 processor and coming with built-in BASIC, the FX90009P is highly sought after, despite fetching high prices at auction. Many surviving units, however, require careful restoration.

This didn't deter Andrew Menadue. "Being a big Casio fan, I decided to buy one, partly to see what it was like and partly to add to my collection," he says. "I knew from the auction photos that vertical lines were appearing on the screen, but I was fairly sure that video RAM would be the problem and I was confident about a repair."



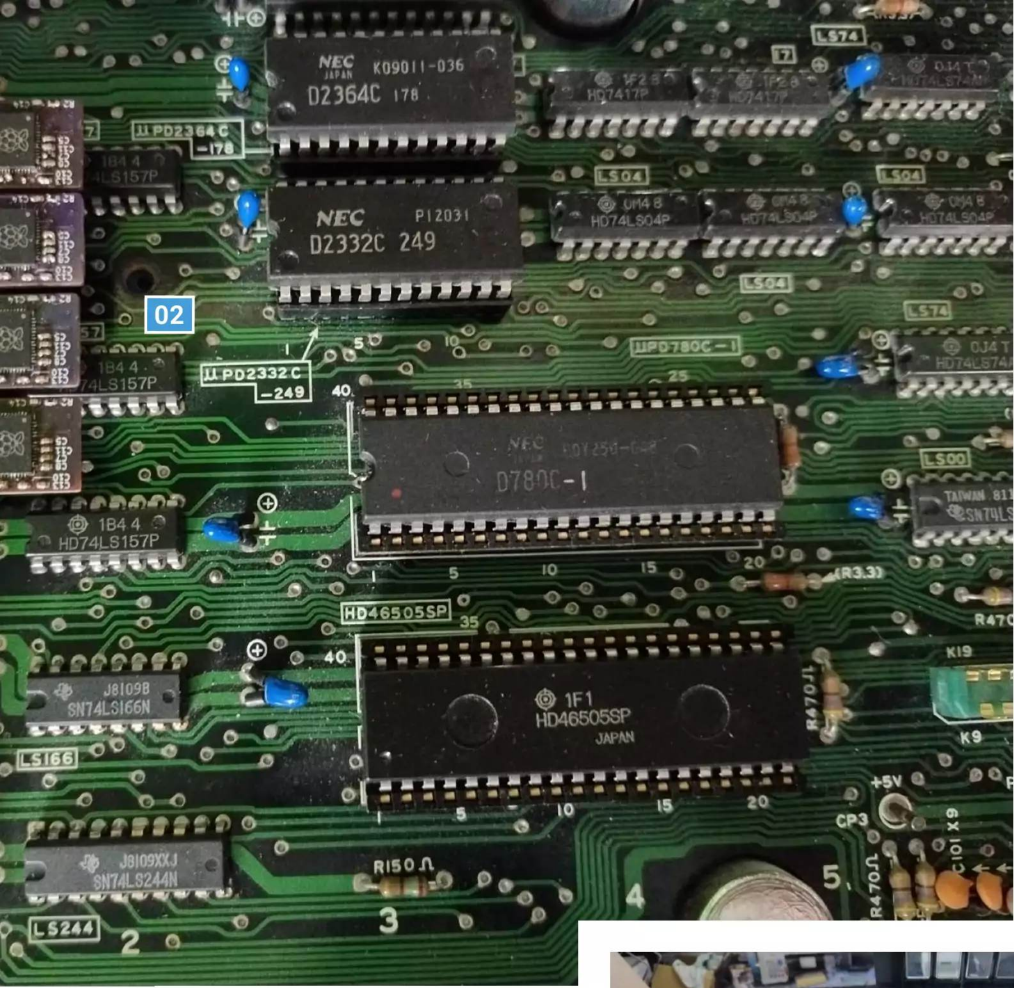
▶ This PCB board is used to replace the FX9000P's video RAM in its entirety. It plugs into all of the RAM sockets

Chips are down

When the machine arrived, two video RAM chips were indeed broken. "The simplest way to fix them would be to replace the faulty ICs, but new ICs aren't available, so replacements – costing about £20 each – would be old and probably close to failure themselves." Andrew decided to look at the RP2040 microcontroller chip as a possible alternative.

> *I was fairly sure that video RAM would be the problem*

"I'd used the RP2040 before in Pico form in other projects and had bought a supply of RP2040 ICs for embedding on PCBs," he recalls. "I contemplated building a replacement using a RAM chip, but the problem with that is the size of the original ICs. There just wasn't room. The RP2040 is small and would just about fit on a PCB that sat in the footprint of the original RAM chip."



01. The video RAM replacement PCBs were designed by Andrew

02. Each one is placed on the FX9000P board and contains an RP2040 microcontroller chip

The Casio FX9000P computer has the characteristics of Casio's calculators and BASIC programmables

Quick FACTS

- The only RAM inside an FX9000P is video RAM
- Programs are stored on plug-in cartridges
- Andrew has used RP2040 to replace the video RAM
- Original RAM is hard to find
- The all-in-one PCB cost £20 to develop

Andrew made a couple of prototype RAM replacements using Raspberry Pi Pico microcontroller boards connected with wires. "I used these to write the code and verify everything would work," he says. "One problem was that the RP2040 is a 3V3 IC and the FX9000P uses standard TTL logic running at 5V. Luckily, the FX9000P seems to run its 5V supply a bit low, which I have seen on other vintage machines, and I decided not to fit level shifters."

Technology for life

The code, written with the C/C++ SDK, proved straightforward enough. "The program is quite small and is basically a loop," Andrew reveals. "The address lines are sampled on chip select and data is presented on the data lines. Data is stored in the RP2040 RAM and plenty is available for this task."

Once everything was working, Andrew designed a small PCB to fit the footprint of a FX9000P RAM chip. "I then removed all the video RAM and put sockets in, then ran the system with two RP2040 replacements and six original RAM chips," he says.

Later, as more video RAM ICs failed, Andrew decided to ditch the small PCBs in favour of a larger one that fits over all of the ICs and replaces the whole video RAM. "This was made with two RP2040 ICs and runs similar code to the small PCB, but each RP2040 provides four bits of data instead of just one."

This also provided an additional benefit. "The 'all in one' video RAM PCB should be able to provide the FX9000P with an alternative way to screenshot the video RAM and also, perhaps send the video data to a second display," he says. This could be used to replace the CRT with an LCD if and when the tube fails, which means this project is not only helping to preserve old tech but future-proof it too.

Sfera Baltic boat

A 60-year-old tug boat is flawlessly piloted by the latest Raspberry Pi and Compute Modules, finds **Rosie Hattersley**

Maker

Ulderico Arcidiaco

Ulde is the co-founder, CEO, and chief hardware engineer at Italy's Sfera Labs.

sferalabs.cc

In 1962, The Baltic, a 26 metre-long Norwegian tug, took to the oceans. Unlike today, boats of the 1960s were largely manually controlled. Little wonder because the most widely used computer was IBM's 1401: it had a 16,000-character core memory and weighed several hundred kilos. The 1401 was so expensive that most companies would generally rent one.

The 1401 computer took 17 seconds just to boot up and lacked both software and interfaces needed to work in industrial, let alone marine, scenarios. Other early computers such as the renowned DEC PDP-1 would have been no more suitable or affordable to control a commercial craft. Yet by the time The Baltic tug had been in operation for 20 years, computing was beginning to make real inroads, and computer-controlled ships came on stream. Around this time, The Baltic was taken out of service and sold on. Eventually, time and computer capabilities caught up with the venerable craft, and in 2021 The Baltic embarked on a new life, with a brand-new control system built around Sfera Lab's Raspberry Pi and Compute Module Strato and Iono hardware: **rpimag.co/piboatrestore**.

Heritage value

By the early 1980s when The Baltic tug was decommissioned, huge strides had been made in computing capabilities and costs had plummeted. After languishing on the south coast of England and then voyaging to the Mediterranean, The Baltic's fortunes changed. "The new owner had the ambitious goal to fully refit the boat and transform it into a yacht, preserving the original design and heritage of the unit," explains Ulderico Arcidiaco, the man tasked with modernising and fully automating the revitalised craft.

Standard PLCs (programmable logic controllers, often described as industrial computers) are used for the power management and some automation aspects of most ships, while everything else is done separately. Ulde was approached to add automation and decided on an audacious plan for an IPMS (Integrated Platform Management System) such as those found on modern military units or very large commercial ships. The fully integrated approach controls of all of the ship's subsystems. The setup would cost a fortune to implement the traditional way, but made sense using Sfera's Raspberry Pi-based PLCs and the firm's own supervision

IPMS-class architecture would cost a fortune to implement the traditional way, but made sense using Sfera's Raspberry Pi-based PLCs and the firm's own supervision software

01. Former tug The Baltic had a Raspberry Pi-based makeover in 2021 to become a fully automated pleasure craft

02. Sfera Labs' Ulde Arcidiaco has now upgraded the 63-year-old Norwegian tug to take advantage of both Raspberry Pi 5 and Compute Module 5

software. He installed Strato Pi units to provide full, integrated control of The Baltic's core and supplementary functions, from power distribution to tanks and pumps control, navigation, alarms, fire, lighting, stabilisers, chargers, inverters, battery banks, and video feeds.

The first phase of the refit saw Ulde install Strato Pi Compute Module Duo and Raspberry Pi Compute Module 3+ on The Baltic. These were replaced in 2022 with two Strato Pi UPS (uninterruptible power supply) modules with 4GB Raspberry Pi 4B. "It was a plug and play replacement," he notes. "The combination of a more powerful CPU and more memory resulted in a 2× to 4× increase in performance, buying us ample margin to address the new needs."

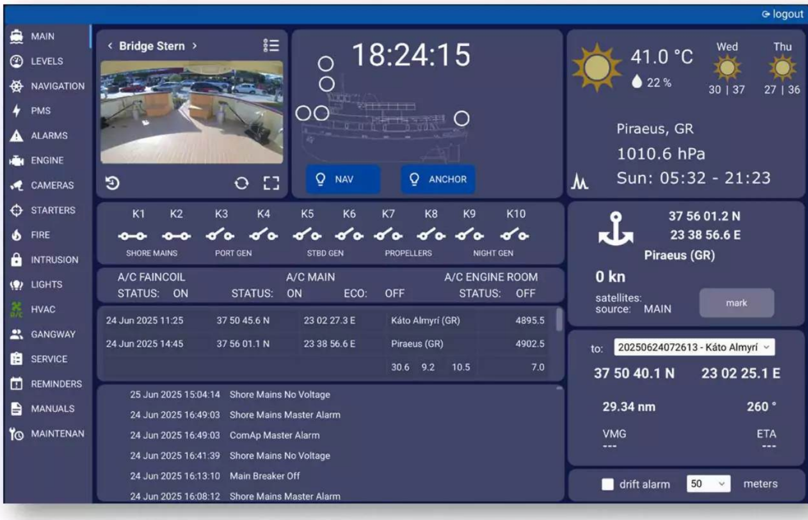
▶ The bridge is very high-tech now, thanks to the Raspberry Pi-powered upgrades

Nonetheless, by this year, Ulde saw the sense in upgrading. “When sailing, the amount of data that has to be processed and stored in real-time is significant,” he observes, and he knew from the outset that performance updates would be needed over the years, especially with new onboard systems and services to support. Two fully redundant Strato Pi Max XL units with Compute Module 5 and 8GB RAM plus a 480GB SSD now function as the core IPMS processors. Performance-wise, The Baltic’s upgraded setup is now around 15× more powerful than the original CM Duo setup Ulde first installed.

Time and tide

Having been born just a couple of years after The Baltic first set sail, Ulde has been an electronics and computer enthusiast since his teens and says he has seen “a meaningful chunk of the computer evolution”. The founder and co-CEO of industrial computer specialist Sfera Labs, Ulde says Raspberry Pi is one of the most important developments that happened throughout this journey. “Raspberry Pi opened a whole range of new opportunities, similar to what Linux did on the software side.

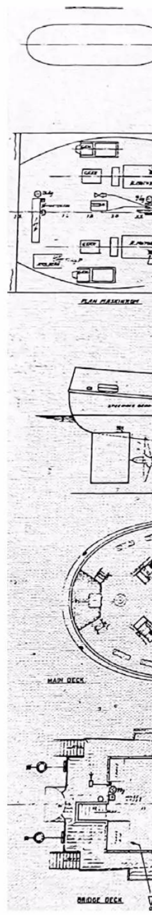
“Just as it was inconceivable to install a computer on a yacht in 1962, so it was unthinkable until a few years ago for a relatively small company such as Sfera Labs to be creating products and software capable of implementing a full IPMS on a complex vessel.”

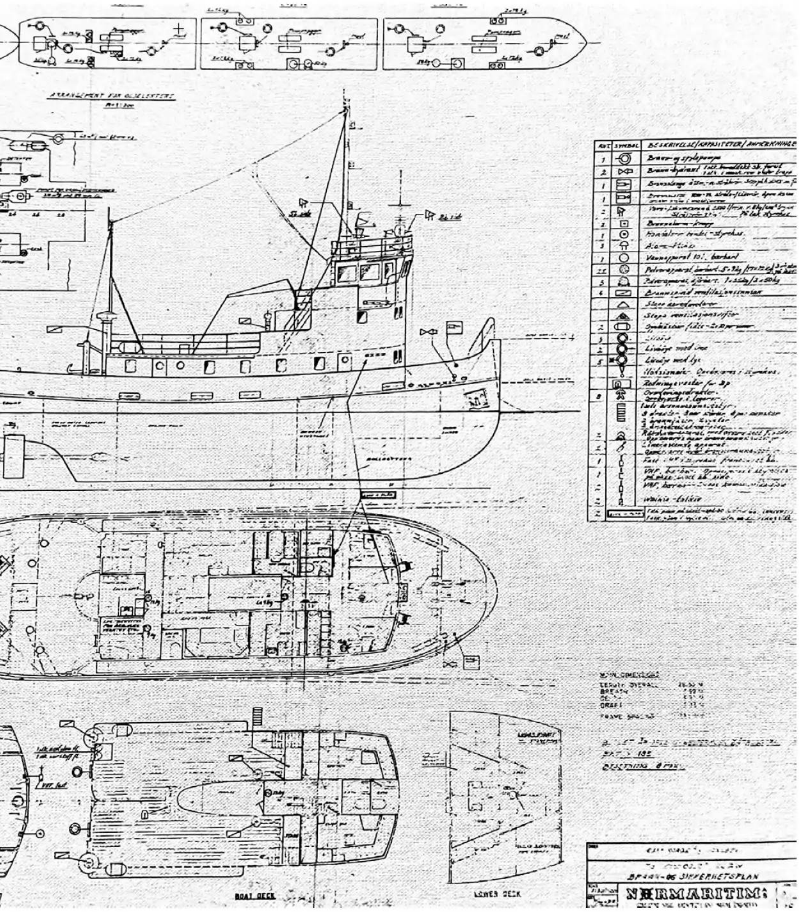


▲ Sfera’s bespoke IPMS interface controls almost every aspect of the ship

Quick FACTS

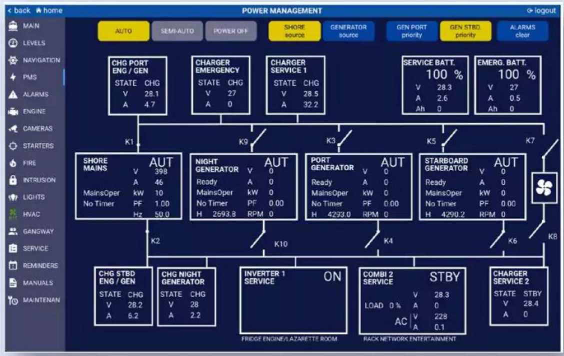
- Navigation involves significant amounts of data processing and storage
- The Baltic has more than 4500 quickly changing data points
- Sfera’s hardware also has to cope with video from ten onboard cameras
- Upgrading to Strato Pi Max XL provided a 5× performance jump
- The upgraded CM5 system now shows a CPU load of just 20%





▲ Original drawings showing The Baltic back when it was a lowly tug-boat

Captain's log



1. Sfera Labs' take on IPMS (Integrated Platform Management System) provides detailed insights into the power being generated and distributed on board.

2. The Baltic's 2025 upgrade saw the Strato Max and Iono hardware that governs the propulsion, power generation, temperature, and automation updated to Raspberry Pi 5 and Compute Module 4: **rpimag.co/baltictug**.

3. Upgrading the IPMS to more powerful Compute Module 5-based Strato Pi Max XL units significantly improved its efficiency and failover support.

Fingerprint-controlled lock

Keep your belongings safe with a device that only unlocks with an authorised tap. By **David Crookes**

Maker

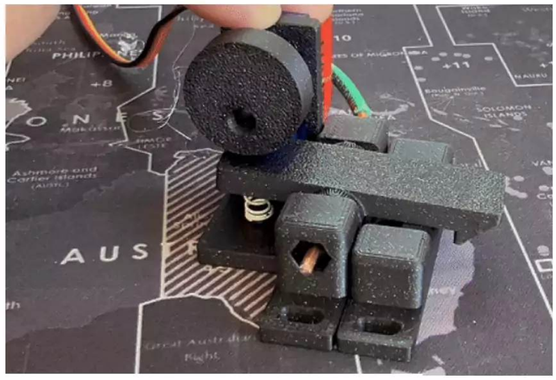
Joyce Lin

Joyce shares hands-on Raspberry Pi builds that teach the fundamentals of robotics.

rpimag.co/fingerlock

In an ideal world, we wouldn't need security devices but, if you have any valuables that you need to protect, you'd be in safe hands if you give Joyce Lin's project a try. She's created a robust lock that lets users scan their fingerprint to gain physical access to a box. Such is its versatility, it could also be used to unlock all sorts of items.

"I was inspired to create the fingerprint-controlled lock after using a fingerprint reader for my garage door," she says. "I loved it so much that I started thinking of other ways to use the same sensor, like unlocking a physical box to store the souls of my enemies – or a geocache."



▲ The servo needs to be mounted so that it can securely close the latch and easily open it

> *You can save a bunch of templates for the same finger to improve matching*

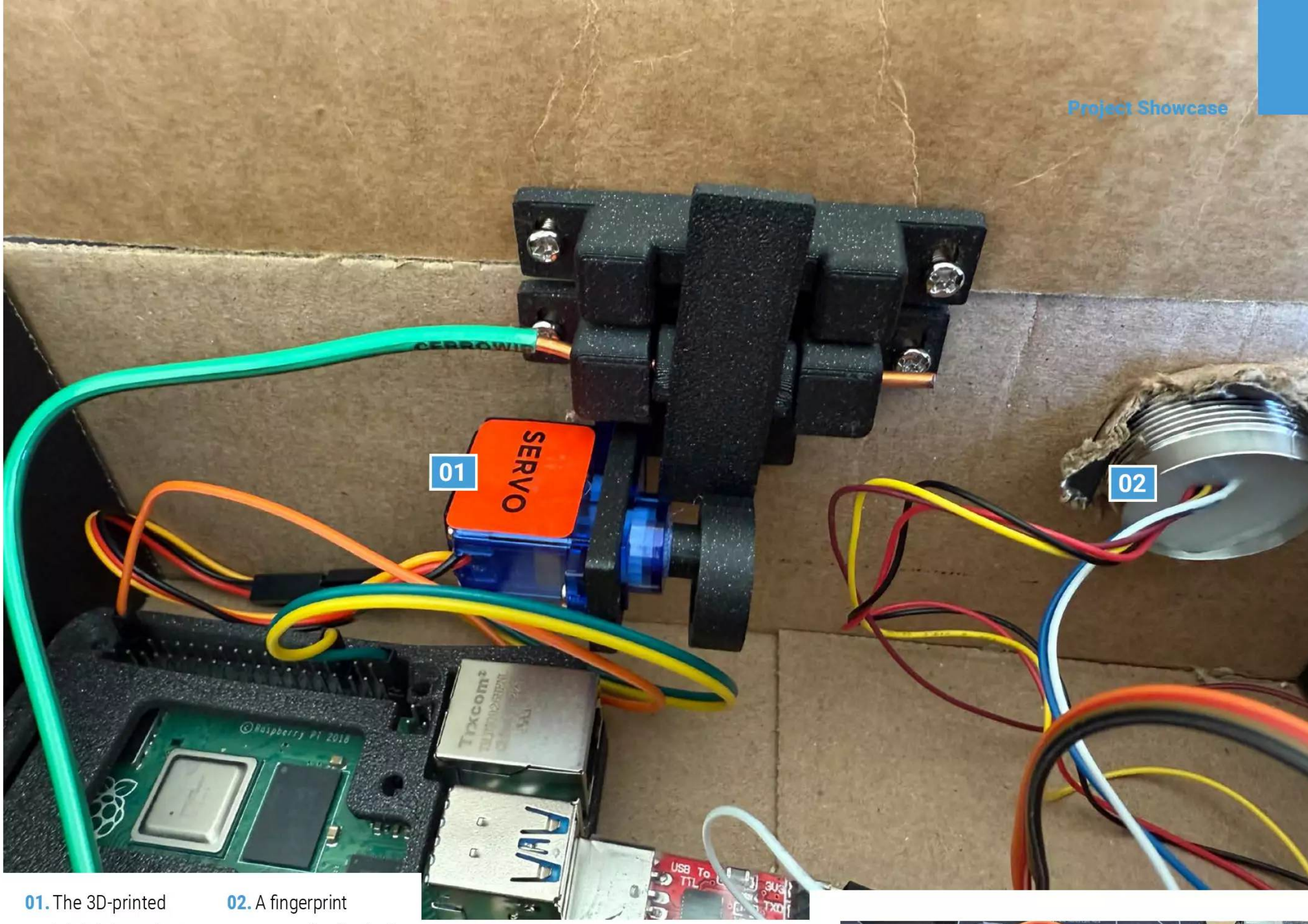
To create the project, Joyce combined an Adafruit R503 fingerprint sensor with a Raspberry Pi 4 computer, and she used a micro servo motor to power a 3D-printed latch. These electronic items were easily wired together, but Joyce then needed a way of gathering and assessing data from the fingerprint sensor.

Find a match

Joyce's chosen method was the Viam platform (**viam.com**). Viam is capable of managing sensor data for use with real-world devices. The aim was to get the sensor to capture a fingerprint image, convert it to a digital template, and compare the image to enrolled fingerprint data stored locally on the Raspberry Pi.

There were some initial challenges. "The Python library I used required exactly two scans to enrol a fingerprint, and I initially thought the reader was just bad at recognising my finger," Joyce says. "The trick was to enrol multiple fingerprints for the same finger using different parts and varying pressure levels. You can save a bunch of templates for the same finger to improve matching."

With the fingerprints stored correctly, the lock could get down to work. A Python



01. The 3D-printed latch is based on a servo mount design that Joyce found online

02. A fingerprint sensor is attached to the device, but located separately from it

▶ The R503 fingerprint sensor has a bi-colour LED ring, allowing it to light up when a finger is placed on it

"algorithm determines if there's a match to an enrolled fingerprint based on a confidence score," Joyce explains. "When there's a match above a certain confidence threshold, the Raspberry Pi sends a PWM signal to the servo motor via a GPIO pin. The servo releases the latch, allowing the box to be opened."

Remote control

A Raspberry Pi computer proved ideal. "Raspberry Pi 4 is my go-to device for spinning up hardware projects quickly," Joyce says. "It has more than enough processing power for image matching along with built-in Wi-Fi for over-the-air updates and remote monitoring."

Viam also had benefits. "Viam has a visual interface that lets me mess around with hardware without writing scripts every time," she explains. "I can just click a button in my browser to test the servo instead of SSHing into Raspberry Pi.

"Each hardware component or service can also be wrapped up in its own module, so you can stitch them together like Lego blocks to make your own project. What's more, I can control and monitor my Raspberry Pi from anywhere with internet access to check logs, grant temporary access, or troubleshoot issues from my pyjamas at home."

Having achieved her main aim of producing a geocaching box, Joyce would now like to go further. "It's possible to add more fingerprint sensors and require a sequence of scans instead of one match," she reveals. "It's also possible to build a mobile app that makes someone solve a puzzle before accessing the box and of course the sensor can unlock other things, like a hidden room, habit tracker completions, or whatever needs securing."

Quick FACTS

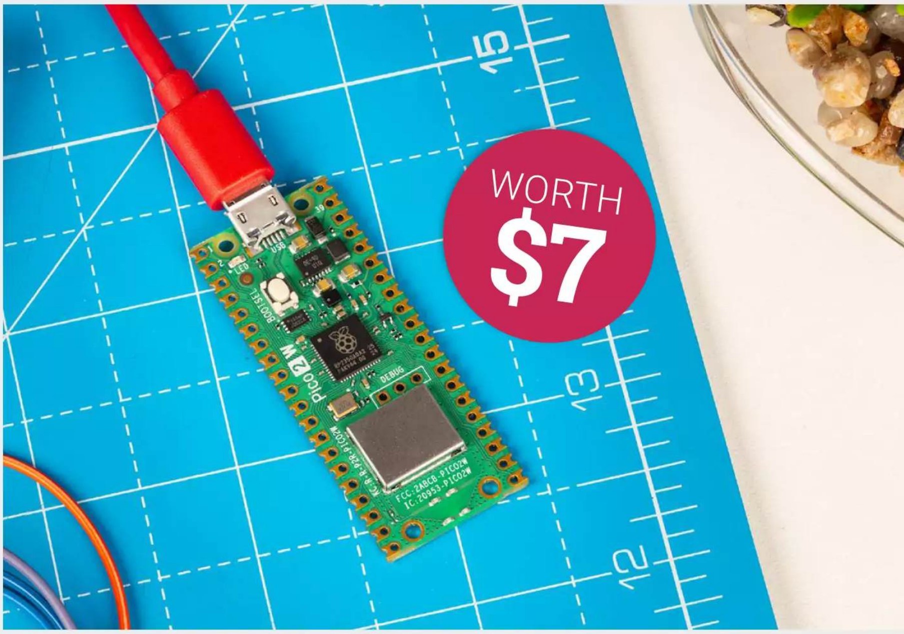
- The lock is being used for a geocaching project
- It uses an R503 fingerprint sensor
- Data is passed through the Viam platform
- The latch opens when a correct fingerprint is detected
- The build cost less than £80

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 RP2040-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

Teka-Sketch

By Tekavou

rpimag.co/Etch-a-sketch

The Etch-a-Sketch was, for this author at least, impossibly difficult, frustrating, and not fun at all. In other, more artistically gifted hands, its constraints lead to brilliant artworks, but for us, the Etch-a-Sketch was an instrument of torture.

Maker Tekavou has taken the Etch-a-Sketch and made it better, by not making it an Etch-a-Sketch at all – his version uses an e-ink screen, rather than the original grey powdered aluminium which gets scraped off to reveal the black underneath. He's also developed a way of turning photographs into Etch-a-Sketch-like images, and has built a Raspberry Pi Zero 2 W-based camera to take images and upload them to the new and improved Teka-Sketch. It takes 10–15 minutes to render an image on the Teka-Sketch when the conversion is performed on the Raspberry Pi Zero 2 W; to speed this process up, the maker has added a Mac mini in between the camera and the display, to perform the work of turning the image from a photograph into a realistic Etch-a-Sketch image.

This all-new e-ink-based Etch-a-Sketch can play Pong, and it also has an undo button

3D-printable Raspberry Pi Cyberdeck

By Nathanael de Jongh

rpimag.co/pi5-cyberdeck

The phrase 'minimum viable product' has been ringing in our ears for the last few weeks for some reason. Maybe it's all the LEDs on the Raspberry Pi 500+ reminding us that more is, in fact, more, but we were looking for something small and functional to show that if an idea is a good one, it doesn't need any additional bells and whistles.

Enter this small 3D-printed cyberdeck, by Nathanael de Jongh. It holds a Raspberry Pi 5, a 3.5-inch display, and a Rii miniature keyboard, and is encased in 0.08mm layer height, 5% infill 3D-printed plastic. Like many awesome projects, it arose because the maker had the stuff already laying about:

"I designed this model because I had bought the screen a while ago, and when I then got a 3D printer I decided to create a simple easy-to-use case to house it and a Raspberry Pi Zero," says Nathanael. "I then created the Raspberry Pi 5 version upon realising that it required a bit more power for some tasks. The Pi Zero version is great for things like Python programming and the Raspberry Pi 5 one is better for things like web browsing and watching videos."

▲ Unlike most cyberdecks, this one still needs mains power

NTRON

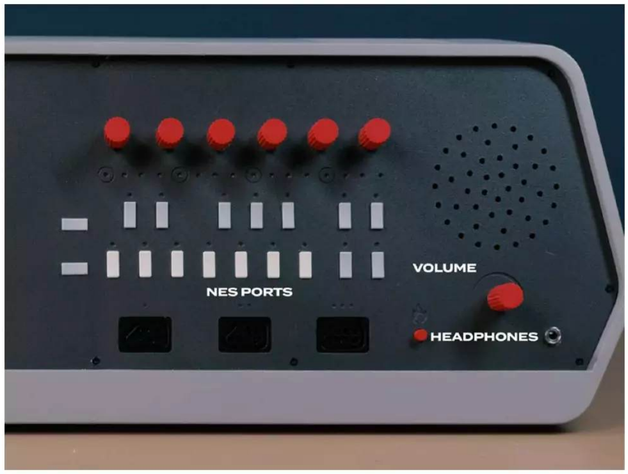
By Artifextron

rpimag.co/NTRON

Here's a wonderful homage to 8-bit games and chiptune music: the NTRON. It comprises a console, gamepad and keyboard, and it's both a RetroPie gaming setup and an 8-bit synth.

Rather than a monolithic, ground-up solution, Artifextron has brought together a collection of existing hardware, wired it together, and popped it into a beautiful 3D-printed shell to create something that looks like you've stumbled across an unreleased bit of Nintendo hardware from the eighties.

At the heart of the NTRON, there's a Raspberry Pi 3B+ and a Waveshare QLED 7-inch screen. Also housed in the 3D-printed case, there's a hapiNES synth by Twisted Electron and an amp, and the MIDI keyboard is provided by a cannibalised M-Audio Keystation Mini USB keyboard. Software-wise there's a custom IMG file for the Raspberry Pi 3B+ with all the software installed and configured, making it really plug-and-play.



▲ The NTRON is compatible with original NES controllers, and there are custom models built on perfboard

E-ink shipping monitor

By Embarrassed Octopus

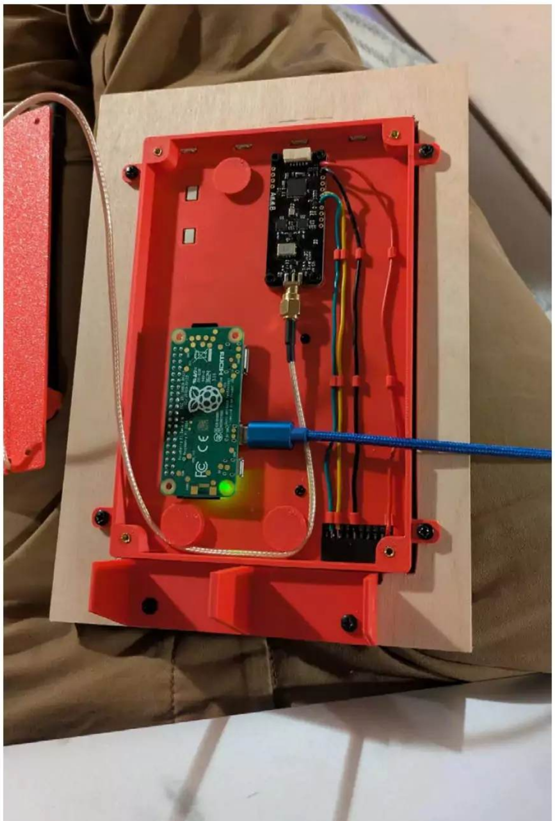
rpimag.co/ShippingMonitor

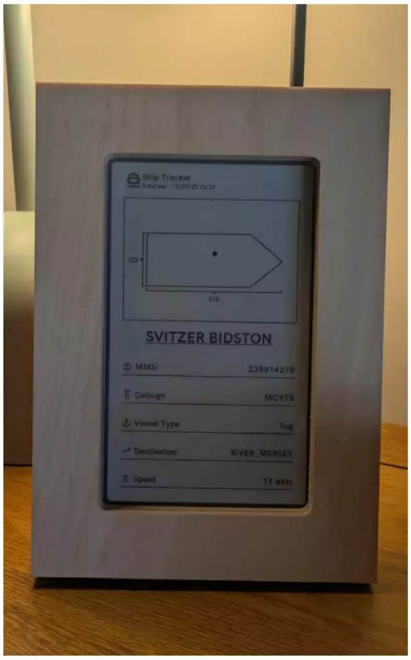
Otis Redding may have sat on the dock of the bay, watching the ships go rolling in and rolling out again, but history doesn't record whether we ever wondered what the ships were called, or where they came from. Unlike Reddit user Embarrassed Octopus, who was curious about the glimpses of the ships on the Mersey that they get from out the window of their home.

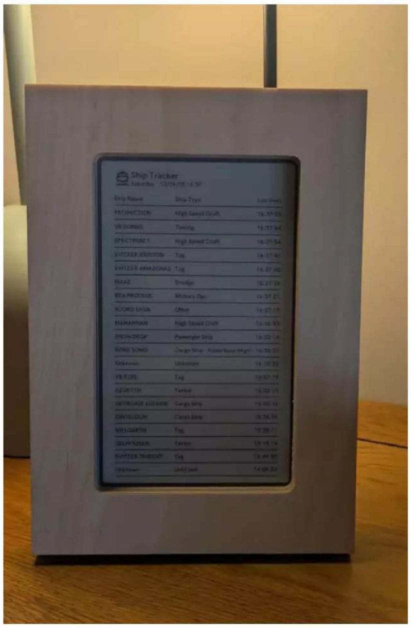
The display comprises a Raspberry Pi Zero 2 W, a Pimoroni Inky Impression 7.3-inch display, and a Wegmatt dAISy Mini AIS receiver. AIS stands for 'Automatic Identification System', and it's this device that picks up the signals coming from the ships themselves, which goes to the display via the Raspberry Pi Zero 2 W. There's probably a website somewhere that displays the location of ships, and you could build a similar system using an API to extract data from that website, but this uses real-time data collected via VHF radio frequencies direct from the ships themselves, which makes it far, far cooler.

According to the maker, there are three screens: "geofence, table, and map. Map will show all vessels that have been heard from in the past five minutes. Table shows the most recent 20 vessels that have been seen and geofence is the most recent vessel to enter a user defined area, which I've set up to be right where we can see from the window."

The geofence screen also outputs a blueprint of the ship, showing the length and breadth of the vessel plus the location of the ship's GPS receiver, which gives you an idea where the bridge is. Ahoy!







▲ There's a newer version of this project in the works, using the (also) newer version of the Pimoroni Inky Impression

3D print

Smaller, cheaper, and lighter than a Lockheed A10 Warthog, this 3D-printed propeller launcher is tons of fun. By **Toby Roberts**

rpimag.co/HSPropellerLauncher

Propeller launchers have been around for generations: simple handheld toys that send a small rotor spinning into the air. Traditionally made from wood or bamboo, and more recently plastic, they've also gone by names like helicopter spinners, prop toys, or just plain propeller launchers. Today, you'll find endless versions on 3D model sites for printing at home, but a new wave of more elaborate, engineered designs has started to appear. And while we've always had a soft spot for over-engineering, this one really takes it to the extreme.

Lukas Hannert has designed a three-stage herringbone planetary gearbox with a 1:64 gear ratio. What on earth does that mean? Well, the planetary system gives compact, efficient load sharing across multiple planet gears, while the herringbone tooth form cancels axial thrust for smooth, quiet, and durable operation. With three stages, it produces serious torque and massive reduction ratios in a compact, vibration-free package… and breathe!

In practical terms, that 1:64 ratio means a tiny quarter-turn of the launcher spins the propeller up to ridiculous speed, shooting it skyward with the velocity of a caffeinated hummingbird on Red Bull.

3D-printing the model is straightforward with just standard PLA and no support required. It simply clicks together. It does use a fair amount of filament and takes a while to complete over several build plates, but the end result has real heft and durability, built to last well beyond a few uses.

Alert!

Safety Goggles

Be mindful of the spinning blade when testing. Wear safety goggles.

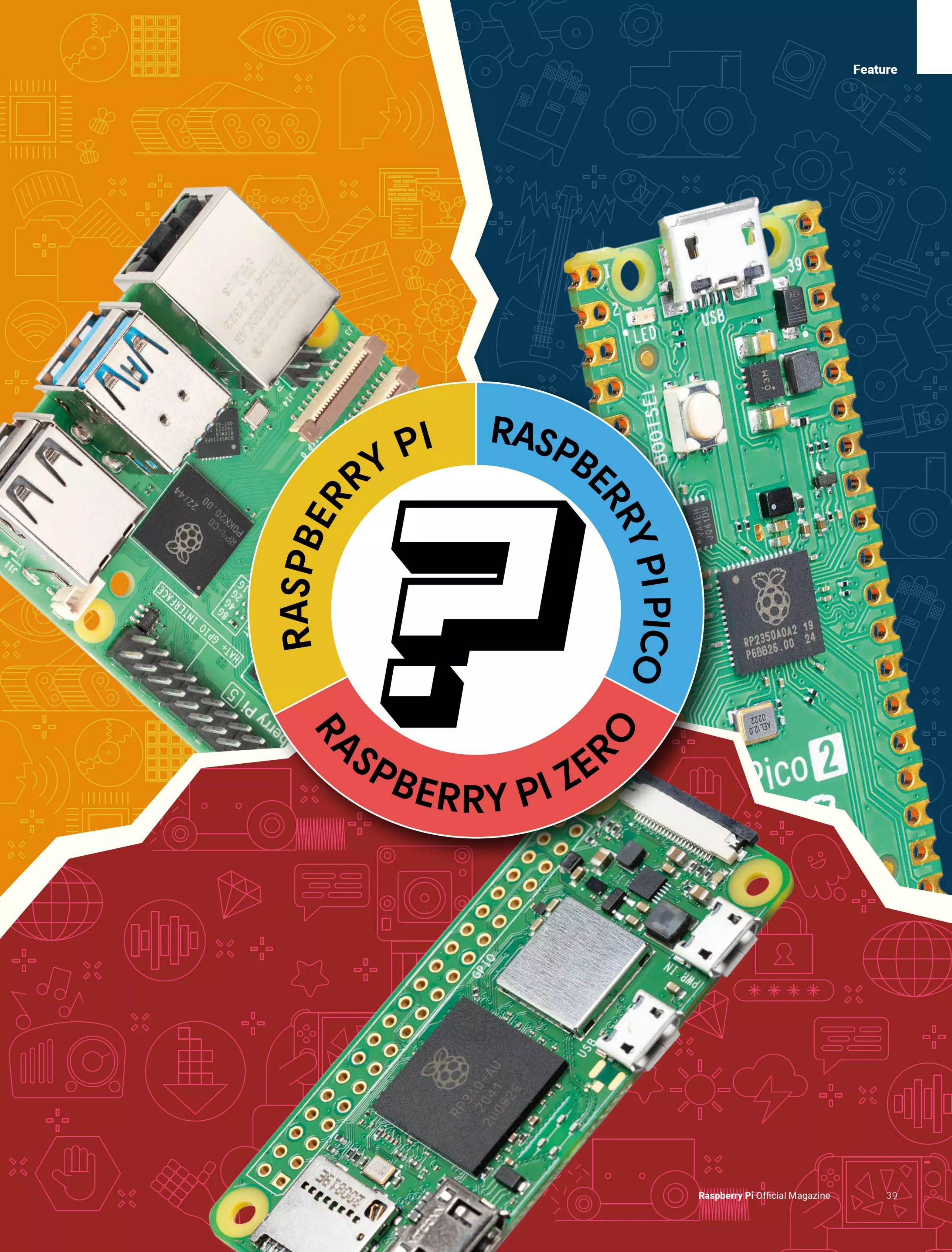
rpimag.co/ppe


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

HEAD TO HEAD

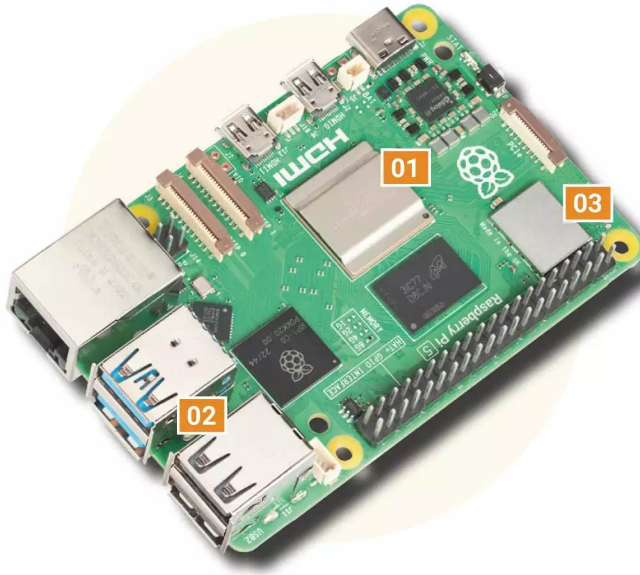
Which Raspberry Pi should you use for your next project? We explore the differences between Raspberry Pi Model B and Raspberry Pi Zero single-board computers and Raspberry Pi Pico microcontroller. Each has its distinct advantages, so you should really own all three!

By **Phil King**


RASPBERRY PI
RASPBERRY PI PICO
RASPBERRY PI ZERO

RASPBERRY PI (MODEL B SERIES)

The flagship models boast superior processing power and connectivity



01. Raspberry Pi 5's BCM2712 Broadcom system-on-chip delivers serious processing power

02. Most Raspberry Pi Model B computers feature four USB-A ports

03. Raspberry Pi 5 comes with up to 16GB of RAM; Raspberry Pi 4, up to 8GB

Raspberry Pi 500/500+

If you are looking to create a desktop computer, consider using a Raspberry Pi 500 or the new Raspberry Pi 500+ instead. It gives you the power of Raspberry Pi 5 inside a keyboard. The new 500+ has a mechanical keyboard, 16GB RAM, and a fast 256GB M.2 SSD.

The very first Raspberry Pi launched back in 2012 was a Model B, and the credit-card-sized form factor has continued ever since for the flagship range of single-board computers. The main change over the years has been the introduction of ever more powerful processors, along with changes to the arrangement of ports and enhanced features.

Ever since 2014's Model B+, the 40-pin GPIO header has been a mainstay, enabling the connection of HAT (Hardware Attached on Top) expansion boards and electronic circuits for projects.

Ideal for more demanding tasks such as AI and machine learning

While the same GPIO header is present on Raspberry Pi Zero (although you may need to solder the pins to it), the main advantage of later models in the flagship range is notably superior processing power. Along with increased RAM, this makes them ideal for more demanding tasks such as AI, machine learning, emulation of more powerful retro games systems, high-quality video playback, and home automation – for instance, a Raspberry Pi 4 or 5 is recommended for running a Home Assistant setup, as we explored in our issue 154 cover feature (**rpimag.co/154**).

Another key advantage is the expanded range of ports compared to Raspberry Pi Zero models. Four USB ports (including two USB 3.0 ones on Raspberry Pi 4 and 5) enable you to connect a mouse, keyboard, and other peripherals or external drives. Along with the superior processing power, this makes it highly suitable for desktop computing, using the standard Raspberry Pi OS, based on Debian Linux, or an alternative OS such as Ubuntu.

Raspberry Pi 5 also benefits from a PCIe 2.0 x1 port enabling you to connect – via the M.2 HAT or similar add-on board – an NVMe SSD (solid-state drive) with much faster data transmission than connecting a drive via USB. The PCIe connector may also be used to add an NPU (neural processing unit) for AI applications, either with the AI HAT+ or AI Kit, boosting processing power for up to 26 TOPS (tera-operations per second).

If you need a versatile single-board computer with plenty of processing power and connectivity, a Raspberry Pi 4 or 5 is ideal.

Spin

This AI music turntable looks and sounds cool. A Raspberry Pi 4 running xwax, an open-source digital vinyl system (DVS) for Linux, deciphers audio timestamps on a time-coded vinyl record. This enables the DJ to scratch (move the record back and forth) to manipulate the sound. The grid of buttons is used to select mood, genre, and sounds, with sliders to set duration and tempo; a text prompt is then sent to a MusicGen AI model in the cloud to generate the track.

rpimag.co/spinai

CinePi V2

This ambitious open-source project transforms a Raspberry Pi 4 and Raspberry Pi High Quality Camera into a high-end cinema camera that can shoot 2K RAW Cinema DNG video at frame rates up to 50fps with 12-bit colour depth. This makes it suitable for capturing cinematic-quality footage in projects such as short films, commercials, and YouTube videos. A 4-inch touchscreen is used to monitor what you're shooting and provides an interface.

cinepi.io

Avanade Intelligent Garden

A great showcase for how Raspberry Pi 5 and AI HAT+ (for up to 26 TOPS of processing power) can be used for computer vision, this project – demonstrated at the Chelsea Flower Show – features camera traps to count how many flying insects are visiting areas of the garden to pollinate plants. This can be used to assess whether the plants are attracting enough butterflies and bees. The system can identify them and automatically take stills and video footage when they're in the frame.

rpimag.co/intelligentgarden

Desert Eye 2.0

This impressive tank-like robot keeps an eye out for hazards and can traverse tricky terrain thanks to its 3D-printed caterpillar treads featuring roller-equipped rocker arms, each with its own mini shock absorber. The maker chose to use Raspberry Pi 4 for its video processing capabilities for the surveillance footage coming from a front-mounted night-vision camera, which it sends wirelessly to its remote operator. Raspberry Pi also controls the robot's motors, GPS, communications, and a three-axis sensor.

rpimag.co/deserteyeyt

KEY SPECS

DIMENSIONS
85.6 × 56.5mm (Model B)

WEIGHT
50g

PROCESSOR
Up to 2.4GHz quad-core 64-bit*

MEMORY
Up to 16GB*

PORTS
4 × USB (including 2 × USB 3.0**), Ethernet, up to 2 × HDMI video, MIPI display / camera, PCIe 2.0 x1*, micro USB / USB-C power input

WIRELESS†
Dual-band 802.11ac** or 801.11n Wi-Fi, Bluetooth 4.1, 4.2 or 5.0**, BLE

* Raspberry Pi 5
** Raspberry Pi 4 and 5
† Raspberry Pi 3 Model B and upwards

MODELS RECOMMENDED

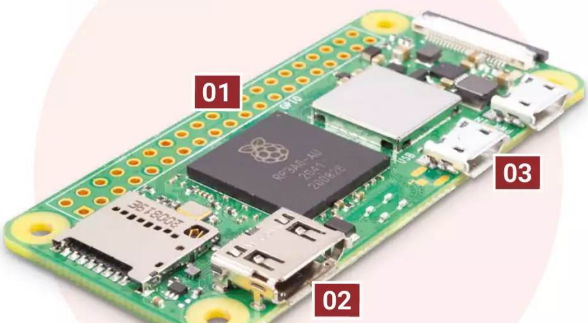
- Raspberry Pi 3 Model B
- Raspberry Pi 3 Model B+
- Raspberry Pi 4 Model B
- Raspberry Pi 5 Model B

IDEAL FOR:

- Desktop computing
- AI / machine learning
- Media centre / NAS
- Larger robots
- Retro gaming
- Home automation
- Magic mirror

RASPBERRY PI ZERO

Smaller, more affordable, yet it can still run a full OS



01. On many Raspberry Pi Zero models, you'll need to solder pins to the GPIO header to use HATs

02. A mini HDMI port enables you to connect it to a monitor, but it's often used headless

03. You'll need a USB hub to connect multiple peripherals to the single user micro USB port

In case it wasn't clear, the W stands for wireless radio enabled, and the H stands for (GPIO) header included

Launched with the aim of making computing even more affordable, the Raspberry Pi Zero range of models also benefit from a smaller form factor – about the size of a stick of gum. This makes a Raspberry Pi Zero ideal for any projects where space is at a premium, such as in drones and smaller robots, or handheld games devices.

While smaller, it still features the same 40-pin GPIO header as the flagship Model B range. So you can use it with the same range of HATs and other expansion boards, as well as connecting your own electronic circuits to its pins. Note that on some Raspberry Pi Zero models, you'll need to solder the pins to the header; others have them pre-soldered, such as the Zero WH.

Ideal for any projects where space is at a premium

Compared to the later models of the Model B range, Raspberry Pi Zero has less processing power and RAM. The standard Zero / Zero W features a single-core processor, but the quad-core Zero 2 W is roughly equivalent to a Raspberry Pi 3 Model B. So, anything you can do with that, you can do with a Zero 2 W.

Unlike Raspberry Pi Pico, all Zero models are fully functional computers that run an operating system, so you can install your favourite applications and tools as per usual. You can even connect it to a monitor from its mini HDMI port, although the single user USB port means you'll need a USB hub to connect a wired keyboard and mouse (or use a Bluetooth connection).

Typically, however, Raspberry Pi Zero is used in a headless setup, connecting to it from a remote computer via SSH over Wi-Fi to issue terminal commands. Its lower power drain (as little as 100 milliwatts) makes it suitable for battery-powered projects in remote locations away from a mains power outlet, such as a weather station or wildlife camera – you can connect a Camera Module to its CSI camera port.

If you need a more compact Raspberry Pi, but with several orders more processing power than a Pico and the ability to run an operating system, Raspberry Pi Zero fits the bill.

Lawny

This remote-control robot mower was originally built using a Raspberry Pi 5, but the maker has since switched it out for a Raspberry Pi Zero 2 W, demonstrating how the smaller and cheaper single-board computer is powerful enough to handle a sophisticated robotics project. A front-mounted Raspberry Pi Camera Module 3 gives the remote operator an as-it-mows Lawny-eye view as they control the robot from a web interface on a smartphone or computer – thanks to Raspberry Pi running a Node.js web server.

rpimag.co/lawny

PiMiniMint

If you want to build a handheld games console, your best choice is a Raspberry Pi Zero model, whose smaller footprint enables it to fit into a compact case with a mini LCD for a display. The lower power drain means your battery pack will last longer, too. One of the first projects to demonstrate the possibilities, PiMiniMint crams a Raspberry Pi Zero into a 60 × 95mm Altoids tin. At the time, the maker used an IoT board to provide Wi-Fi and Bluetooth connectivity, but this isn't needed with a wireless-equipped Raspberry Pi Zero W or Zero 2 W.

rpimag.co/piminimint

Time Machine Radio

Replacing the innards of an old radio to turn it into an internet radio is another very popular Raspberry Pi project. If you can't find a genuine retro model, you can always get a vintage-effect replica, as used in this project. A Raspberry Pi Zero 2 W equipped with a Pimoroni Audio Amp SHIM provides analogue audio out to the speakers. Two potentiometer knobs are used for volume and tuning – or in this case to play custom sound clips from different decades, as per the time-travelling theme.

rpimag.co/timeradio

The Oracle

This miniature version of a Zoltar-style fortune-telling arcade machine is made using a stripped-back Nintendo Game Boy hooked up to a Raspberry Pi Zero W. The latter provides all the I/O necessary for interfacing the pin-pad, coin mechanism, LCD, and relay modules. It also connects wirelessly to the ChatGPT API to generate a horoscope in the style of American writer HP Lovecraft or children's author Dr Seuss, then outputs it on paper using a thermal printer.

rpimag.co/theoracle

KEY SPECS

DIMENSIONS:
65 × 30mm

WEIGHT:
9g (Zero) / 12g (Zero 2 W)

PROCESSOR:
1GHz single-core 32-bit (Zero) / 1GHz quad-core 64-bit (Zero 2 W)

MEMORY:
512MB

PORTS:
2 x micro USB*, mini HDMI video, CSI camera

WIRELESS†:
2.4GHz single-band 802.11n Wi-Fi, Bluetooth 4.0 or 4.2, BLE

* One for input power
† W and WH models only

MODELS AVAILABLE

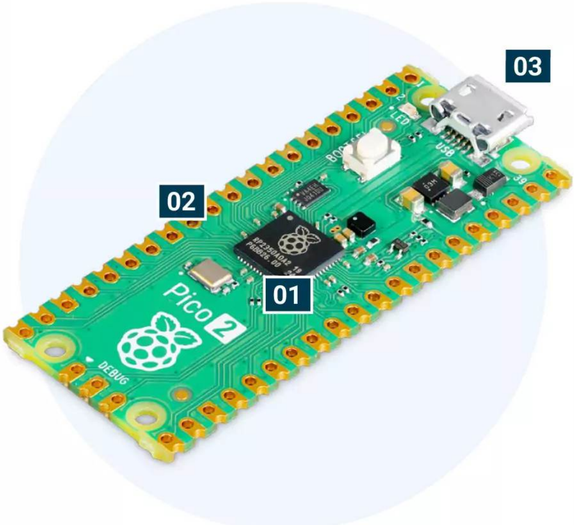
- Raspberry Pi Zero, Zero W, Zero WH
- Raspberry Pi Zero 2 W, Zero 2 WH

IDEAL FOR:

- Small robots
- Drones
- Remote camera / sensors
- Handheld gaming
- Internet radio / music streaming
- Ad blocker / VPN
- Network monitor

RASPBERRY PI PICO

A microcontroller board with plenty of I/O



01. Pico and Pico 2 are based on the RP2040 and RP2350 microcontroller chip, respectively

02. GPIO headers are on opposite edges and include analogue inputs

03. To program Pico, you'll need to connect it via USB to a computer

Like Raspberry Pi Zero, the W stands for wireless radio enabled, and the H stands for (GPIO) header included

While the other Raspberry Pi models mentioned in this guide are single-board computers, Pico is a microcontroller development board, akin to an Arduino or ESP32. While this means it is unable to run a full operating system, it does make for a simpler setup. A connected computer is used to code Pico in MicroPython, CircuitPython, or C/C++. It also boasts very low power drain and some unique features.

First and foremost, a Pico is small. Really small: 51 × 21mm and just 1mm thin without header pins attached – you can either solder them on manually (if needed) or buy an 'H' model with pre-soldered pins. Unlike Raspberry Pi computers, Pico has two 20-pin headers on opposite edges. So it can't be used with standard HATs, but a wide range of third-party Pico add-ons are available, ranging from mini displays to robotics boards and audio outputs – check out last issue's Only the Best section (**rpimag.co/157**) for some of our favourites.

One key advantage regarding Pico's GPIO pins is the inclusion of several analogue inputs. This means you can connect analogue sensors directly without the need to wire up an ADC (analogue-to-digital converter) chip to convert the signals to digital – Pico's onboard 12-bit ADC does that for you. There are also two controllers each for UART, SPI, and I2C protocols, plus 16 PWM channels. You even get a real-time clock and temperature sensor on board.

Pico was the first product built on Raspberry Pi's own silicon. The original Pico and Pico W (with Wi-Fi and Bluetooth) models utilise the RP2040 chip, while Pico 2 and 2 W are based on the upgraded RP2350. While notably less powerful than Raspberry Pi computers, they benefit from ultra-low power drain (including sleep and dormant modes) and are well suited to low-latency control of I/O to read sensors and control electronic circuits. In addition, some processing tasks can be offloaded to run on Pico's innovative PIO (Programmable I/O) state machines.

Very low-cost, tiny board with ultra-low power drain

If you need a very low-cost, tiny board with ultra-low power drain to embed in a project, Pico is perfect.

MIDI Gesture Controller

Powered by a Raspberry Pi Pico, this innovative musical expression pedal for a guitar, or other instrument, rotates and rolls around a ball joint, its position read by a six-axis AHRS IMU sensor. Unlike a standard pedal, this enables it to control three parameters at once using yaw, pitch, and roll, so far more musical effect variations can be produced. It can also operate as a hand controller, so could be used by DJs or in a studio. Or, by swapping MIDI out for HID, as a multi-axis games controller.

rpimag.co/midigesture

Automated Model Railroad

Raspberry Pi has been integrated by many hobbyists into their model railway setups, such as to provide smart, controllable lighting. This project features a 'sensored track' equipped with IR proximity sensors that detect a train passing. Raspberry Pi Pico acts as the brains, reading the sensors and controlling the track voltage – via a motor driver board – to alter the speed of a loco using pulse-width modulation (PWM). In this way, it can make it speed up, slow down, or come to a halt.

rpimag.co/picorailroad

Quadcopter Drone

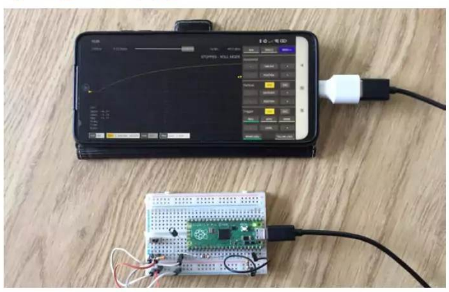
Raspberry Pi Pico's small size makes it particularly useful for really small robots – we've even seen one with beads for wheels! – and drones. Created by Tim Hanewich, this impressive quadcopter drone uses Pico to read telemetry from an MPU-6050 accelerometer and gyroscope, interpret radio commands from an onboard receiver, and control four independent motors through an ESC (electronic speed controller) using PWM. Check out his twelve-part guide to the build process, MicroPython coding, and testing to achieve stable flight.

rpimag.co/picoquadcopter

Scoppy

As covered in our tutorial (**rpimag.co/134**), this project enables you to turn a Raspberry Pico into a working oscilloscope, linking it to an Android smartphone app – via USB or Wi-Fi – to show the graph and controls interface. Once flashed with the special firmware, Pico can read analogue signals from a circuit connected to the ADC GPIO pins and then send them to the app. As a bonus, there's a logic analyser mode for viewing the digital signals on GPIO pins – useful for debugging circuits and programs.

rpimag.co/scoppy



KEY SPECS

DIMENSIONS:
21 × 51mm

WEIGHT:
4g (6g with pin headers)

PROCESSOR:
133MHz RP2040 (Pico, Pico W) / 150MHz RP2350 (Pico 2, Pico 2 W)

MEMORY:
264kB on-chip SRAM, 2MB onboard QSPI flash storage

PORTS:
SDA debug header, micro USB / USB-C power input

WIRELESS†:
Single-band 802.11n Wi-Fi, Bluetooth 5.2

† W and WH models only

MODELS AVAILABLE

- Raspberry Pi Pico, Pico H, Pico W, Pico WH
- Raspberry Pi Pico 2, Pico 2 W

IDEAL FOR:

- Wearables
- Tiny robots and drones
- IoT sensors
- Keyboard / gamepad
- LED lighting
- Weather / info dashboard
- Model railway automation

Part 01

Everything you need to know about NeoPixels: Hardware

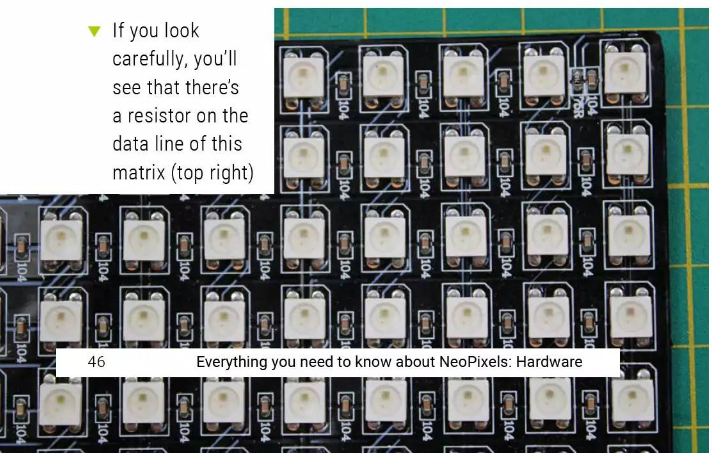
Every project deserves more blinking lights

Maker

Ben Everard

Ben is a light artist who has shown NeoPixel-based art at light festivals around the UK.

glowingart.co.uk



▼ If you look carefully, you'll see that there's a resistor on the data line of this matrix (top right)

Let's start with the very basics – what are NeoPixels? The simple answer is that they're a type of RGB LED that are chainable so that you can control vast numbers of them with a single GPIO pin on a microcontroller or computer. The official product name for this is WS2812B or WS2811; the name 'NeoPixel' comes from Adafruit, which did a lot to popularise the products (and wrote a lot of the early software to control them, so you'll often see that the library used to control them is called NeoPixel).

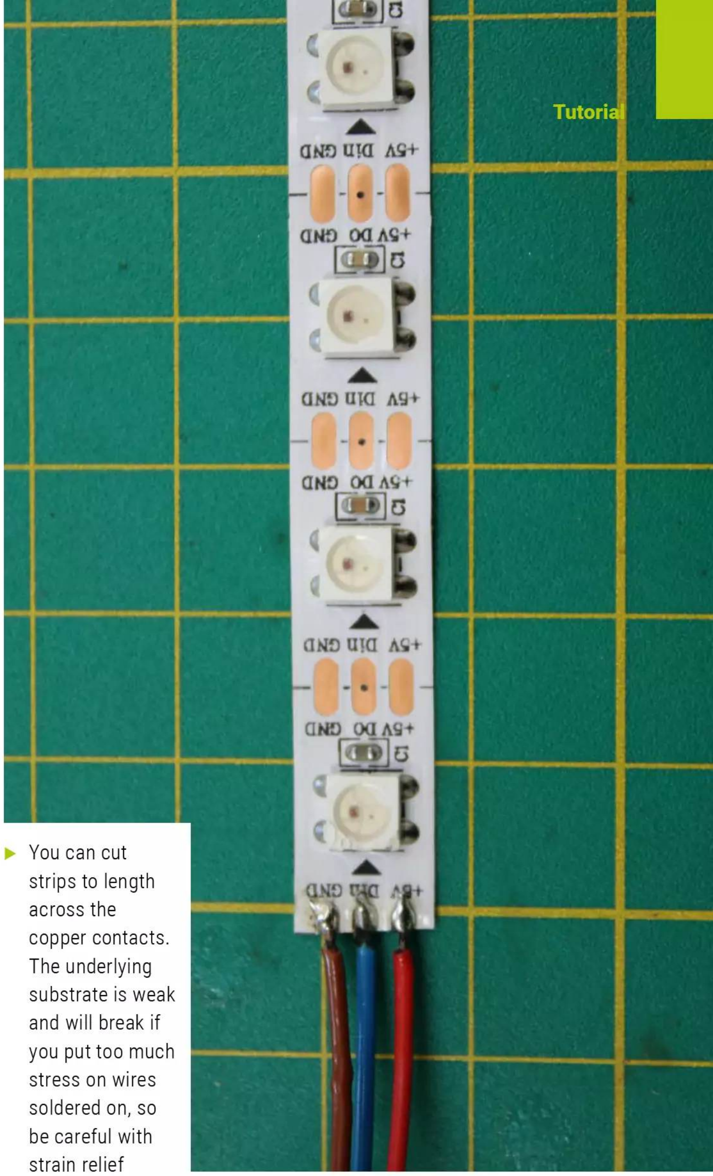
The slightly more complex answer is that there's a whole ecosystem of similar LEDs. The most famous of which is the WS2812B made by a company called WorldSemi. This integrates the LED and the controller into a single package (usually, that's square with a circular hole for the LED in the middle). There's also the WS2811, which is a chip that can control external LEDs and is commonly used on 'fairy light' style strings of LEDs. Both WS2812B and WS2811 are widely copied, and it's possible (or even likely) that if you buy them you might be getting fakes. There are also a lot of compatible products that are broadly interchangeable with WS2812Bs.

It is probably fair to say that WS2812B is used as a generic name for LEDs that follow a specific communication protocol rather than as a specific product. This has led to there being a huge variety in NeoPixel forms; while resulting in low prices, it also means that the electrical properties aren't consistent.

Form factors

Let's now take a look at what physical form this myriad of LEDs can come in.

- **Strips.** Perhaps the most famous type of NeoPixel is the strip. These are sold in spools or a particular length, and the main difference is the number of LEDs per metre. The strip is cuttable into separate lengths, and should have points that you can solder on to. They sometimes come covered in a waterproofing sheath that also provides some diffusion.
- **Strings.** These are your classic fairy lights. They are almost always WS2811 external chips with a moulding designed to fit in a 12mm hole cut in sheet material (if poking through a thicker material, you might need a 12.5mm hole to accommodate the clips). They are typically very waterproof, but not well strain-relieved, so need reasonable support if stringing like fairy lights.
- **LED neon.** This is a strip of NeoPixels covered in silicone to diffuse the light and create an effect similar to neon tubes. Underneath this is just a side-lit WS2812B strip, and you can work with it in the same way. However, if you're cutting the strip, it can take a bit of digging away at the silicone with a knife to figure out where the cut-points in the strip are.



▶ You can cut strips to length across the copper contacts. The underlying substrate is weak and will break if you put too much stress on wires soldered on, so be careful with strain relief

Voltages

Strips, strings, and LED neon come in a choice of different voltages (usually 5V or 12V) but there is more than one way of these being set up. A 5V strip uses one controller per LED (that's usually integrated into the LED itself). With 12V, there are two options: one is less wasteful, the other is more stable on long current runs.

Sometimes 12V strips have one controller per three LEDs. These put the LEDs in series and are more power-efficient and need less current. However, the downside is that it lights up LEDs in blocks of three; that is, you can't control the colour of each LED, but of each block of LEDs. This is a bit less common these days, though you do still see it, particularly on LED neon.

Other times, 12V strips have one controller per LED. These are less efficient (they typically have about the same current draw per LED as 5V strips), but they tend to suffer less from voltage drops on long runs.

For small setups, you're probably best off picking the voltage that's easiest for you to work at (and for us, that's usually 5V). For larger installations, the differences can be significant.

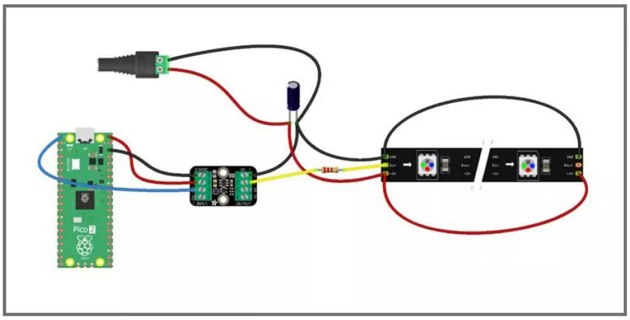
Strips, strings, and LED neon come in a choice of different voltages

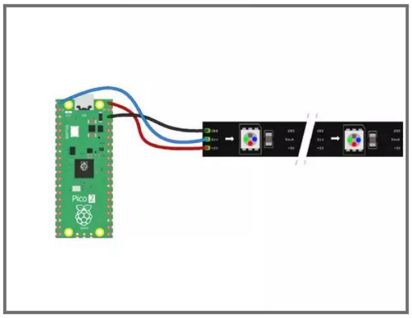
Individual components

You can use WS2812B LEDs on your own PCBs. The only recommended extra is a 100nF capacitor per LED to help smooth the power line. WS2812Bs really aren't designed for hand-soldering. It's just about possible, but we've had more failures than successes when we've tried. If you want to build your own circuits using bare LEDs, then we'd recommend using a hotplate or oven.

There is a type of LED that follows the NeoPixel protocol that is hand-solderable: the APA106 (not to be confused with the APA102 which is a common RGB LED that isn't NeoPixel-compatible) is a through-hole domed LED that looks a lot like a regular LED but with four legs rather than two. It comes in 5 and 8mm variants.

▼ This is the maximum possible setup for NeoPixels, including a level shifter (there are many different types of level shifters with different pinouts), smoothing capacitor, resistor, power injection at both ends of the strip, and an external power supply connector. In practice, this is overkill for most projects





◀ The minimum possible setup for controlling NeoPixels. This is fine for small projects, but may prove unreliable

How long?

There isn't a hard limit on the number of LEDs in a string. The protocol allows you to continue to add more and more LEDs. However, there are a few reasons you might want to limit them:

- **Speed.** The protocol sends bits of data at 800kHz and 24 bits of data per RGB LED (or 32 per RGBW). After each transmission there's a reset pause, so it's slower than this. You can get smooth animations with a few hundred LEDs. If you limit yourself to smaller runs (up to around 60 pixels) you can use some tricks for temporal dithering to get a greater range of colours (particularly at dimmer shades).
- **Errors.** WS2812Bs tend to be very reliable. This author has used them a lot, often in hostile environments, and has had remarkably few errors (the most common of which has been one of the three colours burning out). However, because the signal goes through each LED, the failure of one LED can stop any others after it on the strip working. If you split your lights up into multiple shorter strips, then a single LED failing will do less damage.

Wiring it up

Perhaps the biggest contention in NeoPixels is whether, or not, you can connect them directly to a microcontroller pin. The problem is that according to the datasheet, WS2812Bs need a data signal of at least 3.5V while the GPIO pins on most modern microcontrollers are 3.3V. Does a difference of 0.2V really matter?

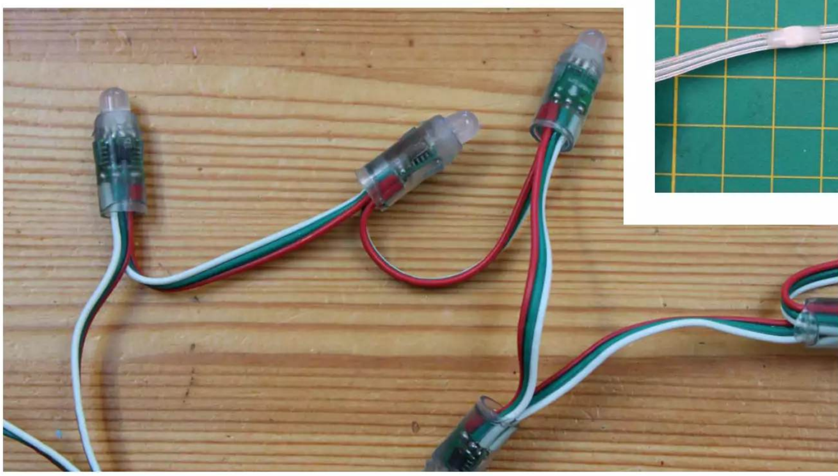
Most of the time, no. If you're setting things up on your desk with good connections and short lengths of wire, you'll almost always get them to work just fine with 3.3V. However, not always. We have several LED strips that don't work with 3.3V signals whatever you do and require at least 3.5V, and many more that will work with 3.3V most of the time, but occasionally glitch.

3.3V is out of spec. That doesn't mean it won't work, but you also can't be sure that it will. For a little project you're making one off that you can set up and test and has no serious consequences if it goes wrong, then driving the data pin at 3.3V is usually fine. However, if you're not setting it up in a perfect environment, then the odds of success start to fall: if you've got a long break from the pin to the first LED; if the project is going to run for a long time; if it'll be exposed to the elements; if it needs to work with different batches of LEDs; if it failing is expensive. As soon as one or more of these things become true, then we would recommend adding a level shifter to bring the data line back into spec. There are quite a few on the market and they only cost a few pounds.

Sometimes, we also recommend adding a capacitor to the power line, and a resistor between the driver signal and the input of the strip. Are these necessary? No – lots of people never use them and also don't have a problem – but occasionally they will be important. They serve slightly different purposes. The resistor (between about 270 and 1000 ohms) is intended to improve the signal integrity, for reasons we don't have space to get into here. The longer the wire between the GPIO and the first LED, the more likely you are to have trouble without the resistor.

The capacitor is there to smooth the voltages on the power lines. Again, like the resistor, the actual value of the capacitor is not important. Around 1000µF is a good range to aim for.

◀ If you look carefully at this WS2811 fairy light, you'll see markings showing Data Out (Data In is the middle contact on the opposite side of the PCB), +ve, and -ve



These tiny LEDs on strings can look great

Everyone has their favourite form of NeoPixel, and this author likes these chunky fairy lights

Imagine that all your LEDs are off, and the next instant, all your LEDs are on. You power supply has gone from having to supply almost no current to several amps very quickly. The capacitor acts as a store of charge so that it can supply some of this current to LEDs before the power supply has had time to react. How important this is depends on a few things, including the power supply you have, the way you're injecting power into the LEDs, and length of the power lines.

Like the level shifter, the resistor and capacitor are things that you can get away without using for many projects, until all of a sudden your LEDs start glitching and you can't work out why. While you can omit them for small projects, we'd strongly recommend you include them for larger or more permanent projects.

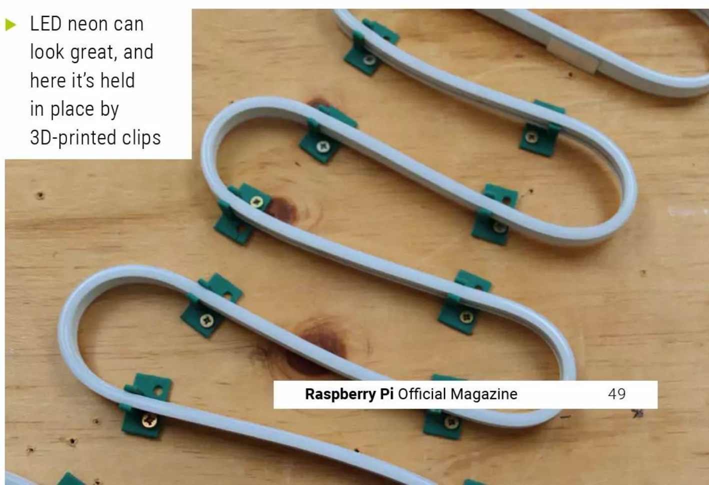
Power distribution

For small projects, you can power your strip directly from your control board. However, you have to be careful with power consumption. For example, on Pico you can draw up to 2A from the 5V pin (or less if the power supply can't manage that much). That's only enough for about 30 LEDs on full brightness. As soon as you go past this, you need to make sure that both your power supply and your wiring (including any connectors) can handle the current. Power can travel in both directions along a strip, so it's good practice to wire your power supply to both ends of the strip (and even some points along the middle if it's a long strip).

Doing projects with large currents does require a reasonable amount of thought and care. As a rough rule-of-thumb:

- Displays drawing 2A are pretty straightforward provided you follow the usual thought you would with any other electronic circuit.
- Displays drawing 5A aren't too challenging, but you need to think about your wire thickness, connectors, power supply, and power injection into your strip.
- Displays drawing over 10A require proper thought and planning.

You can have large numbers of LEDs without breaking the 2A limit if you limit your pattern, and that's something we'll look at next month when we investigate how to write code to control your LEDs.



LED neon can look great, and here it's held in place by 3D-printed clips

Needle-felted singing bird

Needle-felt a songbird, and make it sing like a canary (or, in our case, a robin!)

Maker

Nicola King

Nicola is a freelance writer and sub-editor. This month she's appreciating autumn, as all fibre fans do, and is surrounded by yarn… she's knitting, crocheting, and sculpting it!

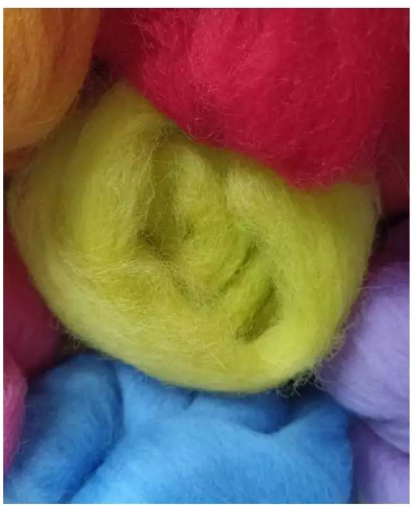
@holtonhandmade

QUICK TIP

For a good reference point, sketch or photograph some birds in your garden to get an idea of shape, size and colour.

Needle felting, or forming some kind of shape from natural wool fibres, is a highly addictive hobby. It's something this author has done before, and in this latest handicraft/Raspberry Pi mash-up, she's returning to needle-felting for a couple of reasons. Firstly, we'll see how easy, therapeutic, and inexpensive a pastime it is and secondly, we'll suggest how you can utilise a Raspberry Pi RP2040-powered microcontroller board to bring your felted item – in this case a garden bird – to life by giving it a voice.

We chose to needle-felt a European robin, a common bird in the British Isles. We've previously had a bird-cam in our garden which took some close-up shots of all sorts of birds, including the robin that we based our design on. From those images, we roughly sketched the shape of the main body on a sheet of paper, and an idea of what the wings would look like.

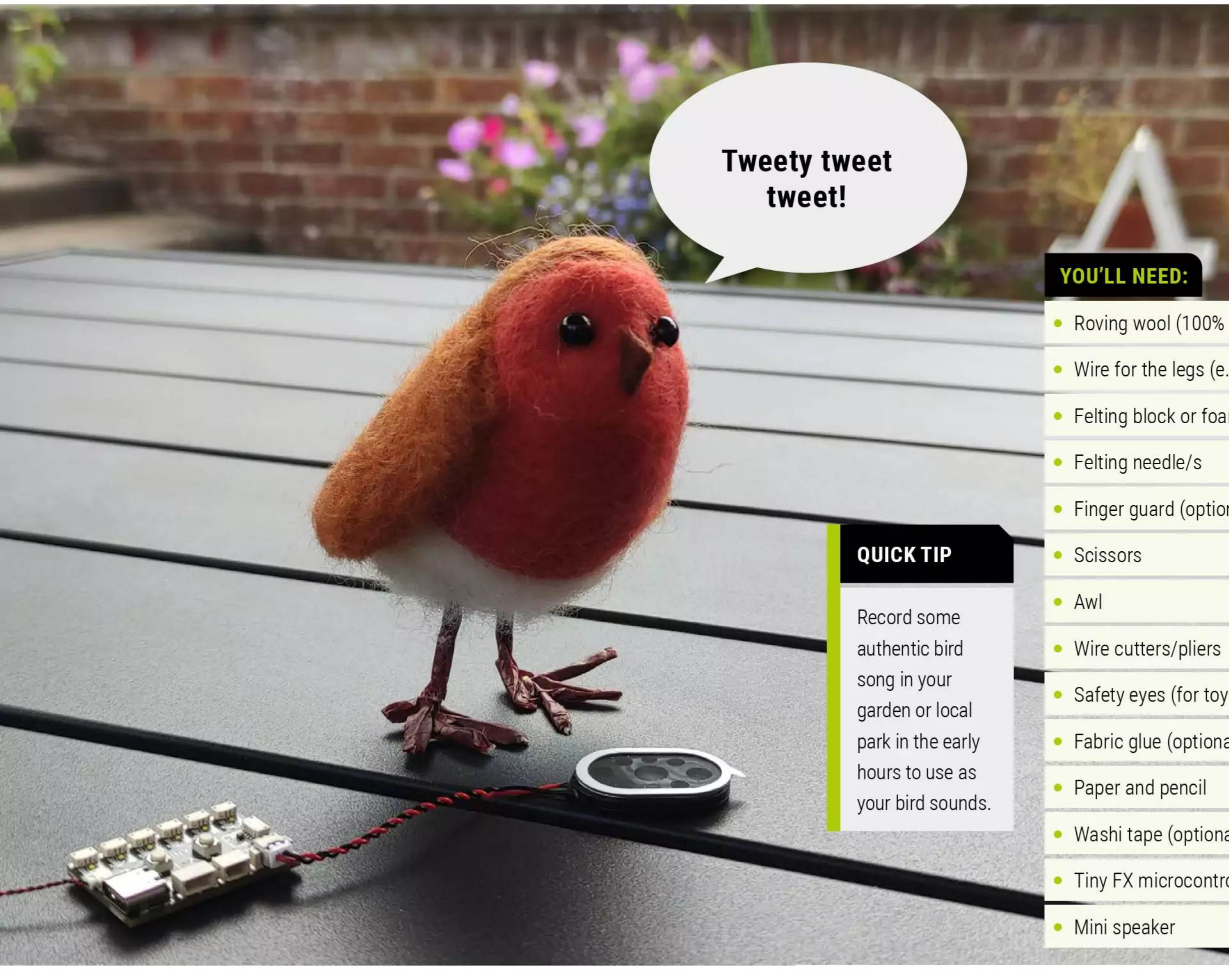


▶ A colourful cacophony of yarn fibres prior to being needle-felted. These are dyed merino wool tops (from non-mulesed sheep, thus sheep-friendly). You can use these for needle felting and wet felting, and you can even spin it on a drop spindle if you want to try yet another addictive hobby!

Give it a stab

We used these shapes as a guide when we began working with our raw wool, and chose a natural colour for the main body to which we'd attach the other colours and limbs. We began by rolling a small amount of wool into a tight ball and, with our needles, started to form the shape required, adding more wool as needed. One thing to be aware of is that your shape will not really look like much for a little while. Quite a bit of needle-felting is required to interlock the wool fibres, so you'll be stabbing the wool for some time.

If you've never needle-felted before, we can assure you that it's not difficult, just a



Our completed robin may need a little more work, but we're happy with the effect

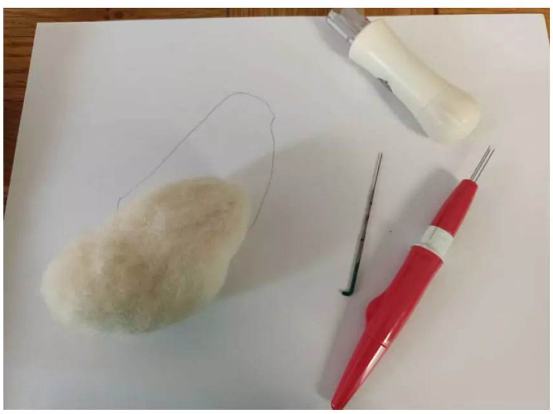
YOU'LL NEED:

- Roving wool (100% wool) in various colours
- Wire for the legs (e.g. floral or jewellery wire)
- Felting block or foam pad
- Felting needle/s
- Finger guard (optional)
- Scissors
- Awl
- Wire cutters/pliers
- Safety eyes (for toy making)
- Fabric glue (optional)
- Paper and pencil
- Washi tape (optional)
- Tiny FX microcontroller
- Mini speaker

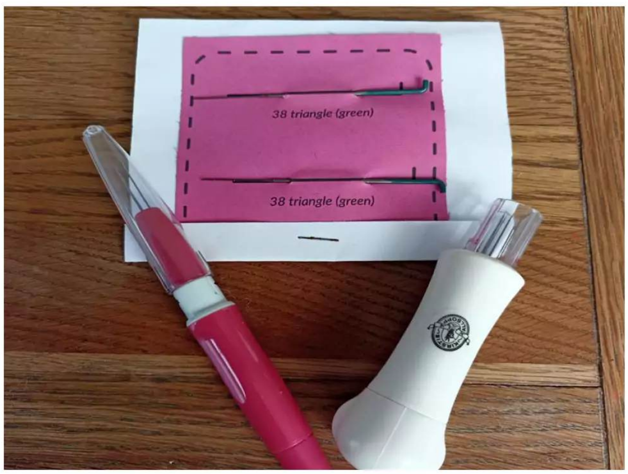
QUICK TIP

Record some authentic bird song in your garden or local park in the early hours to use as your bird sounds.

Quite a bit of needle-felting is required to interlock the wool fibres

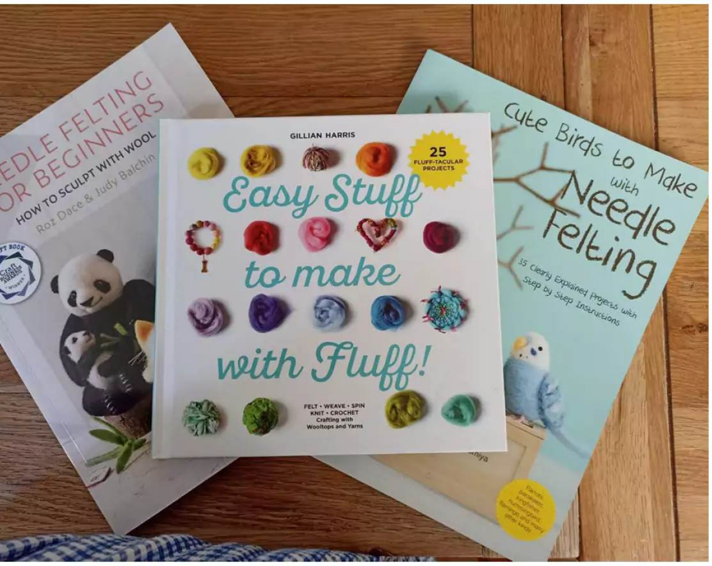


Needle-felting takes time, and it's worth spending a fair bit of it on the body so you have a solid base on which to add colour and other features. Make a paper template and use it as a guide



There's a vast array of needles that you can purchase for felting. We favour the spring-loaded five-needle tool as it speeds up the process a little, especially when you are forming the basic body. A pen-style holder is also a good option as a bare needle can be awkward to hold on its own

▼ As always, we can't recommend enough a trip to the library when you want to learn something new. There are a number of great books on the subject, taking you from beginner through to more advanced stages of needle felting. Written by experienced makers, they have been there and done it!



Tabletop engineering artistry

A singing model bird is nothing new. Automatons are a centuries-old concept and the songbird, in particular, was a highly popular 'parlour toy' which mimicked the sound and movements of birds.

They tended to be in the form of a brass cage containing the bird, and a base covered the intricate mechanism of gears and springs. Many such birds opened their beaks, moved their tails, or tilted their heads. A small bellows system was often used to force air through a pipe/whistle, and the bird's 'sound' was thus created.

A Parisian artisan, Blaise Bontems (1814–1893), was a specialist in the manufacture of singing birds and created a company in his name. He was particularly well-known for the realism of the sounds his birds made. Sometimes his creations were under a glass dome, with perhaps a clock included in the piece too. Often covered in real birds' feathers, the models are highly collectible and very expensive, if you can find a surviving example.

little time-consuming as you sculpt your shape from wool fibres using notched needles that tangle and condense the fibres as you put the needle in, and then pull it out. We began by using a five-needle tool that covered a slightly larger area, then progressed to a finer needle for the more detailed parts. You'll notice that, as you progress, the shape becomes more solid, and it will look worse before it looks better! Also, remember to keep your needle straight or the delicate tip will break.

When we were happy with the body shape, we took some washi tape and temporarily placed it halfway around the body so we had an idea of where we'd add the brown colour signifying the top part of the bird's body. (You could just draw a line with a pen.) Felting a colour on top of another is straightforward – we just smoothed a piece over the area and started jabbing, adding more as needed until the area was covered. We decided to give our bird a rustic look, so left some tufts of wool along the colour join lines, which we think makes it look more authentic and the joins less obvious. We repeated this process with the red for the red breast.

Next, add some character. We used our needle to create two indentations for the eyes, then used an awl to make a more distinct hole, after which we inserted two small safety eyes. We made a small beak from brown felt and glued it into place. How you add facial features such as eyes and beak is completely up to you. Some makers add small felted eyes; some make facial features from polymer clay and then paint them.

Using our wing teardrop template, we then made two thin wings, so not much wool was required. We had a very thin piece of wool larger than the template and worked around it with the needle to indent the shape, then turning the excess in toward the centre and felting until it was complete. Leave loose ends for a rustic look once again. If you want tail feathers, needle-felt those as well. None of these appendages should be bulky, and then you can just needle-felt them onto the main bird body with a few dozen jabs. Finally, we twisted some wire into shape for the legs, using pliers, and covered that with brown washi tape, inserting them into two holes we made in the body.

QUICK TIP

Sit your finished bird on an upside-down terracotta pot, and hide all of your tech gubbins under the pot.

The bird sounds are played through a mini speaker connected to an RP2040-powered Tiny FX board, triggered by pressing its BOOT button

Chirpy chirpy cheep cheep

To make our bird sing, we opted to use Pimoroni's RP2040-based Tiny FX board (**rpimag.co/tinyfx**). While it is possible to output audio from a Raspberry Pi Pico's GPIO pins to a powered speaker, the Tiny FX has an onboard amplifier to output I2S sound to a passive mini speaker. You can power everything with a 3 × AA battery pack or power bank. To trigger each bird call, we used the Tiny FX's BOOT button; alternatively, you could connect an external push-button to the board's Sensor or Qw/ST port.

We downloaded some bird calls from **xeno-canto.org** – a vast database of user-uploaded bird calls, available to download and use for free under a Creative Commons licence. To remove unwanted silences, we used Audacity (**audacity.org**) to trim the audio. We exported our sounds as WAV files, put them in a folder (called 'Robin'), and uploaded them to the Tiny FX using Thonny IDE's Files tab on a connected computer.

Written in MicroPython, and making use of the `tiny_fx` library, our code (**birdcalls.py**) waits for a button press and then plays the next sound file from the `sounds` list, selected using the `idx` variable index number.

You can add extra sound files to your list. To make the code run automatically upon bootup, save it with the filename **main.py**. Optionally, you could fit the board (or external push-button) inside the bird so that when you squeeze it, it starts chirping. Or you could trigger it using a connected sensor.

Why not felt your own unique animal, or even human, creation and then give it a voice too? You could then progress to emulating the automatons of yesteryear, and try and add movement to one part of the project – for example, you could add a jointed tail that moves, and use Raspberry Pi to help make that happen. Just try some mindful needle felting and see where it takes you...

QUICK TIP

Always start with less wool than you think you need… you can't easily remove it if you start with too much! Less, as they say, is more.

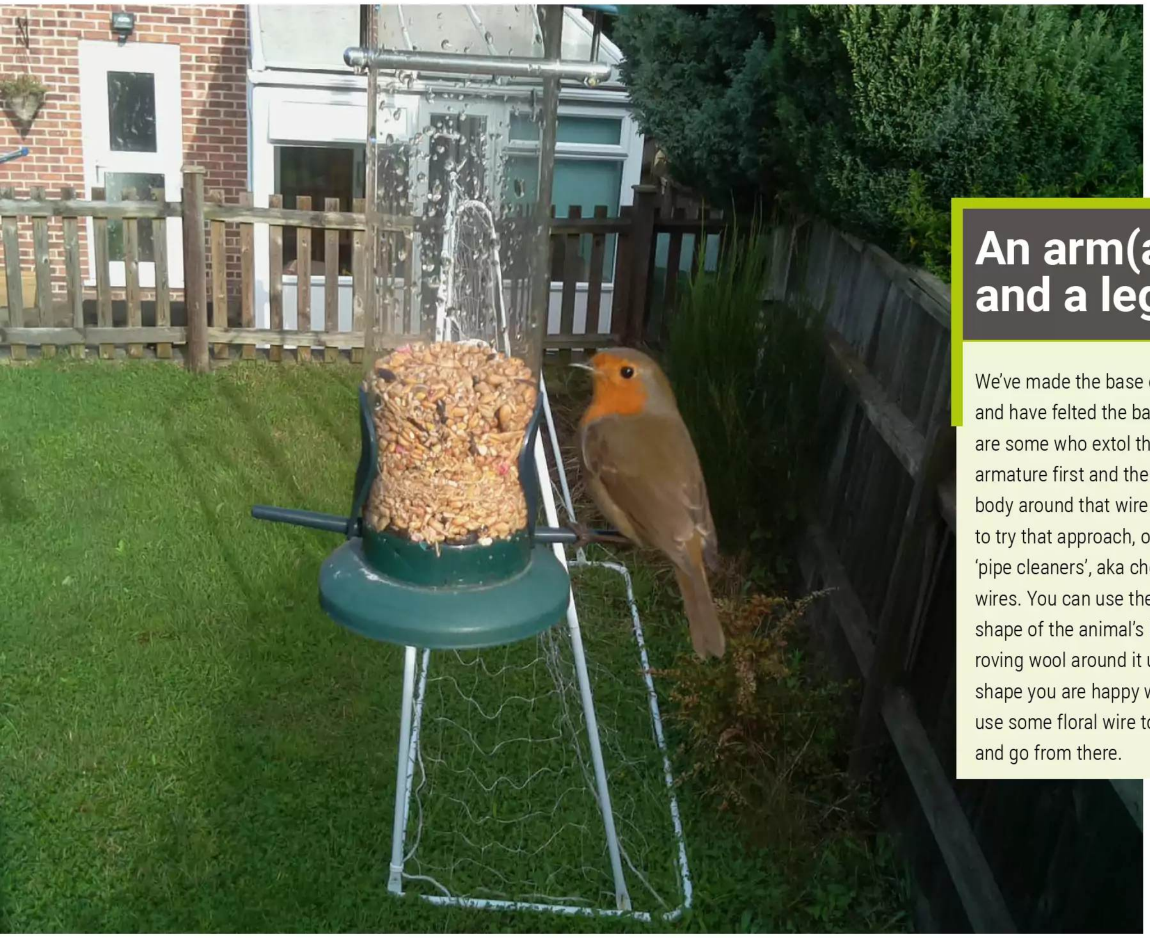
birdcalls.py

> Language: **MicroPython**

DOWNLOAD THE FULL CODE:

rpimag.co/github

```
import time
from tiny_fx import TinyFX

# Create a new TinyFX object and point it to the files
tiny = TinyFX(wav_root="/Robin")

# Define a list of sounds
sounds = ["Robin1.wav",
          "Robin2.wav",
          "Robin3.wav"]

idx = -1 # Set list index number

# Wrap the code in a try block, to catch any exceptions
try:
    last_state = False
    while True:
        pressed = tiny.boot_pressed()
        if pressed and not last_state:
            # Button just pressed: play next sound
            idx = (idx + 1) % len(sounds)
            tiny.wav.play_wav(sounds[idx])
            while tiny.wav.is_playing():
                pass
        last_state = pressed
        time.sleep(0.05)

finally:
    tiny.shutdown()
```



An arm(ature) and a leg

We've made the base of our bird from scratch, and have felted the base body first. There are some who extol the virtues of making an armature first and then building the felted body around that wire structure. If you want to try that approach, one way would be to use 'pipe cleaners', aka chenille stems or furry wires. You can use these to just make a rough shape of the animal's body, then felt some roving wool around it until you have a base shape you are happy with. Alternatively, just use some floral wire to make your 'skeleton' and go from there.

A robin in the author's garden, captured by a Raspberry Pi wildlife camera

Wet felting (on purpose!)

We used the dry felting method in our project, but it is possible to use water, soap, and agitation to felt wool fibres. The science is simple: wool is sensitive to temperature changes, moisture, friction, and alkali in soap, so if you add hot water to wool fibres, the wool will condense and when the fibres are rubbed against each other, they'll bind and mat together creating a dense fabric that cannot be returned to its former state.

Some knitters use this technique to felt their creations. Because the fibres are fused, the final felted item will be stronger and more durable. Virtually all stitch definition will be lost in the felting process once the fibres bind, but felting gives a unique and interesting look to a piece. Once felted, you can even cut the fabric without it unravelling.

Other popular items to felt include oven mitts, purses, phone cases, etc. Technically, this process is known as 'fulling', but the common usage term is 'felting'. The maker will factor in the fact that the item will shrink when they come to felt it, so will likely make it larger to accommodate that fact.

Here's a handy guide to how to felt a piece of knitting, giving it something of a makeover: **rpimag.co/feltknitting**. If you are a knitter, this great book by Arne and Carlos is also worth a glance: *30 Slippers to Knit and Felt*.

QUICK TIP

The more time you spend felting a project, the firmer and smoother your piece will become – invest some time.

W6300

The Ethernet Solution for Faster, Smarter Networking!

Need ultra-fast, stable Ethernet?
W6300 delivers up to 90 Mbps with powerful IPv4/IPv6 dual stack support!

Key Benefits

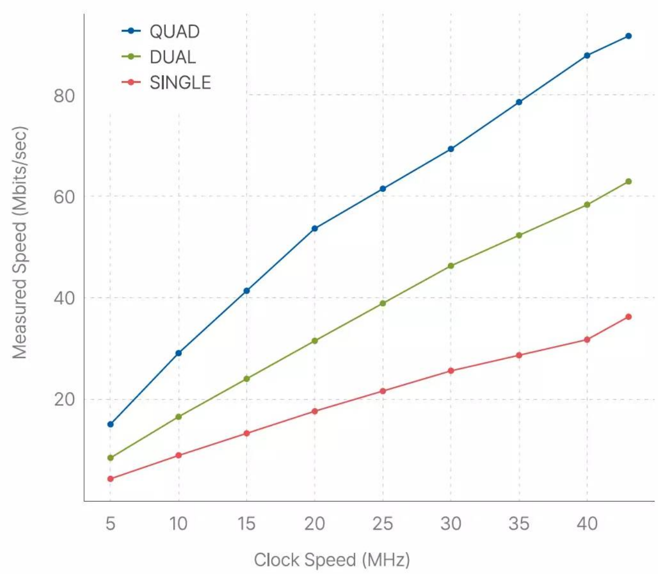
- **High-speed QSPI Interface**
  SINGLE, DUAL, QUAD modes supported
- **8 Independent Sockets**
  Multi-device connectivity made easy
- **Low Power & Wake on LAN**
  Efficient and remote-friendly
- **Seamless IoT & Industrial Integration**
  POS, cameras, automation & more

Where to Use W6300?

- **Smart Home & IoT**
  Network sensors, smart APs, home automation
- **Industrial & Factory Automation**
  POS, network printers, LED displays
- **Security Systems**
  DVRs, CCTV, access control systems
- **Embedded Servers & Cloud Devices**

Performance Test Results

- iPerf3 Performance
  Up to 90 Mbps on RP2040/RP2350 in 43MHz QUAD mode
- Optimized speed across different QSPI modes



WIZ630io

W6300-EVB-Pico
(W6300 + RP2040)

W6300-EVB-Pico2
(W6300 + RP2350)

Unusual tools: cable tracer

The tool that makes your infrastructure sing. By **Andrew Lewis**

Maker

Dr Andrew Lewis

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

lewiswork.etsy.com

▲ Some cable tracers are more like a multi-tool. It isn't unusual to find a unit with a continuity tester, cable tester, and even LED lights built in. Some units also have headphone sockets that can be used to reduce disruption in quiet environments or make it easier to hear tones in noisy environments

Very few people get to live the dream of building their infrastructure from scratch. Dealing with a legacy setup can make it difficult to understand the exact route a cable takes through a building, and finding the right cable when you have bundles of unlabelled copper dangling out of your cable sleeving can be a complete nightmare. In fact, even when you do have the luxury of running your own infrastructure from scratch, it's still amazing how confusing the route even well-ordered data spaghetti can become as a project progresses. That's when you need a cable tracer.

There's little mystery to the purpose of a cable tracer. It's an extremely well-named piece of equipment that helps you to trace a cable. However, it does go by several other names (a singer, a cable toner, a wire sniffer, a wand) which can be confusing if you're not familiar with the particular terms. A cable tracer has two parts: a tone generator (also called an oscillator) and a probe (also called a wand). The tone generator will normally have at least two wires coming out of it, with crocodile clips and sometimes an RJ11 or similar plug.

Turning on the tone generator and connecting the crocodile clips to a pair of wires will make the probe generate a tone when it's near to the wire. Shorting the wires together (it's OK

QUICK TIP!

The tone generator is designed for working with disconnected wires. Never connect it to wires that have electricity running through them.

to short them out, it's an expected use case for the tone generator) will make the tone stop, so you can easily detect that you're tracing the right pair of wires by just touching the wires together at the other end of the cable.

Cable tracers can have different features aimed at different markets, and if you spend more cash you might find that you literally get a few extra bells and whistles. Cheaper tracers will only produce a continuous tone, while more expensive models will be able to produce a range of pitches and different interrupted tones. Multiple tones can be useful in busy environments where multiple engineers are working, or where you need to trace cables from several different sources at the same time. The range of signal can be better with more expensive units, although even a cheap tone generator will normally work through several hundred metres of cable. More modern tracers can inject a digital signal rather than just an analogue tone, which the probe can detect and decode with LED or LCD indicators.

Multiple tones can be useful in busy environments

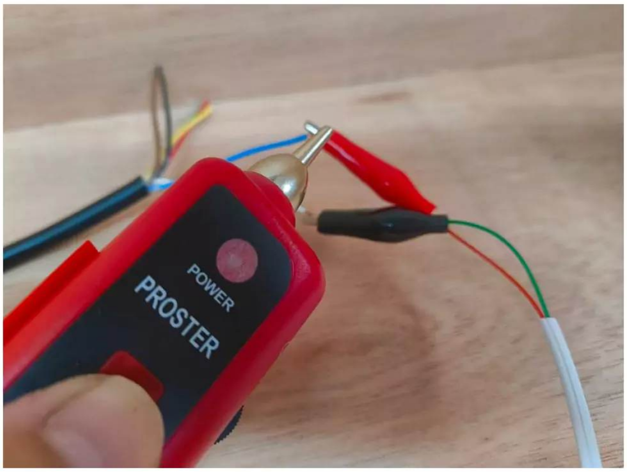
The probe itself is very easy to use, and on the simplest models will just have a power and signal LED, and a speaker with a volume control. Moving the probe close to a wire that's connected to a tone generator will cause the wand to flash and/or the speaker will start playing the generated tone. More modern meters may also have an LCD that indicates a number or letter corresponding to the digital signal injected by a tone generator.

Unfortunately, cable tracers aren't a panacea for wire mapping and they do have some limitations that you should be aware of. Most obviously, cable tracers only trace bare cables. There are a few specialist tools that can work with live cables, but this is an exception and not the normal operation mode for a cable tracer. If there is a break in the cable you are tracing, the tone will not propagate past that point. Although it's a problem if you're expecting to hear the tone at the end point of the cable, it can be used to your advantage to discover exactly where the cable is broken.

A short circuit, signal noise suppressor, or soft-start circuitry wired into electrical cable will mean that the tone generator probably won't work. Additionally, if the cable is joined somewhere to another cable or somehow has a ground connection, you might find that chaos will ensue with every single shielded cable generating a tone.

▲ Some cable tracers can be clipped together to make transport easier, and have removable cable clips with an RJ11 or RJ45 socket built into the tone generator. This can be useful for making your own custom cables with tailored connections, and makes it easier to repair the unit if the clips or cables get damaged

▼ Holding the probe next to the generator when the units are active, you should be able to hear the tone once you push down the 'test' button. On some units aimed at computer networking or telephony workers, the probe will also have a network socket that acts as one half of a network cable tester, with the signal generator having a matching socket on the other end of the wire



Play Steam games using x86-64 emulation

Using Box64, we can play modern Intel/AMD PC games from Steam, GOG, Itch, and more

Maker

KG Orphanides

KG is a game developer, writer, and musician with a special interest in retro gaming. They have worked for Kitsune Games, mentioned in this feature.

rpimag.co/owlbear

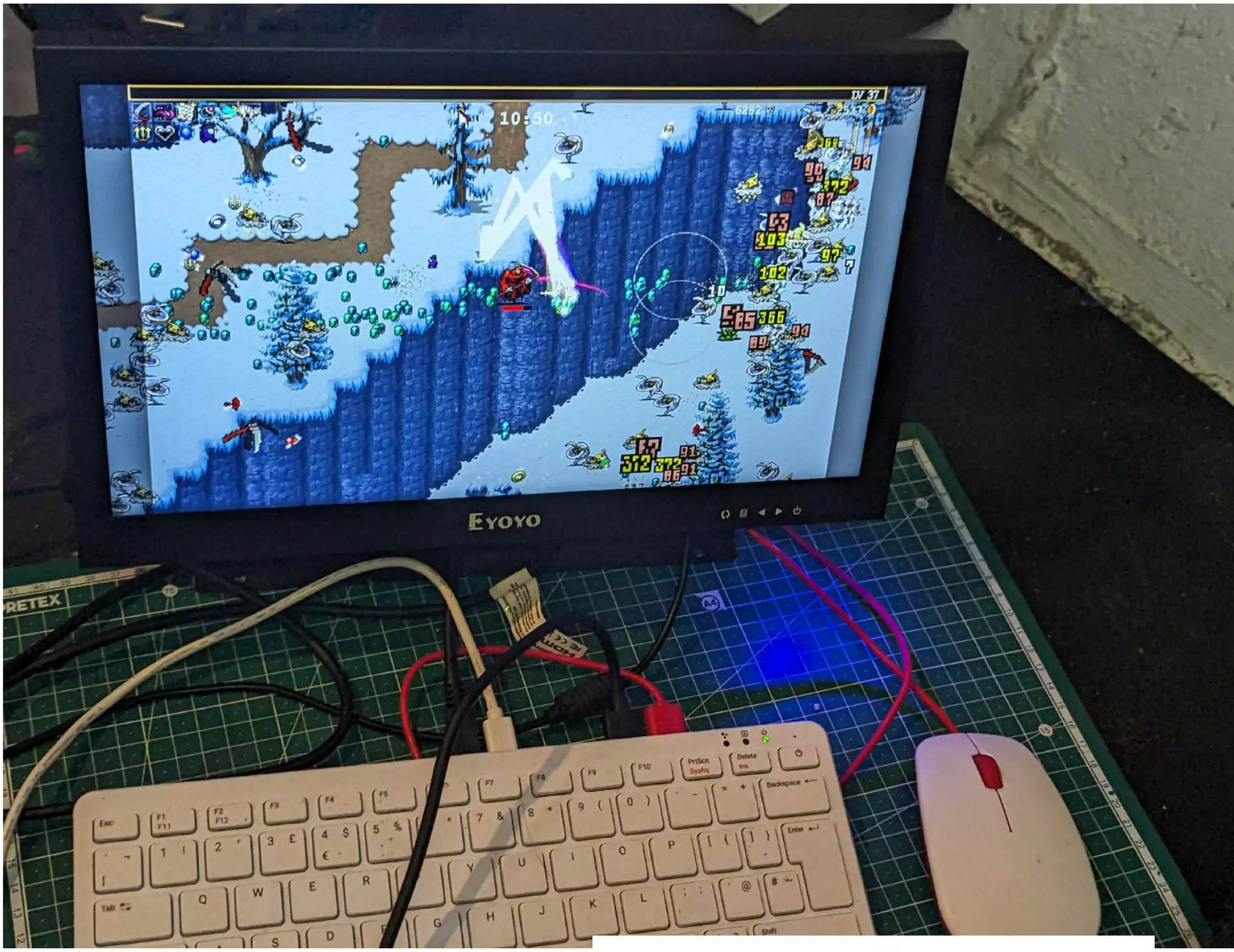
Raspberry Pi is built on ARM64 architecture, which isn't compatible with most games, although some have been specifically ported, and retro games can be played via ARM64-compatible rebuilds of their game engines or emulation tools such as DOSBox.

Although GOG is usually our first port of call, this month we've mostly been playing games on Steam, the most popular and Linux-friendly gaming storefront. We're getting our software from the Pi-Apps repository, which gives you custom builds of popular applications for ARM64 hardware.

> *Manage your expectations when it comes to performance*

Raspberry Pi 5, 500, and 500+, in particular, are powerful enough for everyday use as a desktop PC, but running modern Linux or Windows games can be a power-intensive prospect. We're dealing with multiple compatibility layers in some cases here. Steam is running on top of Box86's emulation of AMD64 CPU architecture; in turn, many games on Steam will use Proton – which allows Windows games to be played on Linux. Despite the performance hit this produces, we've had luck running a number of Steam games.

Manage your expectations when it comes to performance and be prepared to engage in some trial and error. Even if you're only installing the tiniest of games, you'll need to ensure that you have at least 10GB of free disk space to install and run them. Use a high-speed microSD card for your hard disk (or an NVMe HAT) if at all possible if you're not using a 500+.



▲ We were able to get in a round of Vampire Survivors without too much slow-down, although menus lagged

Change your page size

Unlike previous generations of Raspberry Pi, Raspberry Pi 5/500/500+ default kernels both ship with a 16kB page size set by default – this defines minimum unit size for memory management. A larger page size improves memory access speeds. Unfortunately, none of the emulators of Intel and AMD x86/x64 hardware support 16kB page sizes.

Check your current page size, and then open the config file to manually change it if needed.

```
$ getconf PAGESIZE
$ sudo nano /boot/firmware/config.txt
```

Under `[all]`, add `kernel=kernel8.img`

Press **CTRL+X**, then **Y** to save the file and quit, then reboot your Raspberry Pi.

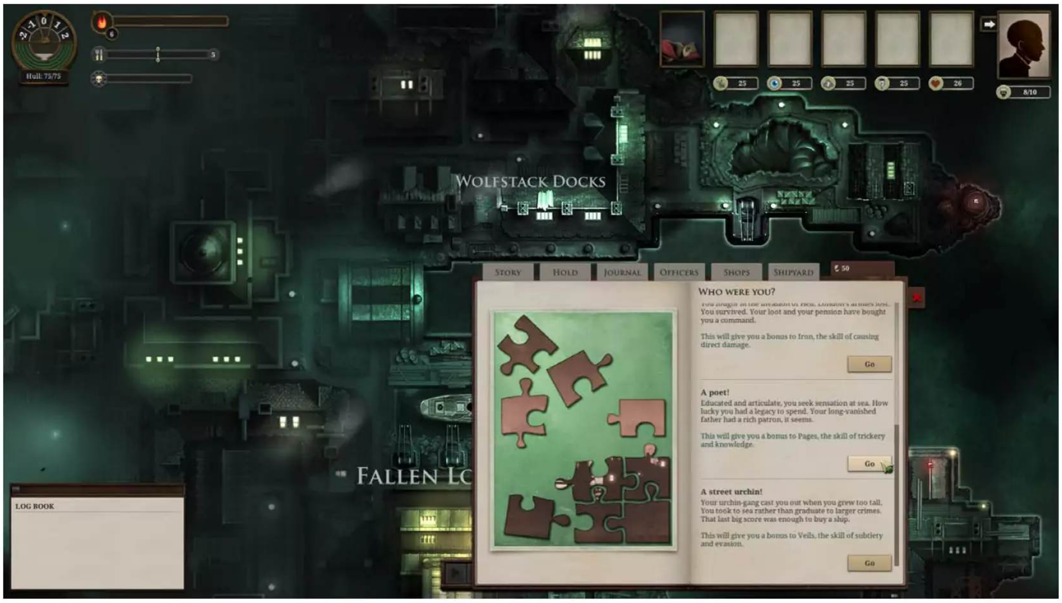
The wisdom of crowds

Running Steam games on Raspberry Pi is, for now, a minority interest, but while researching this article, we found a number of users who'd been kind enough to list the titles they'd got working. We didn't have success with all of these, but they'll certainly give you a jumping-off point if you cross-reference them with your Steam library.

Steam games which work on Raspberry Pi 5: **rpimag.co/steampi**

A list of games which run on a Raspberry Pi 5: **rpimag.co/gamepi**

Gaming via x86 Steam with Box86: **rpimag.co/gamebox64**



◀ We used Wine – running via Box64 – to install and run the Windows version of Sunless Sea from GOG

Run Steam on Raspberry Pi 5/500/500+

The easiest way to get started emulating x86 and AMD64 architectures is to the use Pi-Apps installer (**pi-apps.io**), which provides a repository of popular applications, customised for Raspberry Pi's ARM64 architecture. It will even offer to select the 4kB page size kernel for you when you install an app that requires it.

First, install Pi-Apps. Open a terminal and type:

```
$ wget -qO- https://raw.githubusercontent.com/
Botspot/pi-apps/master/install | bash
```

When it's finished installing, it'll point you at further instructions if you need them. You can now close the terminal window. You'll find a Pi-Apps shortcut on your desktop or in the Accessories section of the main menu. Double-click on the desktop icon to launch it, and choose to run it in the terminal – this means you'll be able to see what it's doing in real time. Make sure you have a decent internet connection, as some of the files we're downloading are large and will fail to download if you've insufficient bandwidth.

To install Steam, select the Games menu and then select Steam from the menu. Read the details of how the application works, then click Install. This will also install Box64, which we can use to run other x86 and x64 applications.

You should also install More RAM from Pi-Apps. This disables the swap file and instead uses on-the-fly compression to squeeze more space out of Raspberry Pi's physical memory.

After rebooting Raspberry Pi, you'll find a shortcut to Steam in the menu under Games. Run the program and it'll download and update the Steam executable. Log into your account using either Steam's 2FA app or your username, password, and the 2FA code that'll be emailed to you. Steam will open on your Friends chat window and game library.

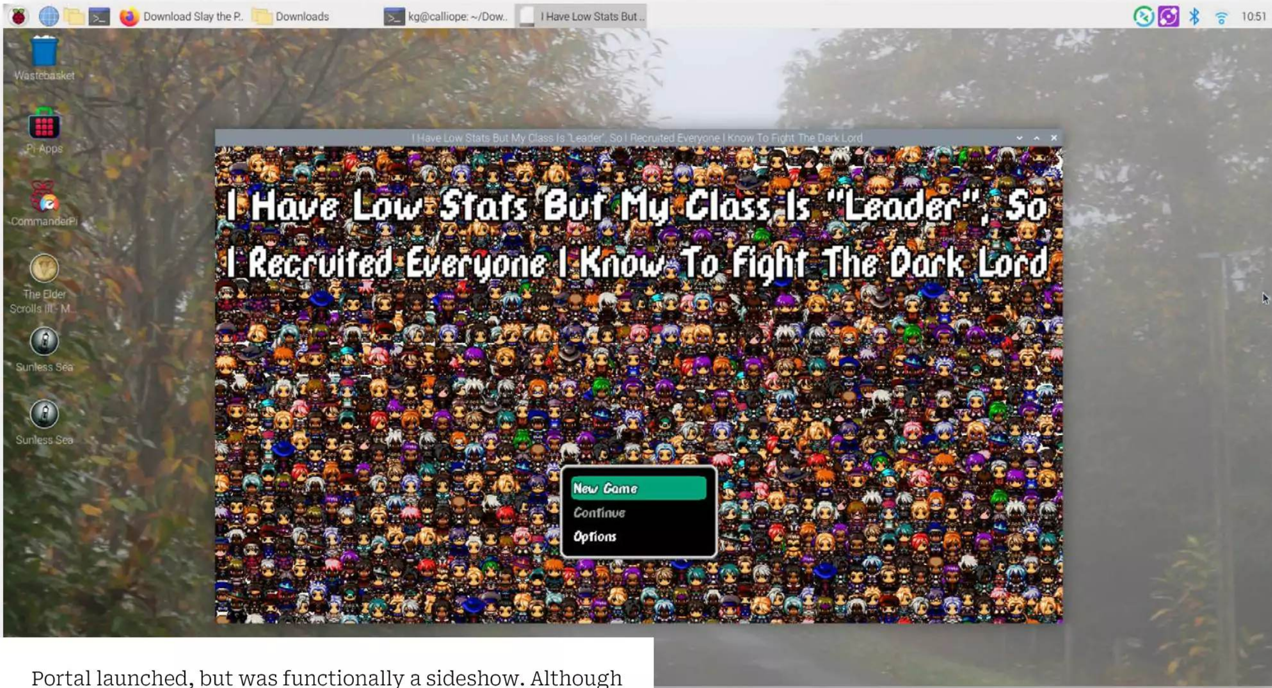
What Steam games can you run on a Raspberry Pi?

Small games are a good start – roguelite card game hit Balatro works brilliantly, and seeing fans of Mario-inspired platformer Kitsune Tales run it on their own Raspberry Pi hardware is what inspired this feature.

Steam itself is sluggish and slow to respond, and games can take a couple of minutes to launch. Once launched, however, those that'll actually deign to run at all often do so surprisingly well.

You'll find that a number of 2D indie games and older titles work. We were able to run Into The Breach, Vampire Survivors (with a little lag when opening menus), and Long Live the Queen.

Unfortunately, other titles that have been reported as functional in internet forums worked less well for us. Valve's Source engine games can be run, but not in a way that we'd consider playable.



▲ Windows versions of RPG Maker games such as I Have Low Stats But My Class Is "Leader", So I Recruited Everyone I Know To Fight The Dark Lord, which we downloaded from Itch, also worked well via Wine

Portal launched, but was functionally a sideshow. Although some users on Steam's forums say they got 3D game performance up to 25–30fps at minimal settings, we were unable to reproduce this by dropping Portal's resolution. Which isn't necessarily to say that trial and error won't produce better results, but we wouldn't get our hopes up.

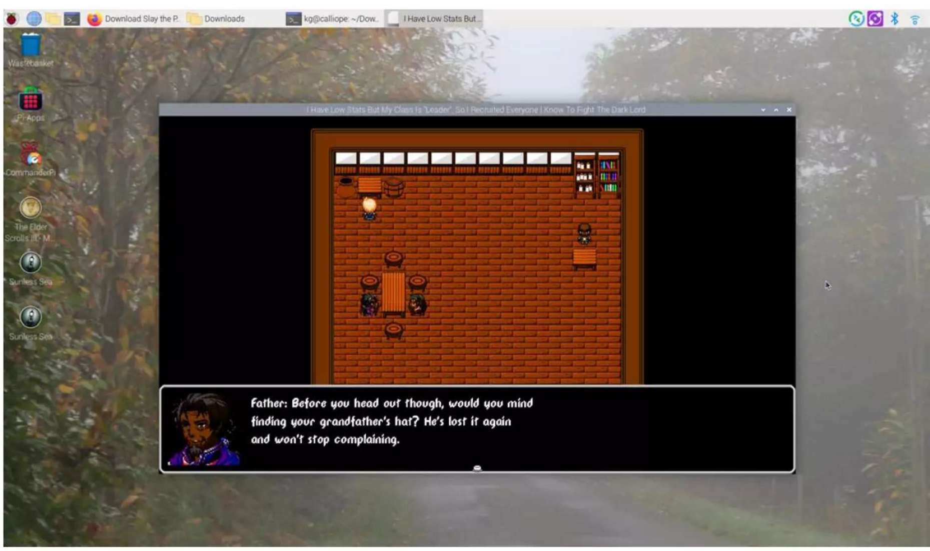
Beyond Steam

Your Steam install can also stream Steam games installed on another, more powerful computer on your network, but bear in mind that there's a native Raspberry Pi Steam Link client (**rpimag.co/steamlink**) that will stream your Steam games from another PC without the overhead of x86/x64 emulation.

In some cases, particularly re-releases (rather than remasters) of retro titles, you can install them via Steam, but – once they're unpacked – you can use an ARM64 native application such as DOSBox-X or ScummVM to run them, without the need to add all that extra weight of compatibility and emulation layers.

You can also use Box64 to run Wine, allowing you to install and play non-Steam games, such as GOG's Windows installers and downloadable Windows and Linux games from Itch.io. When installed via Pi-Apps, Wine automatically runs through Box64. If you want to use a Linux installer or other executable, you can invoke `box64` at the command line before you use `sh` to run it, but we generally had more luck with Windows versions, including Failbetter's Sunless Sea.

▼ Playing the I Have Low Stats… RPG on Raspberry Pi



Part 02

Control LEDs with GPIO Zero

Turn LEDs on and off with just a few lines of code, and build a traffic light system

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
- 3× LEDs (one each of red, yellow, and green)
- 3× 330Ω resistors
- 4× pin-to-socket jumper wires

One of the first physical computing projects you'll want to try with GPIO Zero is lighting an LED. This is very simple with the library's `LED` class and uses few lines of code. Here we'll show you how to wire up a simple circuit connected to your Raspberry Pi's GPIO pins, then light an LED and make it blink on and off. We'll then add two more LEDs to make a traffic light system.

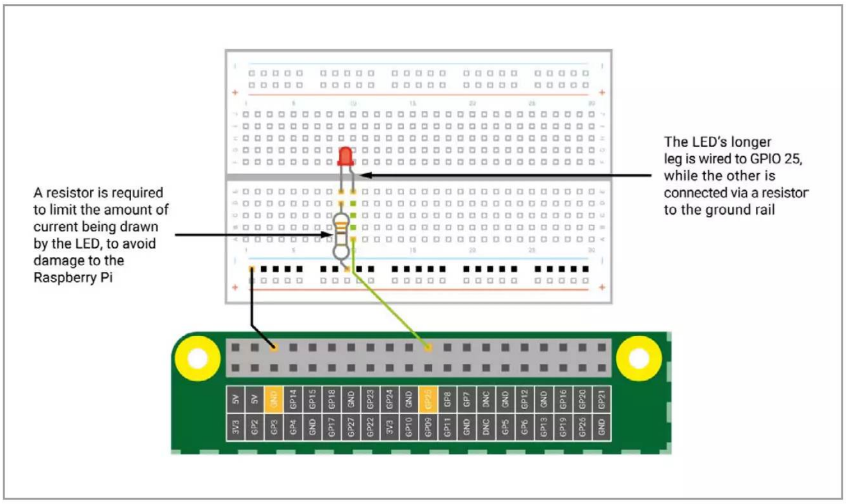
Connect an LED

Before building a circuit, you must shut down your Raspberry Pi and disconnect it from power.

Most breadboards feature numbered columns, each containing one group of five holes, a groove, and another group of five holes. Within each group, all five holes are connected, but the holes on either side of the groove aren't connected. If you're holding your breadboard in a different orientation than shown in **Figure 1**, simply swap 'columns' for 'rows' as you read this section.

Place your red LED's legs in adjacent numbered columns, as shown in the figure. Note that the shorter leg of the LED is the negative end. Next, insert one end of the resistor into one of the other four holes within the same column and group, then place the other end in the outer row marked '–'(the ground rail). Use a pin-to-socket jumper wire to connect any hole in that ground rail to a GND pin on the GPIO header. Despite the gaps in the rails, all the holes in a rail are usually connected to each other. However, some larger breadboards have discrete groups of holes amongst the rails.

Finally, use a jumper wire to connect a hole within the same column and group of the LED's longer (positive) leg to GPIO 25. **Figure 2** shows the assembled circuit.



▶ **Figure 1:** Wiring up an LED

Light the LED

Let's test our circuit with a simple Python program to turn the LED on and off. To edit your Python code, you can use any plain text editor, including console-based editors such as nano, emacs, or vi. If you prefer a more full-featured development environment, try Thonny: click the Raspberry Pi menu, then choose Programming > Thonny. Create a new file by clicking New (the green plus icon). Next, enter the following code and save it by clicking the Save icon, naming it **blink_led.py**.

To use a console-based editor such as nano, open a Terminal window or SSH into your Raspberry Pi. Enter the following code and save it (in nano, press **CTRL+O**, type the filename, and press **ENTER**).

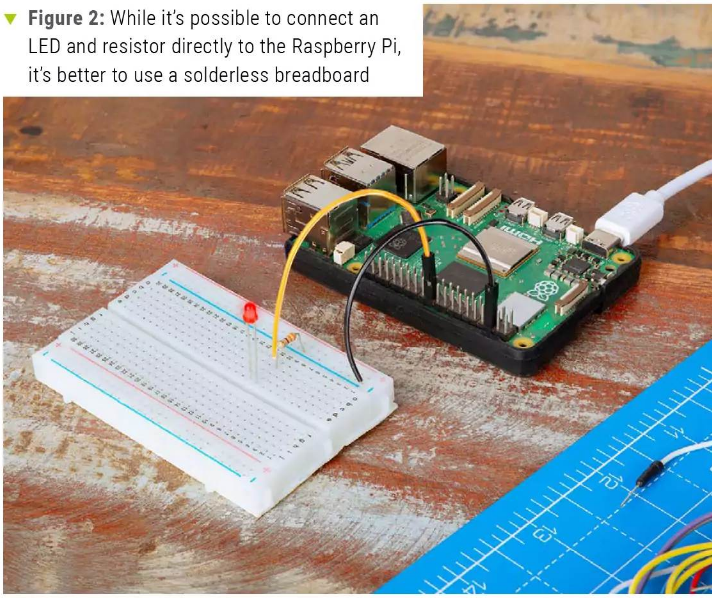
```
from gpiozero import LED
from time import sleep
led = LED(25)
while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

At the start of the program, we import the `LED` class from GPIO Zero, and the `sleep` function from the `time` library (to enable us to pause between turning the LED on and off). We then set the `led` variable to the GPIO 25 pin, which will power it whenever we set it to on in the code. Finally, we use `while True:` to create a never-ending loop that switches the LED on and off, pausing for 1 second between each change.

In Thonny, press **F5** or click the Run icon to run the code, and your LED should be flashing on and off. To stop the program, click the Stop icon. If you're using a console-based editor, exit it (in nano, use **CTRL+X**), then run the code with the command `python blink_led.py`. To stop the program, type **CTRL+C**.

Wire up a simple circuit connected to your Raspberry Pi's GPIO pins, then light an LED and make it blink on and off

▼ **Figure 2:** While it's possible to connect an LED and resistor directly to the Raspberry Pi, it's better to use a solderless breadboard



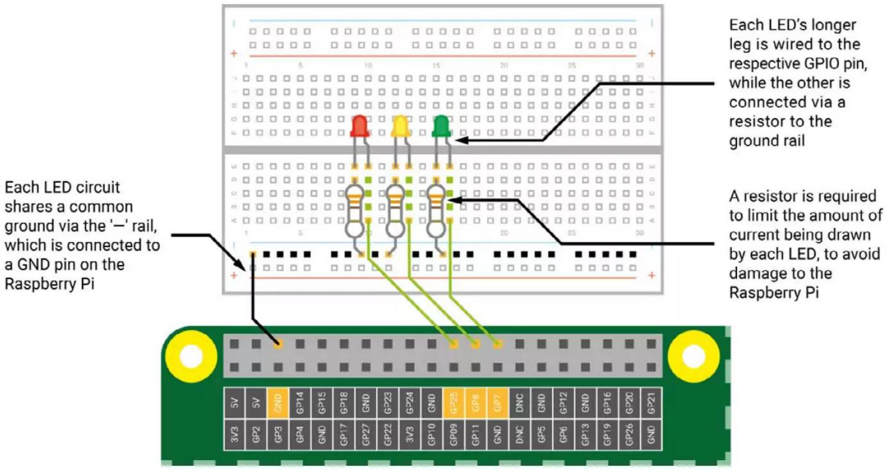


Figure 3: Wiring up a traffic signal

Figure 4: The assembled traffic signal

Easier blinking

Alternatively, to make things even easier, GPIO Zero features a special `blink` method. You could try this code, which does exactly the same thing as the first listing, but with even fewer lines of code:

```
from gpiozero import LED
from signal import pause
red = LED(25)
red.blink()
pause()
```

Note that between the brackets for `led.blink`, you can add parameters to set the on and off times, number of blinks, and determine whether it runs as a background thread or not, as in:

```
red.blink(on_time=1, off_time=2, n=3,
background=True)
```

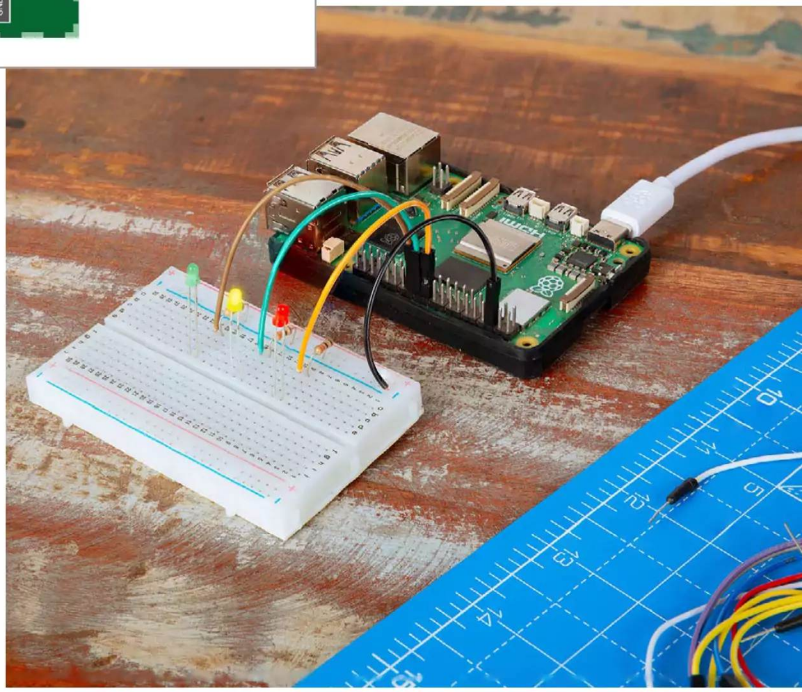
Blink multiple LEDs

Now that we've got the hang of controlling one LED, let's use three LEDs to create a traffic light sequence. You can add the optional push button and a buzzer if you like, but in this tutorial, we'll only explain the LEDs.

Connect them as shown in the wiring diagram (**Figure 3**), with the longer (positive) legs connected to a resistor that straddles the groove in the breadboard. Use jumper wires to connect the side of the resistor that's across the groove to the following GPIOs: 25 (red), 8 (yellow), and 7 (green).

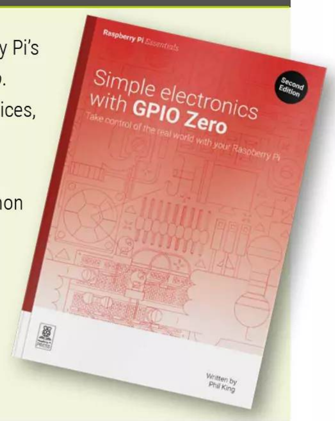
Note that the placement of the resistor is different than the previous circuit: instead of putting the resistor between the LED and ground, we're putting it between the LED and the positive voltage supplied by the GPIO pins. It doesn't matter which way you do it, but the resistor must be part of the circuit.

You'll also need to connect each LED's shorter (negative, or ground) leg to the '–' rail. Like before, the '–' rail must be connected to one of the GPIO GND pins. **Figure 4** shows the assembled project.



Simple electronics with GPIO Zero

This article is an extract from Raspberry Pi's book, *Simple electronics with GPIO Zero*. Updated for the latest Raspberry Pi devices, this book has all the info you need to start creating electronic projects using Raspberry Pi's GPIO pins. Coded in Python with the GPIO Zero library, projects include LED lights, a motion-sensing alarm, rangefinder, laser-powered tripwire, and Raspberry Pi robot.



rpimag.co/gpiozerobook

Enter the code

Open your code editor, type in the following code, then save the file as **traffic_signal.py**:

```
from gpiozero import LED
from time import sleep

red = LED(25)
amber = LED(8)
green = LED(7)

green.on()
amber.off()
red.off()
while True:
    sleep(10)
    green.off()
    amber.on()
    sleep(1)
    amber.off()
    red.on()
    sleep(10)
    amber.on()
    sleep(1)
    green.on()
    amber.off()
    red.off()
```

As before, we import the `LED` class and `sleep` function from GPIO Zero and the `time` library respectively. We then assign `red`, `amber`, and `green` variables to the relevant GPIO pins. To start with, we turn the green LED on and the others off. Finally, we use `while True:` for a never-ending loop; this waits 10 seconds before showing amber then red, then waits another 10 seconds before showing red/amber then green. If you're in Thonny, press **F5** or click the Run icon to run the program and wait for the traffic light sequence to start. If you're using the command line, exit your editor and run the code with `python traffic_signal.py`.

Rather than running through a fixed sequence, you could request that the signal stop traffic with the addition of a push button to alert pedestrians when it's safe to cross. In both cases, you'd need to write some code to work with those components.

DOWNLOAD THE FULL CODE:

rpimag.co/gpiobookgit

traffic_signal.py

> Language: **Python**


```
from gpiozero import LED
from time import sleep

red = LED(25)
amber = LED(8)
green = LED(7)

green.on()
amber.off()
red.off()
while True:
    sleep(10)
    green.off()
    amber.on()
    sleep(1)
    amber.off()
    red.on()
    sleep(10)
    amber.on()
    sleep(1)
    green.on()
    amber.off()
    red.off()
```

Part 02

Make games with Python: Animate shapes and paths

Move shapes around the screen — in different directions and patterns, and at different speeds

Maker

Sean M Tracey

Sean calls himself a technologist, which is a fancy way of saying he still hasn't decided what he wants to do with technology – other than everything.

smt.codes

In the previous tutorial, we looked at creating a variety of shapes in different sizes and colours. Now we're going to be looking at different ways of moving and manipulating those shapes over time. This tutorial covers the fundamentals of moving shapes with code; next time we'll discuss using keyboard and mouse events to control how and when things move. In this tutorial, we won't be using one single Pygame program. Instead, we have a couple of different examples, each demonstrating a different concept.

Moving shapes in time and space

When we think of animation, our minds might turn to cartoons and animated films where subtle changes in shape and colour trick our brains into seeing movement where there is none. It's no different with computers: whenever you move a mouse or minimise a window, nothing has actually been moved; instead, pixels have been drawn, updated, refreshed, and then drawn again, with everything in its new place.

QUICK TIP

By default, our window is called 'Pygame window'. We can set that to anything we like, for example: `pygame.display.set_caption('Pygame Shapes!')`

A game's response to a player's actions is the result of hundreds and thousands of these little checks

Save the following program as **random_rect.py** and run it:

```
import pygame, random
pygame.init()
clock = pygame.time.Clock()

WIN_WIDTH = 640
WIN_HEIGHT = 480
window = pygame.display.set_mode((WIN_WIDTH,
WIN_HEIGHT))
pygame.display.set_caption('Pygame Shapes!')

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            raise SystemExit

    # Begin drawing statements
    window.fill((0,0,0))
    x = random.randint(0, WIN_WIDTH)
    y = random.randint(0, WIN_HEIGHT)
    pygame.draw.rect(window, (255,0,0), (x, y,
10, 10))
    # End drawing statements

    pygame.display.update()
    clock.tick(60)
```

If you run this program, you'll see a bunch of red squares appearing and disappearing all around the screen, as shown in **Figure 1**. Don't worry, nothing is broken! This is just to demonstrate Pygame drawing, destroying, and redrawing things in a window.

Add a `#` to the start of the line that starts `window.fill()`. We use this code to clear the pixel data from the previous frame. Without it, what we see is all of the different frames built up one on top of the other as time passes. `window.fill()` is like the paint that we use to cover old wallpaper before we add the new one: it creates a blank slate for us to work with.

But that's not very useful, is it? Remove the `#` you added, then add the following two lines before the `while True:` line:

```
green_square_x = WIN_WIDTH / 2
green_square_y = WIN_HEIGHT / 2
```

Next, replace the code between the `# Begin` and `# End` comments so that it reads like this:

```
    # Begin drawing statements
    window.fill((0,0,0))
    pygame.draw.rect(window, (0, 255, 0),
(green_square_x, green_square_y, 10, 10))
    green_square_x += 1
    #green_square_y += 1
    # End drawing statements
```

Save the modified file as **moving_square.py**, run it, and you'll see a green square moving to the right of the screen.

So, what's making the square move? In, we were drawing shapes like this using numbers that we would pass through to Pygame, as in `pygame.draw.rect(window, (0,255,0), (40, 0, 50, 50))`, and that's all well and good, providing you never want to change anything about that shape. What if we wanted to change the height, width, or colour of this shape? How could we tell Pygame to change the numbers that we've already entered? This is where variables come in. Rather than passing through numbers to `pygame.draw.rect()`, we pass in variables instead.

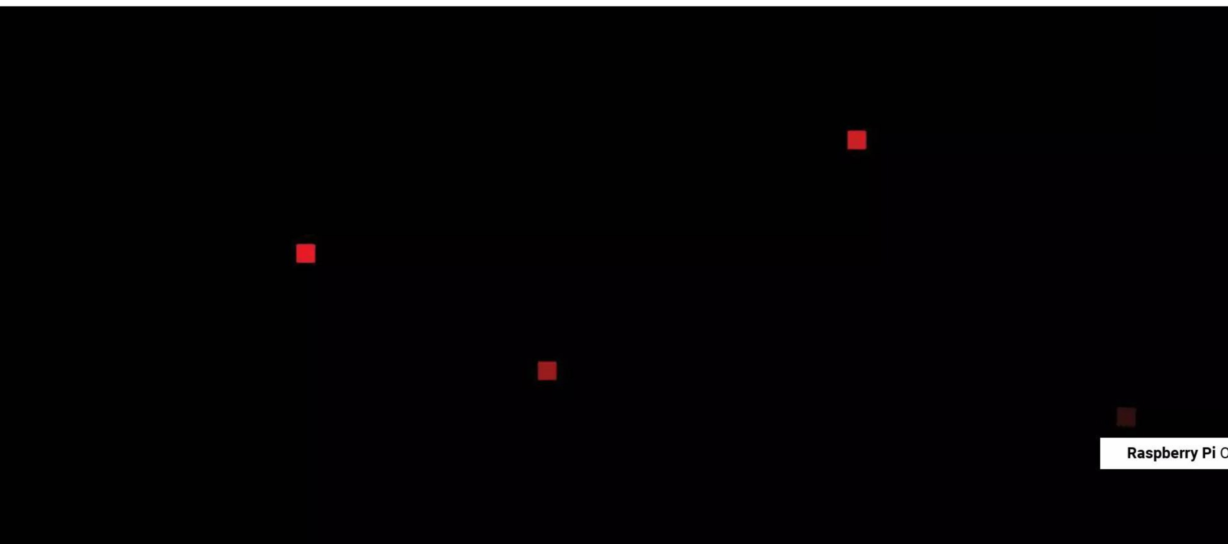


Figure 1: A simulated screenshot showing the random placement of red squares in our window

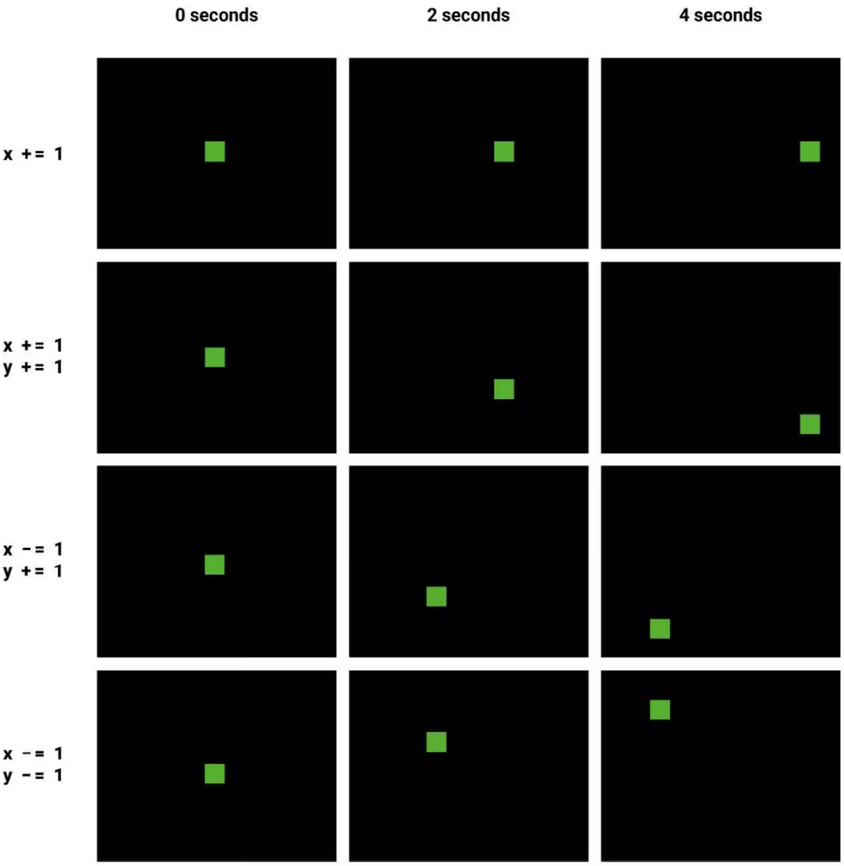


Figure 2: How different motions affect the position of a shape over time

After we've drawn the shapes, we can change the variable so that when it's next drawn, it will look slightly different (see **Figure 2**). Every time we draw our green square, we add 1 to the variable we use to define its X coordinate (how far it is from the left of the screen), `green_square_x`. We do this with `+=`, which basically says 'take the current value of the variable and then add whatever number comes after it'.

If we change that line to read `green_square_x += 5`, every time we draw our square, it will be 5 pixels to the right of where it was the last time it was drawn. This gives the illusion of the shape moving faster than before. If we changed the number we add to `green_square_x` to 0, our shape would never move; and if we changed it to -5, it would move backwards.

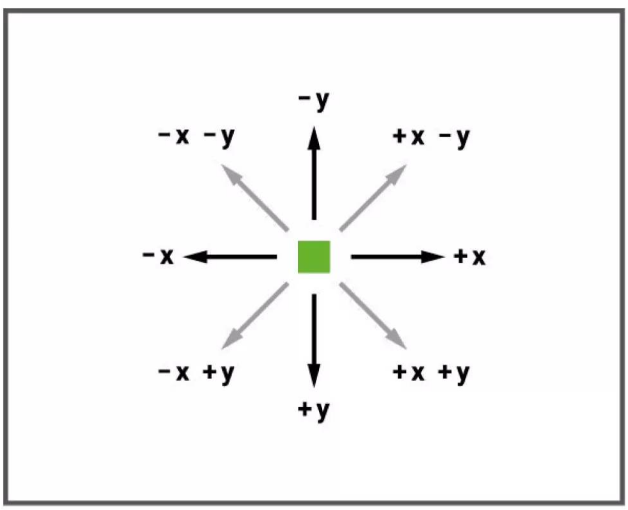
Moving in all directions

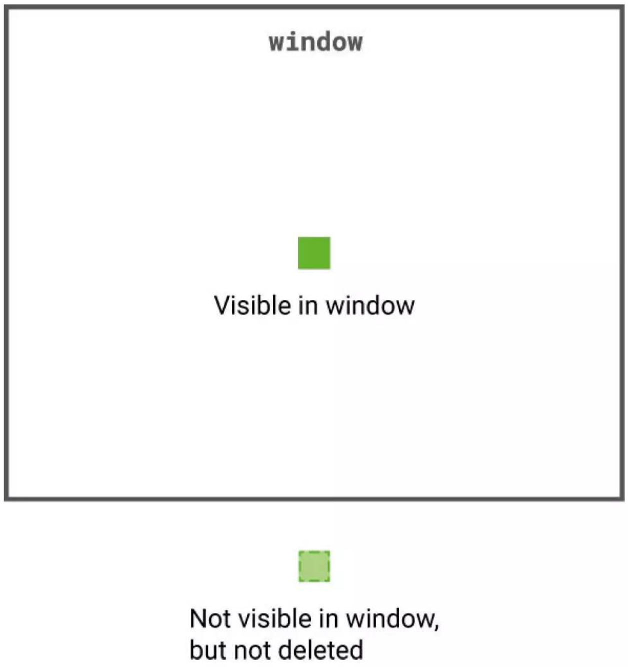
So that's how we move left and right; if we can do that much, surely we can go up and down too? Comment out the `green_square_x += 1` line by adding a `#` before it and uncomment the line below by removing the `#`. Our square will start to travel towards the bottom of the screen. Just like before, we're changing the variable that tells our shape where to go, `green_square_y`, just a little bit each time to make it move. And, just as we saw by changing the X variable, we can make the green square go up by adding a negative number to its Y variable.

So now we can animate things moving in four directions; that's enough freedom to make so many classic games: Pokémon, Legend of Zelda, Space Invaders, and more. These games would only move things horizontally and vertically, but never at the same time. The next challenge would be how to make things move diagonally. Fortunately, this is a pretty simple process too.

If we uncomment both `green_square_x += 1` and `green_square_y += 1` in our code, then our shape will move to the right and down every time Pygame updates the screen. If we add to our X and Y values, our shape will move to the right and down. If we add to our X value and subtract from our Y value, then our shape will move to the right and up. If we subtract from our X value and add to our Y value, our shape will move to the left and down. Finally, if we subtract from both our X and Y values, our shape will move to the left and upwards. That means we have eight directions that our objects can move in (see **Figure 3**) – assuming, that is, that we use numbers that are equal to one another. If we used values that were different for our X and Y values, we'd have more variation. If we use floats (which are numbers with a decimal place, like 2.3 or 3.141) instead of integers (whole numbers), we could get a full 360 degrees of motion.

Figure 3: The eight basic directions in which a shape can move





▶ **Figure 4:** The box is the viewport of a Pygame window

So far, the values we've used to animate our shapes around the screen have been integers that remain constant. With each frame, we would always add 1 (or some other arbitrary value) to move our object. But what happens if we change the values that we use to animate things? What if, instead of adding 1 to X/Y coordinates, we add 1, then 1.1, then 1.2, and so on?

Replace the `green_square_x` and `green_square_y` lines above `while True:` with the following:

```
blue_square_x = 0.0
blue_square_y = 0.0
blue_square_vx = 1
blue_square_vy = 1
```

Next, replace the drawing statements with:

```
    # Begin drawing statements
    window.fill((0,0,0))
    pygame.draw.rect(window, (0, 0, 255),
(blue_square_x, blue_square_y, 10, 10))
    blue_square_x += blue_square_vx
    blue_square_y += blue_square_vy
    blue_square_vx += 0.1
    blue_square_vy += 0.1
    # End drawing statements
```

Save this new program as **moving_accel.py** and run it. What do you notice? We're adding to both our X and Y values, so our square is moving down and to the right, but something is different from our previous bits of code: as our program continues to run, our square moves to the right a little more than it did in the previous frames. It's accelerating. This is because we're using variables to represent the square's velocity, one each for the X and Y velocity. By gradually incrementing those variables, and adding them to our X and Y coordinates, we increase the amount of distance that is added in each frame, which gives the illusion of acceleration.

If changed our code so that it multiplied `blue_square_vx` and `blue_square_vy` by a number greater than one instead of using addition or subtraction, our shapes would accelerate much faster; we'd have hardly any time to see them before they ran off the screen.

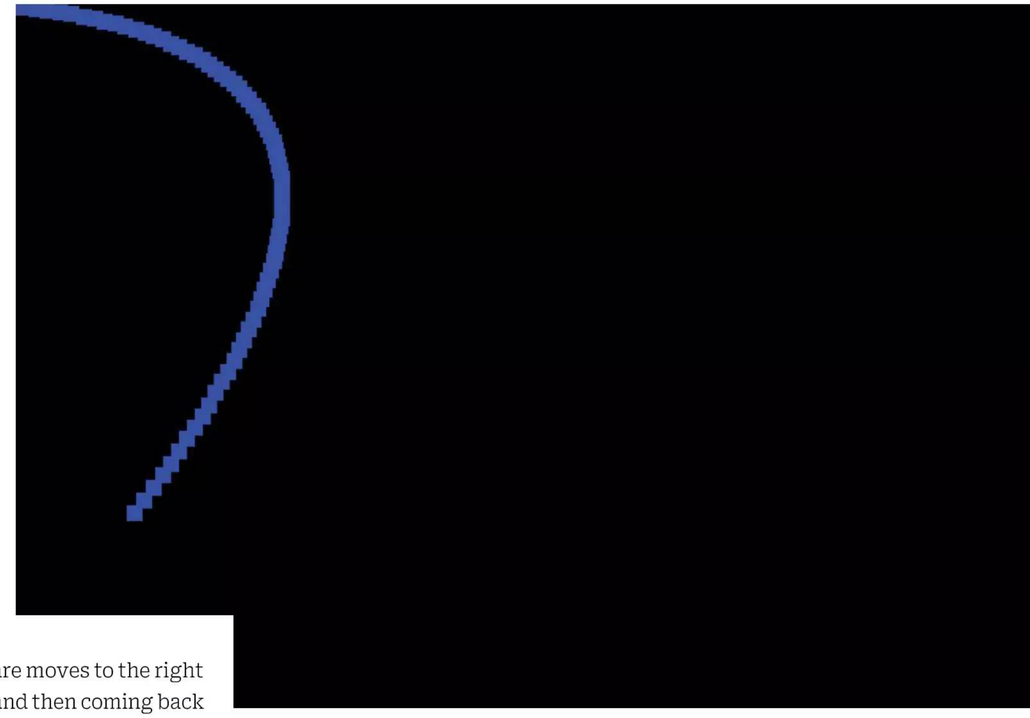
Speaking of which, what happens to our shapes when they run off an edge and are no longer on our screen? Have they disappeared forever? The answer is no. You can think of our window like an actual window in your house, as shown in **Figure 4**. If you look out of the window to see a pedestrian who then moves further down the street so you can no longer see them, they haven't ceased to exist. They're just beyond your line of sight. If our shapes move further across our screen so that we can no longer see them, they don't stop moving or disappear, they keep on going for ever, or until you tell them to stop and come back.

Next, make the following changes:

1. Change the `blue_square_vx = 1` line to `blue_square_vx = 8`
2. Change the `blue_square_vx += 0.1` line to `blue_square_vx -= 0.2`
3. Comment out the `blue_square_vy += 0.1` line

QUICK TIP

If we want to subtract values from a variable, we don't always have to use `-=` for subtraction and `+=` for addition. We can use `+=` for both; simply add a negative number to take away numbers, for example: 4 + -3 = 1.

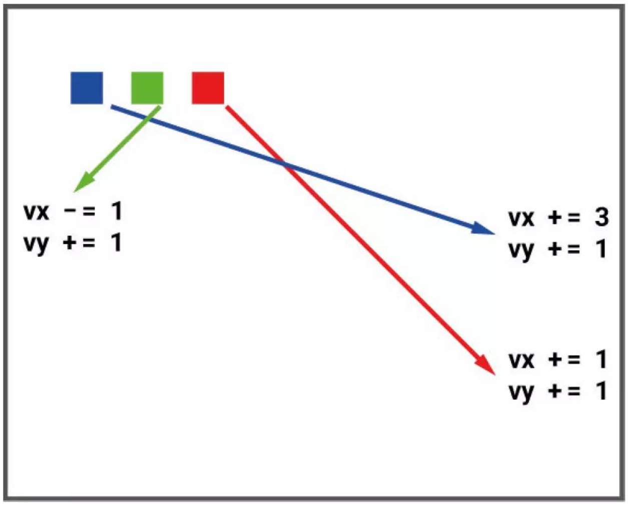


▶ **Figure 5:** This is the path travelled by a shape moving across the window while accelerating

Run it again, and you'll see that the square moves to the right across our screen, before slowing to a stop and then coming back on itself, forming an arcing animation, as shown in **Figure 5**. This is because the `blue_square_vx` variable has decreased to negative numbers, but the `blue_square_y` variable continues to increase.

If we had subtracted the `vx` and `vy` variables in equal values, with equal starting speeds (both `vx` and `vy` being 8, for example), our shape would have continued along its path, stopped, and then reversed along the exact same path, with the same rate of acceleration as it slowed. Play with these values to see what effect they have on how our shape moves (see **Figure 6**). If you like, you can comment out the `window.fill()` line and you'll see the path our shape takes trailing behind it.

▼ **Figure 6:** The varying effects of different acceleration values on shapes



Animating other properties

Animation isn't just about making things move: it's about making things change, too. Until now, we've been animating shapes by moving them, but we can use the same approach of changing variables over time to affect other properties, like the dimensions of our shapes. Replace the four lines you added above `while True`: with:

```
rect_x = WIN_WIDTH / 2
rect_y = WIN_HEIGHT / 2
rect_width = 50
rect_height = 50
```

Next, replace the drawing statements with:

```
    # Begin drawing statements
    window.fill((0,0,0))
    pygame.draw.rect(window, (255,255,0),
                    (rect_x - rect_width / 2,
                     rect_y - rect_height / 2,
                     rect_width, rect_height))
    rect_width += 1
    rect_height += 1
    # End drawing statements
```

Save the program as **shape_change.py**. In the code, `pygame.draw.rect` draws a rectangle just the same as we've done before, but, as in other examples, we've replaced the parameters that determine the width and height of our rectangle with variables that we change.

> *No matter how we change our variables, the shape created will always be in the centre of the window*

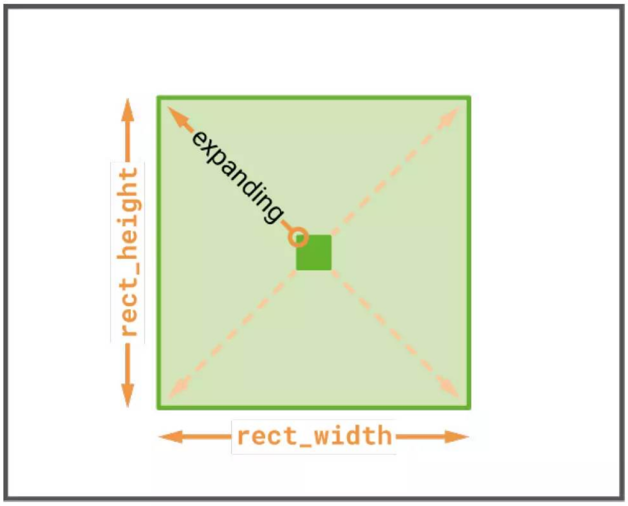
We also do a little bit of maths in our code (see **Figure 7**). As the square gets larger, the point from which it is drawn won't change, so the shape will get bigger, but it will do so off-centre from the rest of the window. By subtracting half of the width and half of the height from the coordinates that we draw our shape at, our square will remain in the centre of the window as it gets larger. The nice thing about using variables in our maths is that no matter how we change our variables, the shape created will always be in the centre of the window. Change the number on the `rect_width += 1` line to any other number between 2 and 10. Now, when our square enlarges, it becomes a rectangle, because its width increases faster than its height does, but it still remains in the centre.

The same effect works in the opposite direction. If we start off with a square that has a width and a height of 50 and change the `+=` inside the `while` loop to `-=`, our square will decrease in size while remaining central to our window.

▶ **Figure 7:** Keeping the square in the centre as it enlarges or shrinks

Changing colour over time

Just like our previous pieces of code, we're using variables in place of values to define what our shapes will look like with `pygame.draw.rect`. The next example, however, has something a little different from the previous examples. Here, we're not adding and subtracting values each and every time we draw our shapes; instead, we're checking the values that we have before we change them, using an `if-else` statement.



For the chapter's final example, here's the code listing (**colour_change.py**) in its entirety:

```
import pygame, random
pygame.init()
clock = pygame.time.Clock()

WIN_WIDTH = 640
WIN_HEIGHT = 480
window = pygame.display.set_mode((WIN_WIDTH, WIN_HEIGHT))
pygame.display.set_caption('Pygame Shapes!')

red_level = random.randint(0, 255)
green_level = random.randint(0, 255)
blue_level = random.randint(0, 255)

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            raise SystemExit

    # Begin drawing statements
    window.fill((0,0,0))
    pygame.draw.rect(window, (red_level, green_level, blue_level), (50, 50, WIN_WIDTH / 2, WIN_HEIGHT / 2))
    if red_level >= 255:
        red_level = random.randint(0, 255)
    else:
        red_level += 1
    if green_level >= 255:
        green_level = random.randint(0, 255)
    else:
        green_level += 1
    if blue_level >= 255:
        blue_level = random.randint(0, 255)
    else:
        blue_level += 1
    # End drawing statements

    pygame.display.update()
    clock.tick(60)
```

This is a key concept of game development: a game's response to a player's actions is the result of hundreds and thousands of these little checks going on every few milliseconds. Without them, there would be no kind of order to any game: it would be like our first bit of code, with the square simply appearing and disappearing at random positions, and there's not much fun in that! With these `if-else` checks, we're making sure that the red, green, and blue values never go over 255 (the maximum value – Pygame will return an error if you specify greater than 255 or less than 0).

If a colour value is about to go over 255, we assign it a random value between 0 and 255. The colour of our square will change and will then continue to slowly work its way through the RGB colour palette by adding 1 to our R, G, and B variables (`red_level`, `green_level`, and `blue_level`) as our Pygame program runs. Just as before, if we added a larger number to each of our variables, we would cycle through the available colours more quickly. Similarly, if we added less to each RGB value every time Pygame updates, we would cycle through all of the available colours more slowly. As well as a great learning device, it looks pretty impressive, too.

colour_change.py

> Language: **Python**

DOWNLOAD THE FULL CODE:

rpimag.co/makegamescode

```python
import pygame, random
pygame.init()
clock = pygame.time.Clock()

# Constants
WIN_WIDTH = 640
WIN_HEIGHT = 480
# Set up display
window = pygame.display.set_mode((WIN_WIDTH,
WIN_HEIGHT))
pygame.display.set_caption('Pygame Shapes!')

red_level = random.randint(0, 255)
green_level = random.randint(0, 255)
blue_level = random.randint(0, 255)

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            raise SystemExit

    # Begin drawing statements
    window.fill((0,0,0))
    # Draw rectangle with current color
    pygame.draw.rect(window, (red_level,
green_level, blue_level),(50, 50,
WIN_WIDTH / 2, WIN_HEIGHT / 2))

    if red_level >= 255:
        red_level = random.randint(0, 255)
    else:
        red_level += 1
    if green_level >= 255:
        green_level = random.randint(0, 255)
    else:
        green_level += 1
    if blue_level >= 255:
        blue_level = random.randint(0, 255)
    else:
        blue_level += 1
    # End drawing statements

    pygame.display.update()
    clock.tick(60)
```

If a colour value is about to go over 255, we assign it a random value

Part 05

Conquer the command line: customise the command line

Make Raspberry Pi OS behave and look just the way we want it – for yourself and other users

Maker

Richard Smedley

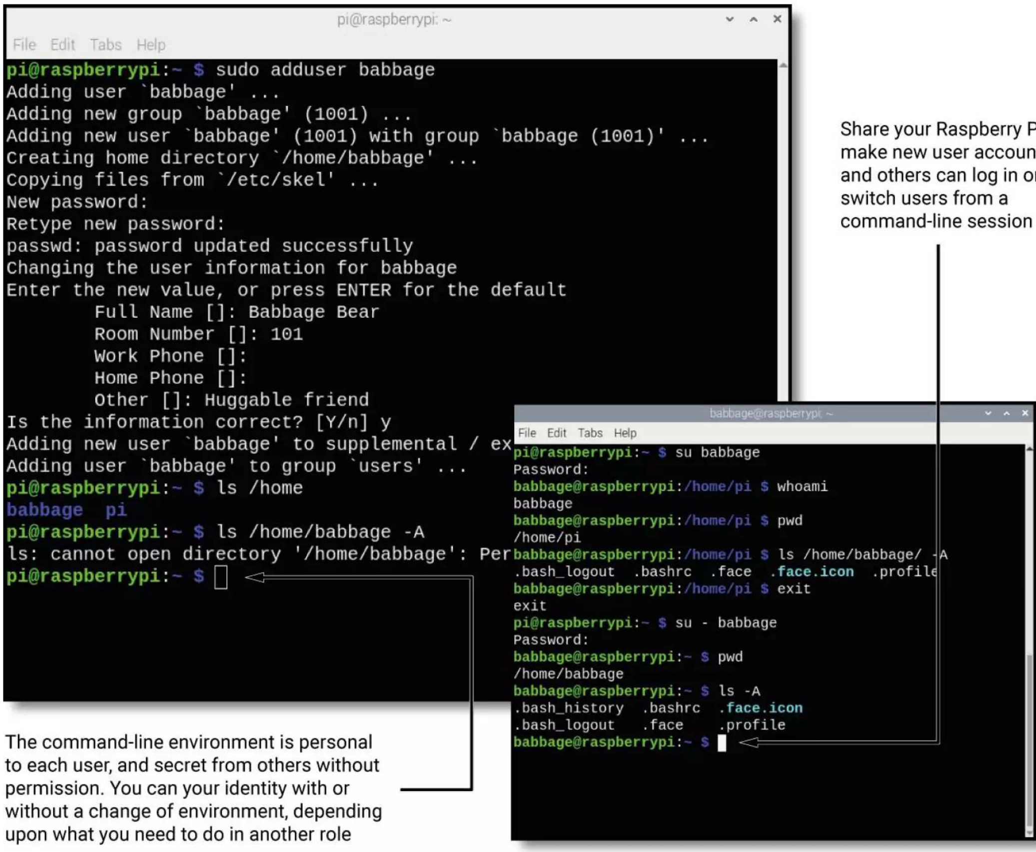
A tech writer, programmer, and web developer with a long history in computers, who is also in music and art.

about.me/RichardSmedley

Take a look at that blinking cursor on your terminal: pi@raspberrypi ~ $. The **$** is known as the 'dollar prompt', awaiting your command; before it you see the **~** (tilde), shorthand for 'home' – which is **/home/pi** in our case (yours is probably different). Before that is [user name]@[computer name], in the form **pi@raspberrypi**. Not only is this informative (at least if you've forgotten who and where you are), but it's also something you can change and personalise.

New user

If more than one person uses your Raspberry Pi, you can set up user accounts for each person. Creating a new user in Raspberry Pi OS is easy: `sudo adduser babbage` will create a new user account named babbage. You will be prompted for a password (pick a good one) and lots of irrelevant info (dating back to shared university computers of the 1970s) that you can safely ignore by pressing **ENTER** at each prompt. Now that we have a user account for babbage, have a look at their home directory with `sudo ls -A /home/babbage`. As babbage has never logged into the computer, you will see it's missing the directories you have in **/home/pi** (for now), but there are a few config files. The first time you log into the desktop as babbage, Raspberry Pi OS creates the missing folders.



Share your Raspberry Pi: make new user accounts and others can log in or switch users from a command-line session

The command-line environment is personal to each user, and secret from others without permission. You can your identity with or without a change of environment, depending upon what you need to do in another role

Files and running programs on Unix-like systems must belong to a user

▲ Customise the command line – it can be different for each user

If you want your Raspberry Pi to prompt for a username and password when it starts up, run `sudo raspi-config`, go into System Options, choose Auto Login, answer no to both questions (console and desktop autologin), choose Finish, and reboot.

Not every user has a home directory and logs in: run `cat /etc/passwd` and you'll see many users that aren't people. This is because files and running programs on Unix-like systems must belong to a user (and a group – see **/etc/group**), as we saw in part one (**rpimag.co/155**) when we ran `ls -l`. If you want to change a password, you'll need to use the passwd command: `sudo passwd babbage` changes the password for babbage. Running `passwd` without a username prompts you to change your password.

Transformations in the virtual world are always easier than those in nature, and this is the case with switching from being 'pi' to 'babbage': we use the change (or substitute) user command, `su`, like so: `su babbage`. Supply babbage's password, and you should see the prompt change to `babbage@raspberrypi`; you can confirm who you are with `whoami`.

Home run

If you're logged in as user pi, then `~` is a shortcut to **/home/pi** – but `~babbage` can be used as a shortcut for **/home/babbage**.

Changing identity

`su - babbage` (note the dash) is best, as you'll gain all of babbage's environment settings, and will change directory to **/home/babbage**.

On older Unix-like systems, running `su` without a username lets you become the root user, with absolute powers (permissions to run, edit, or delete anything). Modern systems like Raspberry Pi OS prefer `sudo` to run individual programs with root permissions. Root's powers may be temporarily attained with `sudo -s` – try it (as your username, because babbage won't have access to `sudo`; run `man visudo` to learn how to change this) and note how the prompt changes (type `exit` to exit) – but it's a bad idea to run with more permissions than you need!

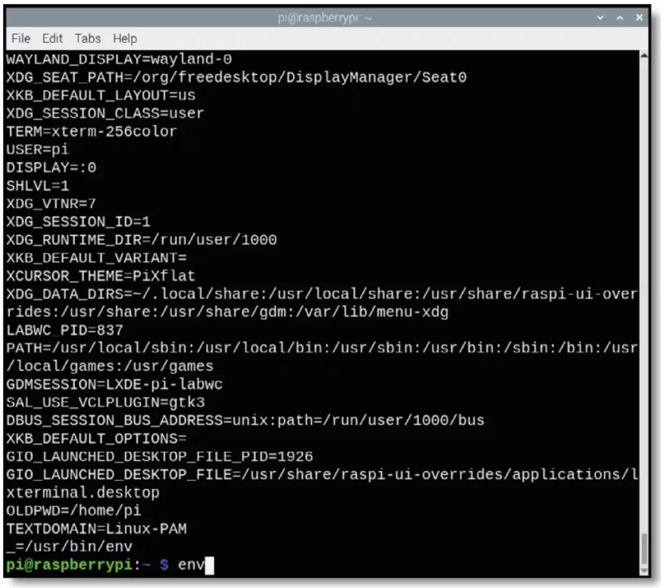
You can customise elements of a user's shell by editing their **~/.bashrc**. Take a look through that configuration file now (as user babbage): `more ~/.bashrc`. Note a number of variables in all capital letters, such as **HISTSIZE** and **PS1**. The last of these controls the prompt you see, such as `babbage@raspberrypi ~ $`. To change it for the duration of your current session, try something like: `export PS1="tutorial@rpom > "`.

This is a temporary change: type `exit` and you've left the `su` value of babbage, so you'll see `pi@raspberrypi ~ $` once more. If you `su` back to babbage, the rpom prompt will still be gone. To make your change permanent, you need to put the **PS1** value you want into **~/.bashrc**. A web search will bring up many options for better customising your prompt.

Bash stores information, from your previous working directory to who you are, in environment variables like **OLDPWD** and **USER**. See individual variables with e.g. `echo $USER`, or view them all with `env` (see **Figure 1**).

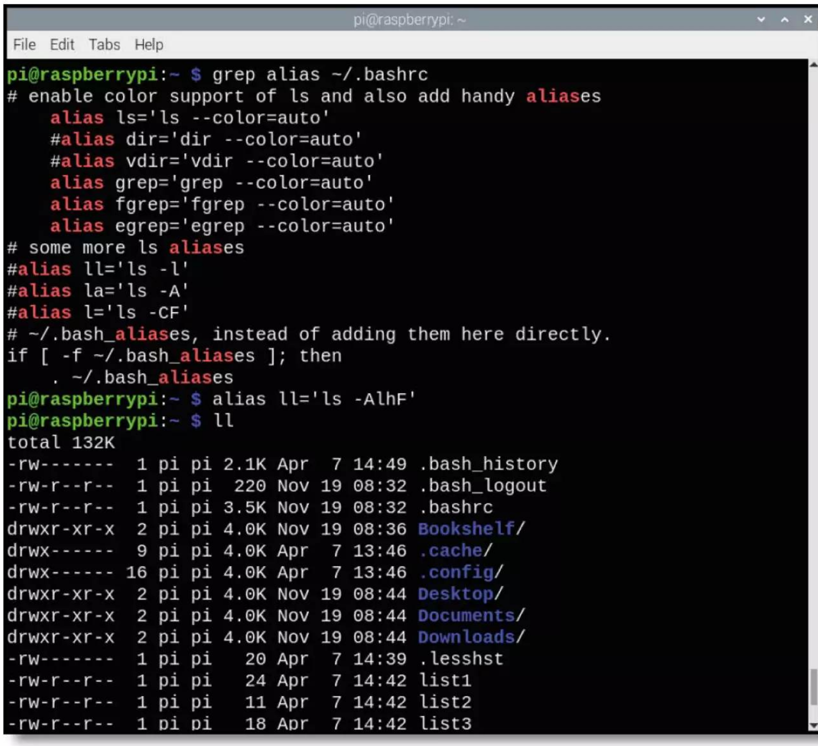
The **~/.bashrc** file is read each time the shell starts, or in other words, every time you log into a console or open a terminal. That's unless you change the default shell in Raspberry Pi OS away from Bash, something you may have reason to do in the future – there are many alternatives available – but let's not worry about that for now. You can put all sorts of commands in **~/.bashrc** to personalise your environment: command aliases are great for regularly used combinations.

▼ **Figure 1:** Viewing all the variables



Who am I?

From the terminal or a virtual console (**CTRL**+**ALT**+**F1** to **F6**), you can `su` to someone else, but start an app from the desktop menu and it will run under whichever user you logged in as.



▲ Use aliases to change what you type to run programs

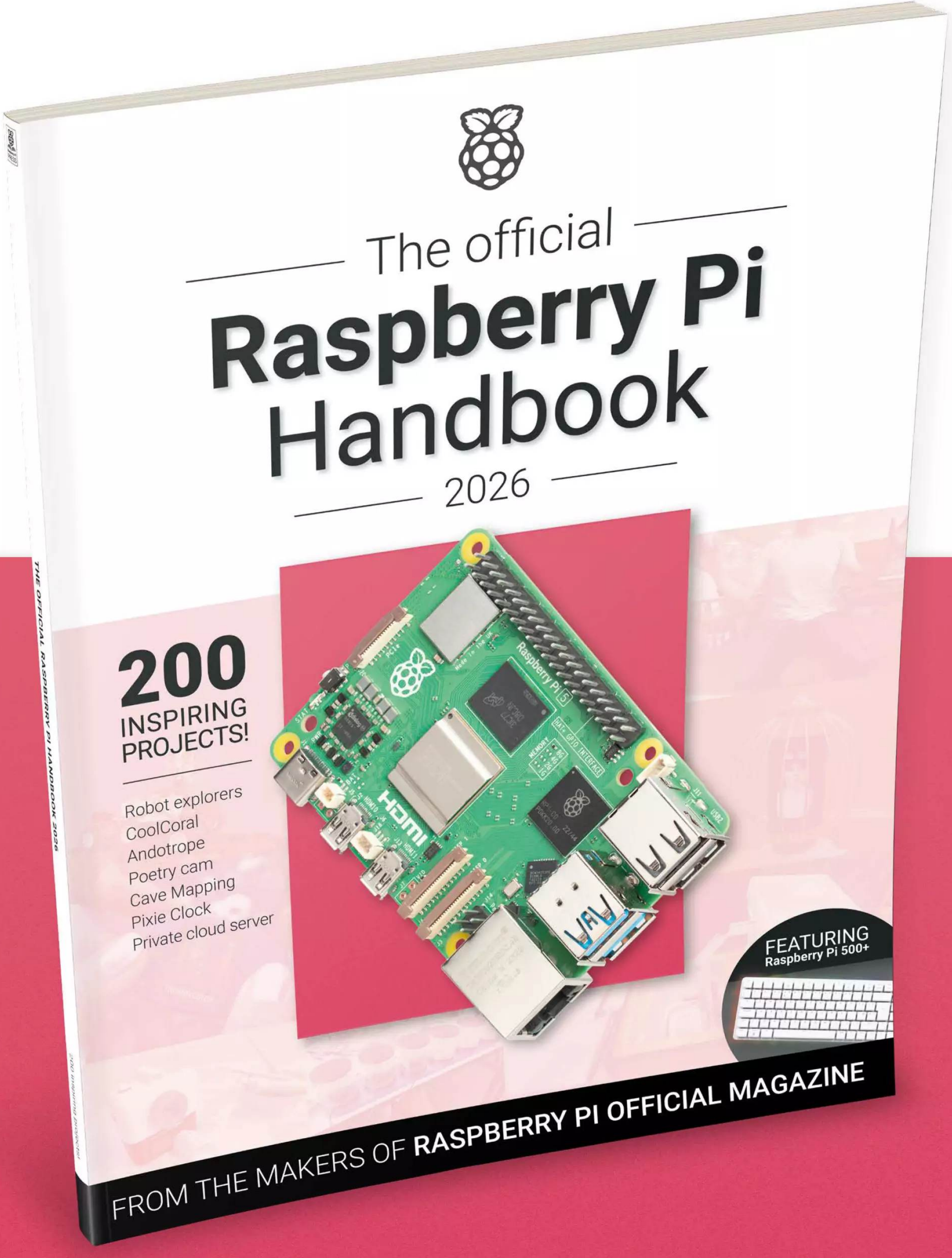
Aliases

Aliases let you change the default command-line arguments to programs. As user pi, you can see the defaults with `grep alias ~/.bashrc`. There are a few aliases already in there, particularly for the `ls` command. One entry is: `# alias ll='ls -l'`. This sounds quite useful, although the **#** indicates that it is 'commented out', which means that it will not be read by Bash. Open **.bashrc** in your text editor (in File Manager, press **CTRL**+**H** to show hidden files, then double-click **.bashrc**) – the simple Text Editor will do for now (we touched on editing text files with nano earlier, and we'll discuss it further in the next part). Remove the #, open a new terminal, and try typing `ll` to run `ls -l`. Handy, but it could be better. Change it to: `alias ll='ls -lAhF'` to get output in kB or MB, rather than bytes, along with trailing slashes on directory names, and the omission of the ever-present **.** and **..** (current and parent) directories.

You can make `rm`, `cp`, and `mv` much safer with these aliases (`-i` requires confirmation when overwriting or deleting files):

```
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

Changes to **.bashrc** take effect in your next terminal session, but you could run the `alias` command in your current session. To disable an alias for a session, use `unalias ll`.



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

Complex Number Calculator

“An organisation of intelligent men” – Frank Jewett, Bell Labs

Author

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

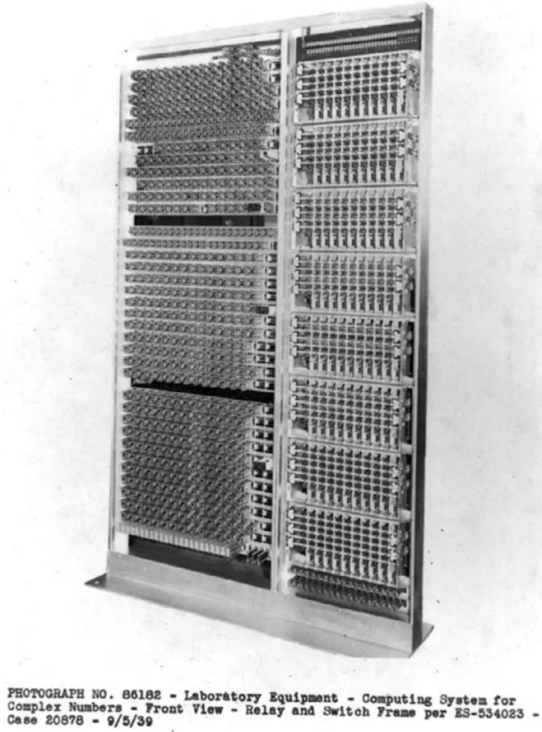
To understand how the Complex Number Calculator[1] came into being, you need to understand the role of Bell Labs in 1930s America. And to understand that, you need to jump in a time machine and travel back to 1909 San Francisco. This was three years after the earthquake that had destroyed swathes of the city. Three men looked on: Frank Jewett, John Carty, and Theodore Vail.

Writer Jon Gertner vividly describes Jewett as “a slight, balding, cigar-smoking physicist” in his celebratory book about Bell Labs called *The Idea Factory*.[2] Jewett had been lured from a teaching position at MIT to join the company five years previously, and had quickly risen up the ranks. Carty was the company’s chief engineer, while Vail had become president two years earlier in 1907.

Gertner describes Vail as “rotund and jowly, with a white walrus moustache, round spectacles, and a sweep of silver hair”, but most importantly as thoughtful. Before he became its president, the American Telephone and Telegraph Company (AT&T) was known for its aggressive policies that often landed it in legal trouble; Vail realised the company would achieve its aims more profitably if it geared its efforts towards technological leadership.

1 Also known as the ‘Complex Number Computer’, ‘Complex Computer’, and (later) ‘Model I’

2 Jon Gertner, *The Idea Factory: Bell Labs and the Great Age of American Innovation* (Penguin Books, 2013, ISBN 978-0-14-312279-1)



▲ Bell Labs Complex Number Calculator (Model I) relay and switch frame
Image: © Nokia Corporation and AT&T Archives

> *Stibitz set to work sketching out what the circuitry might look like*

The three men were in San Francisco to discuss how they could achieve exactly this, and gain some rather nice publicity in the process, by setting up the country's first cross-continental telephone link. One that would span the 2500 miles (4000 kilometres) between San Francisco and the AT&T headquarters in New York. They had five years to make it happen, with the Panama-Pacific International Exposition (better known as the San Francisco Fair) due to take place in 1914.

Carty and Jewett promised Vail they could perform this miracle in time, but rather than rely on their existing engineers they decided to bring in expertise from the academic world. Jewett knew just who to turn to: his old friend Robert Millikan, who he had boarded with when they were both at the University of Chicago.

Best in field

In his autobiography,[3] Millikan remembers his conversation with Jewett in detail. He quotes Jewett asking for help in one specific way: "Let us have one or two, or even three, of the best of the young men who are taking their doctorates with you and are intimately familiar with your field." His field being electron research. "Let us take them into our laboratory in New York and assign them the sole task of developing the telephone repeater."

One particular student, Harold Arnold, had particularly impressed Millikan through his experimental work, and in early 1911 – having completed his PhD – the 27-year-old travelled to New York. He spent the next two years tackling the problem, which boiled down to creating a high-quality amplifier that could act as a repeater across the thousands of miles separating America's west and east coasts.

The key challenge facing Arnold is that when an electric signal is sent down even the best copper wire, its signal drops off. The existing generation of repeaters, mechanical amplifiers, could only work a handful of times before they distorted the sound of the human voice beyond recognition. For a cross-continental call, they needed better technology.

Arnold developed a "mercury arc telephone repeater which worked reasonably well", according to Millikan,[4] but also improved upon an existing invention of Lee de Forest in 1906. Called the Audion, this early vacuum tube was intended to receive "wireless Morse dots and dashes," said Millikan, but Arnold realised that it could be turned into a distortion-free repeater by improving the purity of the vacuum.

Long-distance trials soon revealed the enhanced Audion tube to be the clear winner of four potential repeaters, and AT&T bought De Forest's patent in 1913. With the San Francisco Fair now delayed until 1915, they had two years to turn the trial into a working demonstration. This was no small effort, with four copper wires extending across the expanse of land; simply consider the work to erect the 130,000 poles that kept them up.

It was a colossal effort that also gave the astute marketing team at AT&T a unique opportunity to gain publicity. Alexander Graham Bell had founded the Bell Telephone Company in 1877, a year after his first and now legendary telephone call with his assistant Thomas Watson: "Mr Watson, come here – I want to see you." While the company no longer bore Bell's name, having merged with others to eventually become AT&T, he remained a familiar and popular figure in America.

---

3 *The Autobiography of Robert A Millikan* (Macdonald & Co, 1951), p134

4 As above, p135

For his 1876 telephone call, Watson had been in a different room from Bell but out of earshot. For this encore, Bell would be in San Francisco and his former assistant in New York. Echoing that conversation from almost four decades earlier, Bell said "Mr Watson, come here, I want you." To which Watson replied: "It would take me a week to get there now."

It was a triumph for AT&T on the national stage, but also a personal triumph for Jewett. His newly established research team, which now also included two more of Millikan's students, were breaking new ground for AT&T. And beyond. Millikan describes how in 1915, Arnold demonstrated wireless telephony across 200 miles. Later that year, they would broadcast from the Eiffel Tower in Paris to Honolulu.

▲ Bell Laboratories Building, 463 West Street, New York City in 1925
Image: Charlesbahr, CC BY-SA 4.0

Jewett would later describe an industrial lab, of which Bell Labs was a prime example, as "merely an organisation of intelligent men, presumably of creative capacity, specially trained in a knowledge of the things and methods of science, and provided with the facilities and wherewithal to study and develop the particular industry with which they are associated."[6]

This was in the midst of the Roaring Twenties, a decade that saw huge growth in the USA.[7] The country's gross national product increased by nearly 40%; the proportion of households with electricity shot up from 12% in 1916 to 63% by 1927; the number of cars on the road tripled. There was a feeling that the good times had come and would never end.

But, end it did, and dramatically so, with the stock market crash in 1929 that wiped a third off its value. The Great Depression quickly followed, with millions of Americans losing their jobs and subsequently cutting their phone subscriptions. AT&T's revenues plummeted, forcing Western Electric to lay off 80% of its workforce.[8] Bell Labs wasn't immune to this, with Jewett halting all hires, cutting pay, and instituting four-day working weeks.

Tragically, Arnold suffered a heart attack in early July 1933 at the age of 49.[9] His legacy lived on in numerous ways, however, not least through the Bell Labs Math department that he had instigated. This stemmed from his decision to hire a mathematician to support the engineers with their increasingly complex work. Arnold hired Thornton Fry for this in 1916, and it soon grew into a full department. One that George Stibitz joined in 1930.

By this time, the Math department had outgrown its support roots to being an active participant in research. Members were encouraged to use their brains inventively – and in Stibitz, they definitely had an inventor. In November 1937, as he explained in a 1967 article,[10] "I liberated some relays from a scrap pile at

Go west

In 1916 Jewett was appointed head of Western Electric, the manufacturing wing of AT&T. He commanded a team of roughly a thousand engineers, which included the research group headed up by Arnold. Jewett was relentless in his pursuit for new talent to join this group, writing to one of Millikan's former students – Harvey Fletcher – every year for five years before Fletcher finally caved in and joined the New York team. He was soon joined by Mervin Kelly, who had worked with Millikan on a crucial oil-drop experiment that both decisively proved the existence of electrons and attempted to measure their value.

The goal of the research group was to look beyond the day-to-day improvements and focus instead on the science. Under Arnold, it didn't merely tackle problems set out by senior management but performed research for its own sake – always with the communications in mind, but similar to research in universities the outcomes would not always be obvious. "Of its output, inventions are a valuable part," said Arnold,[5] "but invention is not to be scheduled nor coerced."

With business booming, so had the engineering department under Jewett, to the extent that in 1924 the AT&T board agreed that it should become its own company. Called Bell Telephone Laboratories, Inc., its staff of around 3600 employees would serve both AT&T and General Electric, with a budget of $12 million. That's around $220 million today, adjusted for inflation.

5 Jon Gertner, *The Idea Factory: Bell Labs and the Great Age of American Innovation* (Penguin Books, 2013, ISBN 978-0-14-312279-1), p27

6 As before, p32

7 'Roaring Twenties', Encyclopedia Britannica, **rpimag.co/roaring20s**

8 Figures come from *The Idea Factory*, p36

9 'Harold DeForest Arnold', Encyclopedia Britannica, **rpimag.co/hdarnold**

10 George R Stibitz as told to Mrs Evelyn Loveday, 'The relay computers at Bell Labs', in Datamation, April 1967, **rpimag.co/bellrelays**, p35

Bell Telephone Laboratories where I then worked, and took them home to start what I thought of as a play project."

He had become interested in the electromagnetic relays after being asked to investigate their magnetic behaviour. Whilst doing so, he was struck by the binary nature of the relay contact: "[it] had only stable conditions, off and on, and this I realised was true of the digits in binary notation".[11] This wasn't quite as obvious as might now be imagined. Even as a maths graduate, Stibitz explained, he had to look up the notation to familiarise himself with it because by the 1930s binary was seen as a historic curiosity.

Kitchen input

Stibitz set to work at his kitchen table, fastening two of the stolen relays to a board. He then cut two strips from a tobacco can to serve as sheet metal, then nailed them to the board. These would be his input switches: press on either strip and the connection would become active. For output, he used flashlight bulbs. He then wired his machine so that if you pressed one sheet of metal, a bulb on the right would light up to indicate one; if you pressed both, it would add the two digits together and the left bulb lit up to indicate a carry. The other bulb would be off to indicate zero.[12] In time, this invention would be graced with its own name, the Model K Adder (the K short for kitchen), but its importance was by no means obvious at the time.

"I took it to the Bell Labs and showed it to some of my friends down there and they were satisfactorily amused by the idea that you could use binary notation from the old days to do arithmetic in the modern times, but then I wasn't entirely sure that it was so funny," he recalled in the lecture. What could he achieve with many such adding machines? Could it even mimic the mechanical desktop calculators that were then so prevalent at Bell Labs? He carried on the project during evenings at home, sketching out ever more elaborate circuits for multiplication and division.

All of this was a fascinating but theoretical exercise for Stibitz, but the invention that had so amused his colleagues coincided with a growing problem at Bell Labs: multiplying, dividing, adding, and subtracting complex numbers.

For those who have relegated all mathematics lessons to the farthermost reaches of their mind, a reminder about complex numbers. These include two parts: one is a real number, the other

11 'The Development, Design, and Use of the Bell Labs Relay Calculators, lecture by George Stibitz', Computer History Museum YouTube channel, recorded 8 May 1980, **youtu.be/FipGF3V9obU**

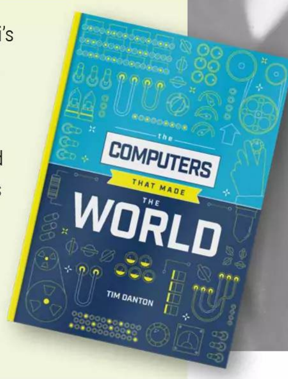
12 'You can see Stibitz demoing a replica in the YouTube video from around 11 minutes in, while the replica itself is now held by the Computer History Museum in Mountain View, northwest of San Jose, California.

▲ Frank Jewett
Image: Underwood and Underwood, New York – MIT Museum, Public Domain

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 12 influential computers built between 1939 and 1950. You can pick up a copy on the Raspberry Pi Press store.

rpimag.co/computersworldbook



imaginary. By 'imaginary', we refer to the odd yet incredibly useful mathematical concept of the square root of -1. Which is, of course, impossible, as any numbers multiplied together will be positive. In maths, the square root of -1 is helpfully denoted by the letter 'i'. A complex number is therefore of the form 'a + bi', where 'a' is the real number and 'b' is the factor of 'i'.

Complex numbers are particularly useful to express the characteristics of alternating currents, which play a crucial role in electrical engineering and telecommunications. Within the Bell Labs Math department alone, Stibitz explained during the lecture, there were a dozen women who did "nothing but calculate with complex numbers using desk calculators."

The problem wasn't merely that this job was tedious and repetitive: calculations had to be written down to save them before being reused. These records, inevitably, were subject to human error[13]. Especially when you bear in mind that calculations needed to be accurate to several decimal places.

One member of Bell Labs proposed the solution of fastening together two calculators by mechanical means. "That plan was circulated around the laboratories and my then supervisor Dr [Thornton] Fry showed it to me and asked whether I had any ideas," said Stibitz. To which his answer was an emphatic yes: rather than mechanically attach calculators, you could connect them using electrical circuits, all based upon his relay-based Model K Adder. And this was how the idea for the Complex Number Calculator – as it came to be known – was born.

Stibitz set to work sketching out what the circuitry might look like. His schematic was handed to Samuel Williams, a relay circuit designer, in the hope that he would confirm that the idea could work. First, though, he had to decipher Stibitz's unorthodox notation and "peculiar squiggles".[14]

"He took the circuits that I had given him and actually traced through something like ten or twelve complex multiplications and divisions, step by step, through every one of the relay contacts and wires that I had drawn up," Stibitz told the Association for Computing Machinery in 1967, "and he came up with the conclusion that it would work."

Relay equipment room of the Model V – a successor to the Complex Number Calculator – installed at Ballistics Research Laboratories, Aberdeen, Maryland
US Army photo, Public Domain

Within a few weeks Bell Labs gave the go-ahead to the project, with Williams handed the job of creating the circuit designs while Stibitz kept charge of the theory. The bulk of time went into the planning, as the pair was breaking new ground. What kind of keyboard would it use? How much storage would it need? How many decimal places would it need to be accurate to?

All this planning took time, with the pair starting the work in September 1938 and only finishing it in April 1939. Remarkably, the manufacturing proved quicker than the planning. Under Williams's careful supervision, the computer was finished by October 1939. All they needed to do now was make it work.

This proved yet another big challenge. The Complex Number Calculator consisted of 450 telephone relays, filling roughly the same space as a tall American fridge-freezer; the larger left-hand side dealt with calculations for the real numbers

13 The tedium behind long calculations, and the likelihood of mistakes, runs through this whole book, all the way from Babbage in our introduction to our final featured computer, the Pilot ACE. It also inspired the ENIAC, which was in effect 20 mechanical computers strung together and then automated.

14 'Computer Oral History Collection, 1969-1973, 1977', part of the Association for Computing Machinery (ACM) Annual Meeting, 30 August 1967, **rpimag.co/acm67**, tape 1

while the right-hand side handled the imaginary part. "The two units operated in parallel," said Stibitz.[15] For example, during multiplication "the real and imaginary parts of the multiplicand were multiplied by digits of the real part and the multiplier simultaneously."

Problems occurred due to the intricate timing involved during calculations. If even one relay didn't activate quickly enough, the calculation would go wrong and nonsense would emerge. They ended up deliberately slowing down aspects of the calculation to ensure that the slower relays would complete in time, but the end result was still a fast machine by existing standards. Stibitz later recalled that addition would take around 0.1 seconds, while multiplication and division were likely to have taken around two seconds.

15 George R Stibitz as told to Mrs Evelyn Loveday, 'The relay computers at Bell Labs', in Datamation, April 1967, **rpimag.co/bellrelays**, p39

Into action

The Complex Number Calculator was put into service in January 1940, at which point it could only do multiplication and division: these were the only functions that Stibitz and Williams thought were worthy of such a beast. However, user feedback quickly made it obvious that addition and subtraction would be heavily used, and by adding a few more relays – and some other relatively minor amendments – they were able to upgrade the machine. "This easy alteration, in so complicated a machine, pointed up its flexibility as contrasted with mechanical machines," said Stibitz.[16]

It was also easy to use, in part because operators didn't need to worry about binary notation. After all, if a PhD holder and maths graduate like Stibitz struggled to remember binary, it would be enormously off-putting to new users of his computer. While both Dr John Atanasoff with the ABC and Zuse with the Z3 invented a decimal to binary converter, so that users could think in normal numbers and the computers could work in binary, Stibitz took a different and innovative approach. In short, the Complex Number Calculator did both.

Its magic trick was to convert decimal numbers into a four-digit binary number. And it was actually even cleverer than this. In standard binary notation, you represent ten as 1010,[17] but this caused "considerable complication when we wanted to carry from one decimal column to another," said Stibitz.[18] To solve this, "I decided to shift all the digits by three units… Decimal zero is to be represented by what we would normally call a binary three, and so on up the line."

If zero is three, then that's 0011 (binary three), while ten would become its opposite: 1100. This symmetry became useful when handling negative numbers, but also meant that "a binary carry occurs at just the right time, namely when the sum reaches ten."

Not that the operators needed to worry about any of this. To use the computer, they sat at a desk with something resembling a typewriter in front of them. Rather than a normal keyboard, however, this included a row of numbers from 0 to 9 and operations in the row above. These allowed the operator to select the type of number (real or imaginary) and

16 George R Stibitz as told to Mrs Evelyn Loveday, 'The relay computers at Bell Labs', in Datamation, April 1967, **rpimag.co/bellrelays**, p43

17 For those unfamiliar with binary, the 0s and 1s read from right to left and are multiples of factors of two. So, in 1010, it's 1 × 8, 0 × 4, 1 × 2, 0 × 1. Add those together and you have ten. 1111 would be 1 × 8, 1 × 4, 1 × 2, 1 × 1, which equals 15.

18 'The Development, Design, and Use of the Bell Labs Relay Calculators, lecture by George Stibitz', Computer History Museum YouTube channel, recorded 8 May 1980, **youtu.be/FipGF3V9obU**

to which operations they were applying: multiply, divide, add, or subtract. They could also call upon useful commands such as clear.

Stibitz's invention did have one quirk. Because it didn't support floating-point numbers, if an operator wanted to input 12.34, then she[19] needed to enter 0.1234 and make the decimal point adjustment after the result came through. For the skilled and experienced operators, however, this would have been trivial.

Once news of the Complex Number Calculator's brilliance spread, others within Bell Labs wanted access. Here, again, we see a sign of the company's innovative thinking: it installed two more input terminals on different floors. If the machine was engaged when someone wanted to use it, they would simply have to wait until it was free, just as they would if they called a telephone number and it was engaged.

Fortunately, the computer took comparatively little time to calculate the results of even tough problems. "The speed of computation was phenomenal for that time: only 30 seconds or so for a complex division comprising three multiplications and addition, and two divisions in eight-place real numbers," remembered Stibitz in his 1967 article.[20] Once complete, the answer was printed out automatically on a teleprinter (often referred to as a teletype in the US due to the dominance of the Teletype Corporation, itself a subsidiary of Western Electric).

The speed of computation was phenomenal for that time

Bell Labs was never shy of publicity, and the Complex Number Calculator made a very public debut at the September 1940 meeting of the American Mathematical Society. This took place at Dartmouth College, Hanover, a good five-hour drive from New York City. And Stibitz had a great idea: rather than showcase its skills by telephoning problems through to an operator based at Bell Labs HQ, as Fry suggested, why not set up one of the teletype machines in Hanover?

One good reason is because it required AT&T to create a "28-wire teletype link between Hanover and New York, and coders and decoders were built at each end," according to computer historian Paul Ceruzzi in *Reckoners*.[21] Just in case anything went wrong, Sam Williams stayed behind in New York.

There was a lot at stake in terms of reputation, with the meeting on 11 September "attended by most of the prominent American mathematicians of the day," to quote Ceruzzi once more. He goes on to list John von Neumann and John Mauchly (co-creator of ENIAC) amongst the names.

First, Stibitz delivered a speech outlining why he had built the computer and what it was used for at Bell Labs. He then demonstrated the machine by running it through a few problems, later declaring that the "pauses during which the computer solved each problem were as impressive as the operating period."

All the problems were solved without any errors, but daringly Fry and Stibitz allowed all the attending mathematicians to try it for themselves. Norbert Wiener, an MIT professor who established the foundations for cybernetics (and coined the term), tried to foil the computer by dividing by zero, but "was told by the machine that this was no good," said Stibitz.[22]

Ahead of time

Viewed through the long lens of history, the Hanover demonstration was more than a simple technology showcase. A decade ahead of its time, it proved that distance was no barrier to users who wanted to access a distant computer's power, foreshadowing the practice of time-sharing – it would take another 24 years for General Electric to announce that all its 600 series computers would support time-sharing. Fittingly, it made that announcement at Dartmouth College.[23]

In its own historic account, Bell Labs even argued that after the demonstration "mathematicians from many parts of the country began, for the first time, to think seriously about new methods in computation."[24]

Journalists, who saw an early demo of the machine a few days earlier, were similarly impressed. "Computing Robot Solves Mathematical Problems Over Long Distance Wire" announced the

19 The operators were almost always women.

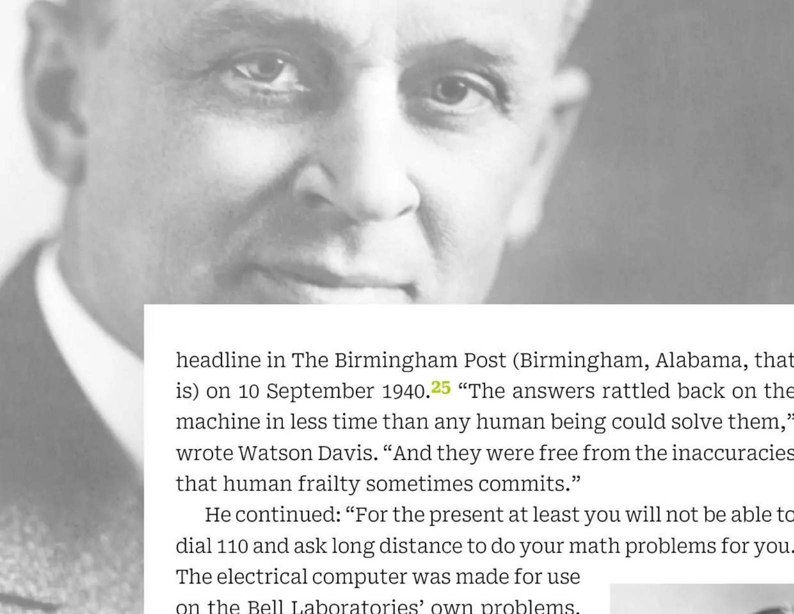
20 George R Stibitz as told to Mrs Evelyn Loveday, 'The relay computers at Bell Labs', in Datamation, April 1967, **rpimag.co/bellrelays**, p44

21 Paul E Ceruzzi, *Reckoners: The Prehistory of the Digital Computer, from Relays to the Stored Program Concept, 1935-1945* (Greenwood Press, 1983, ISBN 0-313-23382-9), p92

22 'The Development, Design, and Use of the Bell Labs Relay Calculators, lecture by George Stibitz', Computer History Museum YouTube channel, recorded 8 May 1980, **youtu.be/FipGF3V9obU**

23 James Pelkey and Andy Russell, 'Timesharing – Project MAC - 1962–1968', The History of Computer Communications, **rpimag.co/timesharing**

24 Bernard D Holbrook and W Stanley Brown, 'A History of Computing Research at Bell Laboratories (1937-1975)', 1982, **rpimag.co/bellhistory**, p6



headline in The Birmingham Post (Birmingham, Alabama, that is) on 10 September 1940.[25] "The answers rattled back on the machine in less time than any human being could solve them," wrote Watson Davis. "And they were free from the inaccuracies that human frailty sometimes commits."

He continued: "For the present at least you will not be able to dial 110 and ask long distance to do your math problems for you. The electrical computer was made for use on the Bell Laboratories' own problems. The one machine constructed so far, when it gets through its demonstrations, will be fully occupied with real computing work already in sight."

▲ Robert Millikan
Image: Nobel Foundation, Public Domain

Those words were prescient. Aside from the occasional day off for maintenance, the reliability of relay switches and the thorough debugging done by Stibitz and Williams before the machine went live meant it put in long hours of service, reliably, until it was finally decommissioned in 1949. And Davis was also correct in hinting that more machines were to come, even though that would not have been obvious in 1940.

For, despite its success, the executives at Bell Labs felt the computer was simply too expensive. "The Complex Calculator was then studied by the group at Bell Laboratories and everyone was horrified to find that it had cost $20,000 [$450,000 in today's money] including the development, design, construction, and debugging," said Stibitz.[26] "The idea that anybody should spend $20,000 for a mere calculator was something that was not acceptable, and the Labs decided that no more things of this sort would be built."

Despite this, Stibitz continued to think about improvements for further machines. One idea was to introduce support for floating decimal points. Another was to add a self-checking code to isolate errors with particular errors; as it was, the mechanics used toothpicks to isolate errors during debugging checks. Similarly to the Harvard Mark I – which Stibitz knew nothing about due to the high levels of secrecy – he wanted to add a tape input so that they could vary the type of programs the computer ran.

These ideas would likely have come to nothing had it not been for the outbreak of war, with the US finally joining the rest of the Allies in December 1941 after the Japanese attacks on Pearl Harbor. Stibitz was loaned out by Bell Labs to the National Defense Research Committee (NDRC), and would go on to create four wartime computers based on relays.

The first, initially called the Relay Interpolator but later renamed the Model II, tied in with an analogue computer also designed by a Bell Labs engineer. The M9 gun director worked in tandem with a radar system to automatically track aircraft, but to work effectively it needed to be trained. This required a huge number of calculations, and rather than rely on desk calculators Stibitz suggested they build a relay machine to do the job.

This time Ernest Andrews "took over the technical work of designing it and having it built," said Stibitz.[27] "This machine had a tape program and it would spend most of its working hours, nights and Sundays, loaded up with enough tape to punch output data and enough paper to produce a wastebasket full of information which we were using in the War program."

It included a similar number of relays to the Complex Number Calculator, later renamed the Model I in light of all its successors, and could only perform addition and subtraction. Technically it could perform multiplication by a small integer, but it used repeated addition and was slow, taking four seconds. Still, it was the first of Stibitz's relay machines to be programmable by tape and due to that flexibility it stayed in service from July 1943 until it was dismantled in 1961.

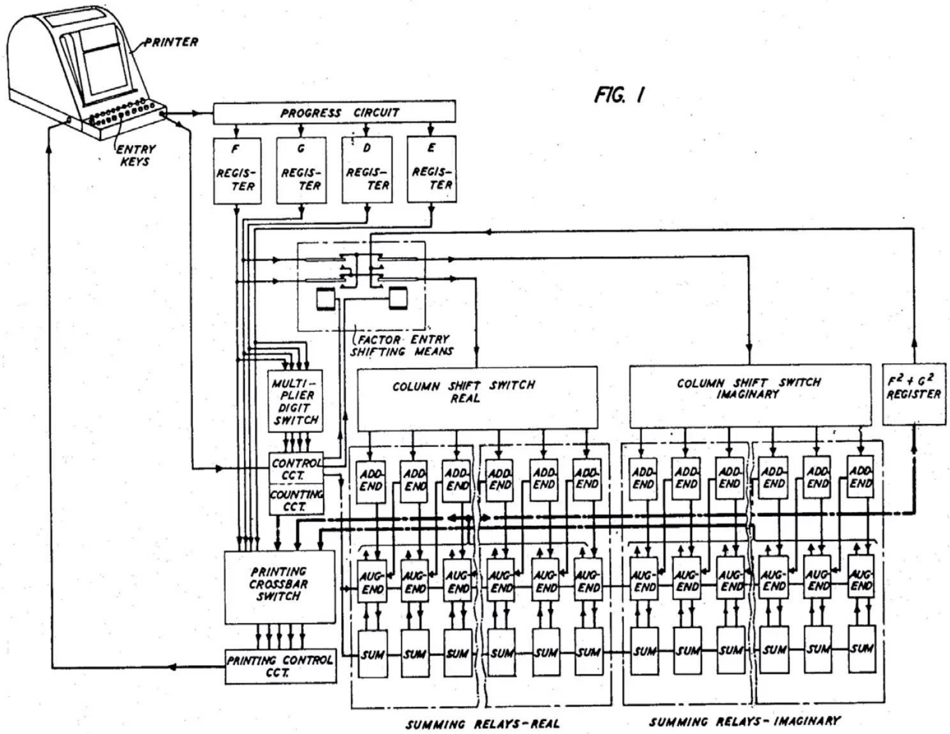
According to the official Bell Labs history,[28] the people of Britain have much to thank these anti-aircraft guns for: "During the month of August 1944, over 90 percent of the [V-1] buzz bombs aimed at London were shot down over the cliffs of Dover."

25 Watson Davis, 'Computing Robot Solves Mathematical Problems Over Long Distance Wire', in The Birmingham Post, Alabama, 10 September 1940, p3

26 'The Development, Design, and Use of the Bell Labs Relay Calculators, lecture by George Stibitz', Computer History Museum YouTube channel, recorded 8 May 1980, **youtu.be/FipGF3V9obU**

27 'Computer Oral History Collection, 1969-1973, 1977', part of the Association for Computing Machinery (ACM) Annual Meeting, 30 August 1967, **rpimag.co/acm67**, tape 1

28 Bernard D Holbrook and W Stanley Brown, 'A History of Computing Research at Bell Laboratories (1937-1975)', 1982, **rpimag.co/bellhistory**, p3



▶ Schematic circuit diagram for the Complex Number Calculator from original US Patent filed in 1941 – **patents.google.com/patent/US2668661A/en** Public Domain

The Ballistic Computer, Model III, was more ambitious still. Completed in June 1944, this packed in 1400 relays and was capable of far more powerful calculations than its predecessor. Its main function was again to train anti-aircraft gun directors, with Bell Labs' Joseph Juley reassuringly stating that "shells obviously cannot be fired at our own planes" in his detailed article about the computer.[29] Instead, planes would go on trial runs and be tracked by the gun director, with measurements such as elevation tracked every second. "Since such a trial run may last as long as 200 seconds, and for each second a large number of computations are required, it would take a team of five [human] computers at least a week to obtain the results desired," Juley added. The Ballistic Computer could perform the same calculations in five or six hours.

The Model III was completed in June 1944, and a year later was followed by the Model IV – or, clumsily, the Error Detector Mark 22 – in March 1945. With almost identical specifications to the Model III, computer historian Paul Ceruzzi argued that they had "just the right balance of features for their relay technology." And both would do sterling service for their military masters, with the Model III retiring in 1958 and the Model IV in 1961.

But arguably the most interesting of all the Bell Labs relay computers is the Model V. The company made two copies, one for the US Army and one for aeronautical research.[30] They weren't completed until after the war, by which time Stibitz had left the NRDC, but it was in many ways the pinnacle of his vision. It was also huge and expensive, with more than 9000 relays and an estimated cost of $500,000. That's well over $8 million in today's money.

Superbranching

You could argue that each Model V was two computers rather than one, though, as they featured two separate arithmetic units that called upon their own memory registers and input-output devices. So operators could run two smaller problems independently or take advantage of both processors (to hijack the more modern term). At a push, you could even say it had its own operating system, with a control unit that dictated the workflow to each processor. Stibitz called this "superbranching", so the computer even wins for hyperbole.

There would be one last hurrah for Stibitz's relay computers in the Model VI, which Bell Labs built for its own use in 1949. This was a stripped-down version of the Model V, with only one 'processor', but it looked flat-footed compared to the electronic rivals sprouting up around it. Ceruzzi describes it as "at least an order of magnitude below even the slowest electronic

29 Joseph Juley, 'The Ballistic Computer', in Brian Randell (ed.), *Origins of digital computers: Selected papers* (Springer Verlag, 1982, ISBN 3540113193), p257

30 The army model went to Aberdeen Proving Ground, as featured in the story of the ENIAC, while the other model was used by the National Advisory Committee on Aeronautics' laboratory at Langley Field, Virginia.

computers", and even with the inclusion of conditional branching it simply couldn't compete with later rivals.

Before we leave Bell Labs, though, we should give due tribute to other contributions to early computing. One is the prioritisation of reliability, something very much lacking in other computers in this book. The longevity of all the relay computers is testament to this, but later models also included error detection – and even error correction.

The problem stems from intermittent errors, such as a dust particle getting in the way of two relay contacts. So, for a few cycles, the calculations will simply be wrong. Then the dust is dislodged and the relay works again. This is quite unlike valves (vacuum tubes): when they failed, they failed for good. That means that even though Bell Labs established a diagnostic procedure to periodically check for faults, it might miss an intermittent problem and not know that it should discard results.

That's why Bell Labs quickly designed independent circuits that would check calculations at "every stop of a computation, like policemen looking over everyone's shoulder," wrote Ceruzzi.[31] From the Model II onwards, they also used seven binary relays rather than four so that they could build in checks. We've tried to avoid printing cumbersome tables in this book, but here we can't resist:

Decimal digit	'Weight of five'	Single relay positions
0	01	00001
1	01	00010
2	01	00100
3	01	01000
4	01	10000
5	10	00001
6	10	00010
7	10	00100
8	10	01000
9	10	10000

So, it's binary but not as we know it. Instead, each decimal digit is represented by two active relays to indicate its multiple of five and how many ones it has. What's clever is that for any number, only two relays will be on: one in the first group, one in the second group. If three are on, or one, the testing circuit knows there's a problem. A separate checking circuit made sure that only one relay was on in each group.

From this clever acorn came the mighty oak tree of parity checking

From this clever acorn came the mighty oak tree of parity checking, which was devised by Bell Labs' Richard Hamming and works in a similar way. As in, it checks that the right number of binary elements are active. "Hamming went on to show that if more redundancy was added to a number code, it could not only detect errors but also correct them as well," wrote Ceruzzi. "[His] work has formed the basis for nearly all computer circuit design ever since."

Our final mention goes to a potentially familiar name in William Shockley, who joined Bell Labs in the mid-1930s after gaining his PhD from MIT. Shockley started his training in the vacuum tube laboratory, he explained in a 1969 interview,[32] "and I was given a lecture by the then research director, Dr Kelly, saying he looked forward to the time when we could get all the relays that make contacts in the telephone exchange out of the telephone exchange and replace them with something electronic so they'd have less trouble."

Shockley had earned his doctorate due to his research in solid-state physics, and Kelly's remarks inspired him to work out a way to use solid-state technology to create a rival to relays. By December 1939, he had developed what he considered a working transistor, only for war to intervene. On his return he picked up where he left off, building on insights about crystals (as the semiconductor materials) he gained during the war to help.

History records the birth of the transistor as 1947, with the shared credit (and Nobel Prize in Physics) going to Shockley and his Bell Labs colleagues John Bardeen and Walter Brattain. While Shockley's name has gone down in computing history in a way that George Stibitz's name has not, we hope that, like a dust mite interrupting a relay connection, this is merely an intermittent fault.

31 Paul E Ceruzzi, *Reckoners* (Greenwood Press, 1983, ISBN 0-313-23382-9), p97

32 William Shockley interview by Jane Morgan, Palo Alto Historical Association, 1969, **youtu.be/LWGVuoisDbI**

PicoMite: a super-accessible boot-to-BASIC computer

If you want to tinker with a BASIC that has a lot of modern features, PicoMite is for you!

Maker

Jo Hinchliffe

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

concretedog.blogspot.com

Back in issue 154 (rpimag.co/154), we reviewed the fabulous PicoCalc, a Raspberry Pi Pico-based device from ClockworkPi, which can be flashed to work with many different firmwares and projects. As shipped, the device has a moderately tweaked version of PicoMite (**geoffg.net/picomite.html**) onboard – an excellent 'boot to BASIC' computer, complete with the MMbasic interpreter. The PicoCalc is a great device, but it's also totally possible to play with PicoMite with just a Raspberry Pi Pico and your regular computer (**Figure 1**).

To create a PicoMite system with a Raspberry Pi Pico, we need to set up a serial console on a regular computer. These install and run inside your terminal emulator and it's possible to do this on most operating systems – Windows, Linux, Mac – and also easy to do on a Raspberry Pi. Numerous different serial console applications exist and all can be configured to work with PicoMite. These include PuTTY, Minicom, Picocom, Telix, Seyon, CuteCom, and more. We opted for Picocom,which we installed onto a Debian machine with `sudo apt install picocom`.

Next, let's set up a Pico as a PicoMite machine. Grab a Pico and connect it to your computer whilst holding down the BOOTSEL button; it should then appear as a portable storage device called 'RPI-RPI2'. Download the zipped PicoMite firmware folder and extract the contents. You'll find a collection of firmware files as well as some additional resources, system fonts, and the user manual inside. Find the **PicoMiteRP2040V6.00.02.uf2**

> *Some older versions of BASIC environments were case sensitive and often commands were written as all caps*

◀ **Figure 1:** Whilst we were first introduced to PicoMite via the PicoCalc, there are a few variants, including just running it on a single Pico

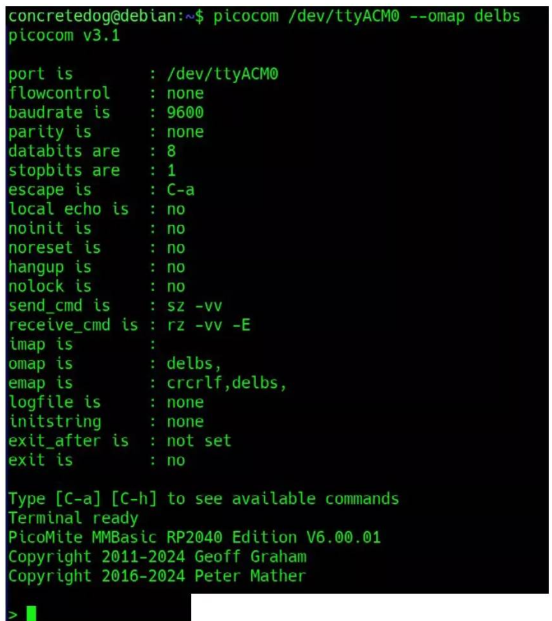
file and copy and paste it over onto the Pico. The firmware should automatically install and the Pico storage device will be unmounted. After a few seconds, the onboard LED on the Pico should begin to blink slowly.

Fire up your terminal emulator. If you have no other USB device connected to your system, it's highly likely on a Linux system that your Pico is connected at **/dev/ttyACM0**; in which case, to connect to the PicoMite via Picocom, issue the command `picocom /dev/ttyACMO`. It it doesn't connect correctly using that, however, you can type `lsusb` in the terminal to try and spot where the Pico is connected.

Hello world

You should now see some information appear listing the Picocom settings (**Figure 2**). Cleverly, PicoMite ignores baud rate settings, so the default Picocom settings should work perfectly without any changes. Try typing a simple BASIC command to check that everything is working correctly, such as `PRINT "Hello World"`. When you press **ENTER**, you should see 'Hello World' printed on a new line (**Figure 3**).

You might notice that if you make a typo and try to correct it, the **BACKSPACE** key isn't working. You can rectify this by relaunching Picocom with an additional argument to map the **BACKSPACE** key function. First, let's quit the current Picocom session; to do this hold down **CTRL** and press **A** and then, whilst continuing to hold the **CTRL** key down, press **X**. This should then exit your Picocom session. Relaunch a Picocom session with the same command but with the additional argument:



▲ **Figure 2:** To use just a Pico flashed with PicoMite, you'll need to set up a serial console in a terminal editor; we used Picocom

▲ **Figure 3:** Typing a `PRINT` command directly into the terminal console

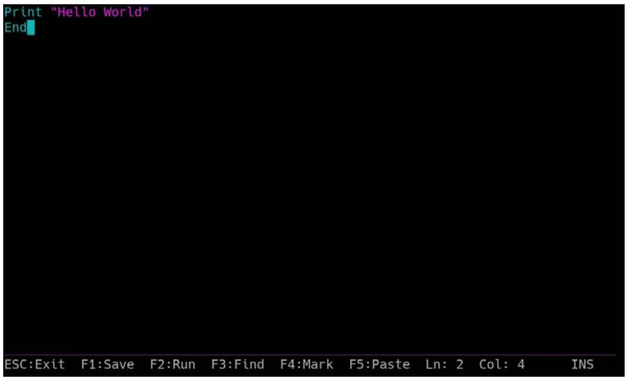


Figure 4: Our first program that we can save and run

```
picocom /dev/ttyACM0 --omap delbs
```

Now that we are up and running with our **BACKSPACE** key working correctly, let's look at some BASIC coding. First of all, let's look at creating a small stored 'Hello World' program rather than just issuing a single command. Typing `EDIT` and pressing **ENTER** launches the BASIC editor. Note that most of the commands are not case-sensitive; some older versions of BASIC environments were case-sensitive and often commands were written as all caps. As such, we've got into the habit of using lots of capitals, but don't worry if you don't; so, for example, `EDIT`, `edit`, and `Edit` are all acceptable commands.

Inside the editor, let's once again write `PRINT "Hello World"` and then on the next line, add the `END` command (**Figure 4**). Note that strings are placed inside `""` and, whilst the second line is not totally needed in this simple example, it's good practice to use `END` to show the end of a program.

We can now exit the editor to return to the terminal to then run this program. PicoMite has some useful shortcuts for this using the function keys. Pressing **F1** from the editor will save the file and exit the editor. Note that we haven't named our project or set a save location, so it's only going to be saved in the active memory. Pressing **F1**, we exit and then we can either type `RUN` to run our program (which should just print 'Hello World'). We don't even need to type the `RUN` command: if you press the **F2** key, it will automatically run the loaded program (**Figure 5**). After you have tried running your Hello World program with **F2**, you can press **F3**. Pressing **F3** invokes the `LIST` command (which of course you can type instead of pressing **F3**). The `LIST` command prints out the contents of a file in the terminal. This is really useful in terms of making a quick check of a file's contents and, as it is read-only, means you can't accidentally make a change – which you could if you launched the editor. Speaking of launching the editor, you can do this by pressing **F4** instead of typing `EDIT`.

Figure 5: The first time we ran the program, we typed `RUN` and hit **ENTER**; the second time, we simply pressed **F2**; to list the program contents without running, we simply sent a `LIST` command

QUICK TIP

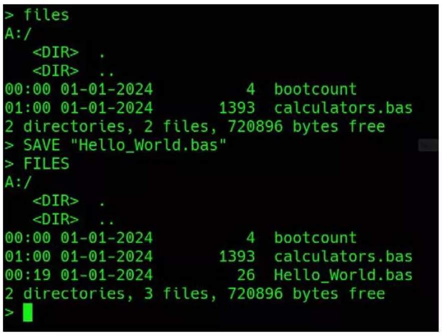
Note that most terminal emulators allow you to set up custom profiles. We went with a larger font size in green which felt sufficiently retro!

You may have noticed that our editor is colour-coding our code automatically, assigning different colours for keywords, strings, numbers, and more. This is an option that can be turned on or off using an `OPTION` command. In PicoMite, `OPTION` commands tend to be saved in the non-volatile memory, so these changes persist and will remain the same whenever you power up the PicoMite until you actively change them. To turn on the colour-coding from the terminal console, enter `OPTION COLOURCODE ON`. Of course, replace the `ON` with `OFF` if you decide you want to revert to standard.

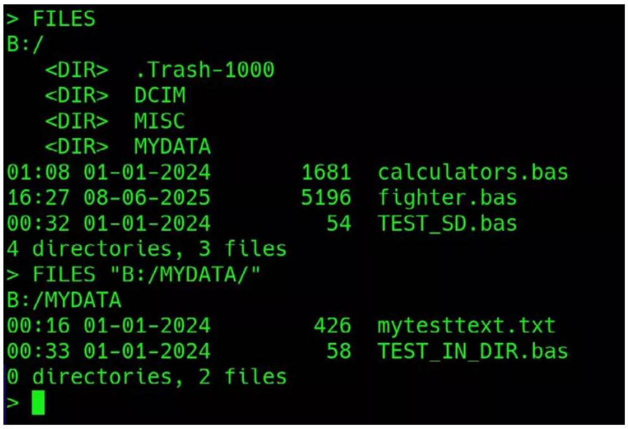
File structure

Having got our Hello World program running, we need to be able to save it to a file so that we can create a different program. To do this, first let's take a little look generally at files and directories/folders on PicoMite.

From the terminal, we can simply type `FILES`. We should then see some listings underneath the label 'A:/' (**Figure 6**). By default, PicoMite has a drive titled A: where we can store some programs. To save our current Hello World program, let's now type `SAVE "Hello_World.bas"` and press **ENTER**. We can then reissue the `FILES` command and we should see **Hello_World.bas** listed. However, if we launch the editor with the `EDIT` command, we can see that we are still editing it. To create a new program, we can issue the `NEW` command, then when we open the editor we should see a blank page. As an exploration of file handling, it's probably worth creating a second program that prints something slightly different than the original 'Hello World'. Create a new program that simply says 'this is the second Hello World', check that it runs, and then save it with a different file name. Now you can use the `LOAD` command with the string of the file name to open a particular file. So for example,



▲ **Figure 6:** Using the `FILES` and `SAVE` commands



◄ **Figure 7:** Using the `FILES` command to list files within directories created using the `MKDIR` command

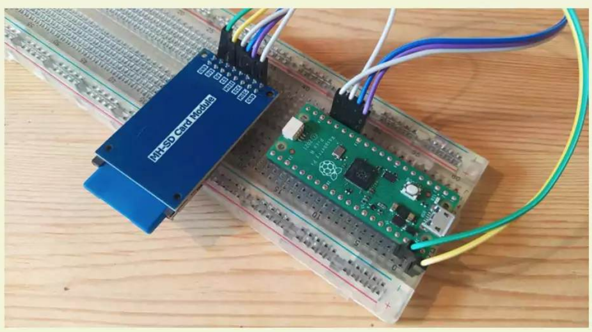
`LOAD "Second_Hello_World.bas"`. Note that if you open a stored file and make changes, you can use **F1** to exit the editor and 'save' the file, but importantly, this only saves the active loaded version of the program, not the file saved in the A: drive. So get into the habit of using **F1** and **F2** to check programs and functionality, but remember to use the `SAVE` command to resave the program to the drive. To remove a file is as simple as using the `KILL` command. So typing `KILL "Second_Hello_World.bas"` will delete that file. Obviously, be careful with this as there is no safety net of a 'wastebasket'-type scenario familiar to more fleshed out operating systems. Finally on the subject of files and organisation, it's possible to create directories. We can use the `MKDIR` command for creation of a directory and use `RMDIR` for removal. You can also specify the drive in this command; so, for example, `MKDIR "A:/MYDATA/"` would create a folder called **MYDATA** on the A: drive. Similarly, we can use the folder location with other commands; in **Figure 7**, you can see we have created a directory called **MYDATA** on the B: drive (an attached SD card) and then we have used the `FILES "B:/MYDATA/"` command to list the files inside the folder. Note that in MMbasic, forward and reverse slashes are treated identically, so it doesn't matter which you use.

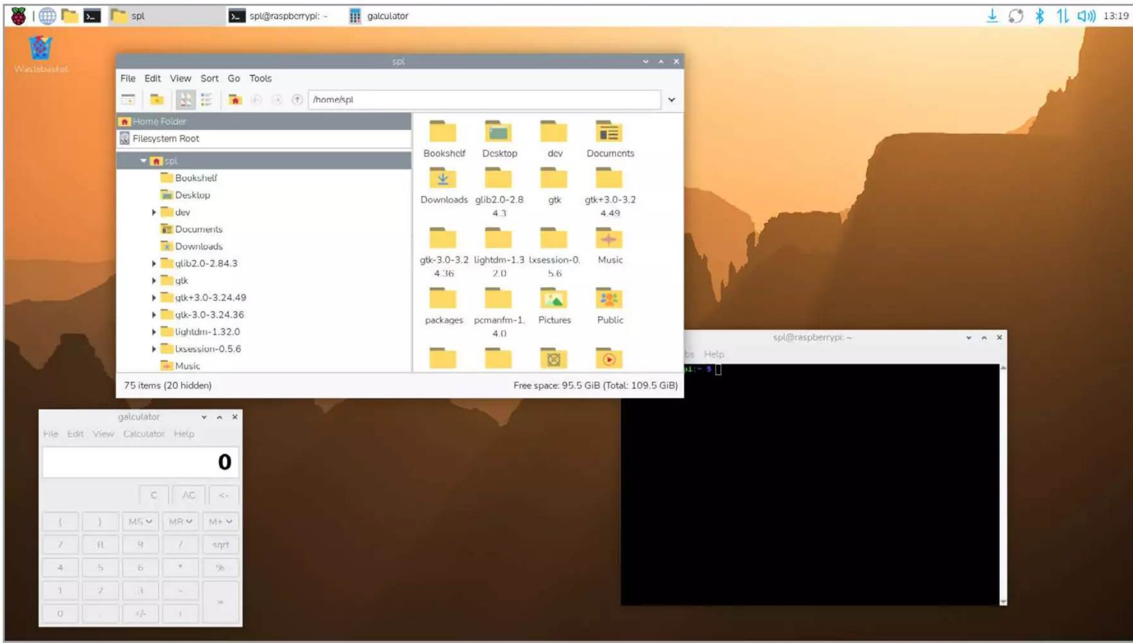
Knowing a little about file structure and navigating PicoMite, there are a lot of potential next steps. We'd advise taking a good look at the user manual (**rpimag.co/picomitemanual**), as there are heaps of examples and an entire appendix that is a tutorial on getting started with BASIC. If you are interested in creating hardware projects and using PicoMite as an embedded controller, then on page 10 of the manual there is the PicoMite version of a 'blink' program where you can connect an LED, define the GPIO as a digital output, and toggle the LED on and off. There are also `OPTION` commands for auto-running a program on boot, so it's as useful as any other Pico coding environment. One area we haven't tried, but is extremely well documented in the user manual, is the Webmite options where you can use the environment to set up web servers, do file transfer, email/SMTP, and even link to MQTT. It definitely isn't just the BASIC you may remember from your youth!

Add an SD card reader

If you are running PicoMite on a PicoCalc or a PicoMiteVGA, then you'll already have an SD card reader attached as a second drive for your PicoMite. However, it's pretty easy to set this up on a standalone Pico setup. Grab an SD card breakout – ours is just a cheap generic full-size 3V3 or 5V compatible SD card device, but a micro SD breakout would work just as well. You can use a variety of GPIO pins to connect it up, but we went with GPIO10, GPIO11, GPIO12, and GPIO13. Connect the 5V to VSYS on Pico (pin 39) and the GND pin on the SD card to any GND pin on Pico. Then we connect GPIO13 to CS, GPIO12 to MISO, GPIO11 to MOSI, and GPIO10 to SCK.

We then inserted a FAT32-formatted 16GB SD card (documentation states up to 32GB is supported) and we could set the default drive with the command `DRIVE "B:"`. Then, when you type `FILES`, you see the file system on the SD card. Note that you can then save and load from the SD card, either by swapping the drive to the B: drive as listed above and then just calling the `LOAD "myprogram.bas"`, or you can call from any drive from anywhere by specifying the drive in the `LOAD` or the `SAVE` command, e.g. `LOAD "B:MyBrilliantProject.bas"`.





Raspberry Pi OS TRIXIE

The latest version of Raspberry Pi's own operating system

By Rob Zwetsloot

For those who may not know, Raspberry Pi OS is based off the popular Linux distribution Debian. Debian is used as the base for other popular operating systems, such as Ubuntu, and a special version of it quickly became the default OS for Raspberry Pi.

Every two years, Debian receives a major update, and now is that time, so say goodbye to Raspberry Pi OS Bookworm and welcome in Raspberry Pi OS Trixie. We've got the lowdown on what's new for this major update – including interface tweaks and behind-the-scenes upgrades – from Simon Long, Senior Principal Software Engineer at Raspberry Pi.

"We've got the lowdown on what's new"

HOW TO INSTALL TRIXIE

Have a Raspberry Pi 4/5/400/500/500+? You can install Trixie over the internet: rpimag.co/netinstall

▲ Like all versions of Debian, it's named after a character from Toy Story. Trixie the Triceratops is one of the toys owned by new child protagonist Bonnie from *Toy Story 3* onwards

Step 1

It's best to install from scratch. From a regular PC, download Raspberry Pi Imager (**rpimag.co/imager**) and insert the microSD card you wish to use into a card reader.

Step2

Launch Imager and select your model of Raspberry Pi with Choose Device, then the top option in Choose OS. This should be Raspberry Pi OS (64-bit) or (32-bit), depending on your device.

Step 3

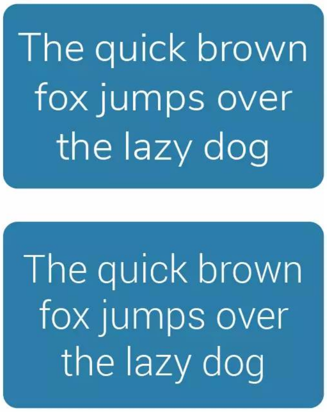
On Choose Storage, select your microSD card. It will be wiped in this process, so back up any files you want to keep. Click Next and it will write to the card. Once it's done, pop it in your Raspberry Pi and turn it on to perform the initial setup. That's it!

A NEW LOOK

Interface elements have been refreshed – meet the new icons and font

New font

For the last ten years, Raspberry Pi OS has used the Piboto font, a version of the Roboto font created for Android 4.0 that was released in 2011; Piboto itself is a custom version created by Simon before Google updated it. From Trixie, the system font will be Nunito Sans Light – you'd have to be really studying the fonts to notice the difference, though. "It was time for a change," explains Simon on why. "It was recommended by the icon designer… and we decided it looked good". Both are sans serif typefaces; however, Nunito Sans uses square dots over i and j, and is very slightly wider.



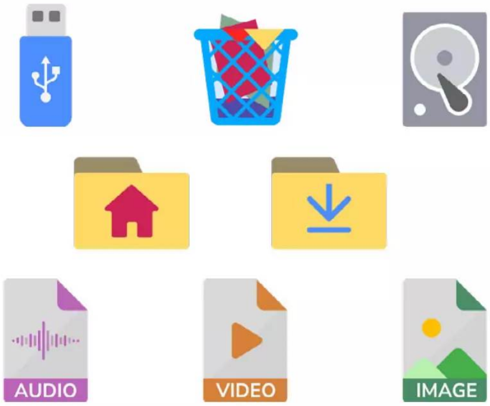
▲ The Nunito Sans font (top) replaces Piboto (bottom)

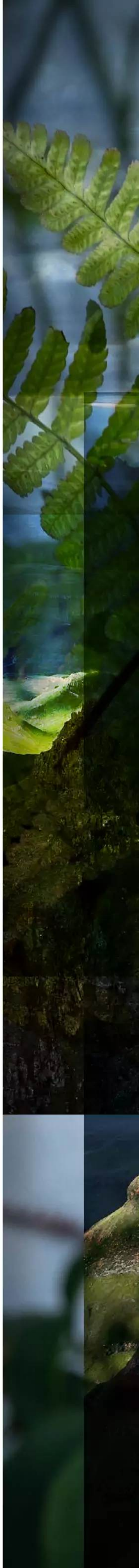
New icons

Many icons around the desktop have been updated, which should be a lot easier to notice for most users.

"Many years ago, at the very start of my career, I worked as a user interface designer at Cambridge Consultants Ltd, and I was very pleased to be able to work on this update with an old colleague from those days," Simon says. "This is the first time we have used an outside designer to provide input on the appearance of the desktop, and I hope you'll agree that the results were worthwhile."

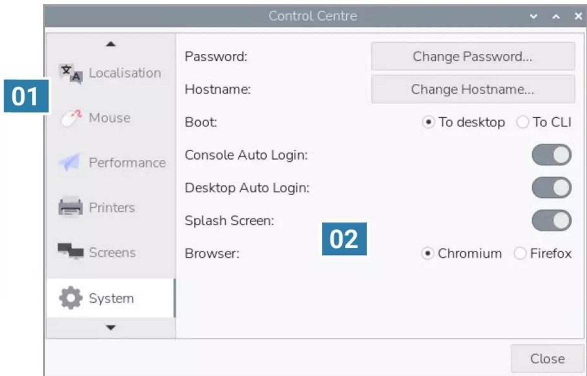
Here's a selection of them – they should all be fairly self-explanatory…





CONTROL CENTRE

Goodbye Raspberry Pi Configuration, hello Control Centre



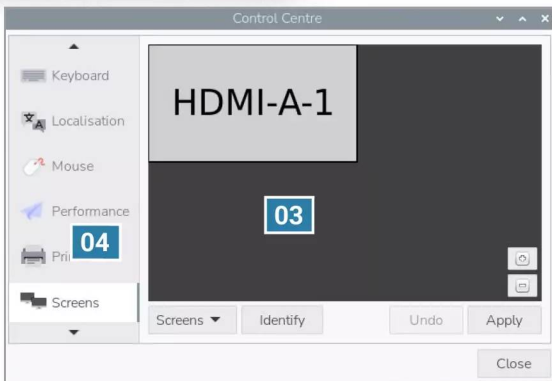
01. Various individual apps such as Raspberry Pi Configuration, Appearance Settings, etc., have been condensed into one app

02. Basic boot options are part of the System tab – including default browser

Before Trixie, multiple apps were used for various settings within Raspberry Pi OS – now, you can find them all listed within one handy app: Control Centre.

"Control Centre is written as a very lightweight core application that loads all its functionality, at run-time, from a set of plug-in libraries, which means it can easily be extended to add new tabs with new controls," Simon explains. "So in the future, we may add new settings pages to Control Centre, but third parties will also be able to write their own settings pages for things like controlling custom hardware."

For interested developers, information can be found on GitHub: **rpimag.co/rpcc**.



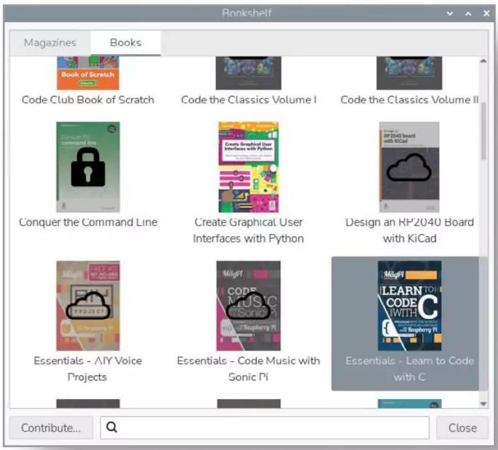
03. Display settings still look and work the same, as do all the tabs that were previously apps

04. More categories can be added by third parties for their specific software or hardware

BOOK SHELF

Find Raspberry Pi Official Magazine and our books in Raspberry Pi OS

> Our books also show up in the app



▲ Handy icons show you which editions are currently free, and/or downloaded

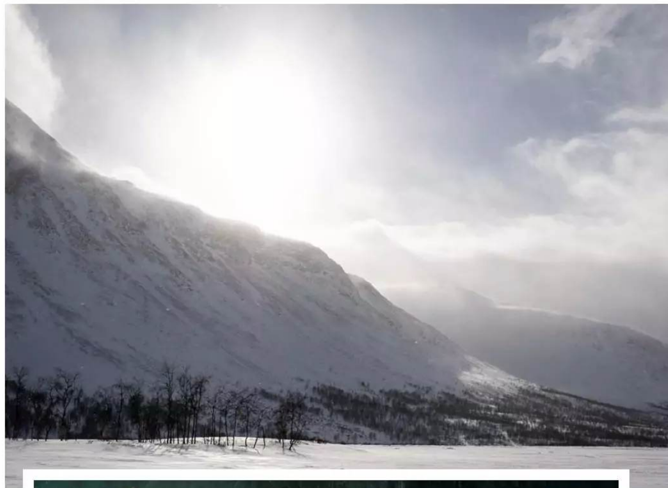
If you didn't know, Bookshelf is an app in Raspberry Pi OS that includes us, *Raspberry Pi Official Magazine*! As magazines go live, you can grab the PDF in Bookshelf and read it directly on your Raspberry Pi.

Since the launch of Bookshelf, we've made the small change to the way the PDFs are distributed – mainly, that they become free after three weeks. If you contribute to the magazine, you can get the PDFs at launch, and now Bookshelf will not only allow you to log in and get your PDFs but also contribute directly from Bookshelf. Our books also show up in the app, not just the magazine and our back issues.

NEW WALLPAPERS

Raspberry Pi Foundation Software Engineer Greg Annandale's new background photos

▼ Sweden Ice

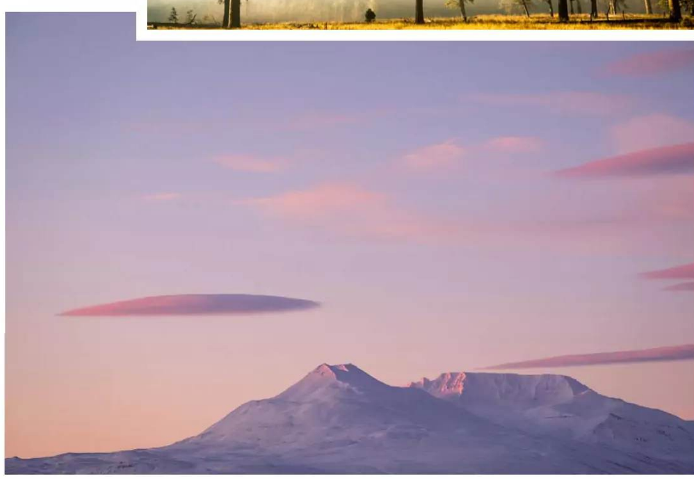


▶ Yosemite Trees

Greg Annandale has been providing incredible photos for Raspberry Pi OS wallpapers for years now, and there's a whole new collection for Trixie. The new default background is a beautiful shot of sunrise at the Drakensberg mountains in Lesotho.

▲ New Zealand Road

▶ Iceland Peaks



BEHIND THE SCENES

Important backend updates you may not notice

64-bit time

Do you know what Unix time is? It's the number of seconds since midnight on 1 January 1970, and is used by a lot of Unix- and Linux-based systems to calculate time. It might seem an odd way of doing things, but it works. A 32-bit variable can be quite large – it has roughly 4.3 billion variations – but it's not quite large enough and a Y2K situation is approaching.

Let us explain:

32-bit = 32 binary numbers (0 and 1) = 4,294,967,296 variables

That is a lot of seconds, and if 0 is midnight on 1 Jan 1970, that means we'll reach 4,294,967,295 at 6:23:15 am on 7 February 2106. However, Unix time can also represent the past. The number is split in half, with 0 at the midway point, which makes the maximum time much sooner.

-2,147,483,647	0	2,147,483,646
20:45:52 13/12/1901	00:00:00 01/01/1970	03:14:06 19/01/2038

Once it reaches that time on 19 January 2038, systems will experience an overflow error, and flip the clock back to December 1901. To fix this, Debian has updated Trixie to a 64-bit time system:

64-bit = 64 binary numbers (0 and 1) = 18,446,744,073,709,551,616 variables

That is over 18.4 quintillion, or 1.84×10^{19} for those more mathematically inclined. It's so large that even if you count it in milliseconds, it won't overflow for another 292 million years. Hopefully by then we're using stardates anyway. Here's the new date ranges, with leap years included:

-9,223,372,036,854,775,808	0	9,223,372,036,854,775,807
01:29:35.53 19/06/292,074,864 BCE	00:00:00 01/01/1970	22:30:24.48 13/07/292,078,803

NB: For any maths corrections, please email magazine@raspberrypi.com

Modular images

As with the previous releases, you can select a full Raspberry Pi OS to install onto your system, desktop and all, or Raspberry Pi Lite which boots to the command line with a minimal selection of software.

"For this release, we have tried to make the installation of the packages that go into the desktop image more modular, to make it easier to create customised versions of the image," Simon explains. "This also makes it easier to convert a Lite image into a full desktop image, and vice versa, something which we have never really supported in the past."

From Raspberry Pi Lite, you can just install eight packages (with Wayland and X variants) with `sudo apt install` to upgrade to the standard image:

Package type	Wayland system	X system
Base desktop	rpd-wayland-core	rpd-x-core
Raspberry Pi desktop	rpd-theme	rpd-theme
Raspberry Pi desktop settings	rpd-preferences	rpd-preferences
Standard applications	rpd-applications	rpd-applications
Utility applications	rpd-utilities	rpd-utilities
Coding applications	rpd-developer	rpd-developer
Command-line graphical apps	rpd-graphics	rpd-graphics
Extra desktop tools (remote desktop, screenshots, etc.)	rpd-wayland-extras	rpd-x-extras

> *Easier to create customised versions*

"Similarly, if you start with a full desktop image, you can remove any of the above by purging the relevant meta package and running `sudo apt autoremove`," Simon reveals. "While to many end-users this functionality won't be of interest, it does make custom images much easier to manage, and is being incorporated into our own image generation tools, like pi-gen."

Powered by Raspberry Pi

We are delighted by the number of products that have Raspberry Pi at their centre, whether as standalone products controlled by our favourite single-board computer, or HATs and accessories designed for hobbyists and makers to make use of. This issue, we feature seven such products from around the world, ranging from a sophisticated controller for theatre and film set technicians to create slick scene changes, to a teeny retro TV screen, plus a rugged handset that pinpoints and reports on the health or otherwise of cattle in remote Canadian ranches.

rpimag.co/poweredbyraspberrypi

APPLY TO POWERED BY RASPBERRY PI

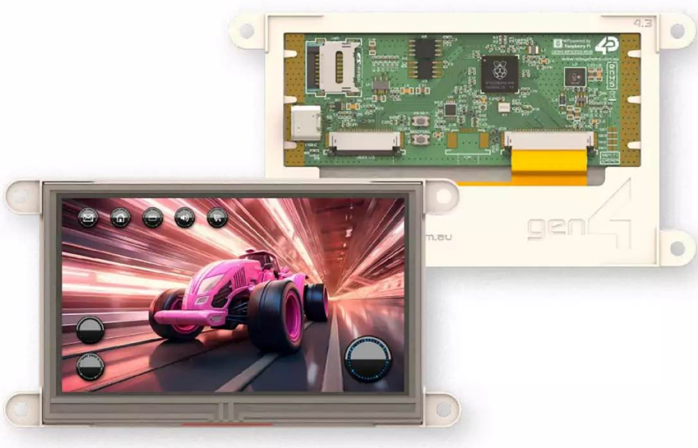
Our Powered by Raspberry Pi logo shows your customers that your product is powered by our high-quality Raspberry Pi computers and microcontrollers. All Powered by Raspberry Pi products are eligible to appear in our online gallery.

rpimag.co/poweredbypiapply

4D Systems Touch Panel

Australia | rpimag.co/4dsystems

Australia's 4D Systems provide a great showcase for the possibilities provided by using dual-core RP2350 microcontrollers. The display specialists offer a range of certified Raspberry Pi screens suitable for mobile gaming and entertainment as well as industrial monitors for installation in challenging environments. Sydney-based 4D Systems has been in business since 1990 and has an enviable distribution network across Asia-Pacific and Europe. The company supplies displays from 2.4in upwards to a host of industries from aerospace to medical. The newest, gen4 Series of professional-grade intelligent displays come in touchscreen and non-touch versions supporting a range of open-source controllers including Espressif as well as Raspberry Pi.



EDATEC Fluid AI camera

China | rpimag.co/aic2k

EDATEC has been one of Raspberry Pi's certified global design partners since 2018 as the hardware partner behind Compute Module 3 LoRa Internet of Things gateway. The firm was part of the bulk rollout of the original Raspberry Pi models stretching back to 2012, as well as the Sense HAT and Raspberry Pi touchscreen. In the past decade, EDATEC has developed numerous industrial hardware designs and computers based on Linux and Raspberry Pi. The Fluid AI camera builds on the success of the firm's CM4-based camera, offering 12MP photography and 70fps video capture alongside 8GB and 1.5GHz system-on-chip quad-core architecture. The camera has an integrated light source and liquid module zooming and comes in a hardy aluminium casing, making it ideal for a range of industrial settings that require ruggedised hardware.

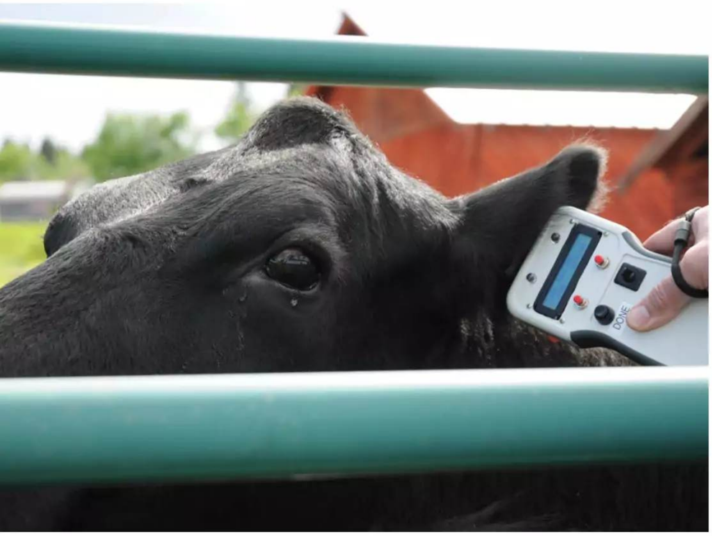
Potteku Cosmo Pico

Japan | rpimag.co/cosmopico

Japan's Potteku wants to enthuse kids about the possibilities of coding, with a board focused on AI and Internet of Things uses that will appeal to newcomers to the world of Python and Raspberry Pi who want to learn how electronics work and begin to create their own gadgets. The Cosmo Pico Editor provides a really simple approach based on CircuitPython (**circuitpython.org**), which is specifically designed for experimenting and learning coding using low-cost microcontrollers. Bluetooth, wireless, or USB connections can be used to connect your chosen microcontroller to the control centre at **code.circuitpython.org**. The board itself – which, in line with Powered by Raspberry Pi requirements has been rigorously tested – also lends itself to manufacturing environments.

Flokk RFID tagging

Canada | rpimag.co/flokkhand



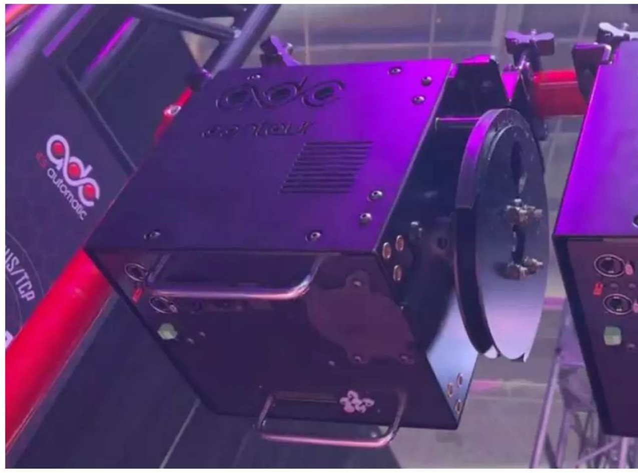
Farming is a hard enough business to make money from, without having to laboriously tag and track every single animal, yet traceability requirements mandate keeping paper records about the health, fertility, and medications taken of each one. Cattle are RFID tagged, but the vast scale of farms in the likes of Alberta in Canada (where Flokk is based) used to be too remote for wireless communications, leading many farmers to log everything in easily lost or damaged paper records. Canada's Mark Olsen worked out that ruggedised handsets that store animal data locally on an SD card and subsequently downloaded to a PC can significantly reduce the time and energy involved in fulfilling herd traceability requirements. With broadband connectivity now available via Starlink and other satellite offerings, data records can be efficiently downloaded and submitted. Flokk handsets based on Raspberry Pi Pico are ruggedised and nearly impossible to break, even if dropped and trampled on.

Melopero Perpetuo LoRa

Italy | rpimag.co/perpetuolora

Italy's Melopero has developed an advanced electronic board with 8MB flash memory, NeoPixel LEDs, and multiple GPIO pins, built around the powerful Raspberry Pi RP2350 microcontroller. The Perpetuo LoRa integrates an Embit EMB-LR1276S LoRa (long range) radio module, enabling robust wireless connectivity in places that may not have fixed communications infrastructure. Melopero designed the Perpetuo LoRa board with sustainability in mind, with parts that can easily be updated. The board comes with a compact LiPo battery charger that can be connected to a small solar panel for recharging. This green energy approach means the Perpetuo LoRa is able to run indefinitely on solar power. This aspect, plus the compact battery and overall dimensions, make it ideal for off-grid projects such as providing communications to campervanners, explorers, and researchers in remote locations.

ADC Axis1 Touch Controller



USA | rpimag.co/adctracks

Automatic Devices Company (ADC) formed in Pennsylvania just as the television and film industries really took off, providing silent and steady scenery shifting as well as judder-free camera tracking. To this day, almost every product in the curtain tracking industry is either made by or based on technology originated by ADC, which prides itself on hand-building. The company won the prestigious Live Design Industry award for best new product for its Raspberry Pi-based Axis1 Touch Controller featuring a 7-inch capacitive display, Power-over-Ethernet and a rugged aluminium case. The controller has sturdy metal handles and can be used on-stage as a handheld device to oversee scenery and curtain load-ins and cueing, or wall-mounted as a central access point. It can be programmed with secure logins, timers, and up to 32 multi-axis groups, and can also be used remotely.

Tiny Circuits TinyTV 2

USA | rpimag.co/tinytv2

Akron, Ohio-based Tiny Circuits specialises in open-source electronics and was one of the first companies to start designing products specifically for use with Raspberry Pi. It set up shop in 2012 – the very same year the original Raspberry Pi Model B launched – making Arduino accessories, but very quickly expanded to cater to the stratospheric success of Raspberry Pi, too. Now focused on working miniature electronics, Tiny Circuits offers two models of teeny Thumby Pico-based mini games console and a fantastic TinyTV that plays real television programmes and video clips with Raspberry RP2040 at its heart. The £46 gadget has a 1-inch 65K colour TFT screen and a 16 × 9mm front-facing speaker. It has 8GB of onboard storage, enough for up to 10 hours of video playback, although you'd probably be squinting long before the two hours of battery-based viewing is up. The TinyTV would make for an intriguing feature in a diorama or doll's house.

ONLY THE **BEST**

Retro gaming

By **Phil King**

Retro gaming is one of the most popular uses of a Raspberry Pi, taking advantage of its low cost, power, and ability to easily run emulation software to replicate many classic consoles and computers. The easiest way to get started is to install an emulation OS/environment such as RetroPie, Lakka, Recalbox, or Batocera – all of which feature emulators for a wide range of systems, from NES to PlayStation 2. While most Raspberry Pi models can be used, if you want to emulate more powerful systems reliably and smoothly, a Raspberry Pi 4 or 5 is recommended.

While you can use a standard Raspberry Pi desktop setup with keyboard and monitor, there are many ways to enhance your retro gaming experience. This could involve anything from plugging in a USB games controller to building a full-scale replica arcade cabinet. In this roundup, we'll take a look at some of the best retro gaming kits, cases, and add-ons around, including a couple of handheld gaming options. Let the games begin...

Warning!
Copyright
Please be cautious when finding ROMs for your retro setup that you are not downloading from a site hosting software illegally. There are lots of other legal ROMs available, including many modern 'homebrew' titles developed for classic hardware.
rpimag.co/legalroms

Arcade Stick

▲ It's really robust, so you can give it some stick!

8BitDo | £82 / $90 | 8bitdo.com

You don't have to build an arcade cabinet to play retro games on Raspberry Pi. By installing an emulation system such as Lakka, Recalbox, Batocera, or RetroPie, you could just use a standard desktop setup with a monitor and keyboard. For serious gaming, though, you'll probably want to add a dedicated controller.

There are countless controllers on the market, but 8BitDo makes some of the best, such as this impressive Arcade Stick. Measuring 303 × 203 × 111.5mm and weighing 2.1kg, it's a bit of a beast, but feels really sturdy and capable of taking some major punishment. You can even swap out the buttons for alternatives if you like.

The stick can be connected to Raspberry Pi in three ways: USB, Bluetooth, or 2.4GHz wireless (using the supplied RF receiver). It uses the standard XInput API and the eight face buttons map to those found on all major console controllers. There are also Start and Select buttons. One slight downside is that the two macro buttons can't be programmed on Raspberry Pi, as the required software isn't available for Linux.

Verdict

A bit pricey, but it's a very solid arcade controller.

GPi Case 2W

Retroflag / The Pi Hut | £75 / $102 | retroflag.com / thepihut.com

With suitably retro styling, this case has everything needed to turn Raspberry Pi Zero 2 W (or Zero / Zero W) into a handheld games machine, including a 3.5mm headphone jack and 640×480 IPS colour display. Controls comprise a D-pad, four face buttons, plus Start, Select, Turbo, Home, and rear L/R shoulder buttons.

To start retro gaming, all you need to do is download Recalbox and install it on a microSD card. If you want to use RetroPie instead, there's a special version with a patch to convert the video output for the mini display. The case contains a rechargeable 2800mAh Li-ion battery that provides up to ten hours of playing time.

One thing to note is that due to very limited space, your Raspberry Pi Zero 2 W must not have a pin header soldered onto it. Instead, the bare board just slots into the GPi Case's plug-in 'game cartridge' and connects to its I/O interface board using pogo pins.

Enjoy gaming on the go with this retro case

Verdict

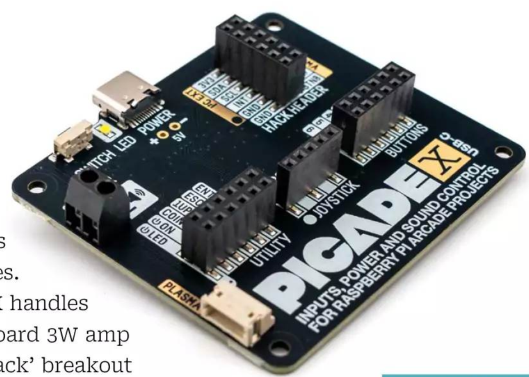
A neat portable gaming solution for Raspberry Pi Zero 2 W.

Picade-X HAT

Pimoroni | £16 / $18 | pimoroni.com

If you want to build your own custom arcade machine, you can buy a Picade-X HAT (as used in the standard Picade kit) to make the wiring process a lot easier. The HAT features several female DuPont headers for all your inputs. Along with a joystick (four directions and ground), you can connect up to six player buttons. A 'Utility' header enables you to wire up four extra buttons (such as Coin and 1-Up), plus a soft power button and LED output. A one-line installer is used for the Python software that converts inputs into keyboard presses.

Better yet, the Picade-X handles audio output with its onboard 3W amp – just add a speaker. A 'Hack' breakout header includes pins for power, I2C, and two additional buttons. There's also a Plasma port to add optional LED lighting using Pimoroni's Plasma buttons. Handily, the HAT's USB-C connector also powers Raspberry Pi.



Lots of connections for your arcade machine setup

Verdict

Makes it easier to connect arcade controls to Raspberry Pi.

Thumby Color

TinyCircuits | £38 / $50 | color.thumby.us

▲ It's so small, you can attach it to a keychain

Another good option for portable gaming is this micro games console powered by a Raspberry Pi RP2350 microcontroller chip (as used in Pico 2 and 2W). It really is tiny, too: the case measures a mere 51.6 × 30 × 11.6mm and has a hole to attach it to your keychain.

The 0.85-inch screen is a 128×128 16-bit backlit colour TFT LCD. Either side of it are a D-pad and two action buttons, plus a menu button and a couple of bumpers on top. It's a bit fiddly but still perfectly playable. A rechargeable 110mAh LiPo battery pack powers around two hours of play, while a tiny speaker delivers audio.

While the Thumby Color comes with a few games pre-installed into its 16MB flash storage (with more downloadable), a big draw is the ability to program your own creations in MicroPython using either Thonny IDE in a computer or an online code editor.

Verdict

A truly tiny handheld console you can program.

PiStation

Retroflag / The Pi Hut | £27 / $37 | retroflag.com / thepihut.com

Verdict

A well-designed case with optional LCD screen unit.

This case will look very familiar to a certain generation of gamers with its nostalgic nineties styling. Made specifically for Raspberry Pi 4, it even has a flip-up lid, under which – instead of putting a game CD-ROM – you'll find a grille for an optional fan and a tray to store spare microSD cards.

Two rerouted USB ports are situated on the front for your controllers, while at the rear are two micro-HDMI ports, audio out, and USB-C power input. By removing a cover on the left side, you can access two more of Raspberry Pi 4's USB ports and its Ethernet port. The microSD slot is on the underside. Stereo sound is output through speakers on either side of the case.

There's also a special version of the PiStation with a foldable 800×480 LCD screen unit that slots into the rear, held in place by a couple of bolts.

▼ Classic console styling for your retro games setup

Picade

Pimoroni | £249 / $276 | pimoroni.com

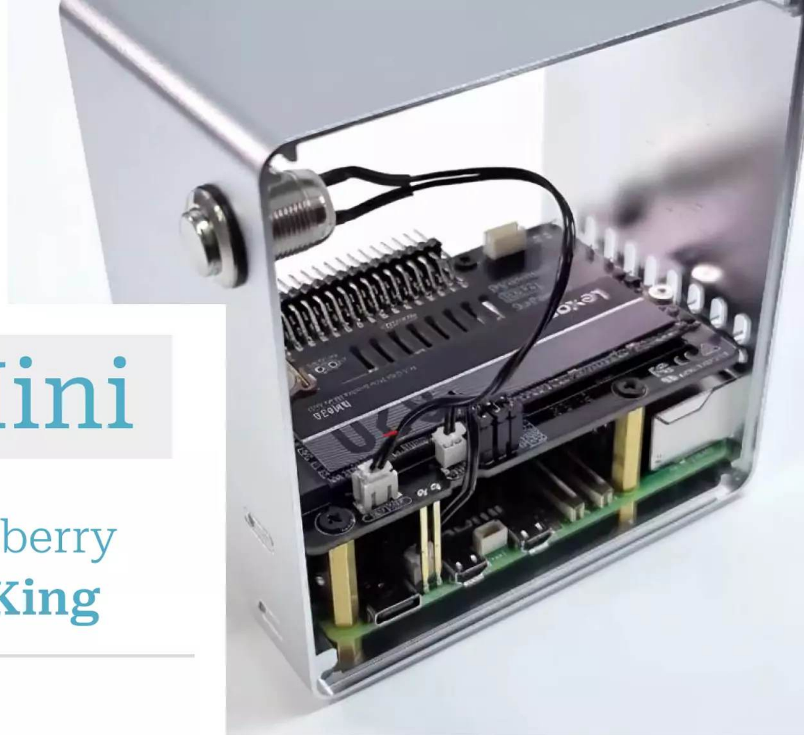
If you find the idea of creating your own custom arcade cabinet daunting, the Picade is an ideal alternative. Using the step-by-step online guide, the kit is fairly easy to assemble, taking 2–3 hours. All you need to add is a Raspberry Pi, power supply, and microSD card. A Raspberry Pi 5 is recommended, supplying the processing power needed to run more demanding games system emulators. The guide includes instructions for installing RetroPie manually in Raspberry Pi OS. In addition, a PICO-8 licence is included so you can start creating your own games.

This recently updated version of Picade features a bigger 10-inch IPS display with 1024×768 resolution and a driver board. As before, the cabinet is built from power-coated pre-cut MDF panels and you can decorate it with the supplied Picade marquee and console acrylic artwork (or your own). For controls, you get a ball-topped joystick and low-profile arcade buttons (which you could swap out for LED-lit Plasma ones). A 3-inch 5W speaker provides audio. Everything connects to the Picade-X HAT, also available separately (as covered in this roundup). There's also a larger Picade Max model with twin player controls and 19-inch screen.

▲ Play retro games and your own PICO-8 creations

Verdict

A bar-top mini arcade cabinet that's easy to build.



Pironman 5 Mini

Compact and bijou, this Raspberry Pi 5 case looks cool. By **Phil King**

Sunfounder **rpimag.co/pironman5mini** £34 / $46

▲ The partly assembled case with its power button connected to the HAT+

SPECS

FEATURES: HAT+ with M.2 SSD slot, RGB fan, PWM active cooler, power button, RTC battery

DIMENSIONS: 67.8 × 98.6 × 101.5mm

Verdict

While not quite as eye-catching or feature-packed as its siblings, it's a quality case with effective cooling.

8/10

We reviewed the original Pironman 5 tower PC-style case for Raspberry Pi 5 back in issue 145 and were very impressed by its looks and cooling performance. Since then, it has been joined by a Max version with room for two NVMe SSDs, and now the Pironman 5 Mini.

As the name suggests, this is a more compact case – with a volume of 0.68 litres vs its siblings' 1.03 litres. This, along with a lower price point, has resulted in several compromises such as the lack of a mini OLED and a single RGB fan rather than two. The tower cooler has also been replaced with an (unofficial) active cooler that sits on Raspberry Pi 5 via thermal pads. Even so, the combination of fans provides good airflow and cooling.

Wear a HAT

There are case cutouts for Raspberry Pi 5's main ports (and camera cable), rather than them being rerouted via an adapter board – so no full-size HDMI ports. The GPIO header is rerouted to the side via the supplied HAT+ board, which also houses an RTC battery and optional M.2 NVMe SSD, and has connections for the RGB fan and a power button on the case exterior.

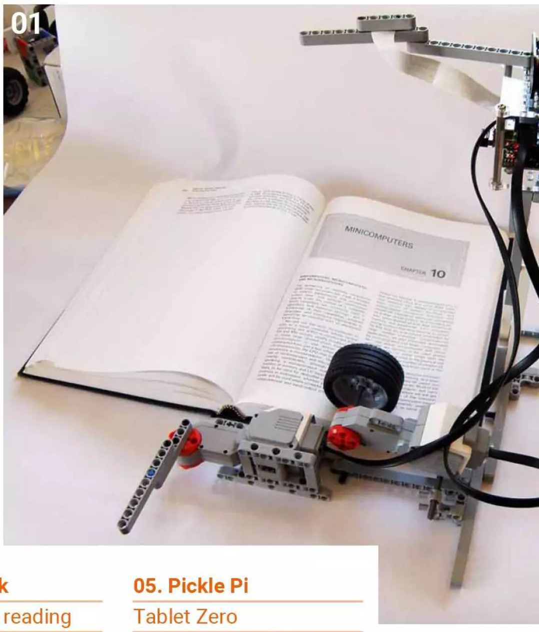
> *The GPIO header is rerouted to the side via the supplied HAT+ board*

With the case itself comprising just a single square aluminium piece and two clear acrylic panels for the sides, assembly is fairly straightforward, aided by good online documentation. As with the other models, installing the Pironman 5 software enables control of the RGB fan/lighting settings, safe shutdown with the power button, and system monitoring via a web dashboard.

▶ The RGB-lit case looks cool and the rerouted GPIO header is clearly labelled

10 amazing: reading projects

Curl up with a good book, comic book, or just have the computer read it to you with these projects



There are numerous ways to read on computers. Endlessly scrolling social media, checking out your favourite ship tag on AO3, or reading through someone's life story before getting to a recipe. You can also use computers for more traditional reading of books (and related). Here's some of our fave reading-related projects…

01. BrickPi Reader

Robot reading

rpimag.co/brickpi2

Using Lego and Raspberry Pi to digitise books is a very clever system. These days you could easily add decent text-to-speech with it too so it reads you to bed.

02. E-ink reader

Kindle competitor

rpimag.co/einkreader

While the main use of this project is daily information, it can easily be modified to read books and such on the easy-to-read and low-power display.

03. DIY e-reader

Retro cool

rpimag.co/diyereader

This tiny e-reader uses mechanical keys to navigate, like a 1970s view of the future. Hopefully they're silent switches so you don't make a huge click every time you turn the page.

04. Open Book

Pico-powered reading

rpimag.co/openbook

A kit you can get yourself to easily assemble an e-reader using a Raspberry Pi Pico. It's a very simple and powerful build.

05. Pickle Pi

Tablet Zero

rpimag.co/picklepi

With a small tablet, you can read books just like you would on an iPad or Android equivalent. This one has a cool keyboard as well, though – checkmate, Tim Cook.

06. Video magnifier

Computer, enhance

rpimag.co/videomag

Need a hand actually seeing the text? This simple project zooms in on paper you place beneath it, making it easier to read and write on.

07. Kavita

Book storage

rpimag.co/kavita

Kavita is like the Plex of books, comics, graphic novels, etc. You can set it up on a Raspberry Pi server and then read your PDFs, EPUBs, CBRs, etc. remotely.

08. Raspberry Pi 5 tablet

Quick switch

rpimag.co/firstrpi5tab

Using a tablet case for a Raspberry Pi 4, Lee of Leepspvideo was able to squeeze a Raspberry Pi 5 into it with minimal problems. Perfect for some e-books in colour.

09. DIY Text-to-speech

A cheaper reader

rpimag.co/diytts

Combining optical character recognition and a TTS engine, this project allows you to put down a piece of paper or a page of a book, and have a Raspberry Pi read it out to you.

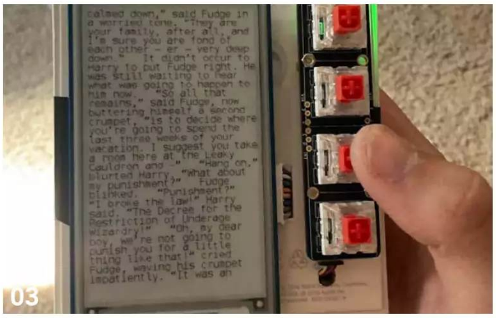
10. Community Jams

Read the notes

rpimag.co/commjams

To keep people on the same page (literally) during a jam session in the park, this group uses a special Raspberry Pi-powered sheet music server.

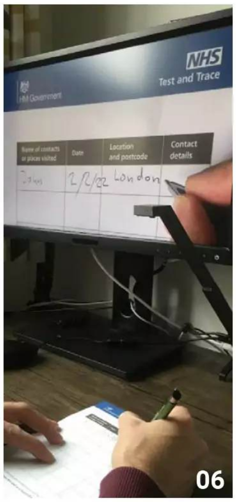
02



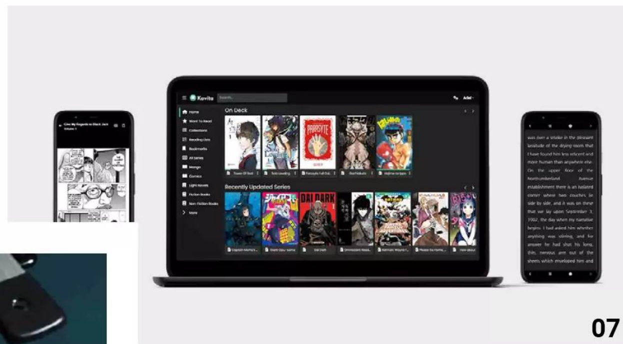
03

04

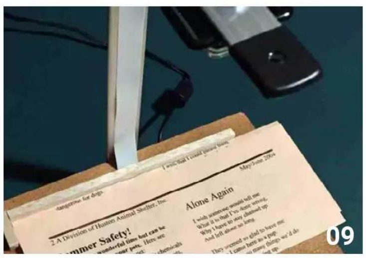
05



06

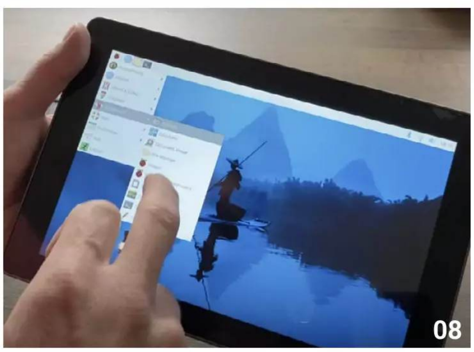


07



09

10



08

Grant Sinclair

Inventor Grant is making it easy to game and code on the go. By **Sean McManus**

- **Name** Grant Sinclair
- **Occupation** Inventor
- **Community role** Creator of GamerCard
- **URL** grantsinclair.com

Sir Clive Sinclair introduced many of us to programming with the ZX Spectrum. Now his nephew Grant is making gaming and coding more accessible to today's kids, with his grab-and-go GamerCard.

It ships with platform game Bloo Kid 2 and shoot-'em-up AstroBlaze DX, both ported from the Nintendo Switch. There will be an app store for more and the PICO-8 Splore browser for new retro-style games. You can add your own emulators, too.

There are two control pads, which each have four dome buttons underneath.

We sat down with Grant to find out more.

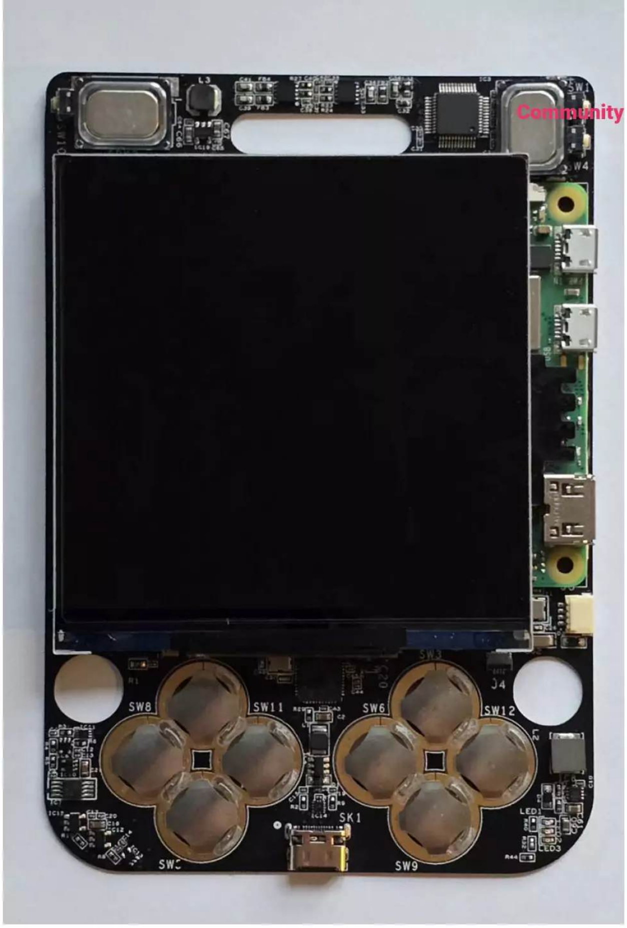
What inspired GamerCard?

I found out that they sell billions of gift cards and they're normally right near the cash register and often gaming focused. I thought if you could do a product which you could grab just before you jump on a plane or a long train journey, and it's got enough games to keep you occupied, it'd be a great product. It's a bit inspired by the ZX Spectrum, which is my family product. Kids will start playing games and then they'll start writing their own.

What influence has Sir Clive had on you?

He influenced me majorly. Also, my father was an industrial designer, so I was influenced by that. I think by the time I was about four, I asked my dad to get me some electronics. He went to see Jim Westwood, Clive's chief engineer, and got me a box of bits.

▶ Grant designed pop-up mail order packaging that doubles as an in-store display stand



▶ Raspberry Pi Zero 2 W is mounted on the right, with a heatsink on top. Speakers are in the top corners **Photo © Grant Sinclair**

What's inside GamerCard?

We did quite an unusual thing in that we've got two separate monoblock amps, one for each speaker, and then we've got a stereo codec chip that feeds them. I wanted to make it so it really pumps. When we were testing, it was just like an arcade. Raspberry Pi thinks that it's got an Arduino-type joystick plugged in. On the initial samples we had a problem with heat build-up because [the device is] so thin. So, I designed a little heatsink made from aluminium. It transfers heat into the chassis, so the chassis is basically one big heatsink. The screen is the same pixels per inch as a MacBook Pro. It really pops off.

Why did you choose Raspberry Pi Zero 2 W?

I looked at lots of different options and all sorts of chips. Raspberry Pi Zero 2 W seemed to have the best ecosystem. There's software already out there and it's very easy to build a launcher on top of the operating system. It's pretty power-efficient and I want to make GamerCard expandable.

What role does the Qwiic connector play?

Education is a big part of what I'm working on. Schools buy Arduinos and Raspberry Pi devices, but they get turned into projects. Because they've been soldered and hacked around, they often get stuck in a drawer. GamerCard is designed as a sealed system. You can buy little breadboards that have a Qwiic connector on, so you can customise your GamerCard without spoiling the product.

What's next for GamerCard?

I've got a distributor and my vision is to get them out to as many people as possible. I've been contacted by lots of gaming companies who want to develop for it. I'm just happy to see already that people love it because you find out straight away if it's a winner or not.

▲ PICO-8's Splore browser gives you access to homebrew retro-style games

> *We've got two separate monoblock amps, one for each speaker… I wanted to make it so it really pumps*

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. It's a big shame that the rotation didn't work! We've only ever done manual rotation stuff ourselves

02. A tidy workspace can do wonders for your productivity and mental health, we find

03. Quite a lot packed into a tiny board here

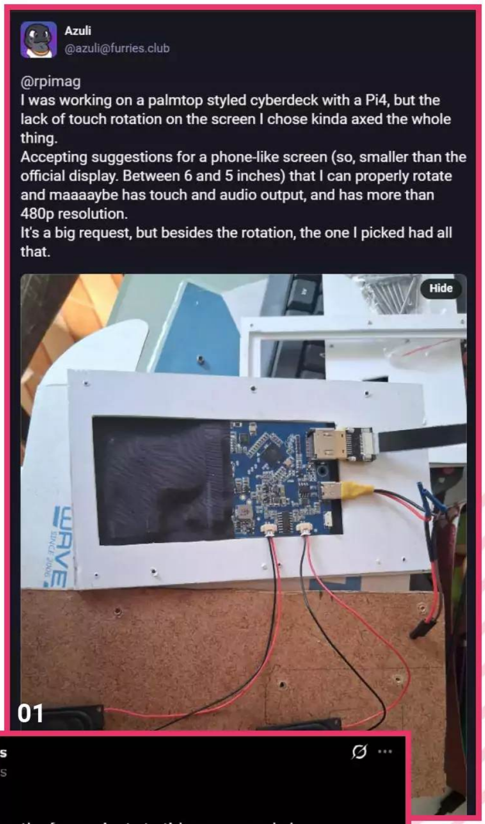
04. Omni wheels, Bluetooth control, and enough ultrasonic sensors to confuse a bat

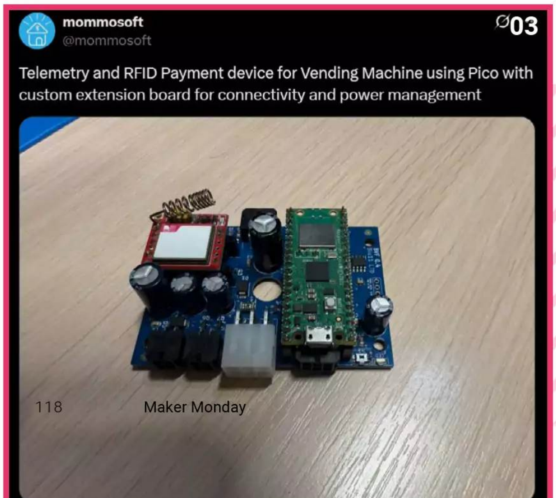
05. When the prototype becomes a bit more final is a wonderful step

06. Could sure go for some tuna nigiri right about now

07. It lives! It moves!

08. The strange case of the AliExpress Red Raspberry Pi





04

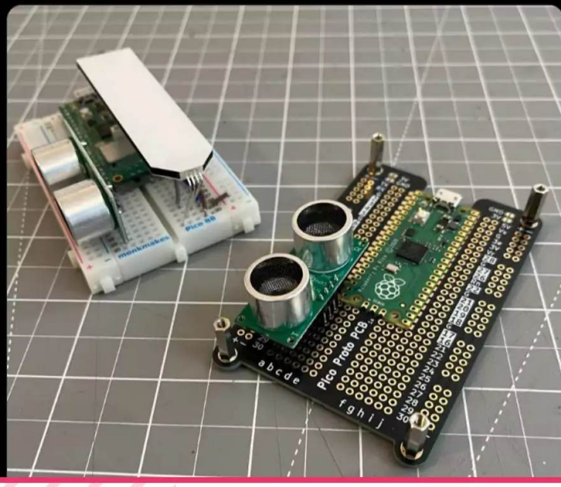
Ulli Sundermann
@ullisun58

Hi today it's very stormy #MakerMonday last week you saw my small #Tiny-4WD with front omniwheels x.com/ullisun58/stat...
in the last few days, I implemented a BLE remote control. Both devices have a Pico W inside. You can clearly see that the omniwheels do their work very well

05

Pater Practicus
@PaterPracticus

Finally making progress on proximity warning sign number 2, planning out the move from Monk Makes breadboard to Pico Proto PCB.
#MakerMonday



06

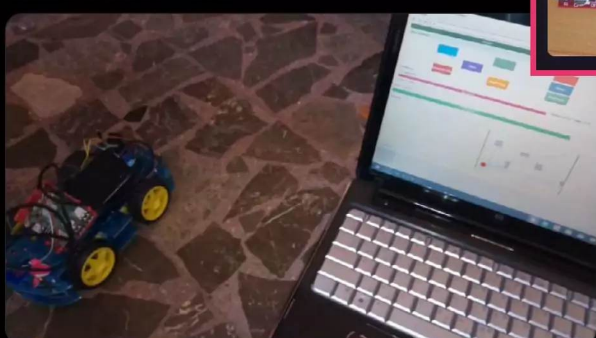
Noboru T
@noborutkhs

@rpimagazine
Here's a very Japanese sushi demo on a Zero 2W, created with CGI Studio.
Video: youtu.be/Mc13dkFuXEY
#MakerMonday

07

Aula Jazmati
@Aula_J2018

EduBot Explorer Part2 : Real-World Control Test 🚗🚗
hackster.io/aula-jazmati/e...



08

あっきぃ
@akkiesoft@social.mikutter.hachune.net

@rpimag Hello!

Last week, I found the Red Raspberry Pi 1B on AliExpress, so I bought it. One of the two was junk that wouldn't boot, but the other was able to boot latest Raspberry PI OS without problem!

I've always wanted one for my collection, so I'm glad I was able to purchase it unexpectedly. 😄

Interestingly, the board had an RJ-45 port, but it was not connected to the LAN9512(Probably a customized board?).

Pico Chess Timer

Check this out: a portable Pico-powered electronic timer for chess matches

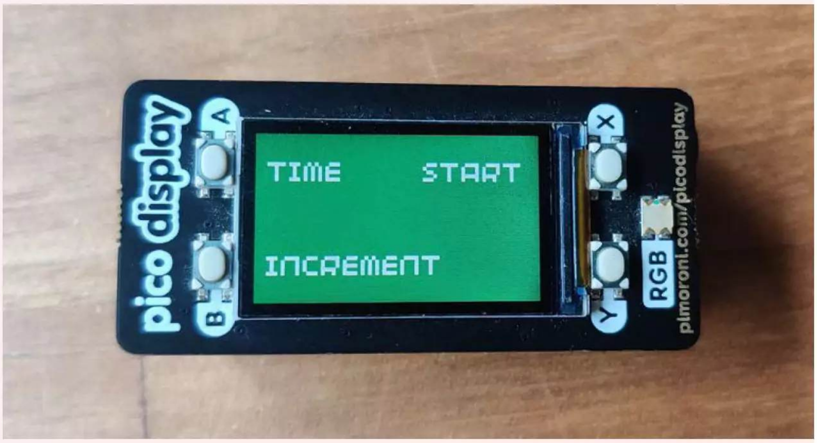
We got an email from Nirvaan, a 14-year-old Raspberry Pi enthusiast who is an avid chess player too.

"I wanted to share a project I made with you – a Pico Chess Timer," his email starts. "I love to play chess. However, many times games can go on for hours without finishing. I wanted to have a way of timing chess games without having to use a phone or computer (which could be distracting). Having created a Pico Pomodoro timer for exams a few months ago, I figured I could reprogram it to keep track of chess games!"

More competitive chess has individual timers for each player, forcing them to think ahead and not spend too much time on each move as the clock ticks down, then satisfyingly pause their timer and restart the opponent's. If the game is not finished by the time the first timer runs out, that player loses.

"I added a menu to set the time for the chess game (e.g. 10 minutes per person), and an option for an increment (e.g. gaining 5 seconds per move) so it could be used for a variety of chess games," Nirvaan explains. "It shows the time remaining for each player and switches players when any of the buttons are pressed. I also added a LiPo battery to make it portable!"

You can get more info about it on GitHub here: **rpimag.co/picochesstimer**



▲ The pressures of a timer might force you to make a wrong move, so plan ahead

◀ The menu is simple and allows for easy customisation

Crowdfund this

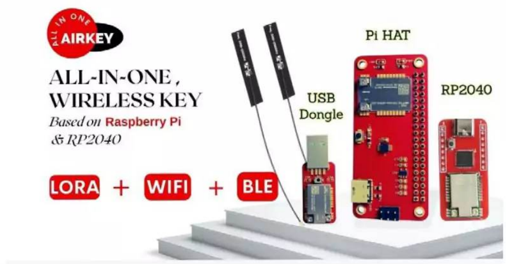
Great crowdfunding projects this month

MakerTiles

While the concept of modular electronic components for educational purposes isn't exactly new, we like the presentation of these MakerTiles. They have plenty of modules available as well, and it's all compatible with Raspberry Pi.

▶ **kck.st/4pZjYuQ**

AirKey



This is a USB dongle, Raspberry Pi HAT, and standalone RP2040 device that allow you to connect to multiple kinds of wireless networks – mainly Wi-Fi, Bluetooth, and LoRa too. The hope is it cuts down on the number of devices you need for connections.

▶ **kck.st/4gSto7l**

Your Letters

Spooky season builds

I laughed when I looked in the magazine and saw not one, but two Halloween prop builds [the other being the Bog Body – actually a theatre prop, but in the spirit of the season we'll allow it]. We've only just had summer – let us enjoy it! That was my first thought, anyway. Now that there are only a couple more weekends before Halloween to make something to entertain my kids, I appreciate you jumping the gun. I've got a cheap skeleton (not the one your writer used, but something that will jump and move), a reused stepper motor, reused PIR sensor, and a battery pack that I wasn't using anyway. I just have to hope one of my pumpkins grows big enough to use as a project enclosure and we're all set for spooky season.

Brian via email

You can never be too early starting work on a project for Halloween – or Christmas for that matter, as we're already working on our twinkly lights. We're delighted that you've been inspired to create your own spooky skeleton, Brian, and it's great that you're repurposing a lot of the parts. Do let us know how you get on with the scaring the kids with it!

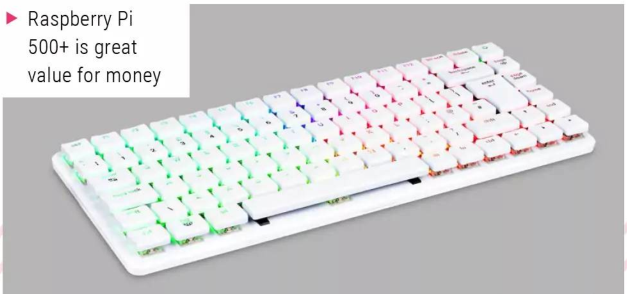
Our animated skeleton from last issue

Raspberry Pi 500+

There's a line in the interview with the developers of the Raspberry Pi 500+ last issue about how one of them remembers waking up with a Commodore 64 under the tree at Christmas. That can't be right – the man in the photograph looks to be around 50 years old, whereas the eighties were only 20 years ago. Everybody knows that!

Sarah via email

Well, quite. Time marches on faster than we can imagine. All our favourite bands – the ones who are still alive, anyway – seem to be doing 30-year anniversary tours for albums that we loved when they were first released when we were kids. However, the one advantage we have now is that as adults we have the disposable income to go and see the bands that we couldn't afford to see the first time around. That's true for computers as well: the Commodore C64 cost $595 when it was released in 1982, whereas the Raspberry Pi 500+ is available for $200 right now. Even without inflation, the better, faster machine is roughly one third of the price – that's absolutely mind-blowing. We're incredibly lucky to have access to so much cool hardware.



Raspberry Pi 500+ is great value for money

IR camera

There's something comforting about the IR camera build in issue 158, in the way that I imagine it makes the user slow down and think about what they're taking pictures of. I have a million shots of my dog, wife, children on my phone that I'll never look at again; if I had to stop and think, and only got to look at them once I'd brought the camera back within range of my home Wi-Fi network, I bet I'd take more care.

Lee via email

This is something that a lot of people forget about technology: it's not just that it can do more, better, faster, more computationally demanding things; often the innovation comes not from the billions of transistors, but from making us think in a different way. There's nothing new in the idea of a digital camera. And in terms of features, the camera we featured in issue 158 is a step backwards compared with the cameras on a bog-standard smartphone, but there's something about the way it uses technology that makes it unique. The way we use technology has had the effect of taking precious photos of our loved ones and making them worthless; anything that reverses that trend has to be good for humankind.

▲ The Yashica IR camera makes you wait and produces some eerie photos

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. Potton Pi and Pints

- **Saturday 1 November**
- **The Rising Sun, Potton, UK**
- **rpimag.co/ppap159**

Fancy doing some computing/electronics stuff in a relaxed, social atmosphere? Fancy a chat with fellow geeky-type people? Fancy finding out about the Raspberry Pi or other devices from first principles? This is an informal family-friendly event from the same team that brings you Cambridge Raspberry Jam and Pi Wars.

02. Melbourne Raspberry Pi Makers Group Meetup

- **Sunday 2 November**
- **Docklands Makerspace and Library, Melbourne, Australia**
- **rpimag.co/mrpmgm159**

This meetup is open to everyone with an interest in electronics, robotics, home automation, 3D printing, laser cutting, amateur radio, high-altitude balloons, space tech, etc. Makers are invited to bring along their projects and project ideas, and come connect with other makers. Get your questions answered, show off the work you are doing, and get support to resolve nagging issues.

03. Cornwall Tech Jam

- **Saturday 8 November**
- **Museum of Cornish Life, Helston, UK**
- **rpimag.co/ctj159**

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.

04. Riverside Raspberry Pi Meetup

- **Monday 10 November**
- **3600 Lime Street, Riverside, CA, USA**
- **rpimag.co/rrpm159**

The purpose of Riverside Raspberry is to share knowledge related to Raspberry Pi hardware in particular, and to promote interest in tech development in the Inland Empire in general. The group is currently meeting on the second Monday evening of each month.

Riverside Raspberry
Raspberry Pi Users Group
Let your computer play in the real world!
Meetup
www.meetup.com/Riverside-Raspberry

FULL CALENDAR

Get a full list of upcoming community events here:

rpimag.co/events

05. Maker Faire Shenzhen

- **Saturday 15 November to Sunday 16 November**
- **Vanke Design Commune, Shenzhen, China**
- **rpimag.co/mfs2025**

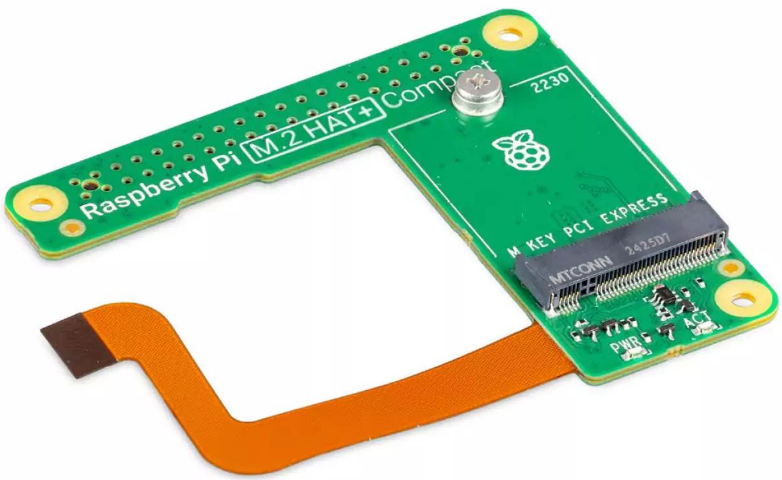
Raspberry Pi are proud to partner with Raspberry Pi Approved Reseller Seeed Studio once again at Maker Faire Shenzhen 2025. Come and meet members of the Raspberry Pi team, learn about latest products, and share what you've made with Raspberry Pi technology.

Official Raspberry Pi Event

Win 1 of 10

M.2 HAT+ Compact

Raspberry Pi's M.2 HAT+ Compact enables you to connect M.2 peripherals such as NVMe drives and AI accelerators to Raspberry Pi 5's PCIe 2.0 interface, supporting fast (up to 500 MB/s) data transfer to and from NVMe drives and other PCIe accessories.



Head here to enter:
rpimag.co/win
Learn more:
rpimag.co/m2compact

Terms & Conditions

Competition opens on **22 October 2025** and closes on **20 November 2025**. Prize is offered to participants worldwide aged 13 or over, except employees of Raspberry Pi Ltd, the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from Raspberry Pi Official magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram, Facebook, Twitter (X) or any other companies used to promote the service.

Official Magazine
#160

Next Month

Christmas Special

- Santa's Raspberry Pi list
- The best holiday projects
- Finish off your Christmas lights

On sale 20 November

PLUS!

Build a working *Aliens* radar tracker

Publish your own zine with Raspberry Pi

Build an image dataset for AI training

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, Ben Everard, Rosie Hattersley, Jo Hinchliffe, Nicola King, Phil King, Andrew Lewis, Sean McManus, KG Orphanides, Toby Roberts

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)

All right all right!!

Artificial Intelligence, Hollywood style. By **Andrew Gregory**

This month, amid the usual cacophony sparked by artificial intelligence, some clarity cut through the noise from an unexpected source. The actor Matthew McConaughey, he of *The Wolf Of Wall Street*, *A Time To Kill*, *Interstellar* and points between, put forward a positive vision for AI that I'd not heard before. McConaughey reckons that he'd like a private large language model, or as he puts it, a version of ChatGPT that's trained only on things that he's written himself.

> *When you want to know what time the last bus home is on a Sunday, you don't need billions of data points: you need one*

Instead of harnessing the internet at large, with its variety of sources ranging in quality from the unimpeachable to the ludicrous, he'd prefer a way of referring to his own thoughts, with a little bit of intelligence to link ideas together. There'd be no more mistakes like getting the number of Rs in 'strawberry' wrong; no more hallucinations, no more AI reacting to videos that are themselves created by AI. Just one man, using technology to help marshal his thoughts.

In one way, this approach is a complete failure to understand what AI is all about. We have the power to distil information from all over the internet, taking in the viewpoints and experiences of millions of people creating billions and billions of data points. On the other hand, when you want to know what time the last bus home is on a Sunday, you don't need billions of data points: you need one.

If you were to create a LLM trained only on words that you'd written yourself – your diaries, your musings, your social media posts – you'd vastly cut down on the amount of data used, which should in turn lessen the processing power taken to churn through that data. Suddenly the environmental damage, the much-publicised water usage of AI data centres, shrinks to nothing.

There would be no ethical concerns over copyright, as you'd own everything. And there would be no problem verifying the accuracy of the data, in part because you'd have no incentive to fill the LLM with nonsense, and also because you'd know your own limitations. There would be no point asking an LLM trained on my writings how to build a nuclear power station, for example, because I know that I don't know.

If this sounds fanciful, maybe it shouldn't. Useful Sensors has a range of sensors with inbuilt AI running on the device, so that it can, for example, recognise when a user is looking directly at a camera, without sending any data back to its parent company. And in issue 158 (**rpimag.co/158**) we ran a project showcase on the Ubo Pod AI assistant, a personal, private LLM that processes your data locally. When we get AI right, odds on it'll be thanks to small firms, motivated individuals, and Raspberry Pi.

Andrew Gregory – Author

Now that the nights are drawing in, Andrew huddles around the RGB glow of a Raspberry Pi 500+ for warmth and to scare away predators.

rpimag.co