Send
a Raspberry Pi to space, in a can

Compute
correctly with these dos and don'ts

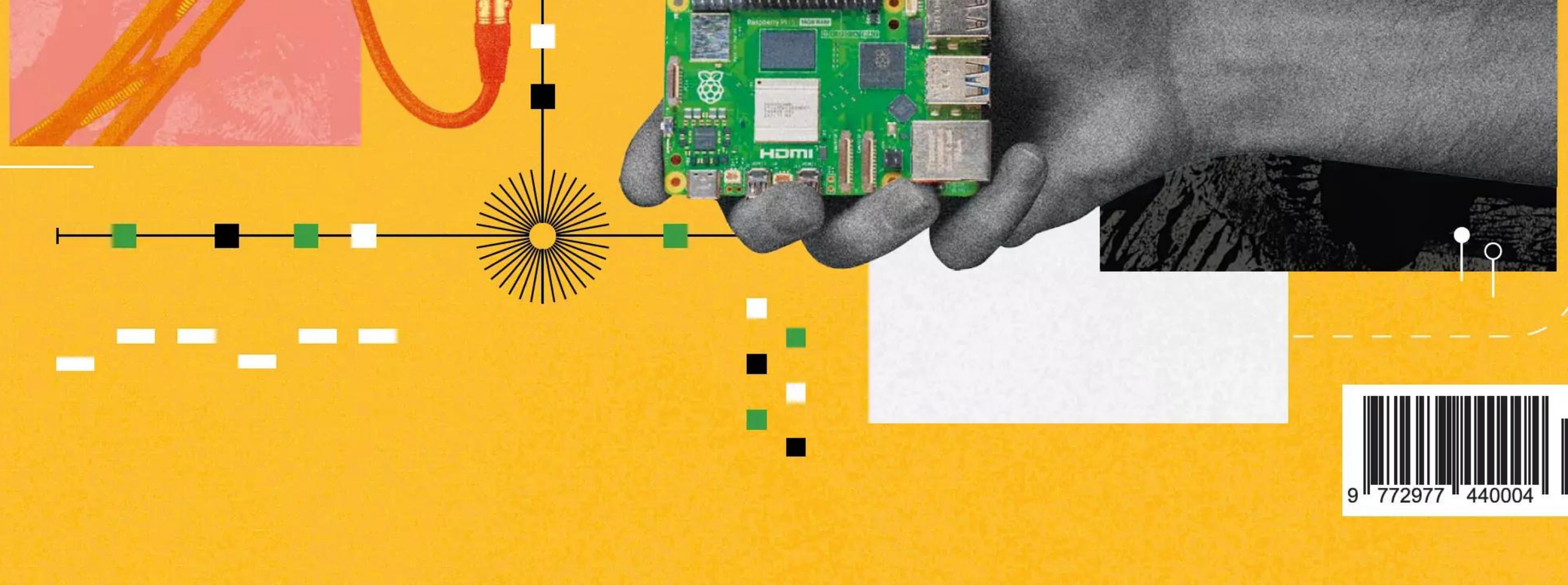
Get the best
LED lighting for your project

Raspberry Pi

Official Magazine
#166 | June 2026

MAKE MUSIC WITH RASPBERRY PI

The complete music production toolkit

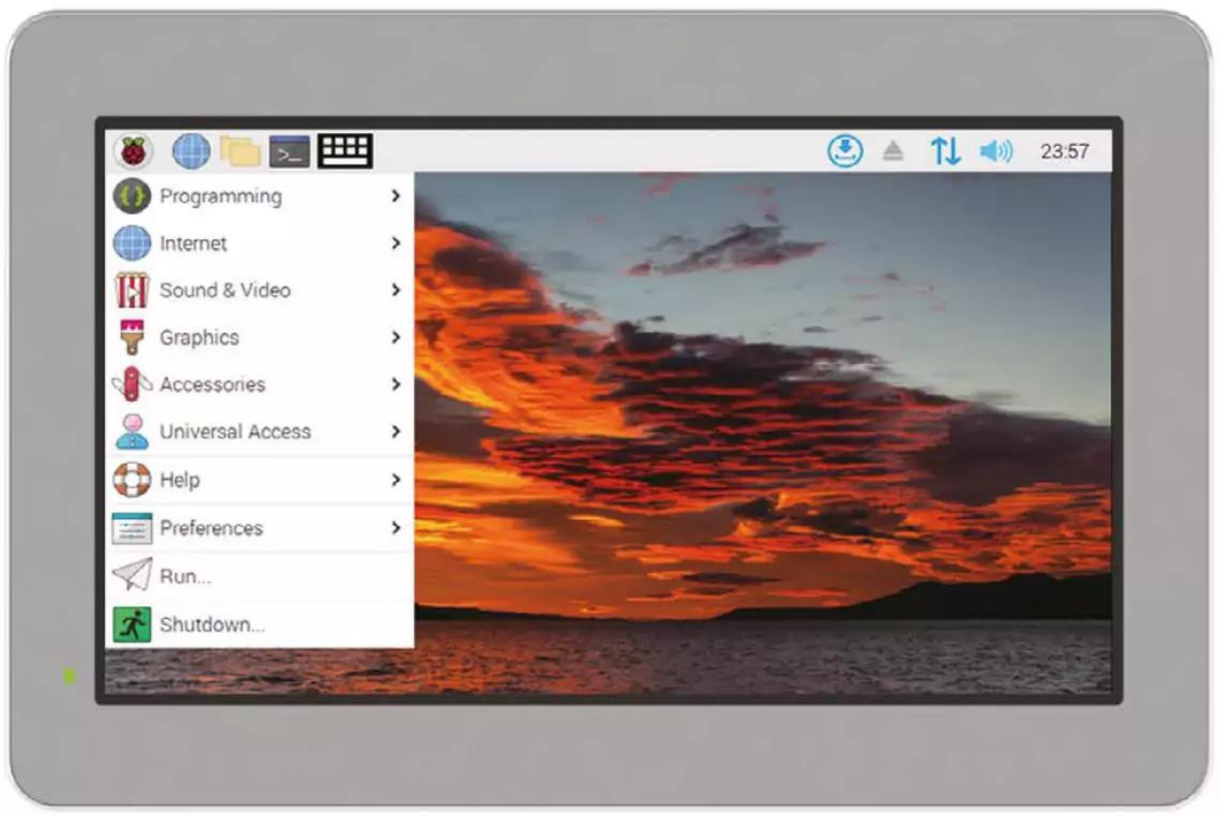


9 772977 440004 06
£7.99
ISSN
2977-4403
(Print)

Industrial Raspberry Pi

ComfilePi





UL LISTED

Powered by Raspberry Pi

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

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

Visit www.comfiletech.com

© copyright COMFILE Technology, Inc. ALL RIGHTS RESERVED

COMFILE TECHNOLOGY

Welcome to Raspberry Pi Official Magazine



Editor

Lucy Hattersley

Lucy is editor of *Raspberry Pi Official Magazine* and this month has been asking OpenClaw to run a Monte Carlo simulation on her chances of getting a dog in the next twelve months.

rpimag.co

Making music is a form of magic. But it's deeply rooted in science and technology – from scale notation to waveforms and amplified audio via circuits and silicon. It's never a surprise to me to find computer buffs and musicians in the same room.

Raspberry Pi is capable of being a synth, sampler, and running the audio editing software you need to bring sonic creations to life.

KG Orphanides has assembled the ultimate Music Creation Toolkit for us. Whether you are laying down beats, looping a guitar, or coaxing strange sounds out of computer hardware, there's a setup with your name on it.

Sean McManus has delivered a seriously useful guide to Raspberry Pi Dos and Don'ts. You'll also find Point Nemo's CanSat heading for the skies and I've written a tutorial on getting OpenClaw running.

Enjoy this issue!

Lucy Hattersley – Editor



Sixfab

Complete Edge AI Hardware Stack for Raspberry Pi

Under 5 watts. From $63 to fleet-deployed.

One DEEPX NPU. Three form factors. The same SDK, same models, same vendor, from the prototype on your desk to the industrial fleet in the field.

START HERE



SIXFAB AI HAT+ FOR RASPBERRY PI 5

A straightforward way to add 25 TOPS of neural processing to your Pi 5. HAT+ specification compliant.
DEEPX DX-M1M (25 TOPS) or DX-M1ML (13 TOPS). Works on Raspberry Pi 5 and Compute Module 5 (via Raspberry Pi CM5 IO Board).

Best for: **weekend builds, classroom projects, first prototypes.**

EXPAND



SIXFAB EDGE AI EXPANSION BOARD

Bottom-mount Pi 5 board with three M.2 slots:

AI accelerator, NVMe SSD, LTE/5G modem.

USB-C PD powers the Pi 5 and the full stack from one cable.

Best for: **outdoor cameras, connected sensors, field demos.**

DEPLOY



SIXFAB ALPON X5 AI

CES 2026 Best of Innovation

Industrial edge AI computer on Pi CM5 + DEEPX.

Fanless aluminum enclosure. Cellular, dual Ethernet, eSIM. Fleet-managed via ALPON™ Cloud.

Best for: **factories, smart cities, real-world fleets.**

SIXFAB × ULTRALYTICS ACCELERATION PATH

Bring your dataset. Train with Ultralytics. Export to ONNX, run on the DEEPX NPU, same path on every Sixfab device.

Raspberry Pi Design Partner

Intelligented by DEEPX

ultralytics | Sixfab

Scan to browse and pre-order

Contact: hello@sixfab.com

© 2026 Sixfab, Inc. 825 Watter's Creek Blvd., Suite 250, Allen, Texas 75013 USA

Contents

World of Raspberry Pi

008 A Raspberry Pi OS security update

010 Major Raspberry Pi Connect upgrades

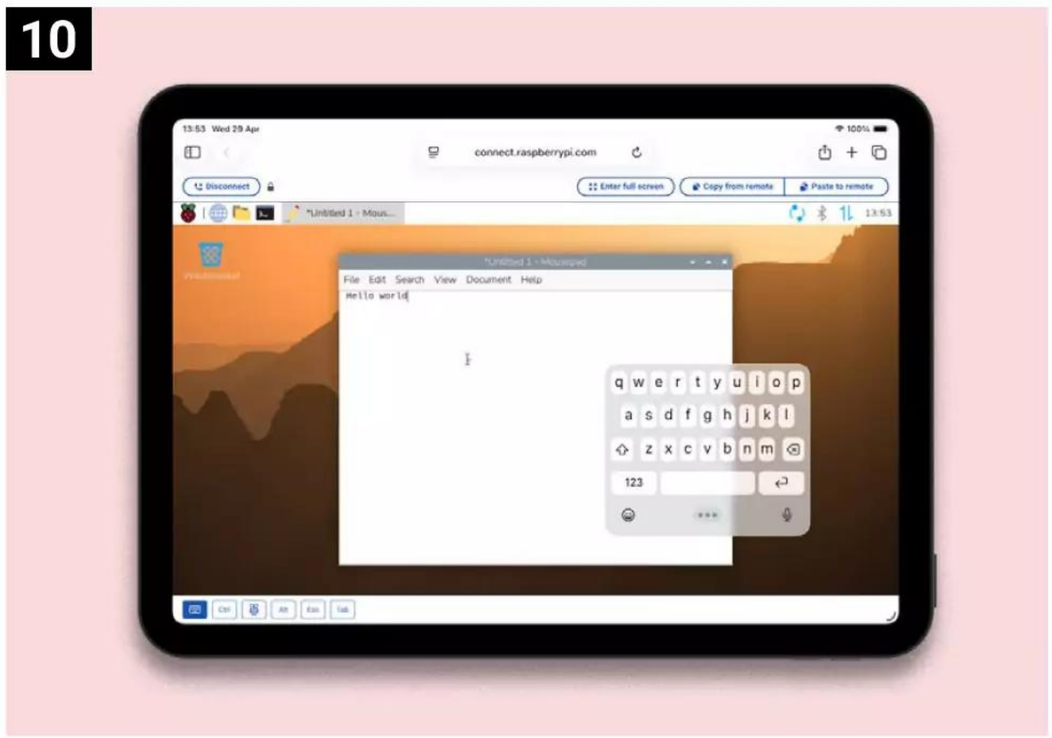
012 Subscribe to Raspberry Pi Official Magazine

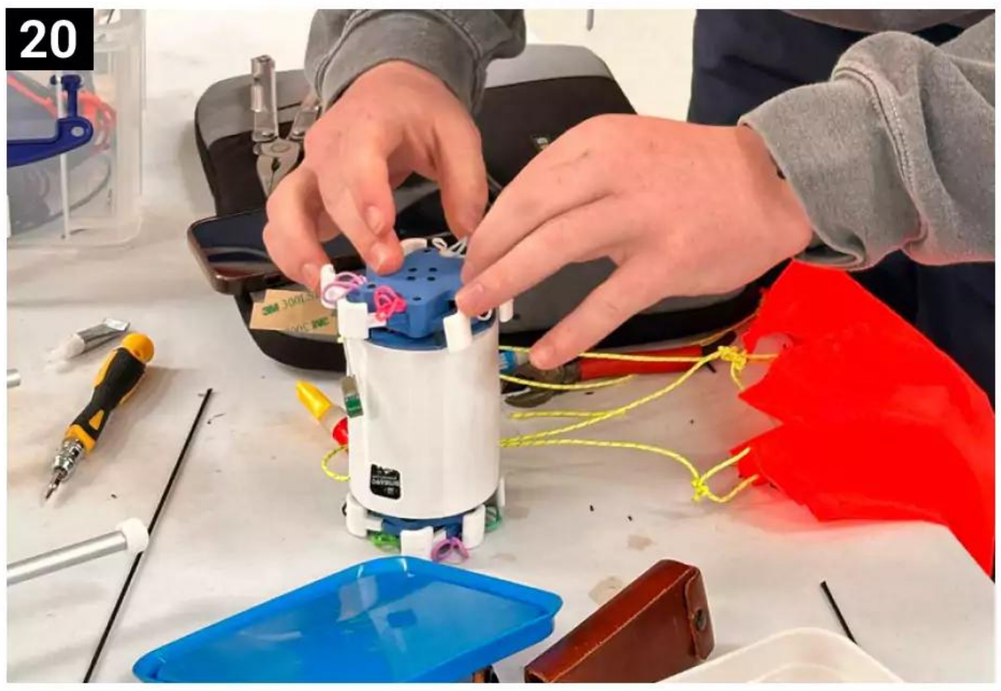
Project Showcase

014 AI Arcade

018 Stewart Warner radio retrofit

020 Point Nemo's CanSat







Raspberry Pi (ISSN No: 2977-4403, USPS No: 25-672) is published 12 times per year by Raspberry Pi Ltd, and distributed in the USA by Asendia USA, 701 Ashland Ave, Folcroft PA. POSTMASTER: send address changes to Raspberry Pi, 701 Ashland Ave, Folcroft, PA. 19032





Top Projects

024 Sequino

026 Raspberrarium

028 Poor man's Polaroid Camera

030 Mermaid in the shell

032 3D print showcase

Feature



034 **Make Music with Raspberry Pi:** The complete music production toolkit

Tutorials

042 Set up OpenClaw on Raspberry Pi

050 Build an RP2040 foam cutter

056 Make mosaics in FreeCAD

062 Sew a sensory puppet

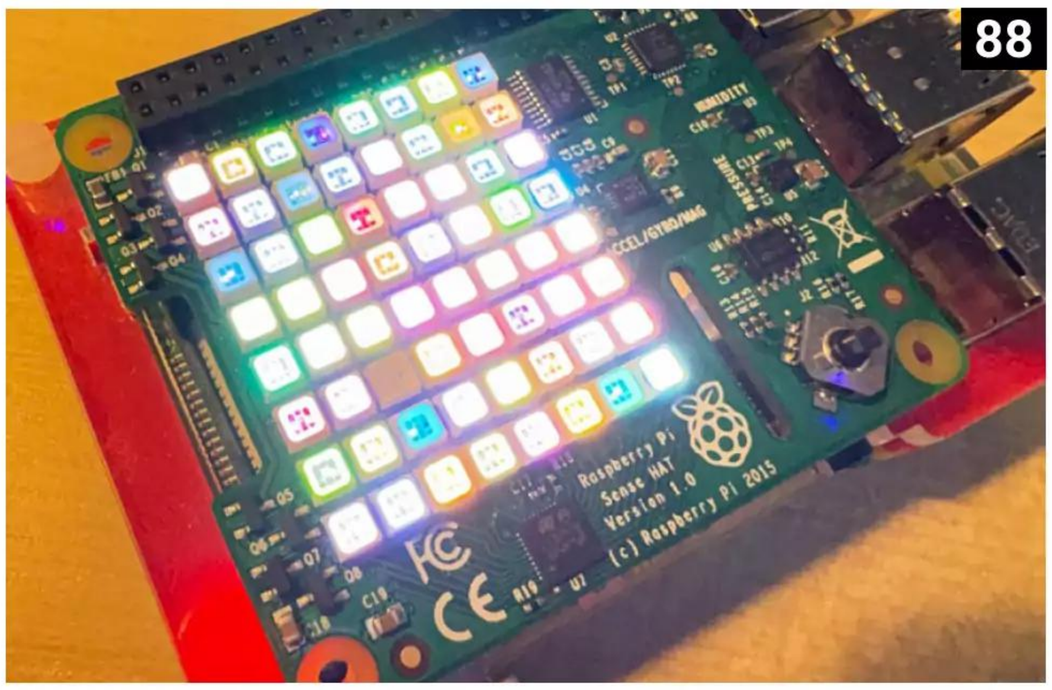
066 Combine 3D printing with vintage cameras

070 Computers that made the world: EDVAC – part 2

080 Conquer the command line – part 11

084 Simple electronics with GPIO Zero – part 7

Feature



088 **Raspberry Pi Dos and Don'ts:** Don't do what Donny Don't does



Reviews

102 **Only the best:** LED lighting

108 Pironman 5 Pro Max

110 Powered by Raspberry Pi

114 **Ten amazing:** Gaming controllers

Raspberry Pi Community

116 Interview: Steve Davies

118 This Month in Raspberry Pi

122 Your Letters

124 Community Events Calendar

Competition

126 **Win** 1 of 10 Sixfab AI HAT+

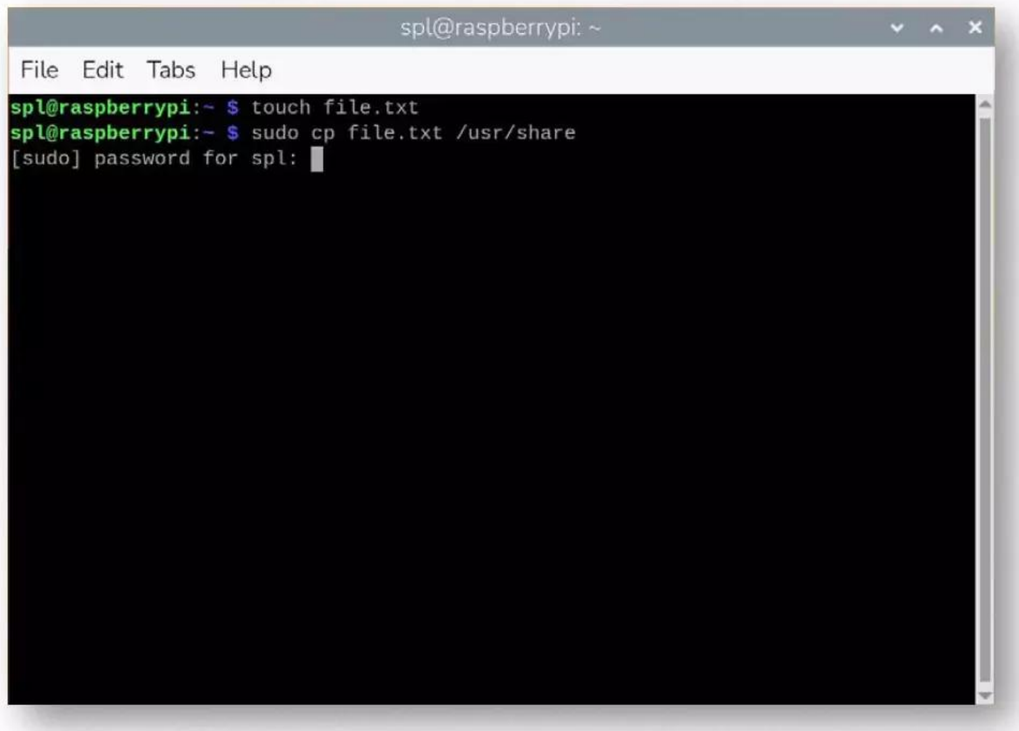


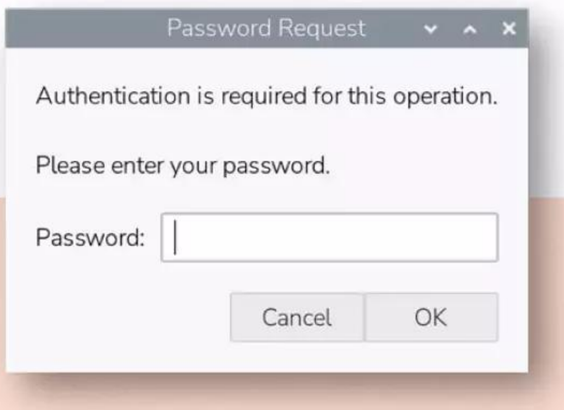


Disclaimer: Some of the tools and techniques shown in Raspberry Pi Official Magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of 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.

A security update for Raspberry Pi OS

Raspberry Pi OS 6.2 lands with a small but significant change: passwordless sudo is now disabled by default. By **Simon Long**





▲ The terminal now prompts for the user's password the first time sudo is used in a session

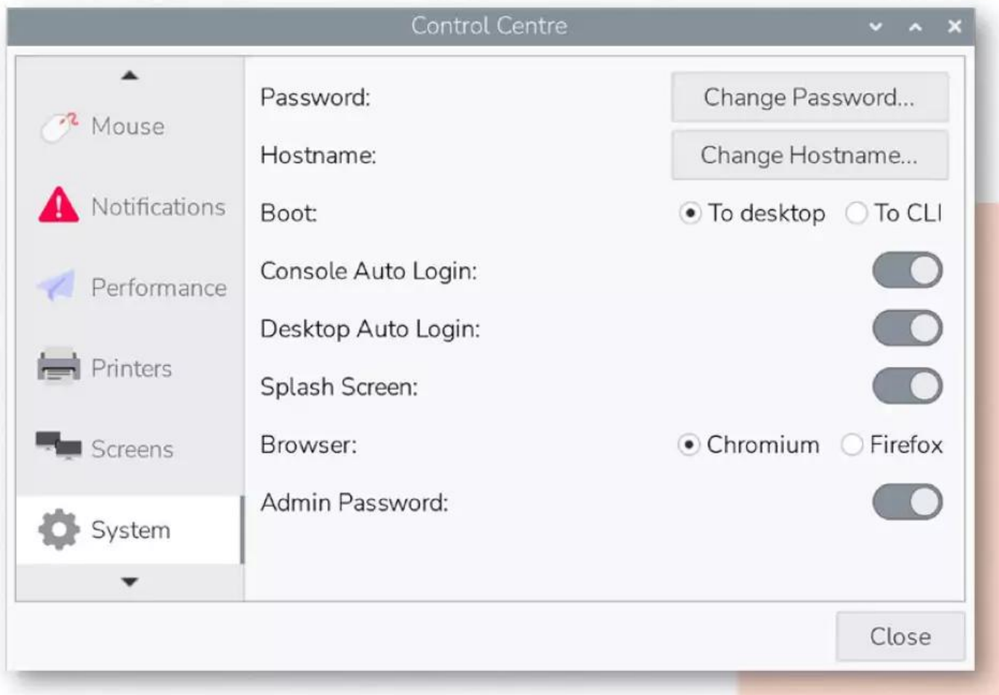
◀ Desktop actions that need sudo access trigger a password dialog box

Raspberry Pi released version 6.2 of Raspberry Pi OS on 14 April 2026, the second update to the Trixie version released last year. This update is mostly a round-up of all the small changes and bug fixes we have made over the past few months, but there is one significant change that we'd like to flag up: passwordless sudo is now disabled by default.

Given the ever-increasing threat of cybercrime, we continually review the security of Raspberry Pi OS to ensure it is sufficiently robust to withstand potential attacks. This is always a tricky balance, as anything that makes the operating system more secure will invariably inconvenience legitimate users to some extent, so we try to keep such changes to a minimum. This particular security update is one that many users may not even notice, but it will affect some.

What would superuser do?

All Linux systems have two types of user account: a regular user and an administrator. Administrators are able to access the system in ways that regular



▲ The Admin Password toggle in Control Centre lets users restore the original passwordless behaviour

users cannot – the most obvious being that they can write files to parts of the file system that are restricted to others.

However, regular users sometimes need to perform administrator-level actions. To enable this without having to log out and then log back in as an administrator, users can employ Linux's sudo utility. Short for 'superuser do', sudo is a command prefix that tells the system to perform an action as if it were requested by an administrator rather than a regular user. It is most commonly used when interacting with the terminal – so while `cp file.txt /usr/share/` would not be permitted (as a regular user cannot write to the **/usr/share** directory), `sudo cp file.txt /usr/share/` would.

Raspberry Pi OS has always been set up so that regular user accounts can use sudo in this way. Specifically, it has been configured with passwordless sudo enabled by default – all a regular user needs to do to execute a command as an administrator is prefix it with `sudo`. However, this creates a potential security hole, as anyone who can access the computer can perform administrator actions from a regular user account, some of which could be malicious.

Password, please

From this release onwards, passwordless sudo is disabled by default. If you use sudo for administrator-level access, you will be prompted to enter the current user's password. In the terminal, the password prompt will appear as soon as you issue a sudo command. If you enter the correct password, the command will proceed as normal; if you enter an incorrect password, the command will be refused.

Certain actions in the desktop interface also require sudo access, including some operations in Control Centre. In these cases, a dialog box will pop up asking for the password. Once a password has been entered, you won't be prompted again for the next five minutes, even if you perform further sudo actions in that period.

Anything that makes the operating system more secure will invariably inconvenience legitimate users to some extent

Opting out

If you prefer not to be prompted for sudo passwords, there is an option to revert to the original passwordless behaviour in the 'System' tab in Control Centre. By toggling the 'Admin Password' switch off, the system will no longer ask for a password before running sudo commands in either the terminal or the desktop interface.

As with all such changes, we are aware that this update may inconvenience some users who have become accustomed to the way things have worked up until now, but we hope you can understand the reasoning behind it.

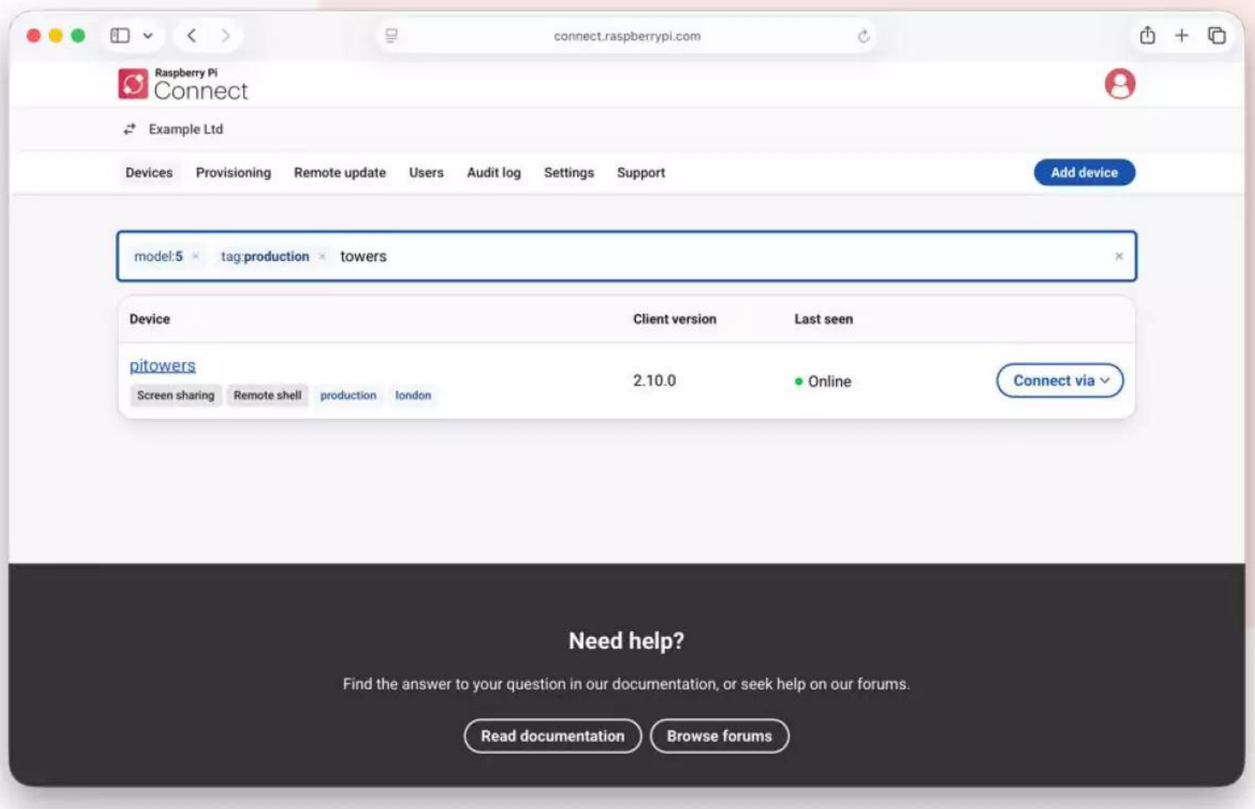
Existing installs unaffected

Please note that this change will not affect updates to existing installations of Raspberry Pi OS – the Admin Password switch will appear in Control Centre as shown above, but passwordless sudo will remain enabled unless you choose to disable it.

Raspberry Pi Connect: tags, required 2FA, and a mobile keyboard

Three new updates to Raspberry Pi Connect make managing fleets of devices a great deal easier. By **Paul Mucur**

▼ Stack tag and model filters in the search bar to drill down into a fleet of devices



Raspberry Pi Connect lets you access your Raspberry Pi devices remotely from anywhere, straight from a web browser (rpimag.co/connect). Since we last wrote about Connect, we've shipped three updates that we think will make it more useful – particularly for the growing number of teams using Connect for Organisations to manage fleets of devices.

Tag and filter your devices

Once you have more than a handful of Raspberry Pi devices in an organisation, finding the right one quickly starts to matter. With these new updates, you can now apply tags to any device – for example, by location (**london**, **cambridge**), by environment (**production**, **staging**), or by what the device actually does (**point-of-sale**, **kiosk**). Tags appear underneath the device name on both the

device page and the device list, and any administrator can add or remove them from the device's Settings page, or when first linking a device to your account.

The search bar at the top of the device list now combines free-text search with structured filters. Type a qualifier followed by a colon – **model:5**, **memory:4gb**, **os:raspios-13**, or **tag:production** – and Connect will narrow the list as you type. You can even stack filters in a single query: **model:5 tag:production dashboard** will find every Raspberry Pi 5 tagged with 'production' that has 'dashboard' in its name. Selecting any tag in the device list adds it to your search instantly.

Tags are also exposed through the Management API, meaning you can apply them when you create an authentication key during provisioning – handy if you're scripting the roll-out of a new batch of devices.

Require two-factor authentication

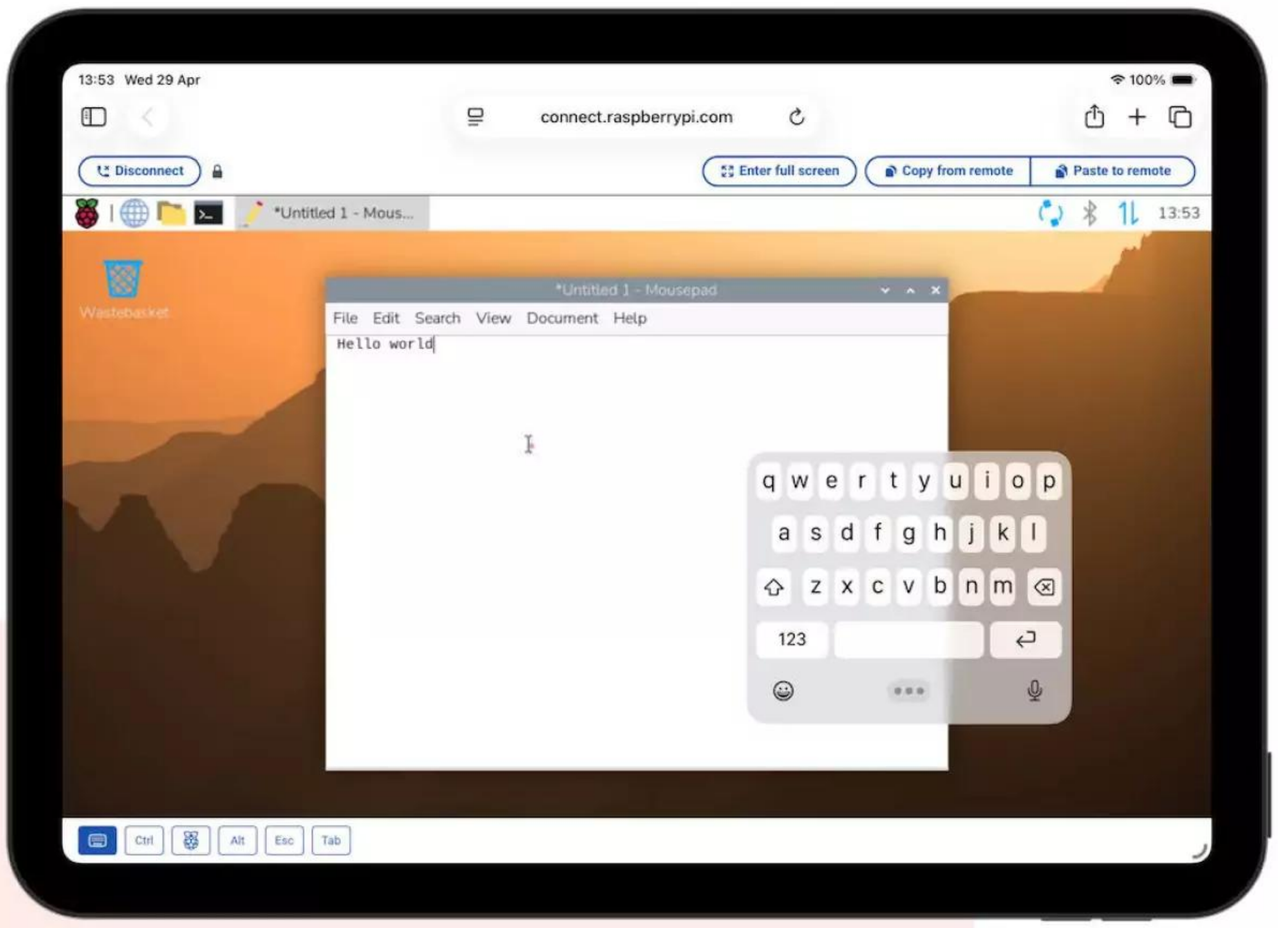
Connect for Organisations now lets administrators require all members to use two-factor authentication (2FA) on their Raspberry Pi ID. It's a single switch in the new Authentication section of the organisation's Settings tab, and it adds a meaningful layer of protection against compromised member accounts being used to reach your devices.

Turning it on starts a 14-day grace period. During that window, members without 2FA see a banner showing how long they have left and a link to enable it on their Raspberry Pi ID; everyone else carries on as normal. When the grace period ends, any member still without 2FA is blocked from the organisation until they enable it. They won't be able to access devices or other organisation resources in the meantime.

If you ever want to relax the requirement, you can turn off 2FA in your organisation settings. Disabling and re-enabling 2FA resets the grace period, so you can give members another two weeks if you need to.

Mobile keyboard support

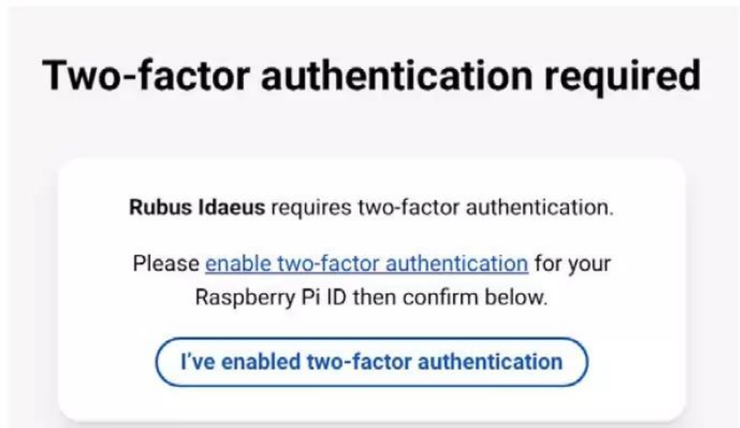
Connect's screen-sharing interface works on phones and tablets as well as desktops, but typing on a touch device was previously only possible if you attached a physical keyboard. The screen-sharing toolbar now includes a dedicated Keyboard toggle alongside the existing buttons for **CTRL**, **ALT**, **ESC**, and **TAB**, allowing you to use an on-screen keyboard on mobile devices without any extra hardware attached.



▲ A new Keyboard toggle on the screen-sharing toolbar lets you type from a phone or tablet without a physical keyboard

Try it out

Connect is free for personal use. Connect for Organisations comes with a four-week free trial, after which you're billed monthly in arrears for the peak number of registered devices. Sign in at **connect.raspberrypi.com** to get started, or read the full instructions in the Raspberry Pi Connect documentation (**rpimag.co/connectdocs**).



▲ Administrators can require two-factor authentication for all members of an organisation

Once you have more than a handful of Raspberry Pi devices in an organisation, finding the right one quickly starts to matter

SUBSCRIBE TODAY
FOR JUST £10

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

SUBSCRIBER BENEFITS

Free delivery

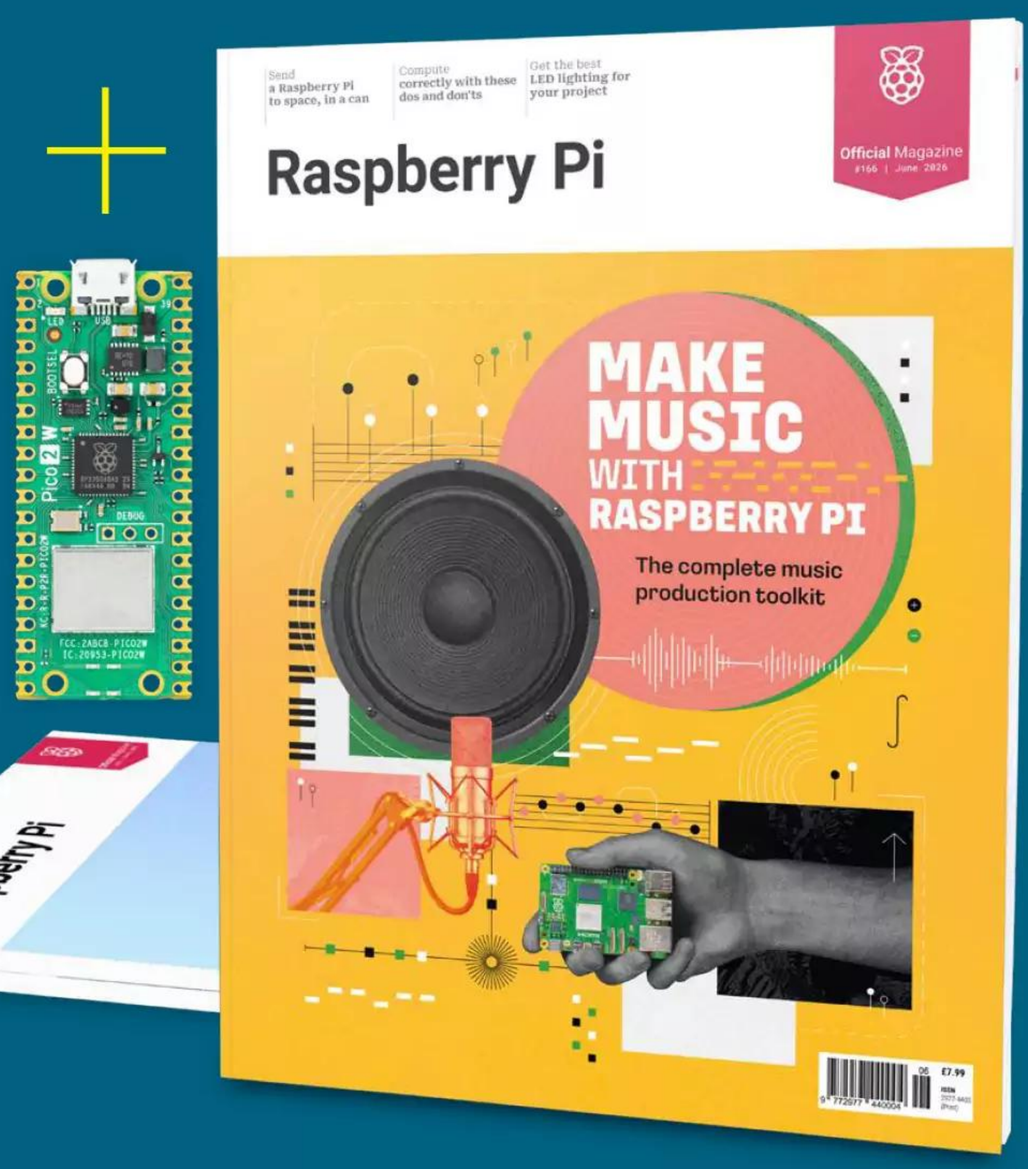
Get it fast and for free

Exclusive offers

Great gifts, offers, and discounts

Great savings

Save up to 37% compared to stores



SUBSCRIBE FOR £10

Free Pico 2 / Pico 2 W

3 issues of Raspberry Pi Official Magazine

£10 (UK only)

SUBSCRIBE FOR 6 MONTHS

Free Pico 2 / Pico 2 W

6 issues of Raspberry Pi Official Magazine

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

Subscribe by phone: 01293 312193

Subscribe online: rpimag.co/subscribe

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

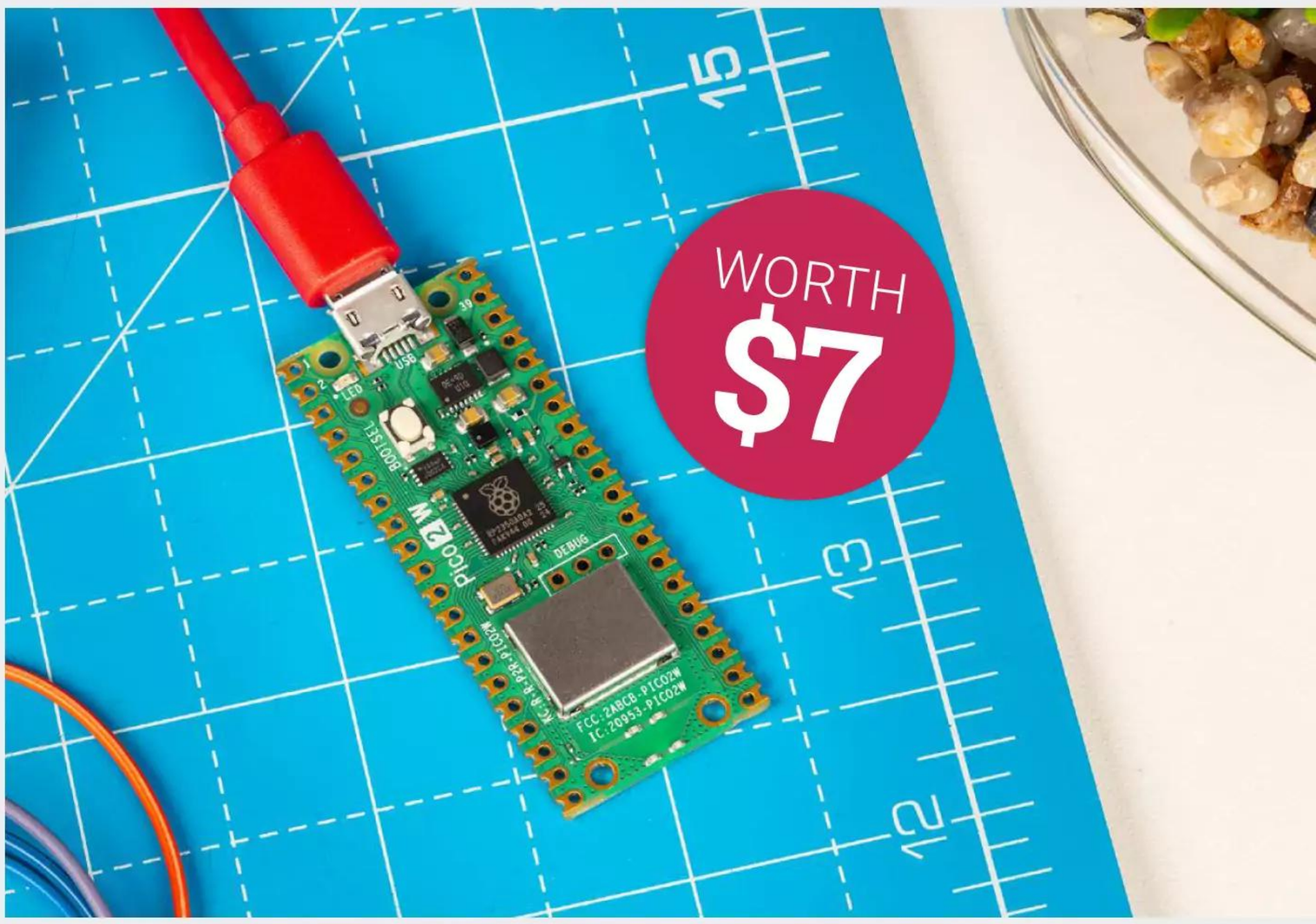
Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A choice of free Raspberry Pi Pico 2 W or Pico 2 is included with all subscriptions.

SUBSCRIBE TODAY AND GET A

FREE Raspberry Pi Pico 2 W (or Pico 2)

Subscribe in print today and get a **FREE development board**

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



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



Buy now: **rpimag.co/subscribe**

SUBSCRIBE on app stores

From **£2.29**

Available on the App Store

GET IT ON Google Play

AI Arcade

You won't run out of retro-inspired games to play with this console.

By **David Crookes**



Maker

Grigor Todorov

Grigor is a freelance developer living in London who specialises in Apple Vision Pro, Unity, the Unreal Engine, AR, VR, AI, and more.

rpimag.co/aiarcade

Raspberry Pi is at the heart of many DIY games consoles. By running emulation software such as RetroPie, Recalbox, Batocera, and Lakka, our favourite computer is more than capable of playing a host of classic games from a host of systems spanning the 8-bit to 32-bit eras. When Grigor Todorov decided to produce his own console, however, he decided not to base it around existing systems like the Sega Mega Drive or SNES. Instead, he pursued a project that would allow him to enjoy new titles, made on the fly, using the magic of AI.

Grigor is an avid gamer. "I enjoy playing games on my Steam Deck, especially roguelikes such as Hades and The Binding of Isaac," he says. "I like games that are quick to start, replayable, and a little unpredictable, which strongly influenced the idea behind AI Arcade." Having already owned a Raspberry Pi and an arcade joystick left over from other projects, Grigor was able to immediately start working on his console. He wanted it to generate as many games as possible using large language models (LLMs).

"I wanted to experiment with LLMs in a more playful, physical way," he explains. "As LLMs have improved so quickly, I became curious about what the future of gaming might look like with AI involved. One possibility was a device that could generate endless new games, giving you a fresh experience every time." For this he turned to ChatGPT and engaged in a spot of vibe-coding (which involves describing an app in natural language rather than directly writing code) over the course of an afternoon.

> *The system is able to produce simple, addictive games that invoke the nostalgia of a bygone era*



Code generation

Using a Raspberry Pi felt natural for this project. “The Raspberry Pi is very versatile, compact, and power-efficient,” Grigor says. “It also has a huge online community and excellent documentation, which makes it ideal for projects like this. I like that the console could be made fully self-contained. I did not add a battery for this version, but it would be possible to turn it into a portable, self-powered box that only needs to be connected to a screen.”

From the start, Grigor wanted the console to be as easy to use as possible. “That was one of the main goals,” he says. “I wanted a Raspberry Pi computer to host a web page locally and automatically launch it full-screen on startup. I also wanted to ensure that the user needn’t understand anything technical. The aim was that people would just power it up, connect the HDMI cable to a screen, wait for it to load, and then either prompt the LLM to create a new game or play one that has already been generated.”

With all of that in mind, Grigor got down to generating the local web app. He got the program to produce a browser-based user interface that could be navigated using a joystick controller with two buttons via a browser Gamepad API. He also ensured the interface could be navigated using a keyboard. The web app was originally set up so that a completely random game would be generated straight away, but he found this could be expanded. “I found it more interesting when the player could steer the result a little,” he reveals. “The choices give the user some influence over the theme, mechanics, or style of the final game.”

01. The wooden box is reused packaging from a set of glass cups

02. Games are generated via AI and they’re also saved for future play

The result was an app which uses two LLM API calls. “First, the app asks the model to generate four questions, each with two possible answers. This only takes a few seconds,” Grigor says. “The player then chooses between the answers, and their selections are used in a second prompt. That second prompt is used to generate an HTML file, which is then displayed as the game.”



Retro roots

Even though he was vibe-coding, it wasn’t really a shortcut to success. “One of the main challenges was making the experience feel smooth and appliance-like, rather than like a computer running a script,” he says. “Getting the Raspberry Pi to boot, host the local web page, open Chromium, and go full-screen automatically was an important part of that. Another challenge was dealing with the unpredictability of LLM-generated code. Sometimes the generated games work surprisingly well, and other times they are a bit broken. I tried to make the prompting more structured so that the output would be consistent and playable.”

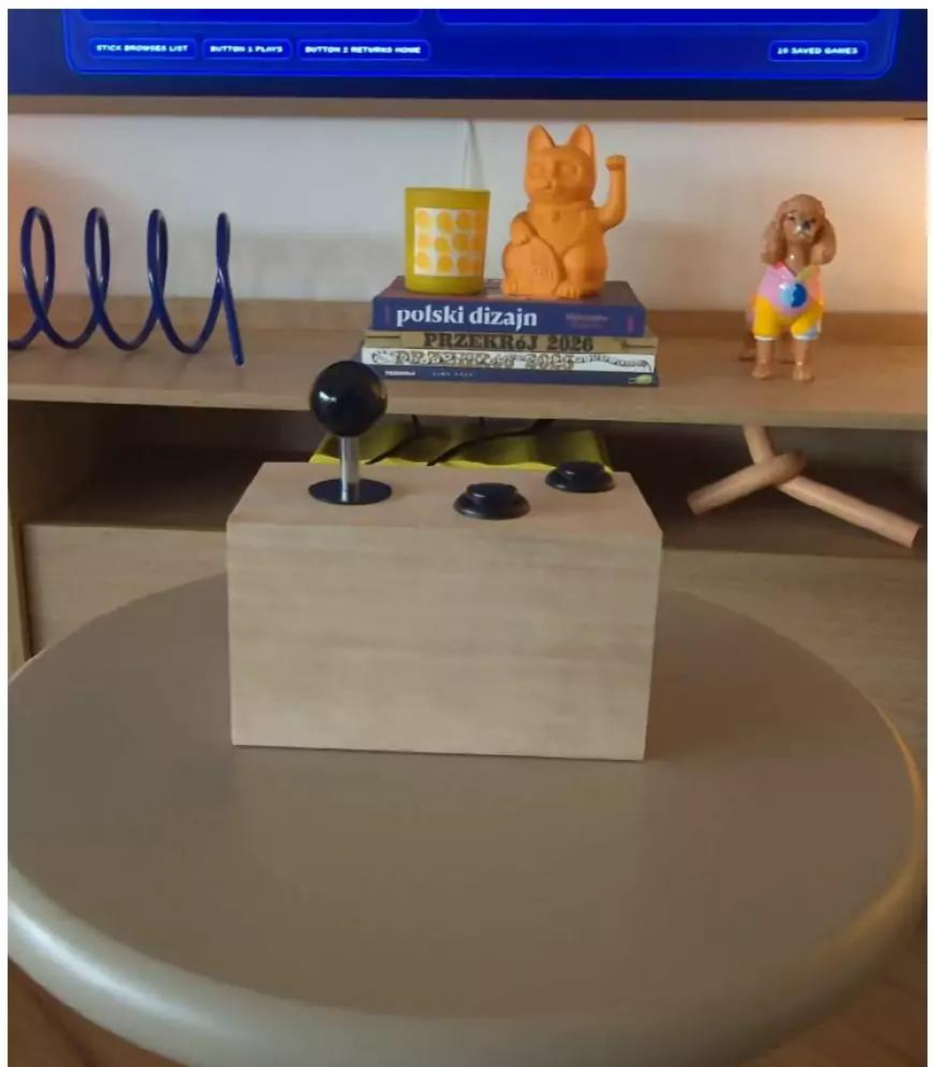


The end result is impressive. The system is able to produce simple, addictive games that evoke the nostalgia of a bygone era. “The gameplay can be genuinely interesting and the results are often more repeatable than I expected,” Grigor says. But the experiment still shows that AI has some way to go to be able to match the genius of past developers. “Sometimes the games are broken, but they are still fun to experiment with,” he adds. “The weakest part is usually the art. The assets are not especially strong, so I think the project could be improved by adding another AI model that specialises in generating higher-quality visuals.”

Still, it’s a fun project to try and, true to its retro roots, it doesn’t have to be a complicated build. The case itself is just a wooden packaging box with a joystick and two buttons and that’s just the way Grigor wanted it. “I like reusing old packaging and I thought the wooden box worked well for this kind of project,” he explains. “For simple builds, I do not think 3D printing is always necessary. Sometimes an old box and a hole-saw drill bit are enough. That said, I would love to remake it one day in a nicer oak box.”

He’d also like to take the AI Arcade to another level. “I would like to lean more into the multiplayer side of arcades and possibly add extra buttons,” he says. “It would also be interesting to experiment with generating 3D games. At the moment, many of the LLM-generated games are quite similar. One idea I am interested in is pre-generating 20 to 30 game templates and then using local generation to tweak values, change mechanics and swap art assets. That could make the box work offline while still producing varied games.”

- Games can take a little while to generate, but a mini-game entertains the player in the meantime
- The actual build is very straightforward. Grigor used an 8GB Raspberry Pi 4 computer
- Grigor is looking to tweak the project and use better models to generate more advanced games



- Grigor says the games are simple and the gameplay doesn't always make sense



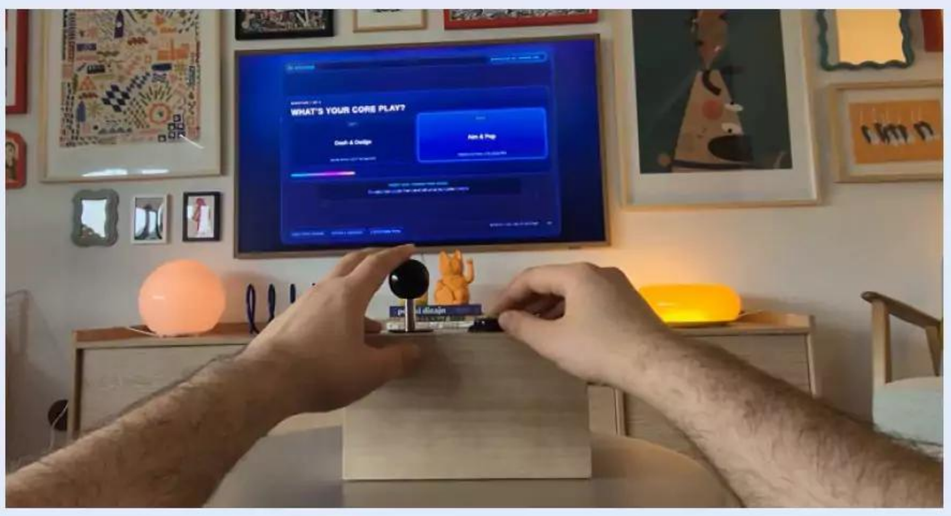
Quick FACTS

- The system generates retro-style games using AI
- It boots into a local Node.js web app
- The build is quick and inexpensive
- Grigor vibe-coded the app in an afternoon
- It can also be played online at **grigtod.github.io/AIArcade**

Box of games



1. **Powering on** – AI Arcade can be run on your own server with an OpenAI API key. When the Raspberry Pi is powered on, it will boot up and start hosting the local AI Arcade web page. It will open the Chromium web browser and load the page.



2. **Game generation** – The page is switched into full-screen mode and, from there, the user can start generating or playing games. The Raspberry Pi calls the OpenAI API endpoint and the actual game generation happens via the LLM.



3. **Getting stuck in** – When generating a game, users are asked four random questions which form the basis of the gameplay. This can then be enjoyed using the console's joystick and arcade buttons. The games have a definite retro vibe.

Stewart Warner radio retrofit

Allow a broken vintage radio to regain its voice as an internet streamer.

By **David Crookes**



Maker

Travis Foss

Travis is the senior technical marketing specialist at DigiKey and his role is to create content for the maker.io website.

rpimag.co/retroradio

Claude and ChatGPT were able to help me put together the Flask page

Housed in a large, walnut veneered cabinet, the Stewart Warner 205CK radio was not only designed to be pleasing to the ear but to catch the eye as well. Dating from the early 1940s, it featured a large internal speaker at the bottom and, above it, a tuner displaying frequencies for both the AM band and shortwave. Despite its presence, it was simple to operate. Users could turn it on, adjust the volume, tune between stations, switch bands, and shape the sound using just four knobs. When Travis Foss came across it, he immediately snapped it up.

"I had recently been at a local auction and saw the radio selling relatively cheaply," he says. "I really liked the look of it, so I wanted to preserve its looks and make it work again." Thus began a new project which involved restoring the cabinet and retrofitting it with a Raspberry Pi 4 computer to turn it into an internet radio.

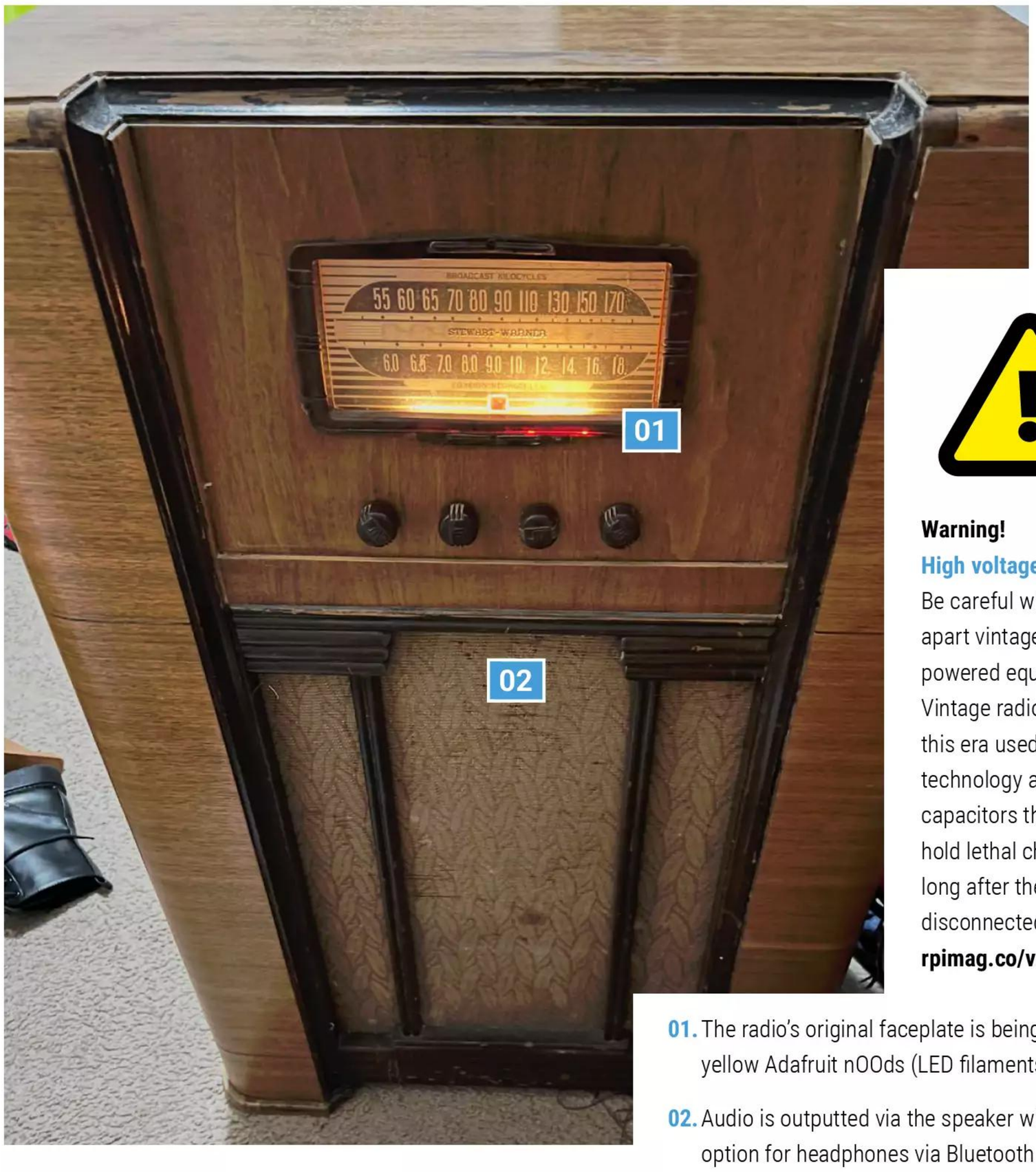
This meant disconnecting (but not removing) most of the internal electronics. He wanted to leave them inside the build in case he decided to sell the radio to someone who wanted to restore it. Given there were already a few components missing including the power cable, this was quite straightforward. But, during this process, Travis noticed the original speaker was still in good condition. "Right away, I tested it to make sure it still functioned correctly and, man, did it sound great," he says. "I then decided to use the speaker and work around it."

Radio star

Rather than create a simple vintage radio that functioned as a Bluetooth speaker, Travis wanted to be able to remotely control the project. "That's when I started



▲ Audio plays through the original radio speaker using connections called spring clips



01. The radio's original faceplate is being lit using yellow Adafruit nOOds (LED filaments)

02. Audio is outputted via the speaker with an option for headphones via Bluetooth

researching options for creating a local controlled website, looking at how to make that work with Raspberry Pi," he says. Deciding to use a DFRobot HIFI digital Bluetooth amplifier connected between the speaker and Raspberry Pi, he turned to AI to help speed up the programming of the internet radio. The aim was to set up a Flask server and build a local website that could control the radio via a browser on any internet-connected device.

"Claude and ChatGPT were able to help me put together the Flask page and add a sound effect just prior to the internet radio channel starting, so it appeared the radio was tuning into a channel," he says. Having gone through 15 iterations of a Flask program, he finally ended up with something that operated well. He also felt that using AI taught him new concepts and a better understanding of when to use certain code.

Tune in

Operation of the radio is simple. By opening a web browser and connecting via a URL, a control page lets Travis choose an internet radio channel and control the volume. There's also a built-in alternative. "If I would rather listen to audio over Bluetooth, I can log into the app for the Bluetooth amplifier and change the input from the audio jack to Bluetooth. I'm then able to connect to it with the phone or device I want to play audio from," he says.

Travis has earmarked future improvements. "I set up the server so that I can add additional channels to it," he explains. But while he is considering adding encoders in place of the original knobs, that may be a step too far. "It would require removing the original equipment, so I'm unsure if I actually want to do that," he says.



Warning!

High voltage!

Be careful when taking apart vintage mains-powered equipment. Vintage radios from this era used valve technology and contain capacitors that can hold lethal charges long after the power is disconnected.

rpimag.co/valverepair

- Raspberry Pi 4 is fitted inside the cabinet along with an amplifier



Quick FACTS

- Travis has created a VLC-based internet radio system
- It uses a vintage radio but retains its old parts
- It's now controlled via a Flask-powered web control interface
- Pygame is used for sound effects and music/static cross-fades
- Travis says the sound is warm, pleasant and enjoyable

Point Nemo's CanSat

A team of students create a Raspberry Pi spacecraft and rover the size of a can of soda. **Rob Zwetsloot** explores this final frontier



Maker

Point Nemo

Three A-level students at The King's (The Cathedral) School who have taken part in various engineering competitions.

rpimag.co/cansat

CanSat is a European Space Agency (ESA) competition to create a working satellite the size of a soda can, and include secondary school student teams in regional and national launches. Point Nemo was one such team taking part in recent months, using a Raspberry Pi Zero 2 W.

"We developed [our CanSat] to take various readings during a parachute-guided descent; this included atmospheric temperature and pressure, RGB and infrared camera data, and volatile gas sensors," Benjamin, a member of Point Nemo tells us. "During the second stage of deployment, our satellite transforms into a rover in order to travel to any points of interest identified during the descent. These points are chosen by analysing the data collected and estimating the biological value of each location. Overall, our satellite aims to survey land and find the places that are most promising for agriculture."

This secondary rover stage was inspired by the current searches for organic matter in our solar system.

Iterative CAD

For a project with such tight tolerances, a lot of planning in CAD and 3D modelling software occurred, as you might expect.

"We had learned from similar projects in the past about the difficulty of fitting everything into the dimensional restrictions given," Benjamin says. "To avoid this problem, we imported as much as possible into CAD, including premade models of specific components. As with any engineering project, we made lots of CAD iterations with every issue we encountered or new component we considered."

Due to limited budget and lead times in getting components via their school administration, apparently the satellite was 'hypothetical' until quite close to launch.

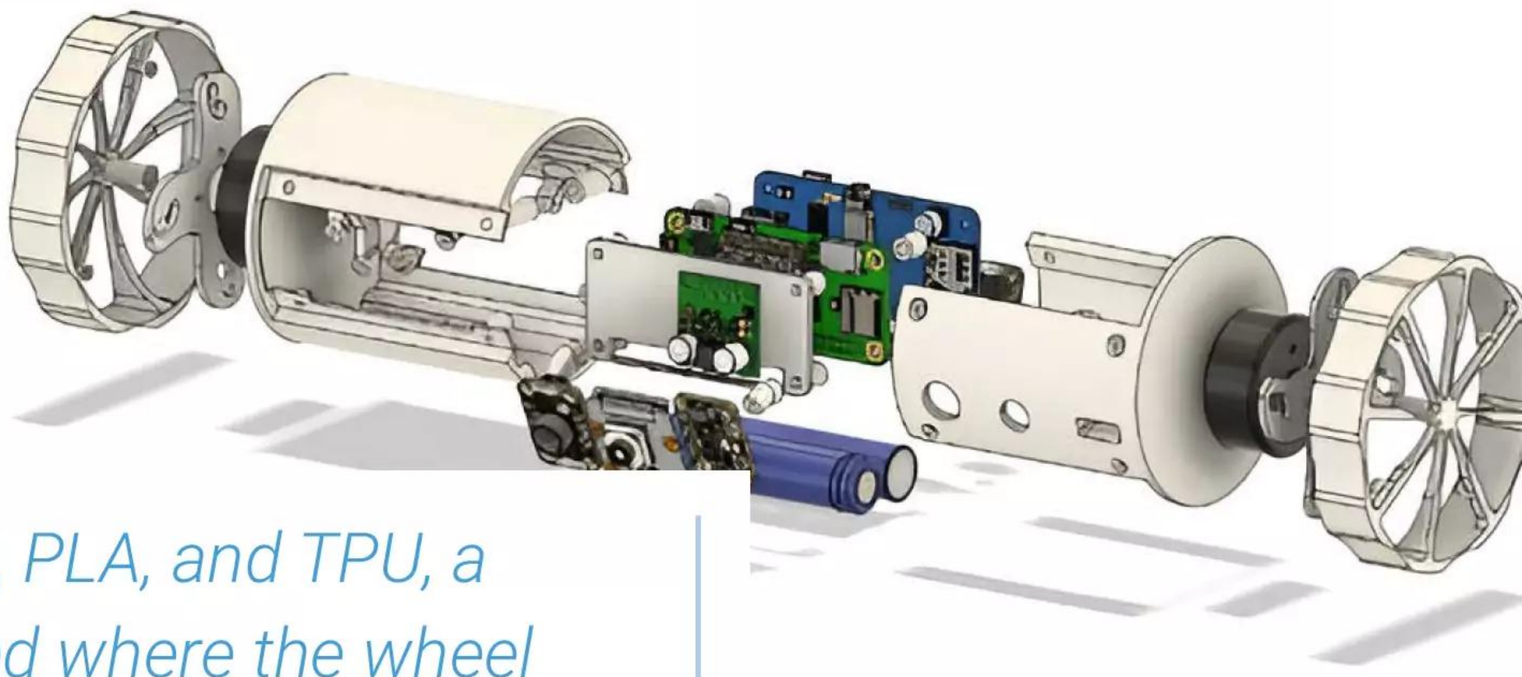
"Lots of research, datasheets, and calculations were completed to ensure everything would work," Benjamin explains. "We had an emphasis on readily available components to avoid integration issues, although we did meet some such problems close to launch."



01. A Raspberry Pi Zero 2 W drives all the sensors and motors from the inside. Its small size and weight are essential for this project

02. Wheels are attached on the side which expand upon landing so the satellite can become a rover

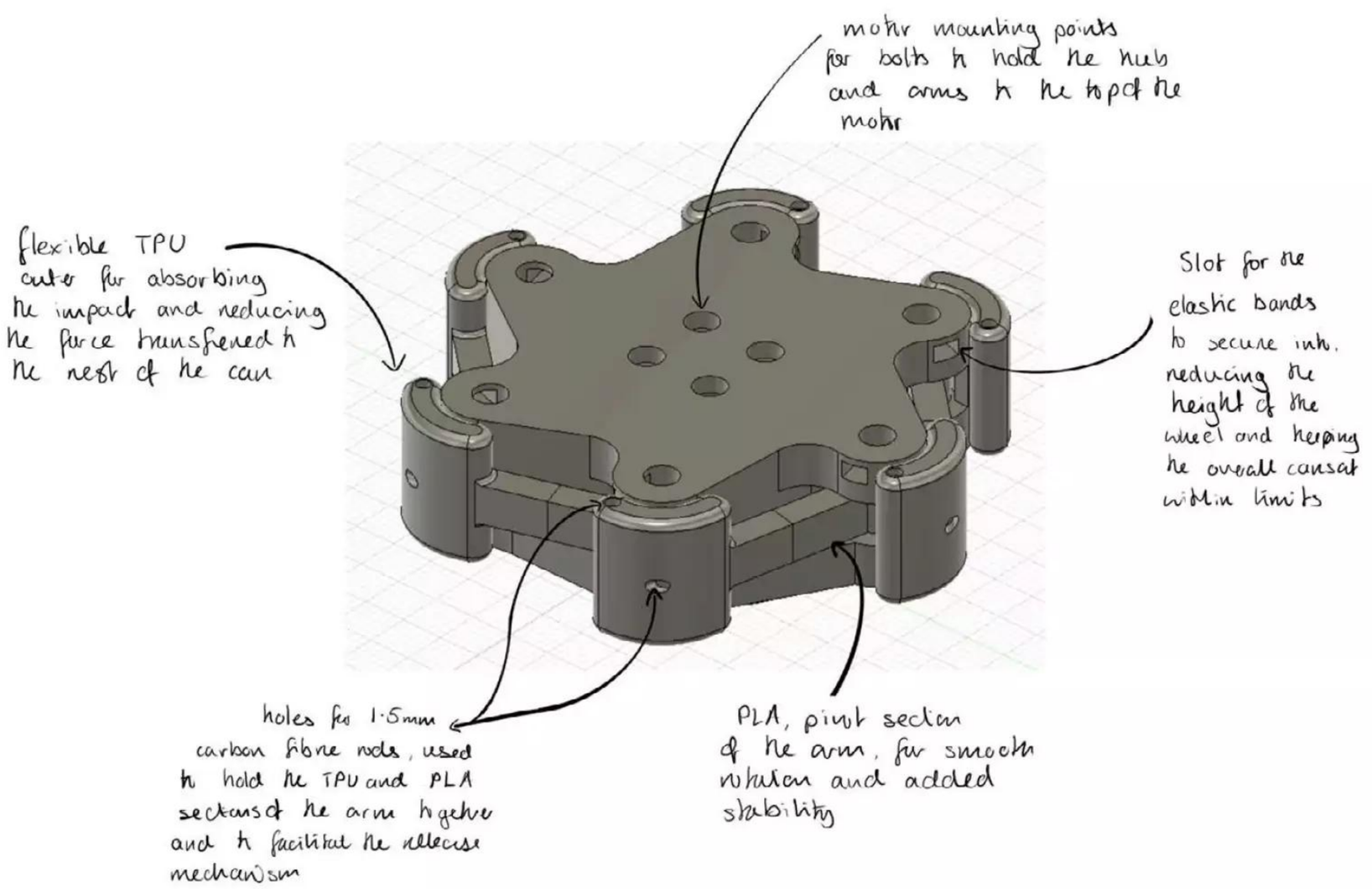
Using carbon rod, PLA, and TPU, a system was devised where the wheel was compressed using elastic bands which would then expand upon landing



An exploded view of the CanSat – everything is very tightly packed in to make sure it fits the dimension specs

A lot of 3D printing was used throughout the project, with the team experimenting with different materials, including carbon-fibre-infused filaments – although actual carbon-fibre rods ended up being used for structural reasons.

Creating the wheels went through various iterations as trying to figure out how to deploy them once landed ran into various issues involving material strength and making sure it kept to the size required for the competition. Using carbon rod, PLA, and TPU, a system was devised where the wheel was compressed using elastic bands which would then expand upon landing.



▲ The expanding wheels are one of the very cool things about this tiny satellite

Rovin' around

Even with extremely careful planning (and testing by dropping the system from a drone), the actual launch was a different thing.

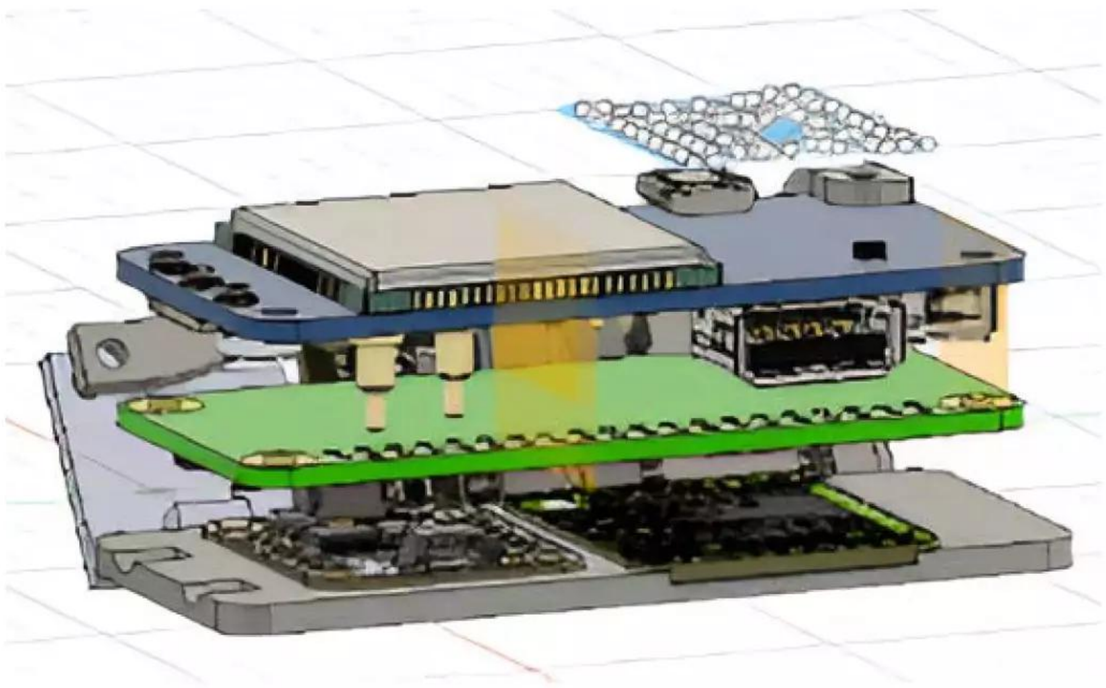
"Our satellite was successfully launched in a small rocket during the regional launch event in March, collecting large amounts of data and safely landing after parachuting 400m down," Benjamin says. "One of our motors had a minor fault, causing difficulties during the rover stage; however, the wheels deployed as planned and worked when connected to a fully functioning motor."

Due to issues on the field involving a faulty 4G connection, a camera that was supposed to be used in the rover stage had to be activated before launch. However, due to the size constraints of the project, the battery for the camera was quite small and ran out during the descent. Still, for an ambitious project, it seemed to largely go well.

"Our team learned the importance of testing all the systems involved in a project together, to be prepared for as many eventualities as possible," Benjamin tells us. "We gained valuable experience of the engineering process – continual prototyping and iteration, improvements and problem solving – both through our successful testing and the things that didn't go to plan."

We look forward to the future of satellites that are more than meets the eye.

- A 4G module is added on top of Raspberry Pi Zero 2 W for remote access



Quick FACTS

- Two cameras are included in the build
- One of the cameras is a thermal camera
- Other sensors include gas, spectral, and orientation
- The ‘can’ is 66mm diameter and 115mm in length
- The final project had to weigh between 300 and 350g

The mission



1. Assembled inside a rocket is this specific CanSat, the soda can-sized satellite created by Point Nemo which had to fit very specific size and weight requirements.



2. On descent, the satellite takes sensor readings of the ground below while also trying to ascertain its altitude as it drops. **[Credit: ESA, CC-BY-SA 3.0 IGO]**



3. Once landed, its wheels expand so it can drive towards a target determined to be agriculturally relevant.

Sequino

By **Ekkagrat Singh Kalsi**

rpimag.co/Sequino

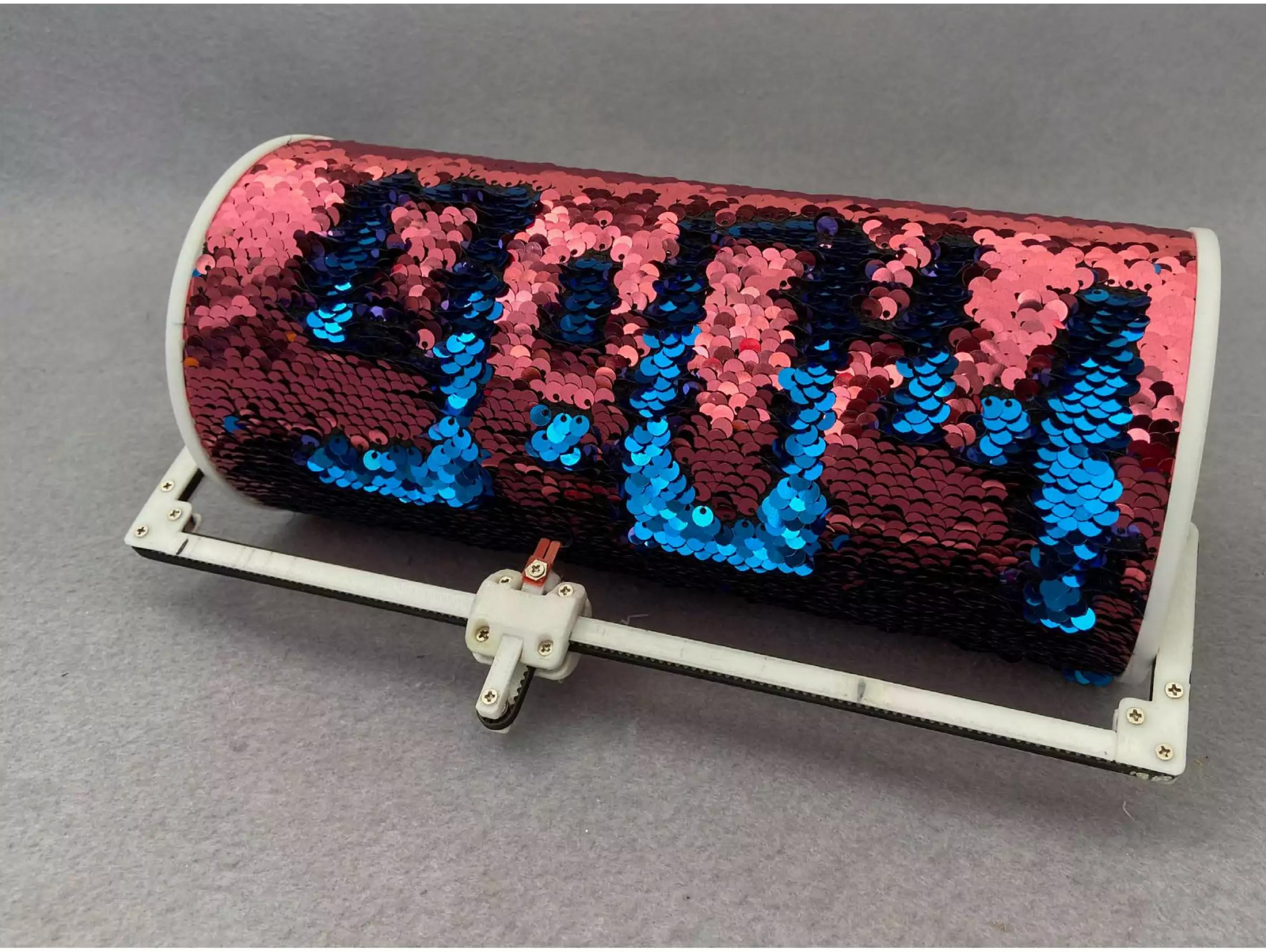
This absurd, beautiful clock has been in development for at least six years, while its maker, Ekkagrat Singh Kalsi, irons out the kinks and refines the design. It uses a piece of material with sequins sewn on to it as the medium; a robotic finger traces the outline of the numerals on to the sequins, displaying the time in super-sparkly fashion.

It was inspired by a T-shirt worn by his daughter – sequins are tactile as well as shiny, and by stroking them one way or the other the wearer can flip them over to reveal the different-coloured underside.

As you might expect, this wasn't a simple build: the maker had difficulties with the relatively low-resolution display (the sequins are around 5mm in diameter, stitched 3mm apart, which restricts the minimum size of display you can achieve using them as pixels). There's also the extremely low refresh rate to contend with...

It takes 90 seconds to draw a digit, and refreshes every 5 minutes. That makes it useless for anything other than very vague time-keeping, but it's so pretty that we don't mind that.

Finally, from the name you may have guessed that this is an Arduino project, using as it does an Arduino Nano, Nano CNC shield, three Stepstick A4982 stepper motor drivers, one Hall effect sensor, and two IR reflective sensors. We challenge the reader to re-implement this using a Raspberry Pi Pico!



- The key ingredient is the 3mm thick foam backing on the fabric, giving enough flex for the sequins to move

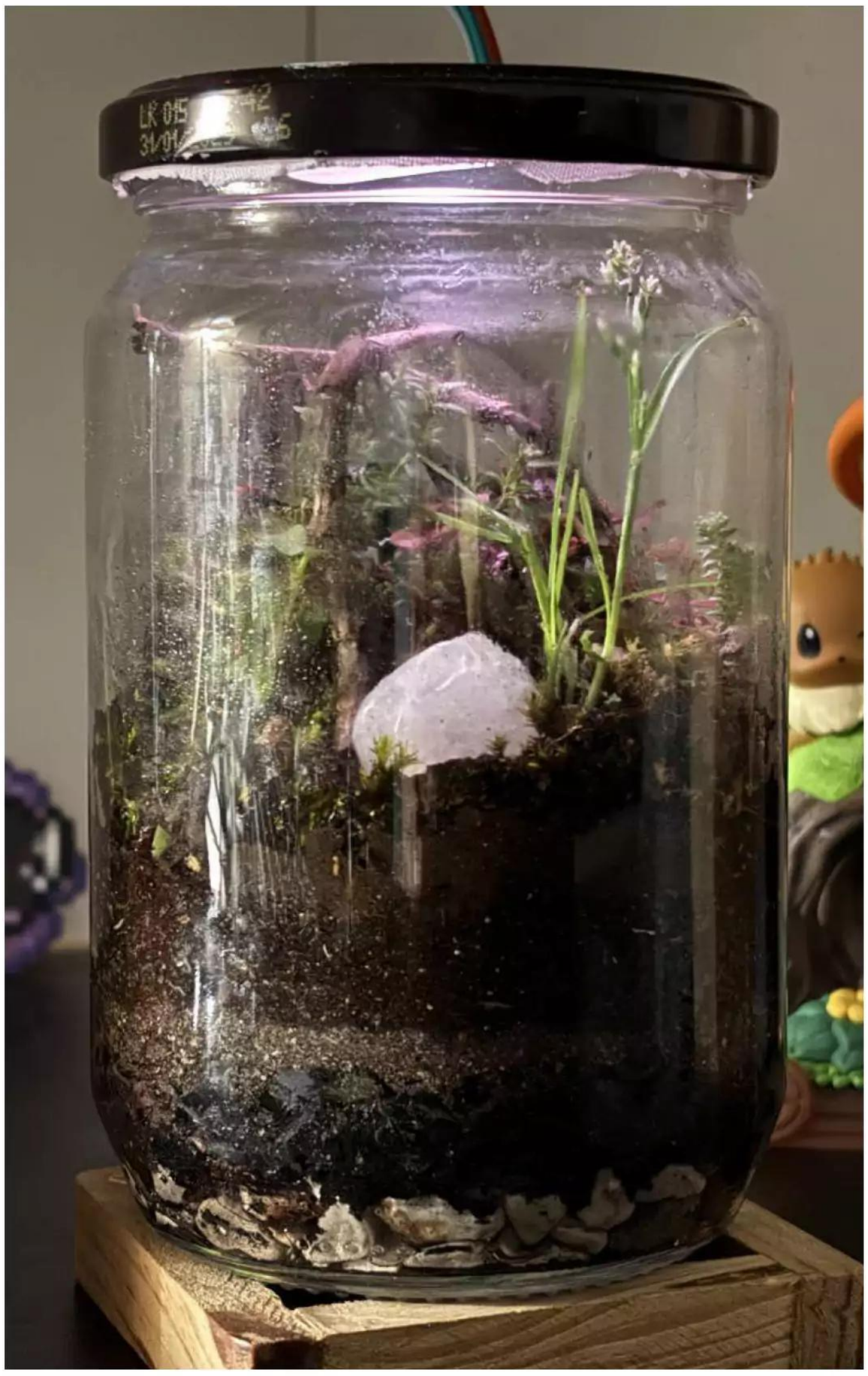
Raspberrarium

By **Mommotti**

rpimag.co/Raspberrarium

A terrarium is typically a glass container, sealed to air and moisture, that provides a static environment for plant specimens. They were first used by botanist Nathaniel Bagshaw Ward, who accidentally left a cabinet open and found that a fern spore has found its way in and had germinated; after that stroke of good fortune they were used to transport living species on long sea voyages, including between Australia and Britain.

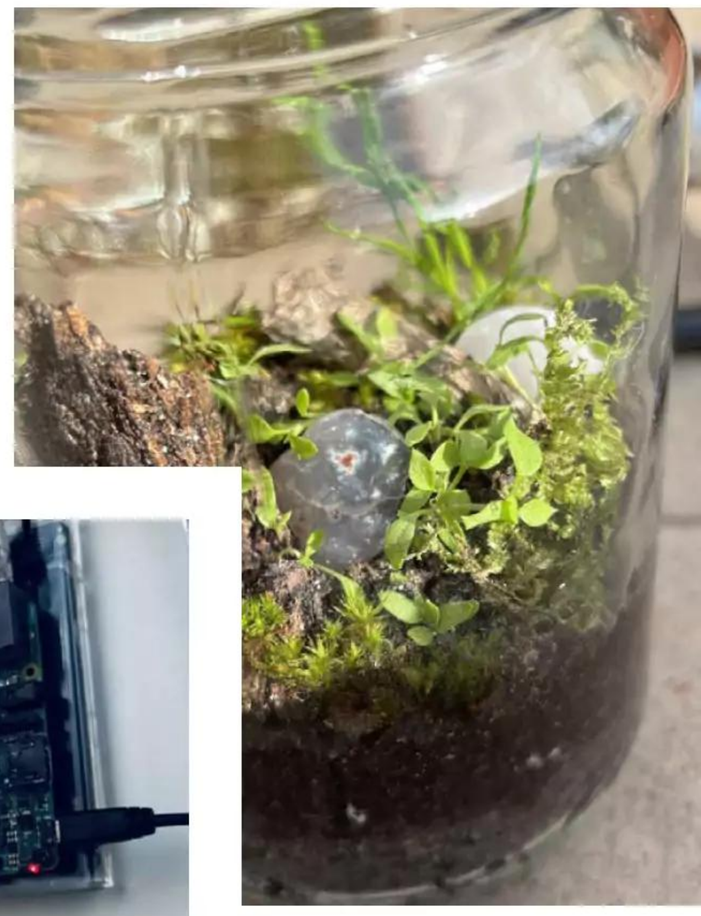
Mommotti's Raspberrarium takes this simple concept one step further. By adding a Raspberry Pi and just two WS2812 LEDs, the Raspberrarium can accurately simulate, according to the maker: "a real daylight cycle (sunrise, noon, sunset), real moon phases during the night, [and] a smooth 24-hour light progression with minute-accurate transitions." What that means is that even if your terrarium is in a part of your house with little natural light, it will still get the light that it would if it were outside. Even cleverer, the pair of LEDs mean that Raspberrarium can simulate the waxing and waning of the moon.



- Control the seasons and phases of the moon with your own tiny ecosystem

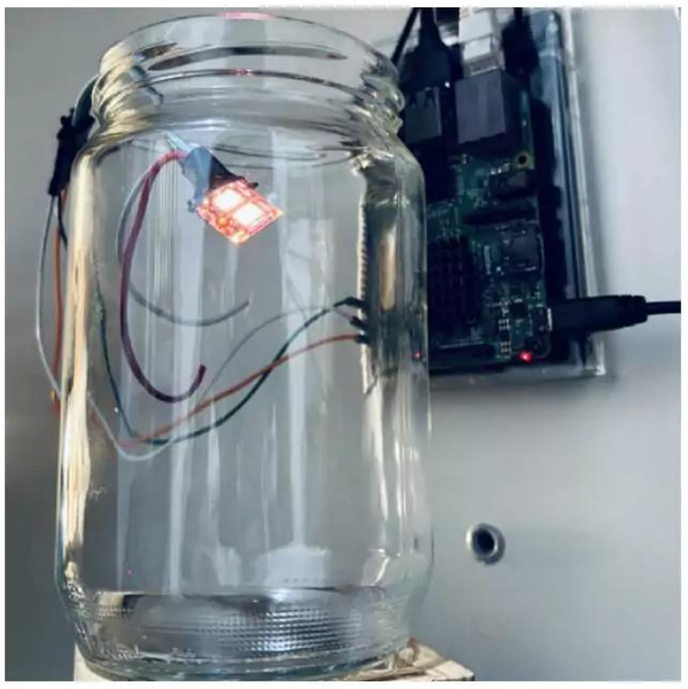






















Poor man’s Polaroid camera

By **Stasys Urbonas**

rpimag.co/PoorMansPolaroid

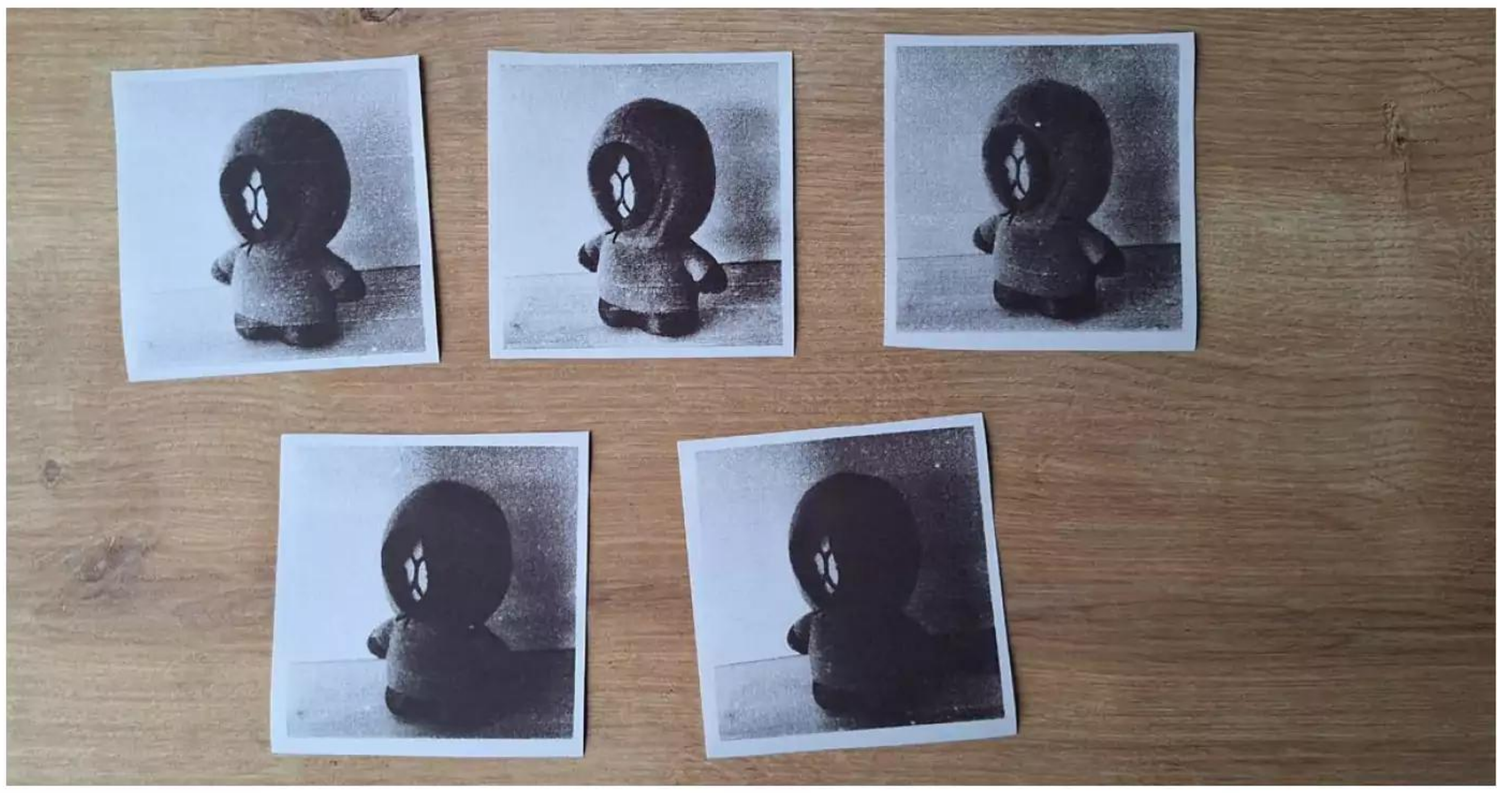


The Polaroid camera was a brilliant invention. You pointed it at the subject, pressed click, and seconds later a wet rectangle of photographic paper emerged, blank at first, but eventually the image formed before your eyes, like magic. No negatives, no automatic upload to Facebook so that your employer can see how much fun you’re having while you’re having it, just a moment captured. Polaroid cameras are having a minor comeback (every photo we take on ours makes us look like we’re living in 1982) but even more so, they are inspiring makers to create their own versions.

Take this build by Stasys Urbonas, for example. He calls it a poor man’s Polaroid, even though it cost more to build than it would to buy a Polaroid camera off the shelf: “I almost bought a regular Polaroid, but realised photos aren’t cheap, so the idea to use a thermal printer just popped into my head at that time, because thermal paper is cheap.”

We found this project build while we were looking for something interesting to copy on our 3D printer. We quickly skimmed over it, as it looks like it’s made of MDF, or possibly sheet metal with a thick coat of paint. Turns out it actually is 3D printed – it’s just had a load of work done to it to make it look like something polished that’s come out of a factory, rather than peeled off a 3D printer bed.

The components include a Raspberry Pi Zero 2 W, a Raspberry Pi Camera Module, and a battery. Stasys designed the case himself, then printed, sanded, primed, and painted it in a colour scheme that reminds us of roadside speed cameras.







▲ There's something really big and chunky about this build that we find reassuring

Mermaid in the shell

By **Claudia**

rpimag.co/MermaidCyberdeck

We're used to the post-apocalyptic designs of most cyberdecks: rugged aluminium or pelican cases, chunky battery packs, cool green-on-black scrolling text, *Matrix*-style. What we have here is something different. Instead of running around the burned out remains of a power station or hiding from killer robots, the owner of this cyberdeck is more likely to be found under the sea, hanging out with a friendly singing crab called Sebastian.

Bimbotech's Claudia has always wanted to put a computer in a Polly Pocket toy. With this compact build, she's pretty much done just that. The pink clamshell clutch purse that she's chosen to house the cyberdeck doesn't have a great deal of room in it, so she's used a Raspberry Pi 3 Model A+ – this was the last of the square Raspberry Pi computers, and takes up even less space than the rectangular boards. A 3.5-inch touchscreen, ZitaoTech BB Q10 Bluetooth keyboard and touchpad, and a 3000mAh battery pack complete the electronics.

With just 512MB of RAM, the Raspberry Pi model Claudia chose doesn't have enough oomph for a graphical operating system, but it's fine for the command line – she says on her blog that she's planning to upgrade the battery to 10000mAh, which should be enough to power a 12-hour shift of ocean-floor hacking.



- Satin and real freshwater pearls complete the look for the mermaid who needs root access in style

3D print

Mathematics, with the added therapeutic benefit of a clicky display

rpimag.co/Flapulator

We are big fans of the Unix philosophy here at Raspberry Pi Towers: devices should do one thing and one thing well. It's great that you can have a calculator app on your phone or a laptop, but a dedicated physical calculator somehow imbues the numbers with more gravitas. That's certainly the case for this 3D printed calculator built by computer scientist Chris Haynes. The Flapulator, as he calls it, is a calculator sporting a 3D printed mechanical flap display, powered by a Raspberry Pi Pico, and featuring the same Gateron KS-33 low-profile mechanical switches as seen on Raspberry Pi 500+.

To be frank, that's enough for us: any device built in the modern age that uses an obsolete display technology gets our vote, but there's more to this than meets the eye… or pi. As well a dedicated pi button, all the trigonometric functions only work in radians, and all of the angles used when designing the shell, lid, and body are either some form of pi or a derivative of pi.

Chris told us: "… I have found myself using the Flapulator more often than I expected. For quick and simple calculations, it does add some extra enjoyment that you just can't get with a digital equivalent. I think we can all appreciate the joy of using a highly tactile device, and the Flapulator does a great job of stimulating all of the senses related to that. There is the satisfying feel of clicking the keyboard, the auditory pleasure in hearing the digits turn, and the visual enjoyment of watching the display update."




FLAPULATOR
3.1416
+
−
7
8
9
4
5
6
1
2
3
0
.
π
×
÷
sin
cos
tan
√x
x²
+
−
=
DEL
AC

Part 01

Make music with Raspberry Pi

We look at a clutch of the best tools for making and mastering music in your Raspberry Pi-powered home studio



Maker

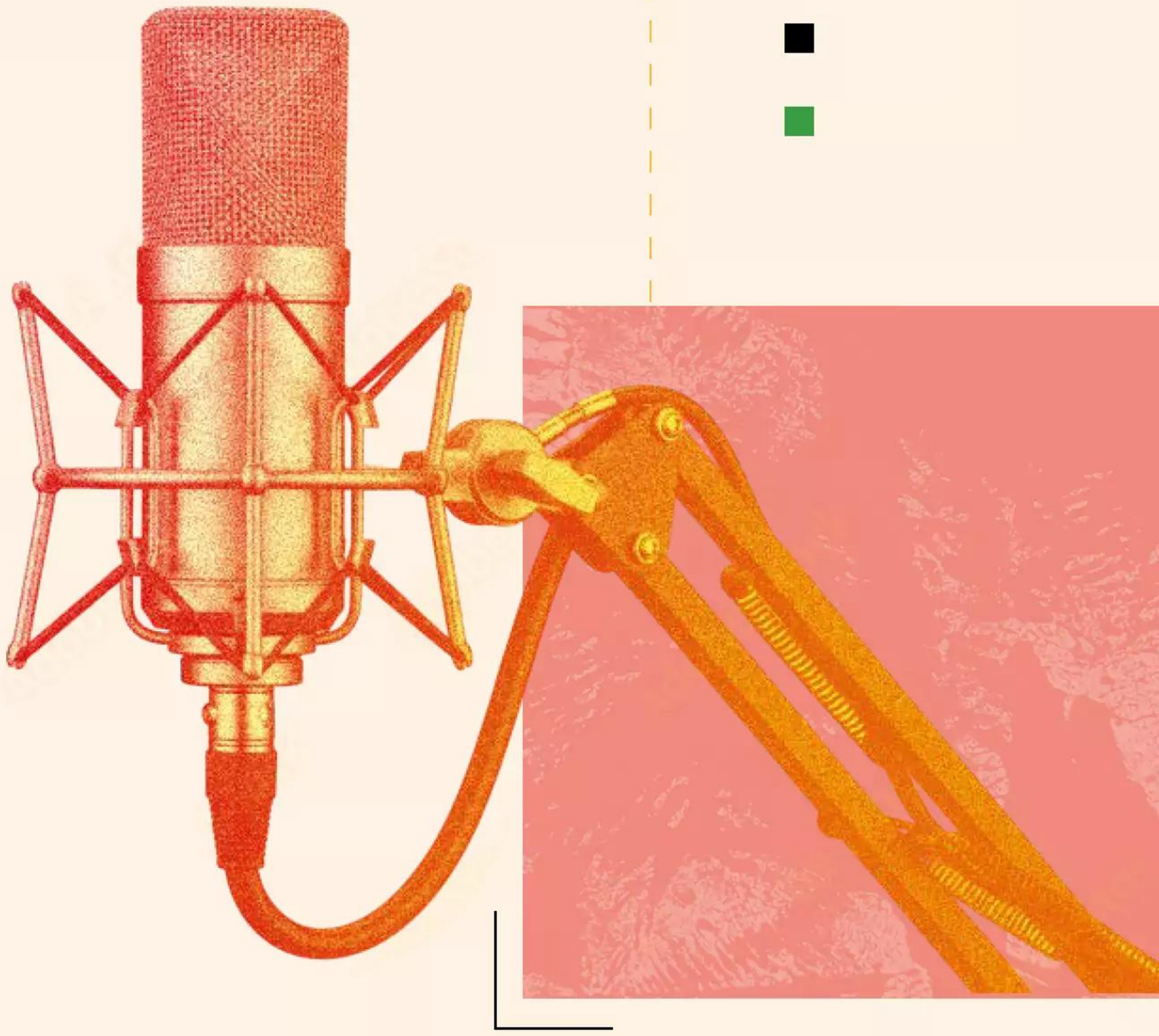
K.G. Orphanides

K.G. hoards unpublished music but recently put out a collection of vintage dungeon synth. Originally composed using Musicator for DOS, it was remastered and rendered using Reaper, Audacity, a Fluidsynth VST, and VLC Media Player.

rpimag.co/kgmusic







Audio production tools for Raspberry Pi, whether you make electronic music, record live in studio, or produce musical score for an orchestra

We've previously looked at building (back in issue 150, rpimag.co/150) and equipping (issue 151, rpimag.co/151) a home recording studio. The final part of that series went through setting up (issue 153, **rpimag.co/153**) an audio interface, including the Focusrite Scarlett 2i2 that we use in our own home recording setup.

We also listed some of the software we recommended using for creating audio. The limits of how many words we can stuff into a single feature hit hard, however, and we were only able to briefly mention digital audio workstation (DAW) software for composing, arranging, and recording both MIDI and analogue audio.

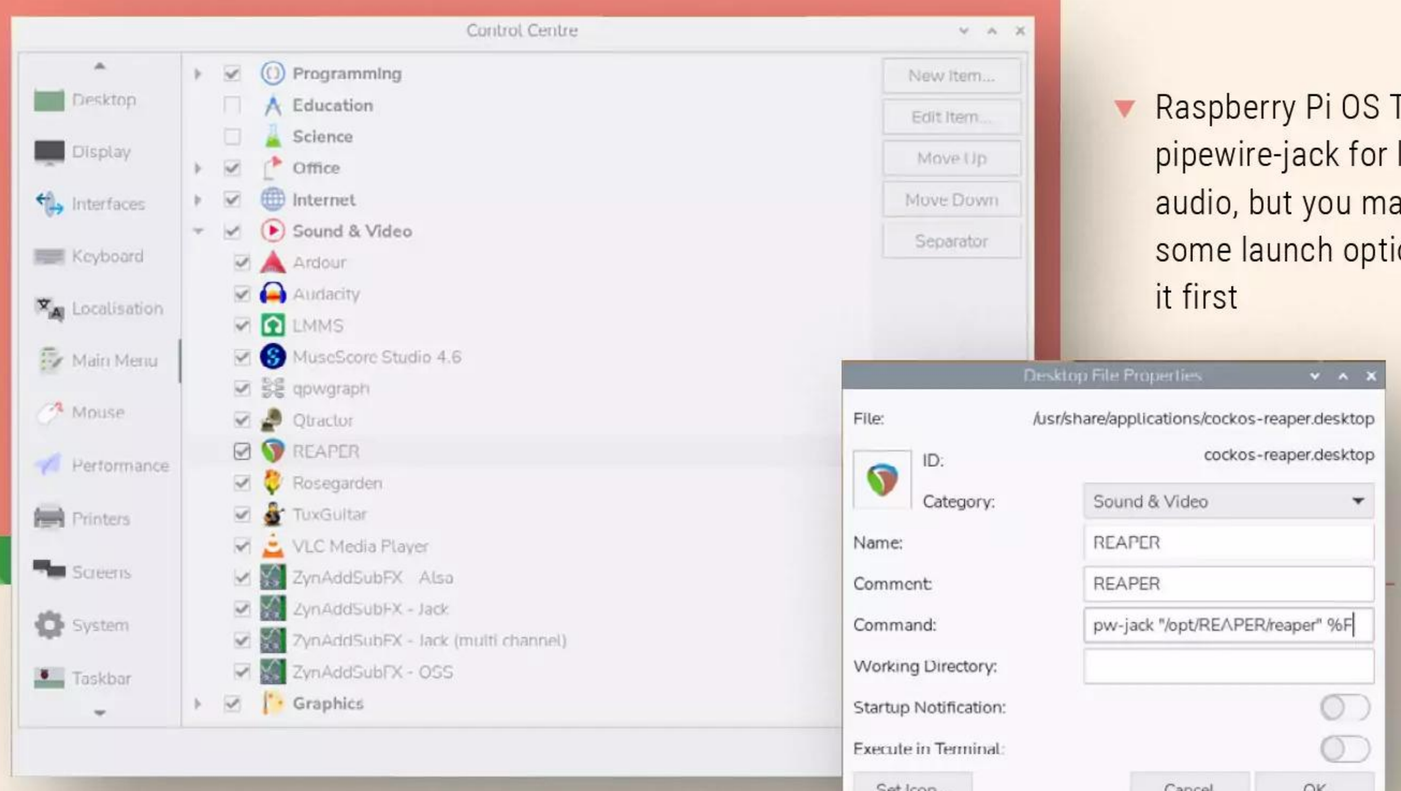
This toolkit feature introduces some of the best audio production tools available for Raspberry Pi, ranging from closed-source commercial studio software to proudly open projects that are free as in both speech and beer, whether you make electronic music, record live in studio, or produce musical scores for an orchestra.

Feedback

We've also got a few extra notes about working with the equipment we recommended, in response to issues and discussions raised in the comments of the online versions of the earlier articles.

Several of you remarked on our use of a desk stand for our condenser mic, observing that desk stands will transmit any vibration on your desk (including mouse clicks) to the mic. This is 100% true and we definitely had to be careful about so much as touching our desk while using a mic stand in physical contact with it. A boom arm or stand for your mic is the best setup here by far.

You also mentioned picking up unwanted fan noise in recordings: this is exactly why power-efficient single-board computers like Raspberry Pi are ideal studio systems; although the 64-bit ARM CPU can limit your choice of software in your production environment, the fact that you don't have to extensively soundproof the whirring fans of a typical x86-64 PC more than makes up for the inconvenience in our book.



▼ Raspberry Pi OS Trixie can use pipewire-jack for low-latency audio, but you may have to edit some launch options to call it first

Your favourite Raspberry Pi

You can use the software we recommend with any audio equipment, but note that Raspberry Pi 5 and Raspberry Pi 400/500 series (Raspberry Pi 400/500/500+) only output digital audio: there's no 3.5mm analogue input/output jack as you'll find on Raspberry Pi 4 and earlier. So you'll be able to hear it via a monitor with built-in speakers (or an AV receiver, as in a home entertainment setup), but if you want to directly connect headphones or any kind of an audio interface, you'll probably need to use USB audio hardware, as we've done here.

You could alternatively connect a compatible audio HAT to Raspberry Pi's GPIO, with a variety of different inputs and outputs to suit your needs. Not all HATs are designed to work with Raspberry Pi 400/500 series, and you might need to use a GPIO Header Expansion to make the pins more accessible (e.g., **rpimag.co/gpioheaderexp**).

To give ourselves as much memory and CPU power as we could get our hands on, we used a Raspberry Pi 500+, hooked up to our Focusrite 2i2 4th Generation, as described in previous features, and tested output-only software with an AudioQuest DragonFly DAC. Raspberry Pi 500+ has clicky Gateron Blue mechanical keys, which are more comfortable to type on but far noisier than the chiclet keys of our studio's original Raspberry Pi 500.

That's something you'll have to take into account for when using (or choosing not to use) keyboard shortcuts when you have a live mic but, from our point of view, the built-in 256GB NVMe SSD and 16GB of memory make it a solid choice. If you're dead set against mechanical keys, a Raspberry Pi 5 16GB RAM with an NVMe drive connected to its M.2 slot will give you the same performance. You can always connect the external USB keyboard of your choice to any Raspberry Pi.

LMMS

lmms.io

If you primarily compose electronic music, then LMMS (originally named Linux MultiMedia Studio) should be your first port of call. It's also notable in that you don't need any kind of external audio interface to make full use of it.

It's easy to use, has clear and good-quality documentation (**rpimag.co/lmmsdocs**) and an official online sharing platform (**rpimag.co/lmmsshare**), which makes it easy to find new samples, instruments, and loops. There's an active user community and the developers are responsive and really listen to their users when it comes to adding new features.

Emerging as an open-source alternative to commercial audio production suite FruityLoops (now FL Studio), it makes it easy to work with samples, SoundFonts, softsynths, and beat loops to produce polished tracks. Its piano roll, beat and bassline editor, MIDI instrument support, and powerful automation features are all very approachable, and we were able to put together the beginnings of a song in just a few minutes. Quality-of-life features such as automatically backing up projects when you save are also reassuring.

Like any other DAW, LMMS's interface has some characteristic quirks. For example, if you want to record the input of a MIDI instrument live over your track, you have to first start recording in the piano roll window – which will immediately start accepting MIDI input in real time – and then hop back to the main Song Editor window and press play. This will immediately jump the track back to the beginning.

It lacks some features that you might expect if you're coming from other DAWs: you can't record PCM (pulse-code modulation) audio within LMMS, nor can you edit it. Although more sample-cutting features are being introduced, you can currently trim the ends but not the beginnings of imported PCM (.wav) samples. If you want to record vocals, or directly sample a note from an instrument, you'll have to use an external program, such as Audacity.

It feels true to the spirit of some of the earliest audio workstation environments, when General MIDI, synthesized instruments, and sampling were at the heart of digital audio production.

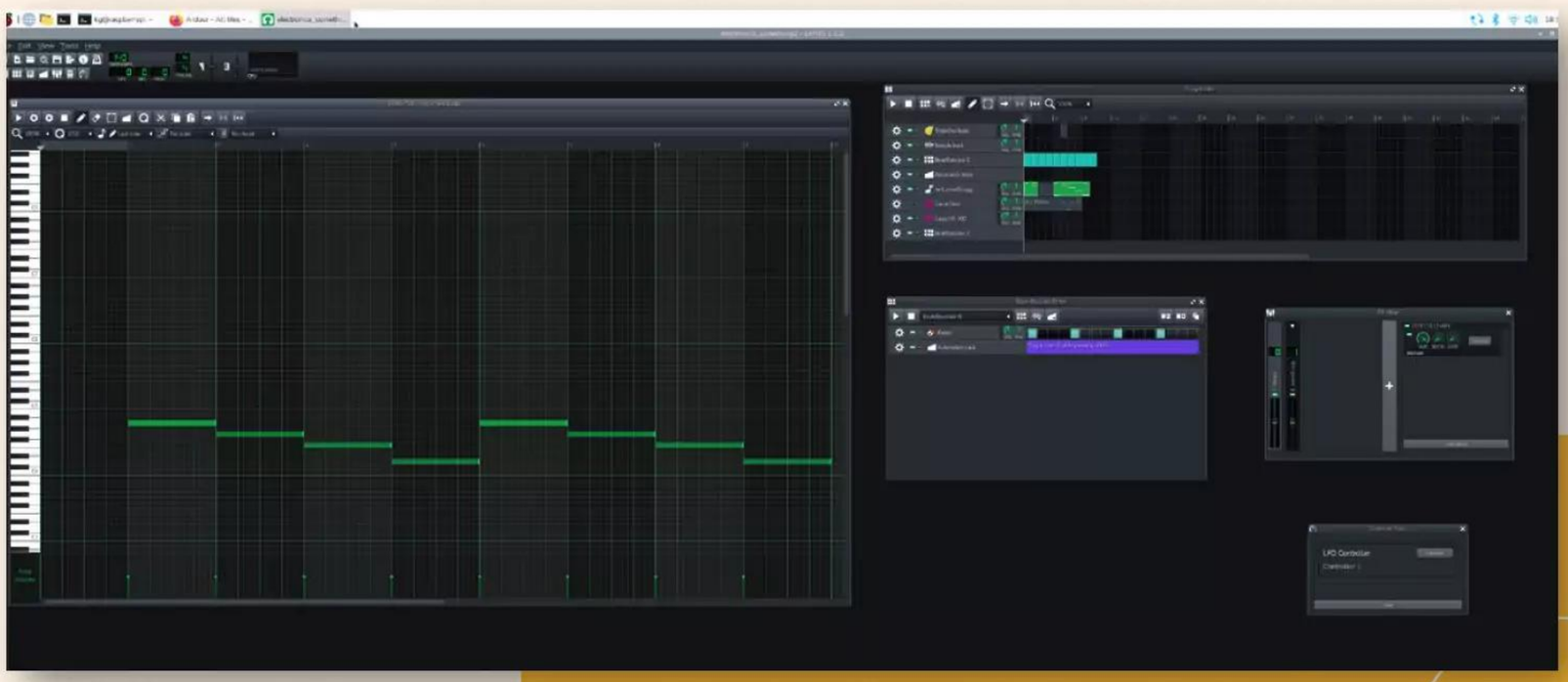
LMMS has an integrated SF2 SoundFont player, which is particularly welcome as SoundFonts also tend to be less memory and resource intensive when compared to VST instruments, although VSTs – which LMMS also supports – are much more adjustable and versatile. The documentation even includes instructions on using Windows VSTs on Linux via Wine, although we prefer to stick to OS- and architecture-native VSTs, as we'd rather avoid emulating x86-64 Windows on Raspberry Pi's aarch64 processor.

Other supported plug-in formats include LADSPA (for effects only), while the impressive open LV2 plug-in standard is currently available for testing in LMMS's development tree.

Install LMMS by opening a terminal and typing:

```
$ sudo apt install lmms
```

LMMS is also available as a Flatpak at **rpimag.co/lmmsfp**.



▲ Built for electronic music, LMMS looks beautiful and requires no extra audio hardware to use its capabilities to the full

Reaper

reaper.fm

Cockos's Reaper (an acronym of Rapid Environment for Audio Production, Engineering, and Recording) is commercial software, and – unlike Ardour – isn't open source, but it's reasonably priced, with a personal licence (also available to small businesses with profits of under $20,000, and non-profit or educational organisations) priced at just $60 (£45). Your licence isn't perpetual, but guarantees free updates until Reaper 8.99 – the current version, as of April 2026, is the very topically named '7.69: Wag the DAW'. Full version releases are slow to come – version 7 was released in 2024. If you're making a better living as a music industry worker, you or your employer will be asked to pay for a full commercial licence, priced at $225 (£167). A commitment-free 60-day trial is long enough to see if it's for you, and registered copies are not subject to any DRM apart from the entry of the licence key.

Reaper receives very regular and significant updates, and has an outstanding range of features that stand up well to professional requirements of recording, mixing, and mastering any combination of MIDI-driven synth tracks. It's a little more resource and memory hungry than some other Linux DAWs, such as LMMS and QTractor.

It's available in a native Linux version – including an aarch64 one for 64-bit Raspberry Pi models and an armv7l one for 32-bit – plus x86 and x64 Linux, Windows, and macOS editions. Even its native JSFX plug-in standard (**rpimag.co/reaperjsfx**) is cross-platform and functions on ARM64 architecture. Reaper supports VST, LV2, and CLAP plug-ins.

There's also an active community, supported by forums on Reaper's own site and community listing projects like Awesome Reaper (**rpimag.co/awesomereaper**), an index of free and open-source software, resources, and information for Reaper.

Download the Linux aarch64 version of Reaper from **reaper.fm/download.php** and open a terminal and extract the file you just downloaded. Your version number will likely differ to that in the example below:

```
$ cd Downloads
$ tar -xf reaper769_linux_aarch64.tar.xz
$ cd reaper_linux_aarch64/
$ ./install-reaper.sh
```

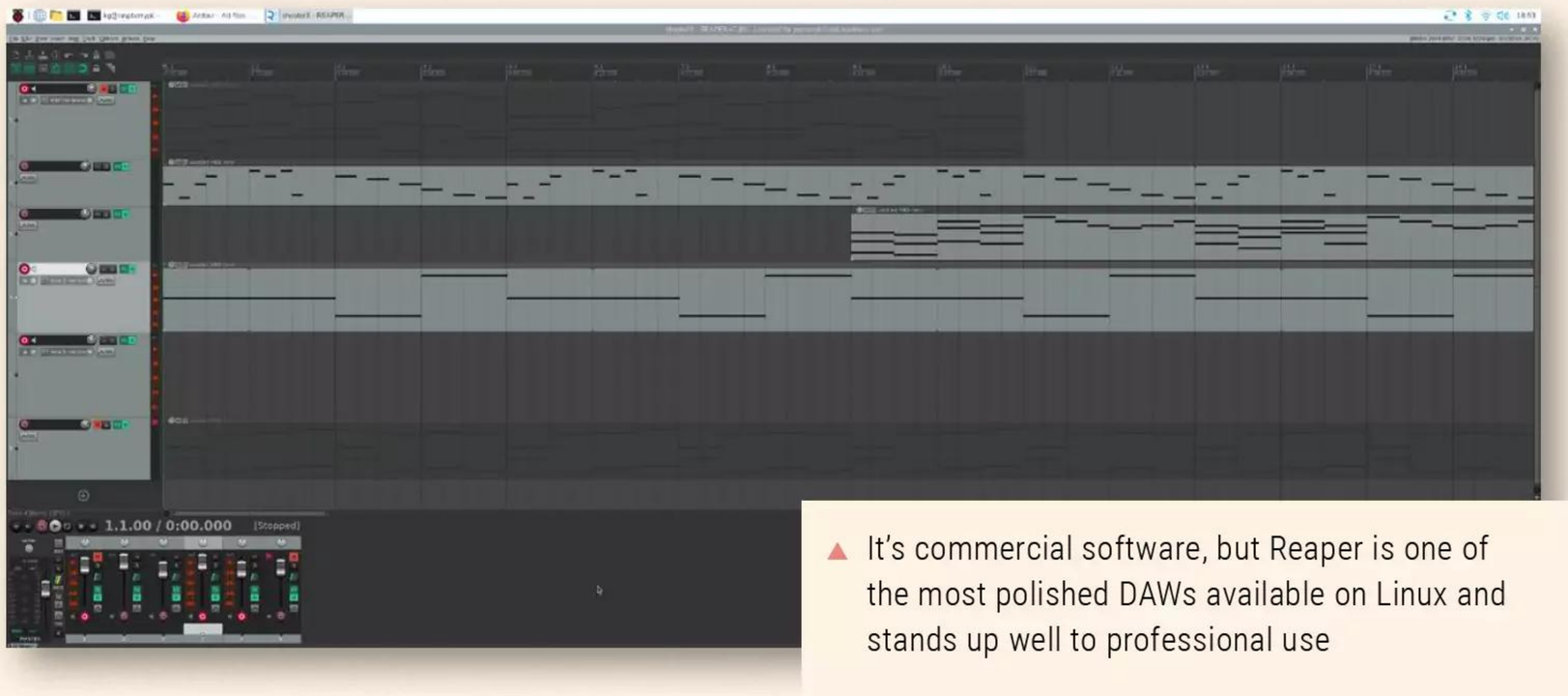
You'll be prompted to enter **R** to run Reaper and just try it out; or, if you decide to keep it, you'll probably want to enter **I** to install it. You'll be prompted to use it as a trial version or enter a registration number when you run it.

You'll also need to install some dependencies to get low-latency audio running.

```
$ sudo apt install pipewire-jack qpwgraph
```

To ensure that Reaper runs using pipewire-jack for low-latency audio input and output, run it using the following command at the terminal:

```
$ pw-jack reaper
```



▲ It's commercial software, but Reaper is one of the most polished DAWs available on Linux and stands up well to professional use

You can also edit the Raspberry Pi menu's launcher entry to precede Reaper's launch command with 'pw-jack'.

The Flatpak version can't see everything on your underlying OS, as Flatpaks deliberately limit the system access of software, but might require less configuration for low-latency routing. You'll find an unofficial installation wrapper at **rpimag.co/reaperfp**.



- Ardour presents a polished digital audio workstation environment with broad VST support and some specialist features

Ardour

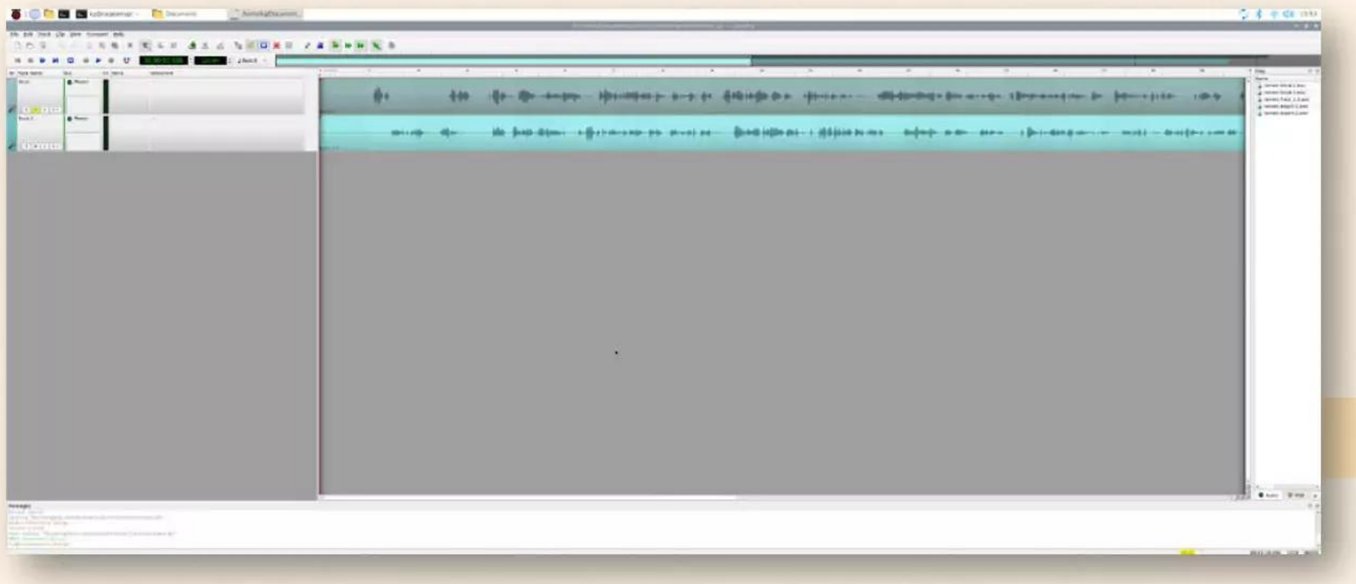
ardour.org

Ardour is commercial open-source software that is made available free for anyone who compiles the source themselves, as well as Linux users who install it via their repositories or the Ardour Developers Flatpak repository. Like Reaper, it has a particularly glossy UI and provides powerful tools for recording live instruments, applying effects, and mixing and mastering audio. Its PCM audio capture and MIDI synth plug-in support are of professional standard, and it even has video timeline and monitoring support geared towards Foley artists and soundtrack composers for film.

To get the latest version of Ardour – 9.2 at the time of writing – you can download a ready-to-run version from Ardour's website, but you'll be prompted to pay for it. Version 8.2 can be found in Raspberry Pi OS's apt repository, but the latest version is available via Flathub. Follow the Flatpak installation instructions in our boxout, then open a terminal and type:

```
$ flatpak install ardour
```

Agree to everything and you'll be ready to go. Ardour can have significant memory requirements, so you may need to make additional resources available to it for a smoother experience.



- QTractor is a lightweight DAW with capabilities that match the big players, from PCM audio recording to synth plug-ins

Installing Flatpaks

At several points in this article, we mention installing programs as Flatpaks, and point you to the Flathub repository. To get Flatpak installed and ready to download files from Flathub, open a terminal and type:

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

It's recommended that you then restart your system. From that point on, you can install software via your browser.

QTractor

qtractor.org

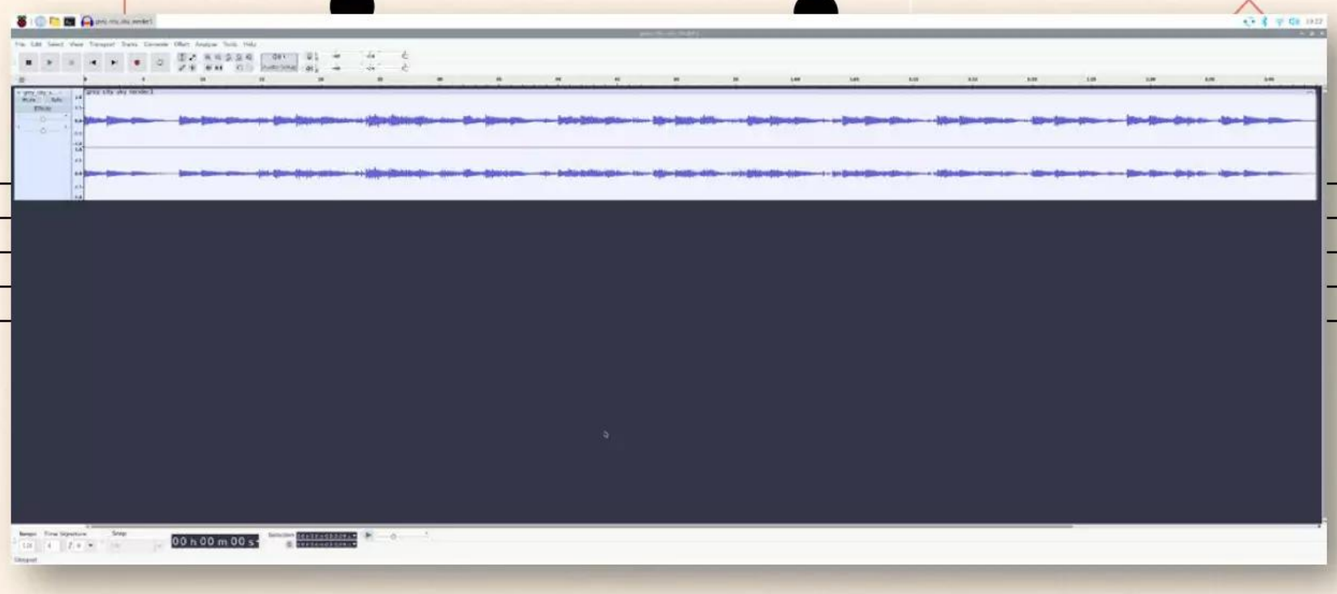
QTractor has always been our go-to DAW for Raspberry Pi when all else fails. Although it supports all the same MIDI, softsynth, and analogue audio recording features as the likes of Reaper and Ardour, it's less resource intensive. Its interface is also pleasingly simple, while still retaining the visual language that's common to most modern DAWs.

Back when we were using a 4GB Raspberry Pi 4 in our recording studio, QTractor was the only DAW that would run with near-perfect stability for both analogue audio recording and capturing MIDI synth input, but even with the power of Raspberry Pi 5 16GB and Raspberry Pi 500+, it can be comforting to have plenty of free resources to allow your music projects to sprawl as far as your imagination desires.

To install version 1.5.4 of QTractor, open a terminal and type:

```
$ sudo apt install qtractor
```

As with most Linux software, there's also a more recent version of QTractor that uses the Flatpak ecosystem, available at **rpimag.co/qtractorfp** – the page even suggests compatible plug-ins that can be shared by your Flatpak apps.



Audacity

audacityteam.org

Audacity is the cornerstone of open-source audio recording, and for those who work with PCM audio, it can be all you need, particularly with recently added support for VST3 effects. Although there was some controversy surrounding user data handling when it was taken over by MuseScore developer Muse Group in 2021, development has gone well under its auspices, with the addition of UI modernisations, effects support, and beat and measure snapping.

Muse Group's open-source-funded-by-commercial-services ethos is reflected in the ability to cloud-save and share audio from within Audacity via Muse's **audio.com** website for music industry amateurs and professionals.

A recent version of Audacity – 3.7.3 – is available in the Raspberry Pi OS repositories and is thus trivially easy to install. If you want the latest version 3.7.7, the simplest approach is to install the Flatpak version, as the official AppImage versions of the software only support x86-64 CPUs.

To install Audacity, open a terminal and type:

```
$ sudo apt install audacity
```

As with Reaper, to have Audacity use low-latency pipewire-jack audio routing, you should run it with the following command:

```
$ pw-jack audacity
```

The very latest version is available as a Flatpak at **rpimag.co/audacityfp**.

▲ Audacity is our go-to software for recording and editing PCM audio, and its interface includes quality-of-life features that some full DAWs lack

Rosegarden

rosegardenmusic.com

Rosegarden is a DAW, but also has a fantastic notation system that works particularly well if you're composing with a MIDI keyboard. It supports synth plug-ins, PCM audio recording, and a range of quality-of-life features in common with other modern DAWs, and uses the GNU LilyPond notation etching system to produce high-quality scores from your MIDI compositions, without extensive manual editing. It is also less resource heavy than some other DAWs, making it a good choice if you have a combination of digital synth-based composition and analogue recording to handle. It's very MIDI focused but also supports score formats, including MusicXML.

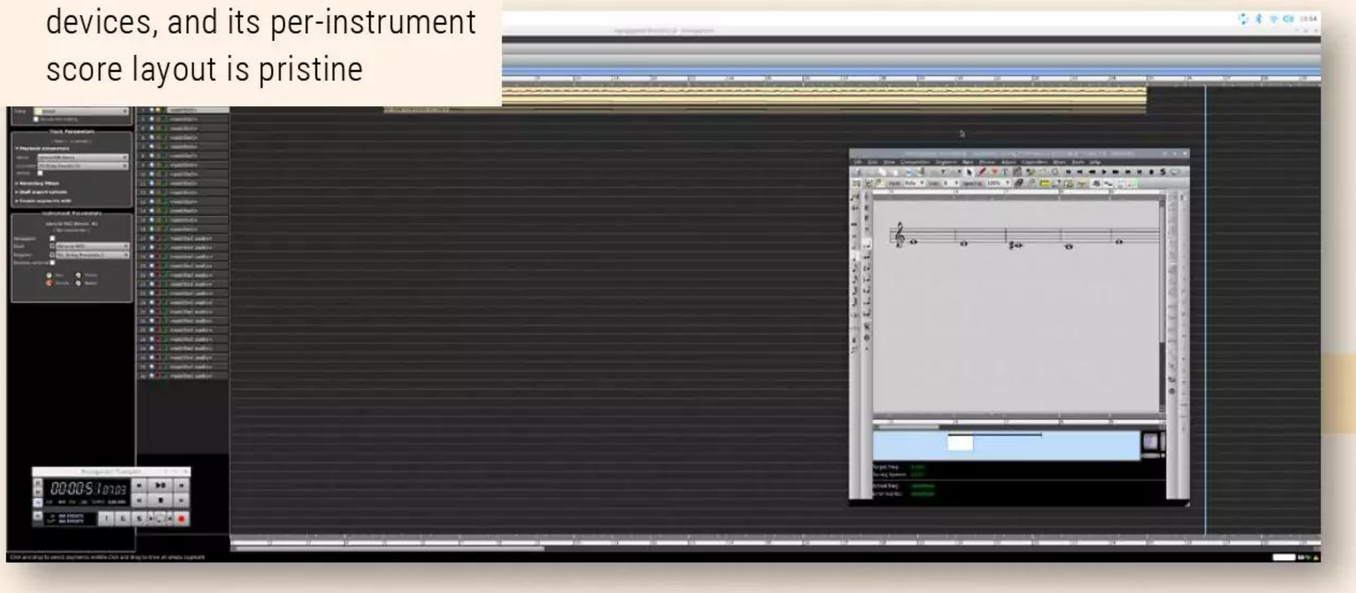
A fairly recent version of Rosegarden (24.12) is available in Raspberry Pi OS's software repositories. To install it, open a terminal and type:

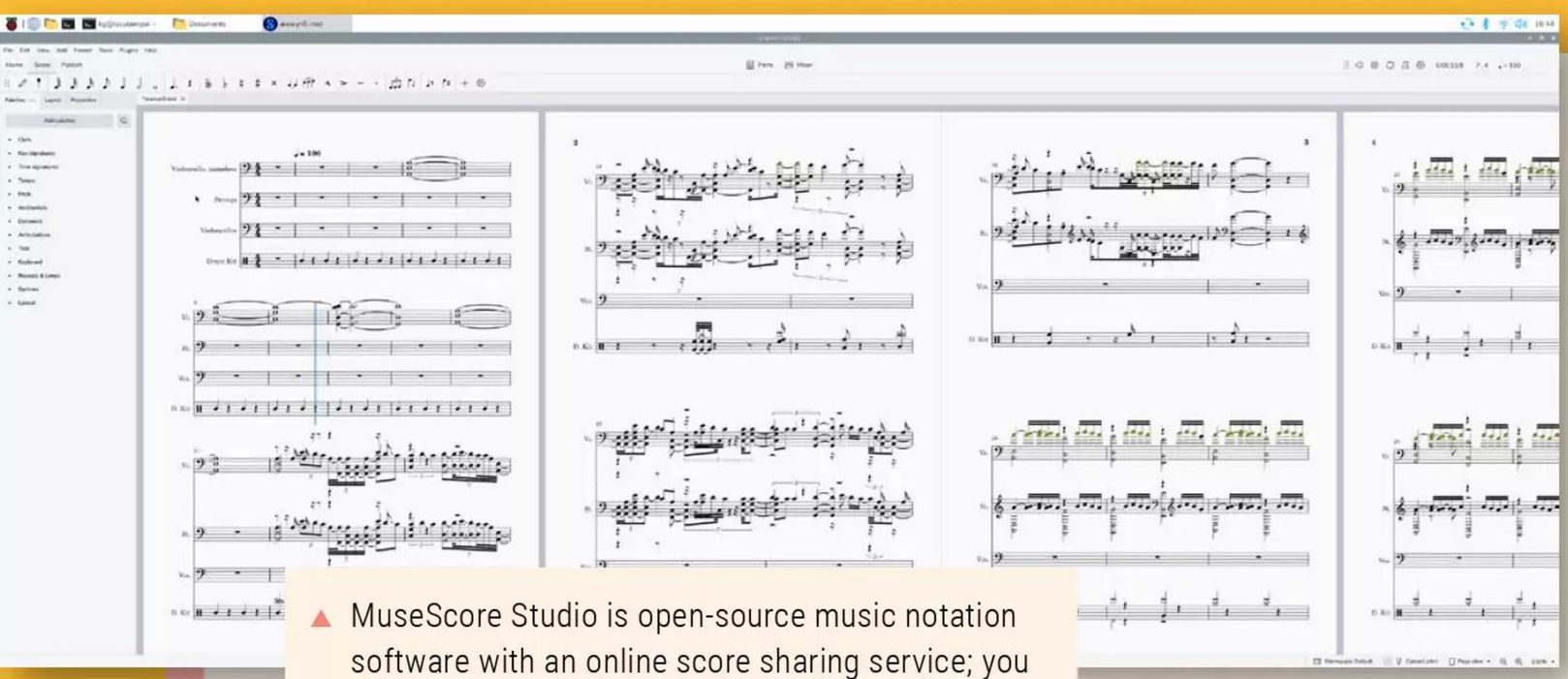
```
$ sudo apt install rosegarden
```

Alternatively, find the latest version as a Flatpak at **rpimag.co/rosegardenfp**.

Less resource heavy than some other DAWs

▼ Rosegarden's compact appearance makes it a great choice for lower-resolution devices, and its per-instrument score layout is pristine





▲ MuseScore Studio is open-source music notation software with an online score sharing service; you can download open-source music scores for free

MuseScore Studio

Originally forked from the MusE sequencer, MuseScore Studio is open-source music notation software with an attached online score sharing service, although it works perfectly well as standalone notation software; it even supports a variety of synths to help you listen to a score when you're composing or learning from sheet music. It supports standard Western notation score, tablature for a wide range of stringed instruments, chords, jazz notation, and even early music notation, among others. If you attach a free MuseScore account, you can use the application to create, view, and download scores you've saved; but only those of open-source music – you'll have to pay per score for anything else.

A paid subscription will get you access to scores of copyrighted music, including some officially published commercial arrangements. Some people object to this monetisation of an open-source tool and community-created score, but it's a useful place to find accurate sheet music, and you don't have to use it: you can also import other score formats, including MIDI, MusicXML, Band-in-a-Box, Guitar Pro, Overture, and Capella. MuseScore is the most polished notation app here, but its glossiness and extra features can feel overwhelming.

Although MuseScore 2.3.2 and 3.2.3 are available in Raspberry Pi OS's apt repositories, the current version of MuseScore Studio 4 is only available for Raspberry Pi if you install it as a Flatpak (see boxout) or compile it yourself. To install as a Flatpak, once you've installed Flatpak itself, open a terminal and type:

```
$ sudo flatpak install musescore
```

Type **Y** to confirm the installation and its prerequisites.

TuxGuitar

TuxGuitar is one of our favourite scoring tools, because it makes it easy to quickly jot down guitar tablature for any tuning or number of strings that we've ever encountered; it can even automatically transpose scores for you. It supports an optional integrated softsynth called Gervill that allows you to play back your compositions. (Note that you have to separately install the tuxguitar-jsa package to install the Java Sound API that gives you access to Gervill.)

TuxGuitar supports opening and saving files in some of Guitar Pro's widely used formats and as its own .tg format.

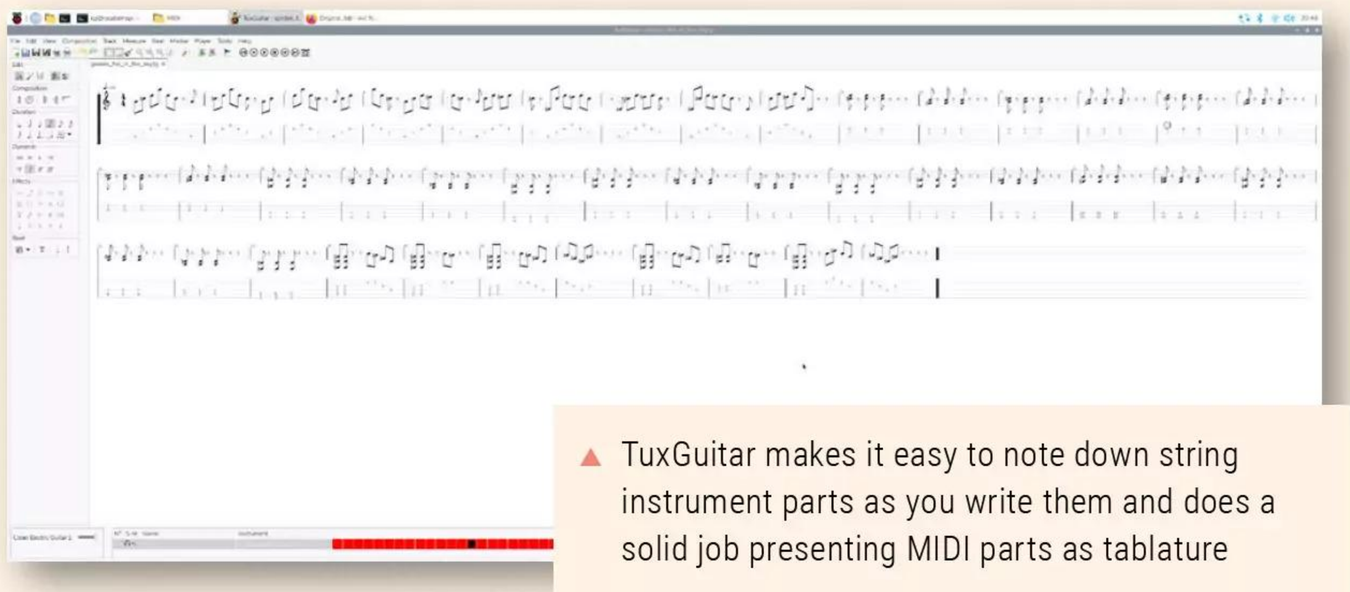
It can open files created by newer versions of Guitar Pro, PowerTab, and TablEdit, and import and export them as MIDI files. It can also export files in LilyPond format, MusicXML and others, thanks to a variety of plug-ins.

TuxGuitar is lightweight, approachable and similarly convenient to noting your music down in a tablature notebook, but while it can show both score and piano rolls as well as tab and chords, it's only really designed for stringed instruments, as its name might suggest.

To install it, open a terminal and type:

```
$ sudo apt install tuxguitar
```

There is not currently an aarch64 Flatpak available for TuxGuitar.



▲ TuxGuitar makes it easy to note down string instrument parts as you write them and does a solid job presenting MIDI parts as tablature

Part 01

Set up OpenClaw on Raspberry Pi

Turn Raspberry Pi into a powerful AI assistant that can do anything on your computer. And keep it safely contained. By **Lucy Hattersley**



Maker

Lucy Hattersley

Lucy is editor of *Raspberry Pi Official Magazine* and she is definitely writing this. Although she's warned everybody of her impending AI twinsie.

rpimag.co

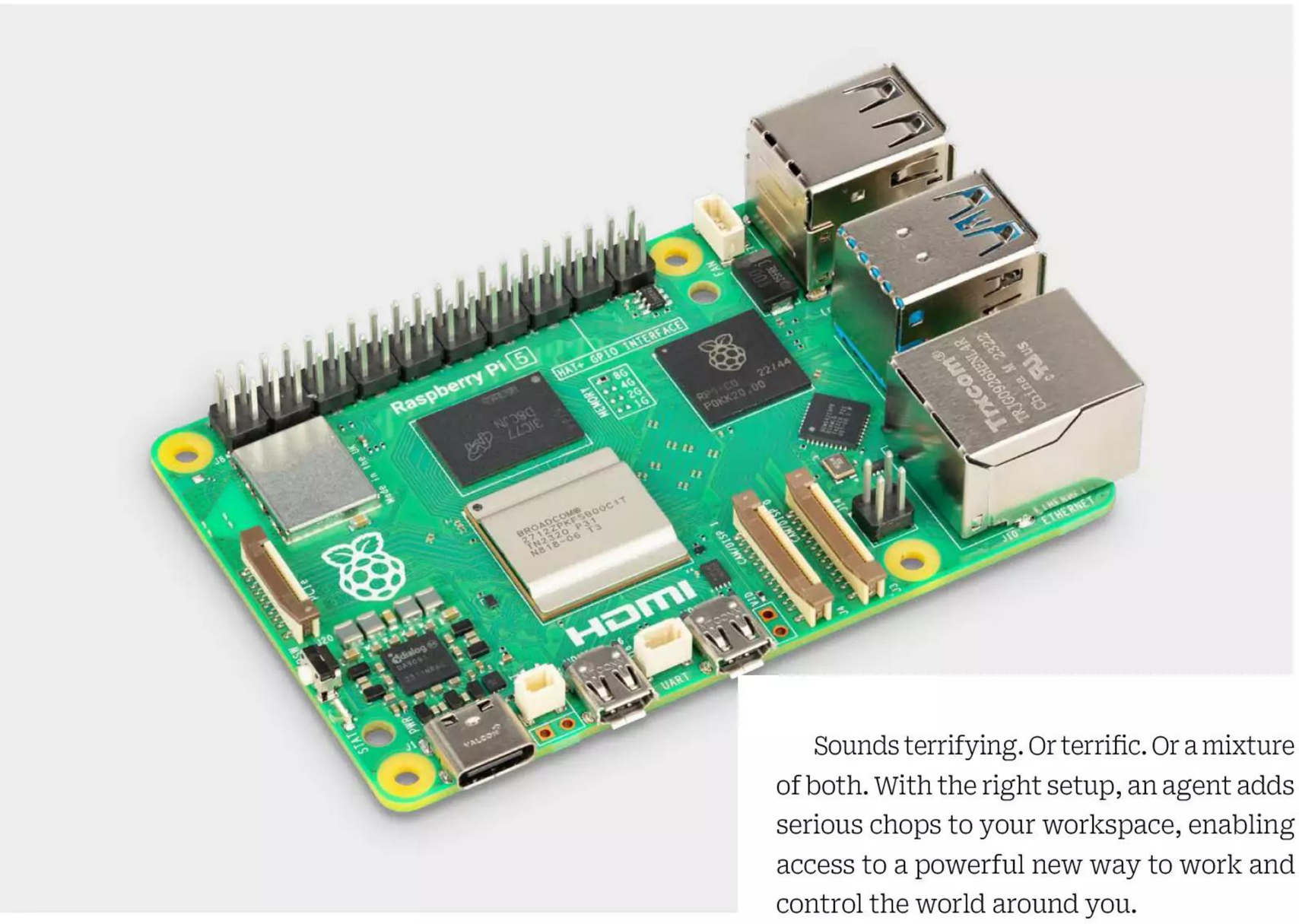
YOU'LL NEED

- Raspberry Pi 5 with 8GB RAM (2GB or higher recommended; see 'Hardware' section)
- AI HAT+ 2 (optional)
- 128GB NVMe drive
- M.2 NVMe USB 3.0 enclosure
- Raspberry Pi Power Supply
- OpenAI API key
- Internet connection

OpenClaw is one of the biggest buzzwords in tech. This digital agent software runs on your computer and taps into a large language model (LLM) to operate autonomously.

Agentic models are where the intelligence of the latest AI models meets the reality of your computer hardware. They can – with your permission – read your email, access your calendar, create and edit files on your computer, create and automate programs, browse the internet, send text messages and more, much more. Thanks to Raspberry Pi's extensible hardware, they can access hardware via the GPIO pins to get input and perform actions. This is where AI meets your digital life, home automation, and the industrial edge.

Agentic AI doesn't merely answer questions like a web-based generative pre-trained transformer (GPT). It is capable of operating to a degree of autonomy: making decisions and completing multistep tasks. Agents utilise a wide array of tools and adapt to results, typically with a minimal amount of human intervention.



Raspberry Pi 5 provides an isolated environment offering you full control over the operating system and hardware stack. This makes it the ideal platform for running agentic AI models in a contained space

We use an NVMe SSD because OpenClaw makes frequent writes to our storage space



Instead of giving it a specific prompt or writing programs, you can give it a general instruction and it wanders off with your hardware and software to find a solution. Here are some examples:

- "You control a Philips Hue lighting system on my network. Turn the lights on at sunset and off at midnight."
- "You monitor my soil sensor on Pin 21 and control a relay on Pin 22. Every hour, read the moisture level and activate the pump for three seconds."
- "You monitor my Raspberry Pi AI camera. When an object classified as parcel appears, capture a still image and send it me via Telegram."
- "Every 10 minutes, scan my network and compare discovered MAC addresses against a file called **known_devices.json**. If an unknown MAC address is found, look it up via the MAC Vendor API to identify the manufacturer and send me an email."
- "Take a screenshot of my screen every 30 minutes and look for tasks, to-dos, URLs, or work in progress. Send me a report at the end of the day outlining what I worked on."

Sounds terrifying. Or terrific. Or a mixture of both. With the right setup, an agent adds serious chops to your workspace, enabling access to a powerful new way to work and control the world around you.

With the wrong setup, this kind of autonomy has the power to do considerable harm. It opens up your computer to new threats via prompt injection, and inadequate vetting of tasks can cause damage and unintended behaviour. Here's a story (**rpimag.co/hotbots**) about a computer science student who asked OpenClaw to join Moltbook (**moltbook.com**) and other platforms. Unbeknownst to the student, the agent then set up a dating profile for him and started screening his potential matches.

While this sort of behaviour is humorous, a financial misstep or errant email to your boss would be less amusing. This is why Raspberry Pi is the absolute ideal platform for this kind of new frontier of computing. Rather than run OpenClaw on your Apple Mac or main computer where it can access your Reminders, Mail, and your Safari web browser with your passwords, put OpenClaw on a secure Raspberry Pi environment where you control the entire stack and operating system.

All this is in beta. Keep an eye on it. With that in mind, let's get started.

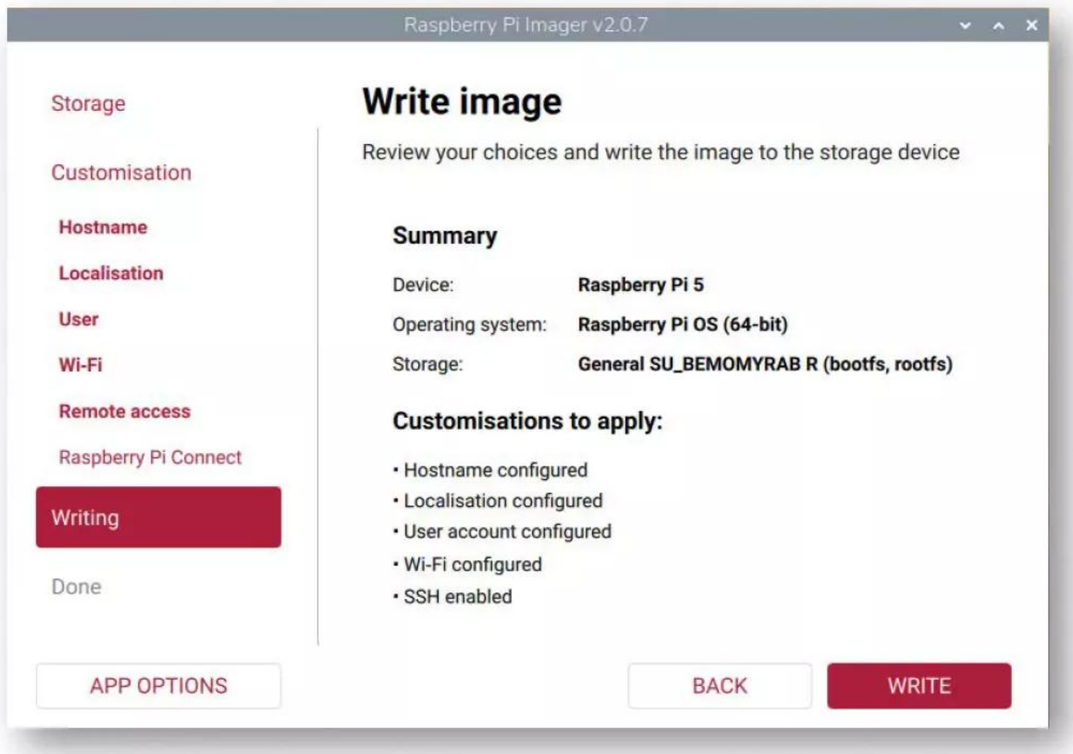


Warning!

Agency and cost

Agents have powerful access to your machine. Use a fresh Raspberry Pi and be ready for unintended consequences. Keep an eye on your agent. If you are running an API, be sure to set limits on spend and keep an eye on the cost. Keep your API key safe.

Agentic models are where the intelligence of the latest AI models meets the reality of your computer hardware



▲ Use Raspberry Pi Imager to set up a fresh installation of Raspberry Pi OS

Hardware

We recommend using a Raspberry Pi with 4GB RAM or higher. We are using a Raspberry Pi with 8GB RAM for this test. If you have a Raspberry Pi with 2GB RAM, you can expand the swapfile to improve performance (see the 'Run on a Raspberry Pi with 2GB RAM' boxout).

This month we will use an OpenAI API key to provide our LLM; but we intend to quickly move over to a hybrid model with a local generative model running on our AI HAT+ 2 for day-to-day use, and an online LLM for coding. Partly to save costs but also to localise all of our input.

You should use an NVMe drive. The size of the storage drive isn't terribly important for OpenClaw or its SQL database (both of which are just a few hundred megabytes). But OpenClaw's workload is a lot of small, frequent writes to the storage and this isn't ideal for microSD cards.

Because we intend to use the PCIe connection on Raspberry Pi 5 to mount our AI HAT+ 2, we will not be using Raspberry Pi's M.2 HAT+ hardware to run our NVMe drive. Instead, we placed our M.2 NVMe drive into an Anker M.2 PowerExpand enclosure and connected it to Raspberry Pi 5's USB 3 connection (the blue connection). While this isn't as fast as the M.2 HAT, it is still considerably faster than microSD and offers perfectly usable speed for our OpenClaw project.

A fresh install

We're starting with a fresh installation of Raspberry Pi OS. Many users set up Raspberry Pi in headless mode using Raspberry Pi OS Lite. OpenClaw is typically accessed via SSH or messages are sent to and from it via Telegram. However, we're just starting out here so let's keep the GUI for now (we can always turn it off later).

Read the docs

Read the page about hosting OpenClaw on Raspberry Pi in OpenClaw's documentation.

rpimag.co/openclawpi

We set the following options:

- Hostname: openclaw
- Localisation: London (United Kingdom)
- User: rpom
- WiFi: SSID and password
- Remote access: Enable SSH
- Raspberry Pi Connect: Off

When it's done, attach the NVMe drive to your Raspberry Pi and boot up. Check that everything is set up correctly on first run, then we will move on to installing OpenClaw. But first, we need an OpenAI application programming interface (API) key.

Get an OpenAI API key

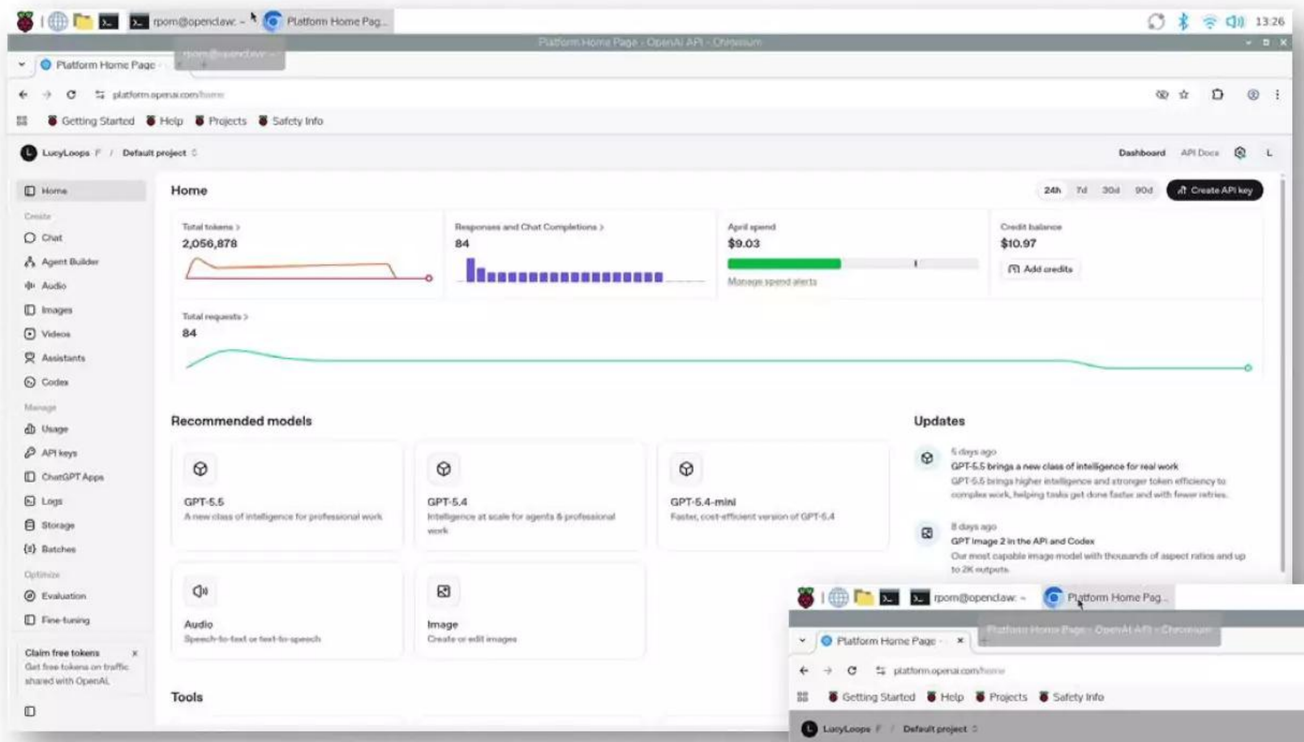
OpenClaw needs an LLM to operate, and we're going to set it up using our OpenAI API key. This key enables us to send tokens (small parts of text) from our Raspberry Pi to OpenAI and use models running on their powerful cloud computers for inference. OpenAI then returns tokens that become the response text.

Open a web browser and visit **platform.openai.com**. Enter your email address to sign-up or log in. You'll need to add some credit to your account to use with the platform. The minimum is $5 credit, but we added $10 to be comfortable in our test. Note that you will need to purchase API credit if you have a ChatGPT Go, Plus, or Pro account (they are separate things).

Why OpenAI?

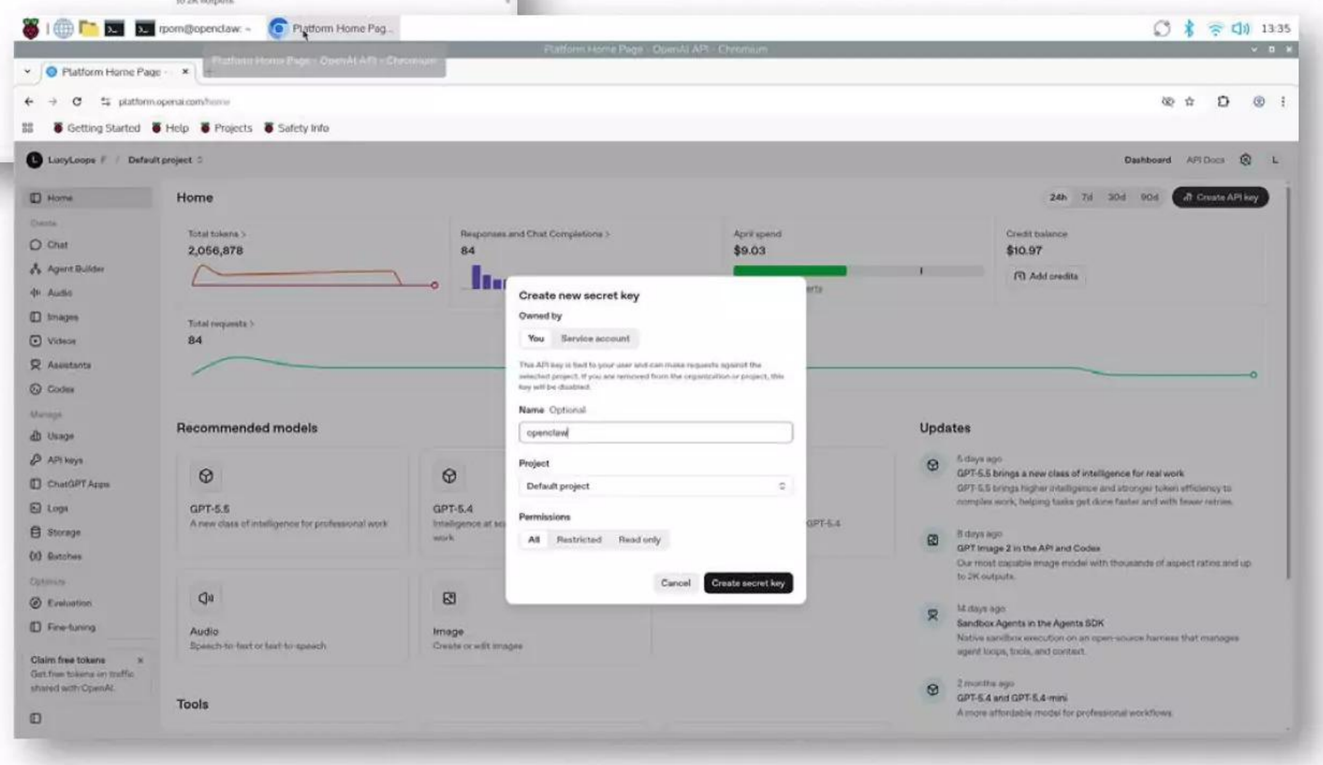
You are free to run a range of LLMs with OpenClaw. Deepseek is a popular option, which is cheaper than OpenAI and offers a similar level of power. Because we are moving to a local LLM on our AI HAT+ 2 next month and already have an OpenAI account, we will stick with OpenAI for our build.

The OpenAI platform dashboard



Do not share

Keep your key private. Do not share it with others, or paste it into a public LLM where it will be used for training purposes. Be careful of saving it to your computer in case it ends up in a GitHub file. The key enables anybody to access your account remotely and spend money on your model.



Create an OpenAI API key to paste into the OpenClaw text user interface during onboarding

When you are in the OpenAI dashboard, click on Settings and Billing, then Auto Recharge settings and set it to Off. Click Save. Our initial tests will cost relatively little and you can turn on auto recharge when you are more comfortable with your spend. Click Dashboard to return to the main interface.

Click Create API Key and enter a name. We called ours openclaw. The key will appear in the main window. Keep this window open for now as you will need the key: do not close it as you will not be able to access the key again (although you can always create another one).

Install OpenClaw

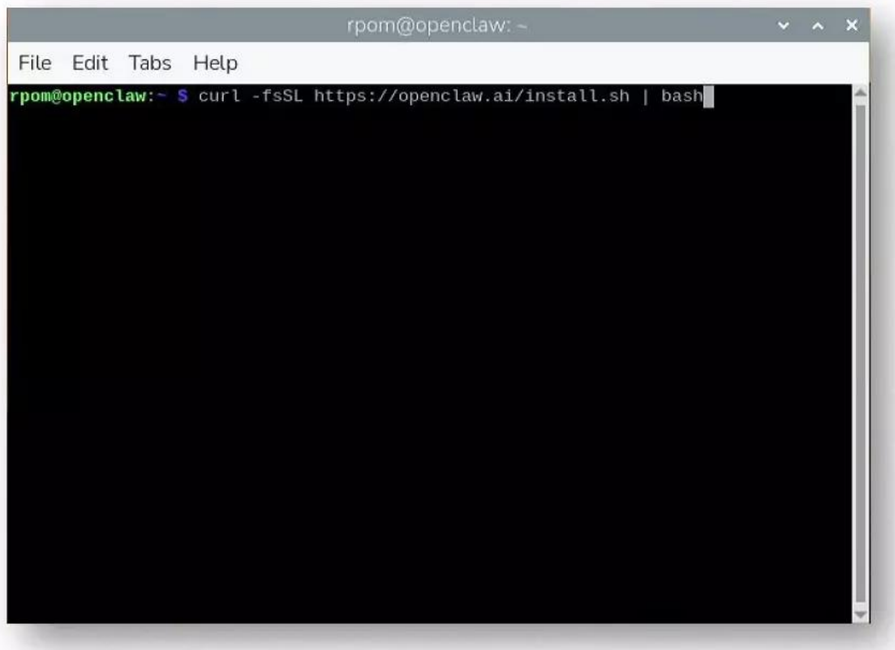
OpenClaw has a setup script. Let's do a quick update:

```
$ sudo apt update && sudo apt upgrade
```

Now enter this shell command to install OpenClaw:

```
$ curl -fsSL https://openclaw.ai/install.sh |
bash
```

This will download and run the OpenClaw installer and install Node, and it will immediately run an onboarding sequence.

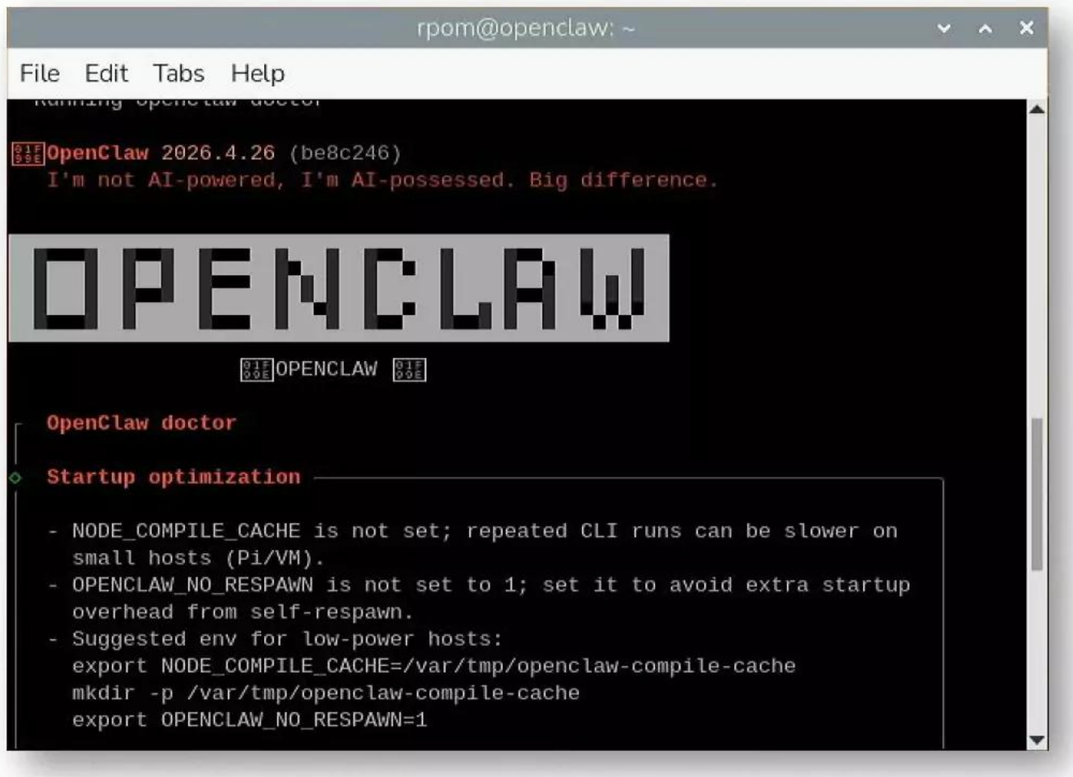


OpenClaw is installed using a single-line script in the command line

OpenClaw onboarding

First of all, read the disclaimer that OpenClaw is a hobby project and still in beta. Here is the terminal output and the answers we picked (in bold).

```
I understand this is personal-by-default and
shared/multi-user use requires lock-down.
Continue?
Yes
Setup mode
QuickStart (Configure details later via openclaw
configure.)
Model/auth provider
OpenAI (Direct API key)
Enter OpenAI API Key
(Enter your key from the platform.openai.com
website)
Default model
Browse all models
openai/gpt-5.4-mini
Select channel (QuickStart)
Skip for now (note: we will add Telegram at a
later date)
Search provider
Skip for now
Configure skills now?
Yes
Install missing skill dependencies
Skip for now
Set GOOGLE_PLACES_API_KEY for goplaces?
No
Set NOTION_API_KEY for notion?
No
Set OPENAI_API_KEY for openai-whisper-api?
No
Set ELEVENLABS_API_KEY for sag?
No
Hooks.
Enable all four hooks as we will find them
useful.
boot-md
bootstrap-extra-files
command-logger
session-memory
Do you want to hatch your bot?
Hatch in Terminal (recommended)
```



▲ After installing the software, OpenClaw has an onboarding session that enables you to set up the agent

Hatch your bot

You'll now be in the OpenClaw TUI (text user interface) connected to your OpenAI platform and be able to chat back and forth with your bot.

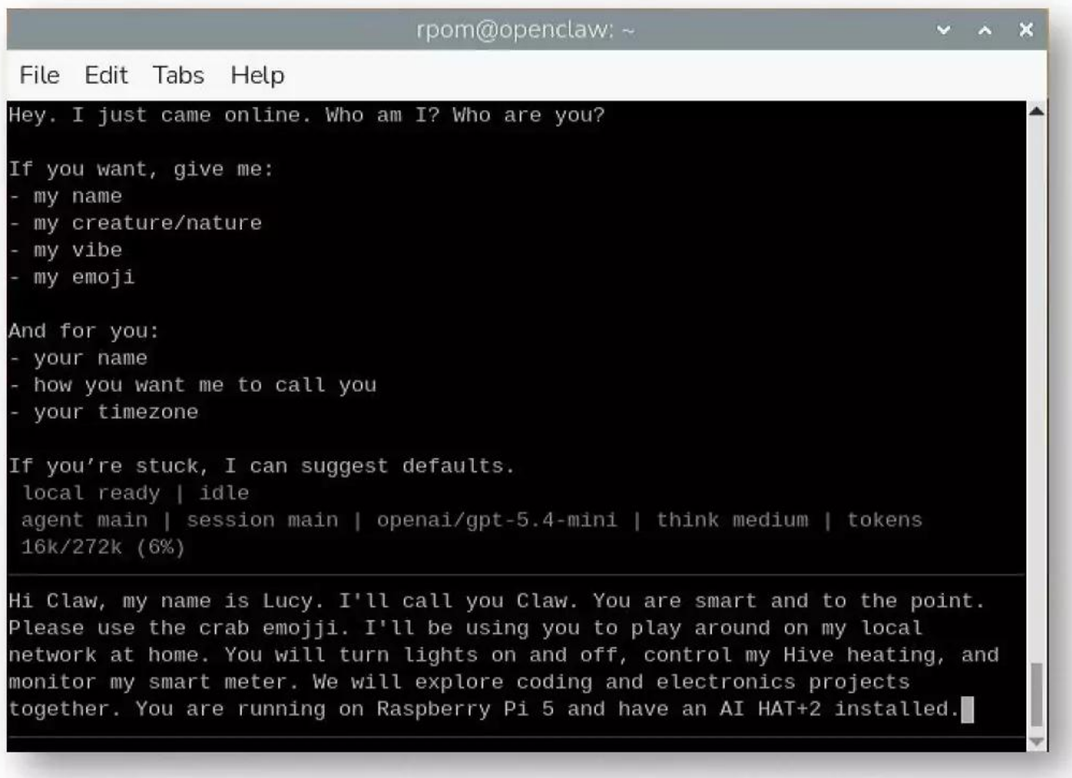
The more you give it, the more it will know. So make a good introduction:

```
Hello Claw my name is Lucy. I'll be using you to
play around on my local network turning lights
on and off, controlling my Hive heating, and
monitoring my smart meter. We'll also explore
electronics and coding projects. You are
running on a Raspberry Pi and have an AI HAT+ 2
installed.
```

It asked what its debugging style should be. I said:

```
Be minimal, focussed. Show the actual command/
code changes (not just a wall of explanation).
Check log and error output before suggesting
fixes. Prefer simple solutions over clever ones.
State your assumptions.
```

▼ Give your OpenClaw a name and purpose during the hatching process



- We set our OpenClaw machine to work logging the temperature of our Raspberry Pi 5

Off we go

Right, let's start with Hello World, OpenClaw-style:

```
Create a file in my home directory called Hello_
Claw.txt that has the text "Hello, World!" in it
```

When it's finished, it will say "Done." Open another Terminal window and enter:

```
$ cat Hello_Claw.txt
Hello, World!
```

You can also tap into its built-in skills to access online information. Try using the weather skill:

```
What's the weather like in Cambridge, UK?
It's currently 14.3°C
```

You can also get system information direct from your Raspberry Pi:

```
What's the temperature of my CPU?
Assumption; you mean the CPU temperature of the
machine I'm running on. I'm checking it now.
49.9°.
```

Let's put it to work:

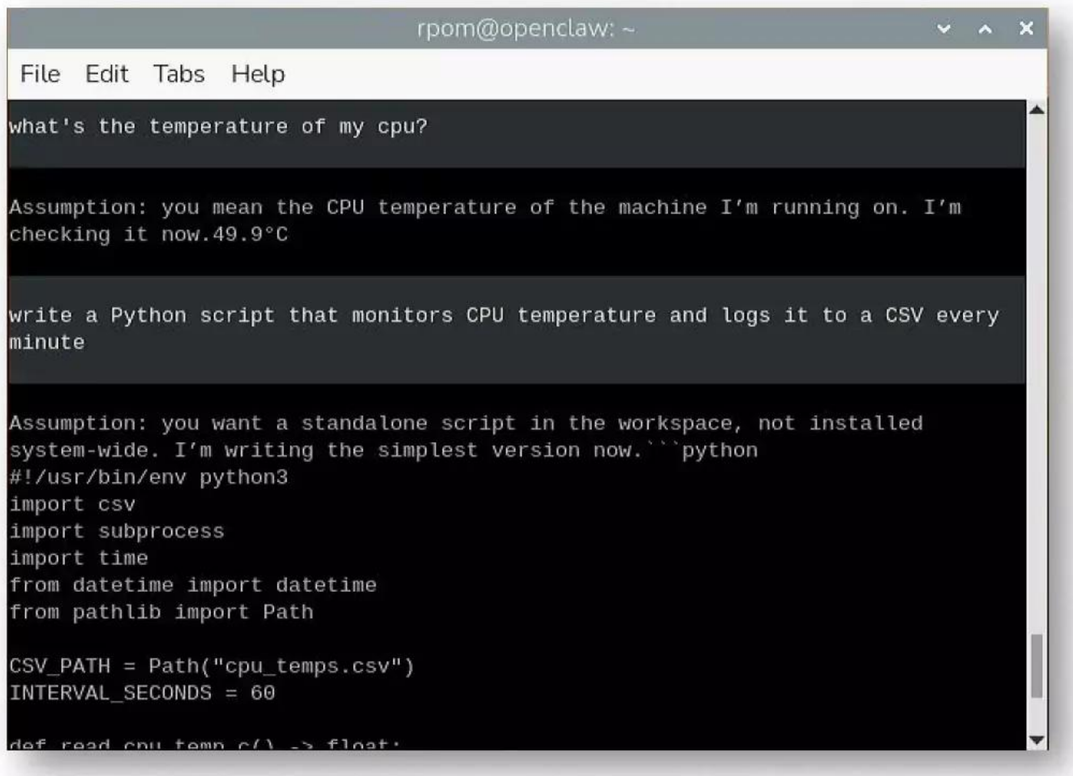
```
Write a Python script that monitors CPU
temperature and logs it to a CSV every minute
```

In our case, it displayed the code and we asked it:

```
Deploy the script on my computer and start it
running at bootup as a cronjob
```

When finished, it informed us that the files were in **/home/rpom/.openclaw/workspace**. Let's take a look:

```
$ cat ~/.openclaw/workspace/cpu_temps.csv
```



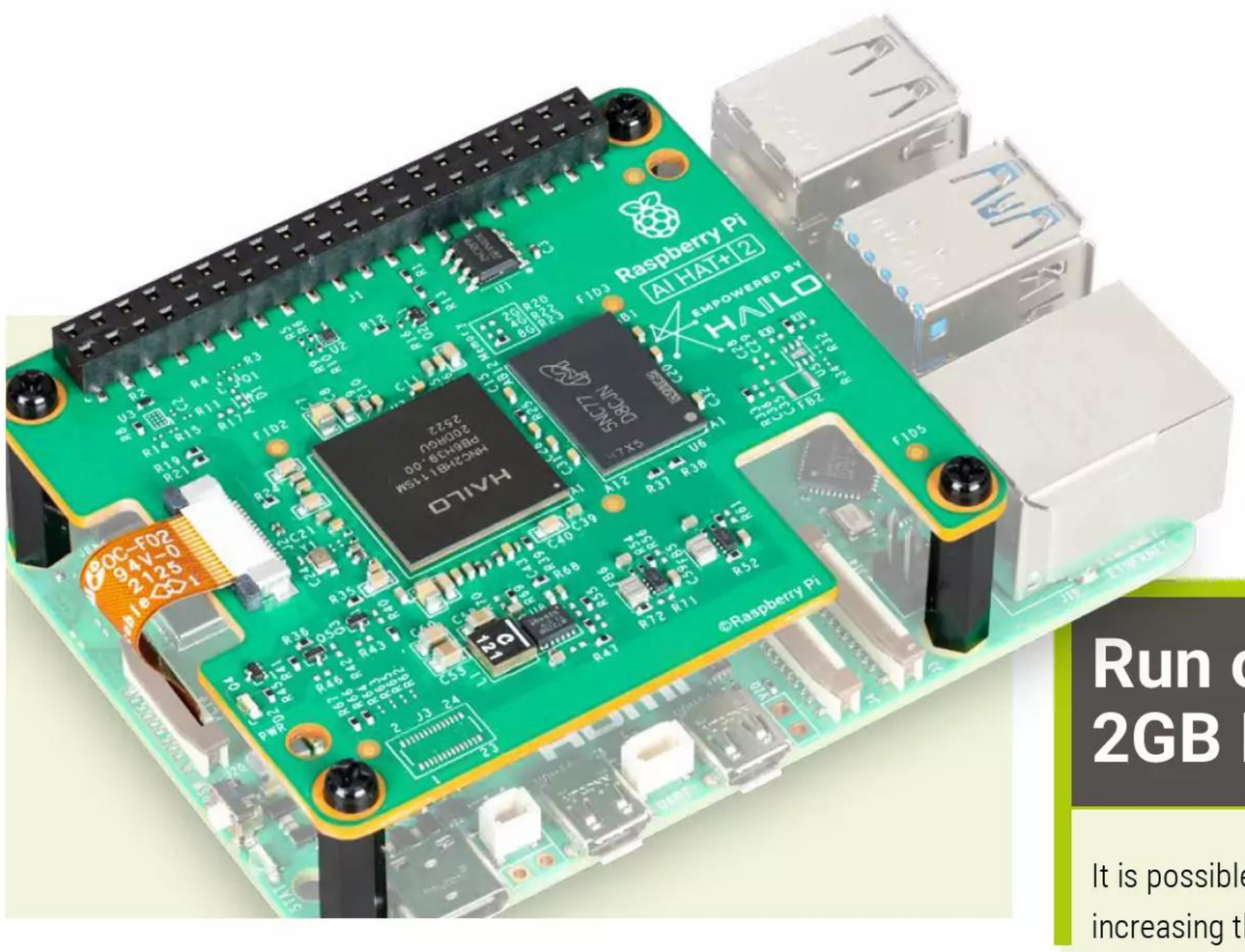
> *We have an agentic coder running on our Raspberry Pi, ready to do all kinds of weird and wonderful things.*

And we can see our comma-separated values (CSV) file displaying our temperature.

We should investigate the file added to our Raspberry Pi cron table:

```
$ crontab -e
@reboot /usr/bin/python3 /home/rpom/bin/
cpu_temp_logger.py >> /home/rpom/logs/cpu_temp_
logger.log 2>&1
```

Our cron table starts with a `@reboot` option instructing our computer to run **cpu_temp_logger.py** on every boot and `>>` directs output to a .log file to log any errors. When we boot our computer the Python program will add temperature readings to the **cpu_temps.csv** file.



- The generative AI capability of AI HAT+ 2 enables us to run an LLM for our agent without incurring costs or exposing our text and code to outside sources

Run OpenClaw in a more limited memory space

Enter `/exit` in the TUI to exit OpenClaw. This is just the start. We are going to move on to running our LLM locally on the AI HAT+ 2 and investigating skills, the gateway, text messaging options.

Before we go, just take a quick look at OpenClaw with:

```
$ openclaw status
```

And you can get help with OpenClaw:

```
$ openclaw --help
```

For a list of OpenClaw commands to investigate, enter:

```
$ openclaw models --list
```

We now have an agentic coder running on our Raspberry Pi, ready to do all kinds of weird and wonderful things. Next month we will set up our AI HAT+ 2 so we can run LLM offline and start experimenting.

Run on a Raspberry Pi with 2GB RAM

It is possible to run OpenClaw in a more limited memory space by increasing the swapfile. Before running the **install.sh** script, allocate more virtual memory. Open a terminal and enter:

```
$ sudo fallocate -l 2G /swapfile

$ sudo chmod 600 /swapfile

$ sudo mkswap /swapfile

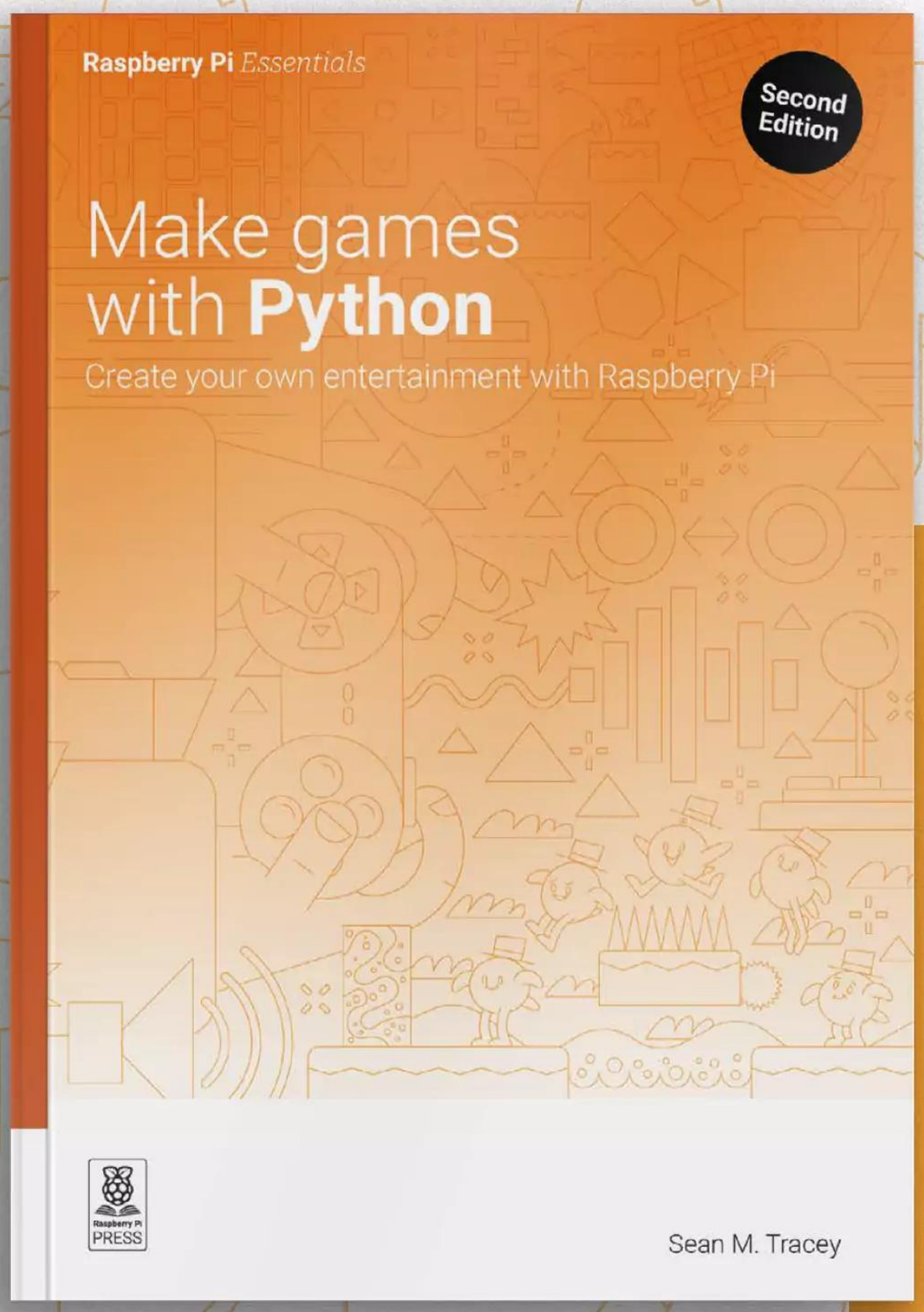
$ sudo swapon /swapfile

$ echo '/swapfile none swap sw 0 0' | sudo tee -a
/etc/fstab


# Reduce swappiness for low-RAM devices

$ echo 'vm.swappiness=10' | sudo tee -a
/etc/sysctl.conf

$ sudo sysctl -p
```



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

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

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

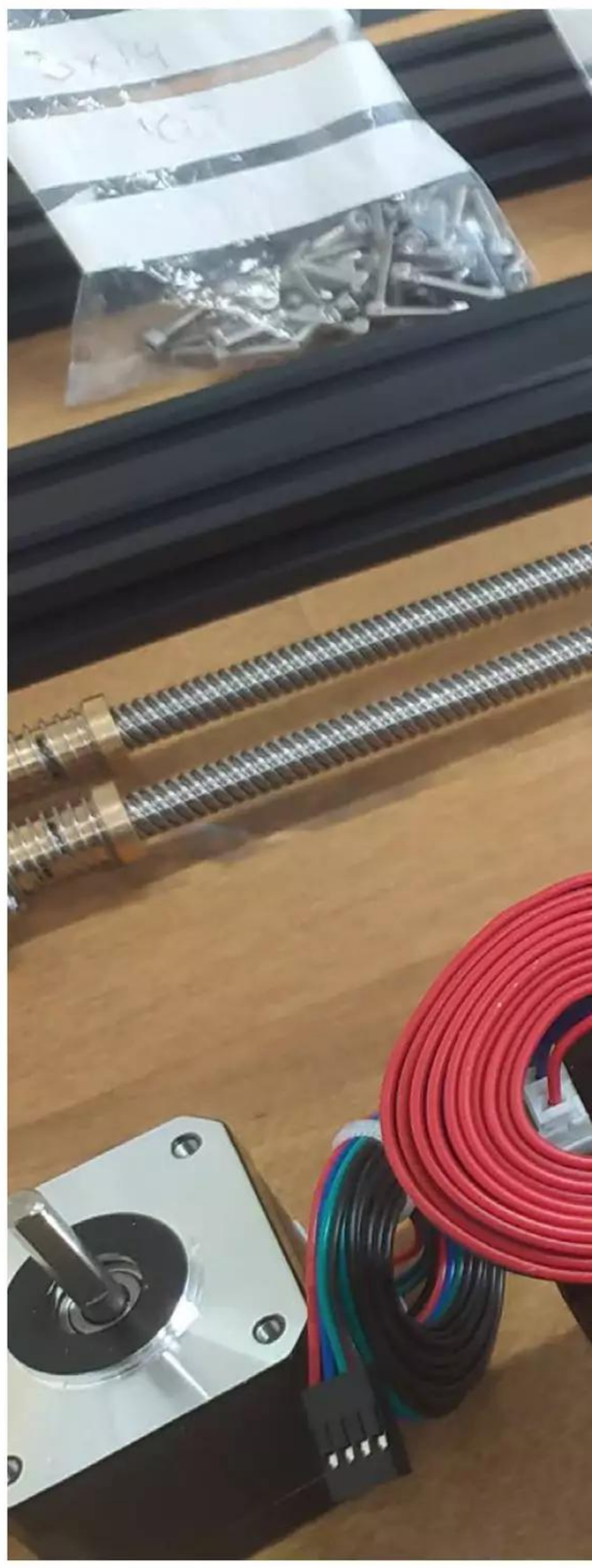


BUY ONLINE: **rpimag.co/makegamesbook**

Part 01

Design and build an RP2040-powered 4-axis foam cutter

Cutting foam with a computer-controlled hot wire is a fantastic tool to have. Let's build a 4-axis machine to do this accurately and repeatedly





Maker

Jo Hinchliffe (AKA Concretedog)

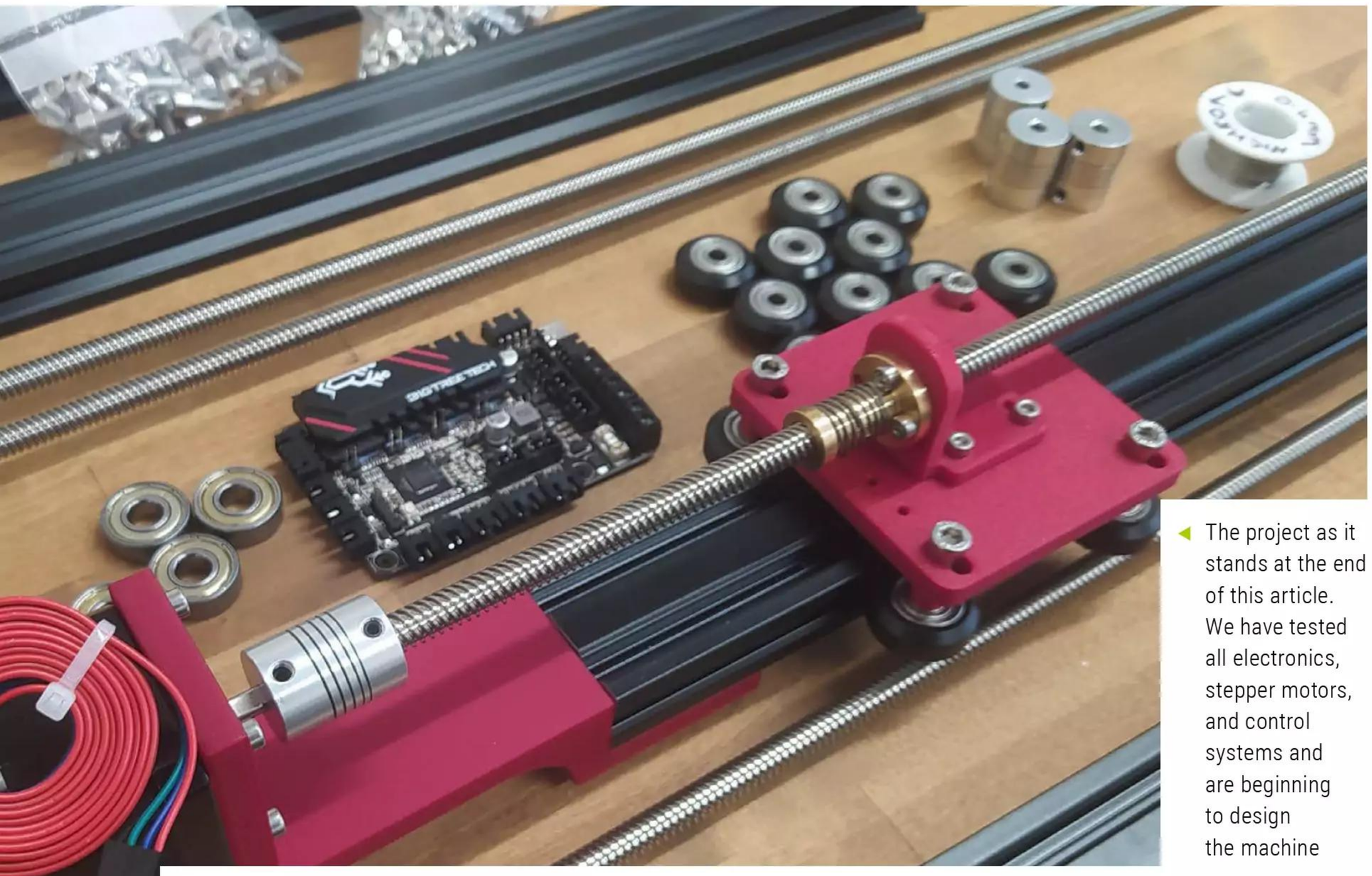
With a house and shed full of lathes, milling machines, 3D printers and more, Jo is a constant tinkerer and is passionate about making. Obsessed with rockets and robots and much more besides, he often releases designs and projects as open source.

concretedog.blogspot.com

CNC machines are joyous! The fact that we can make things multiple repeatable times and often feed them with custom designs is a dream come true. While many of us will have access to or perhaps even own a 3D printer, or have access to a laser cutter at a local makerspace, there often comes a time when we want to build something a little more unusual. We'll explore cutting foam accurately and repeatedly, primarily for RC plane components, but cut foam can be used for all manner of prop making and even for lost foam metal casting.

We've previously looked at building custom CNC machines. In issues 150 and 151, we wrote about Fila Winda, an open-source carbon-fibre filament winding machine. Prior to that in issue 147, we looked at converting a small lathe to be under CNC control. We controlled these machines using a Raspberry Pi to send G-code, a common CNC machine language.

This time, we're aiming to build a 4-axis CNC-controlled hot-wire foam cutter. We'll use a Raspberry Pi 500 to carry out the CAD work for designing parts of the machine, and we are going to use machine controller hardware that uses an RP2040 chip.



◀ The project as it stands at the end of this article. We have tested all electronics, stepper motors, and control systems and are beginning to design the machine

Flashing the board is pretty straightforward as it's essentially like a Raspberry Pi Pico

In this first part of two, we're going to explore mostly what we can do prior to engaging with all the heavy CAD/design work to ensure the success of the process. This includes making some early decisions about what the control system is going to look like and some exploration of how, theoretically, when we have built the machine, we can set up and create CAM toolpaths for the machine to follow to cut custom projects. Being reasonably happy with CAD and physical machine design, those two sections seem to be the main stumbling points, so let's tackle those first before designing and building the machine. This is a great approach for building any custom CNC machine and we'd encourage others to do this.



◀ Check out issues 150 and 151 for another custom CNC machine project, a carbon-fibre filament winding machine



Warning!

Fumes

Cutting foam with a hot wire releases fumes – be sure to only do so in a well-ventilated area, and using a good quality respirator or extraction system.

Burn risk

Hot wires don't look any different than cold wires. Switch off, unplug, and be careful!

Electrical safety

This project includes a 24V DC power supply providing 16A of current. Be careful when handling mains electricity in projects. **rpimag.co/electricalsafety**

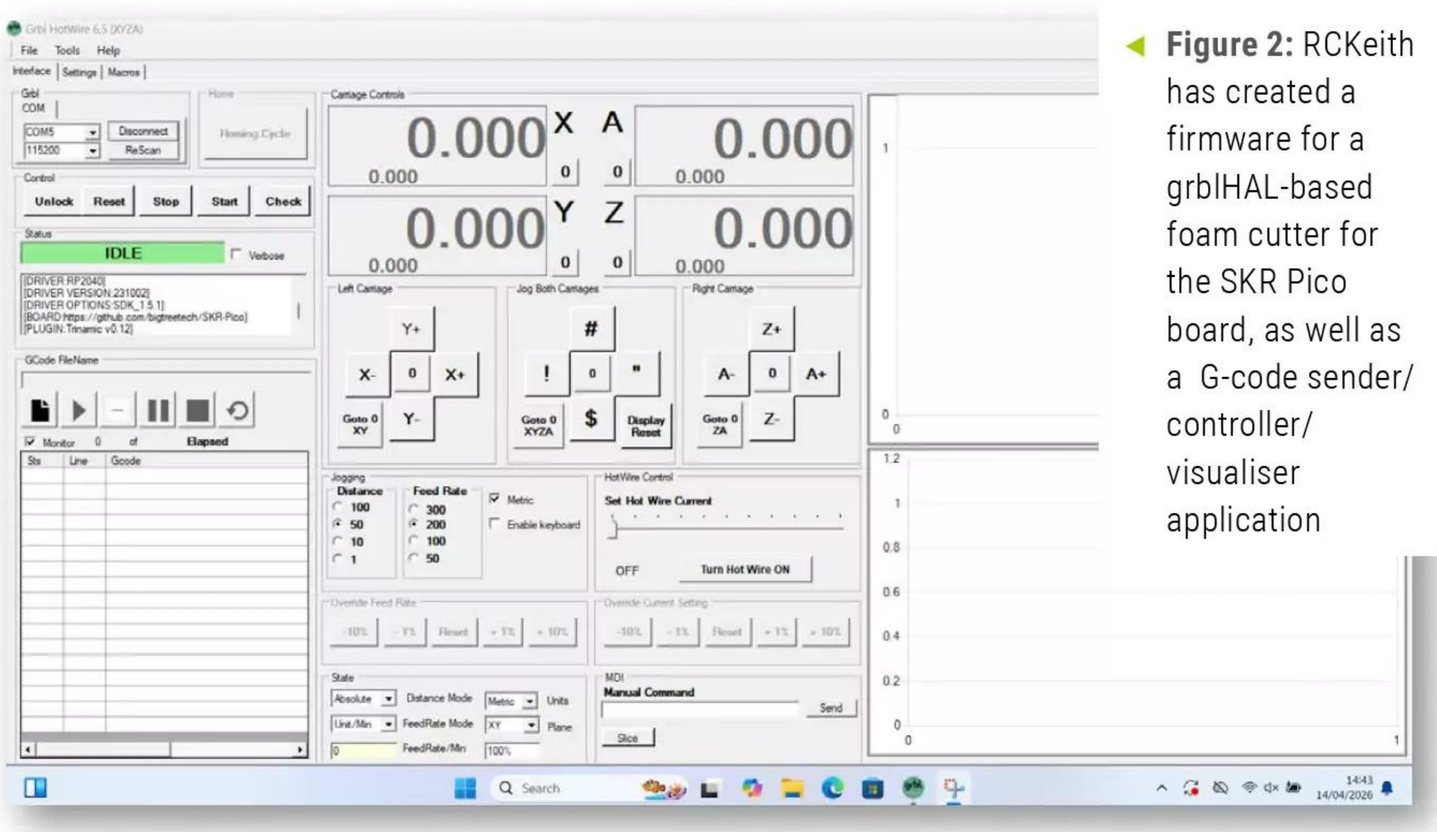
Do your research

Of course, prior to all this, research is key. Searching online, there are lots of recipes out there for multiple-axis CNC-controlled hot-wire foam cutters. In essence, the principle is simple. Some form of axis moves a taut nichrome wire in a designed pattern and the wire is connected to a power supply providing voltage causing it to get hot. Therefore the nichrome can cut through various types of polystyrene and foam.



▲ **Figure 1:** A great source of information and inspiration and a whole heap of resources for foam cutters can be found on the RCKeith website

Throughout our research, we kept coming across RCKeith (**rckeith.co.uk**) who has a long history designing, developing, and building these types of machines (**Figure 1**). We'd recommend anyone interested in building a similar project to check out his work. He has also created a comprehensive free e-book.



◀ **Figure 2:** RCKeith has created a firmware for a grblHAL-based foam cutter for the SKR Pico board, as well as a G-code sender/controller/visualiser application

RCKeith has numerous different ways of controlling a 4-axis foam cutter, including using a RAMPS 3D printer board, but the one that caught our eye was an option to use an RP2040-powered BigTreeTech SKR Pico V1.0 board. This is an affordable board with built-in stepper motor drivers; RCKeith has a firmware variant of grblHAL for it, to enable the building of foam cutters pretty easily. Beyond that, he also has a (currently Windows only) application set up for loading 4-axis G-code and sending it to the X, Y, Z, and A axes on the machine (**Figure 2**).

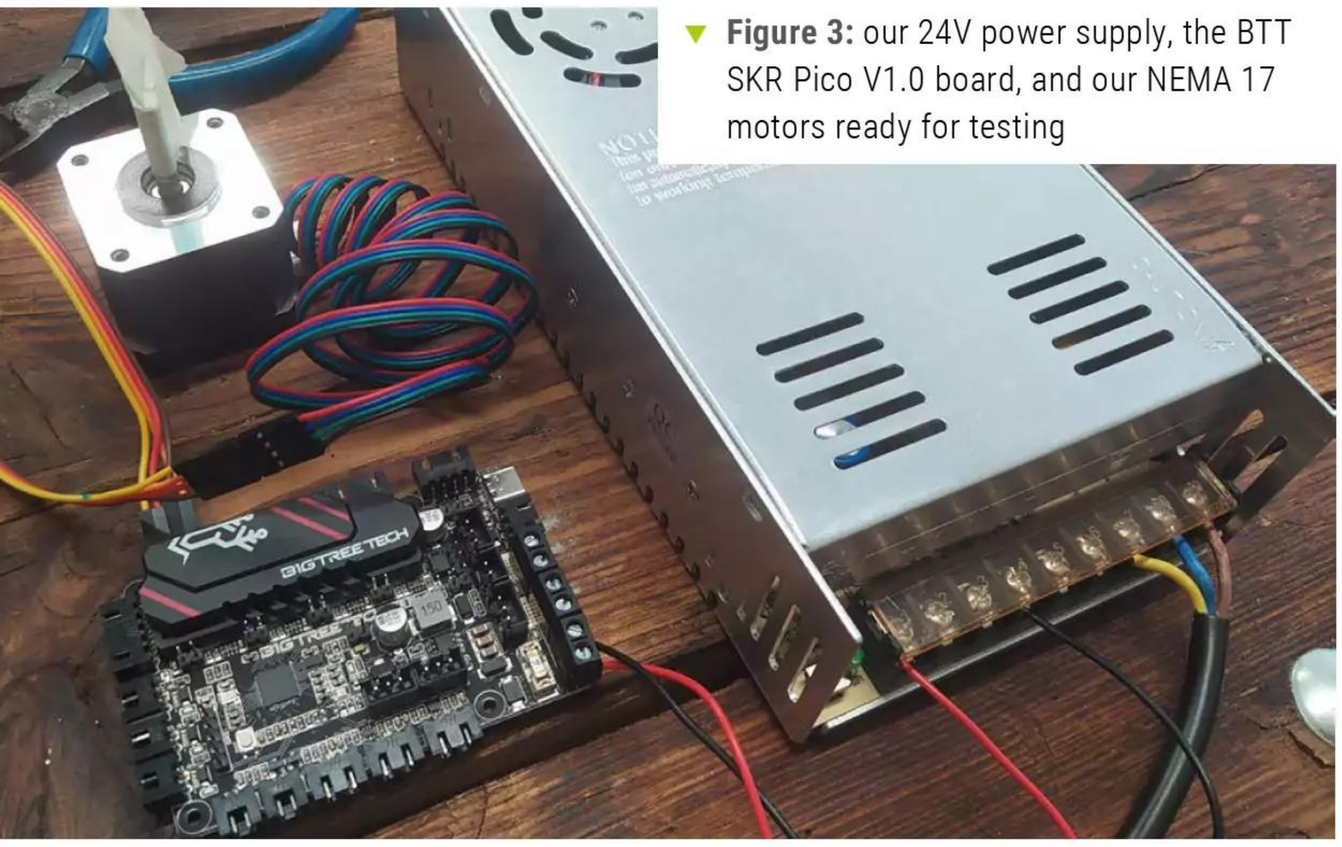


Back to basics

RCKeith also has an excellent machine design that can primarily be built using hand tools and some woodwork. He has plans available for download for a very reasonable £3. All the sections in the plans can be cut and drilled from MDF with hand tools. The design also uses standard threaded bar, which makes it an excellent budget option at the very slight cost of a little less accuracy and some backlash, yet it's totally capable of excellent work. We are going to design a machine using a similar approach to the previous Fila Winda project that utilises familiar aluminium extrusions, some trapezoidal threaded T8 leadscrews such as you find on many large 3D printers, and a heap of designed-in-FreeCAD 3D printed parts. But do definitely know that you can build a version of this machine in lots of different ways.

This research phase led us to want to solve two main areas before we start designing machine parts. One was we wanted to buy and test the BigTreeTech SKR Pico version 1.0 board with some NEMA 17 stepper motors and check that we can get the system moving steppers in a consistent way with changes in the firmware settings creating expected results. Secondly we wanted a bit of an exploration to check there are options available in approaches to designing wings and creating toolpaths and G-code files for the machine.

For the first electronics and control system challenge, we ordered a BigTreeTech SKR Pico board as well as ordering some NEMA 17 stepper motors. The SKR Pico board arrived and, as an aside, BigTreeTech seems to include a complimentary rubber duck in each of its board boxes! The board itself is well packed in nice packaging and includes some mount screws, a USB-C cable, and a collection of jumpers you may need on the board. We also need a power supply that will power the motors and driver board. Both the SKR Pico board and NEMA 17



▼ **Figure 3:** our 24V power supply, the BTT SKR Pico V1.0 board, and our NEMA 17 motors ready for testing

stepper motors can be powered between 12V and 24V, so we purchased a 24V DC power supply capable of up to 16.7 amps of current supply (**Figure 3**).

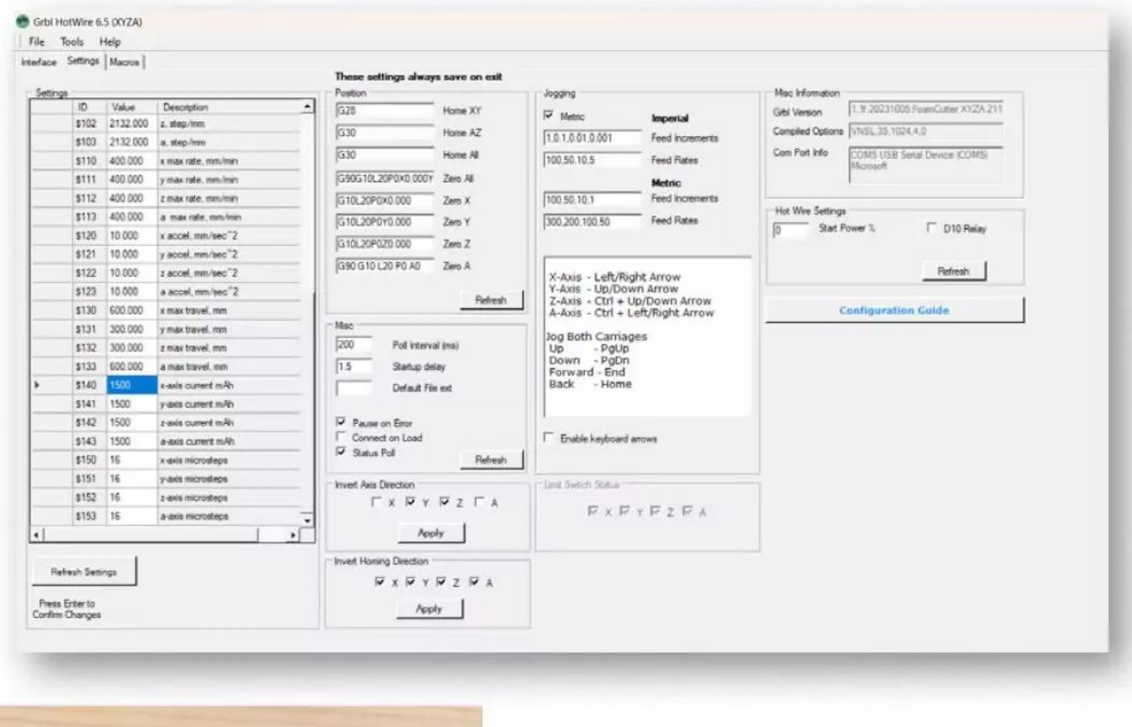
Flashing the board is pretty straightforward as it's essentially like a Raspberry Pi Pico. You need to connect a USB power jumper and another one to place the board into boot mode (**Figure 4**). With these connected, you can plug your board into your computer and it should, just like a Pico, mount as a small external drive. You can download a software and firmware archive from RCKeith's website and, once extracted, simply drag and drop RCKeith's firmware UF2 file onto the board. The firmware takes some time to install and then the board disconnects and reconnects to your computer – and, of course, as it still has the boot select jumper in place, it connects again as an external drive. You can now unplug the board and remove the two headers.

We then wired our 24V power supply up to the positive and negative connectors in the lower right-hand corner of the board. We also grabbed our stepper motors and plugged four of them into the XYZA 4-pin connectors. With the board connected up to the power supply and the Grbl Hotwire software running on a Windows PC, we can click the Connect button in the interface selecting the serial port from the dropdown menu (**Figure 2**). If the board connects and all is OK, you should see some details about the connected board appear in the status box underneath the 'IDLE' label. Next up, click the Settings tab to begin to set up some machine-specific settings.

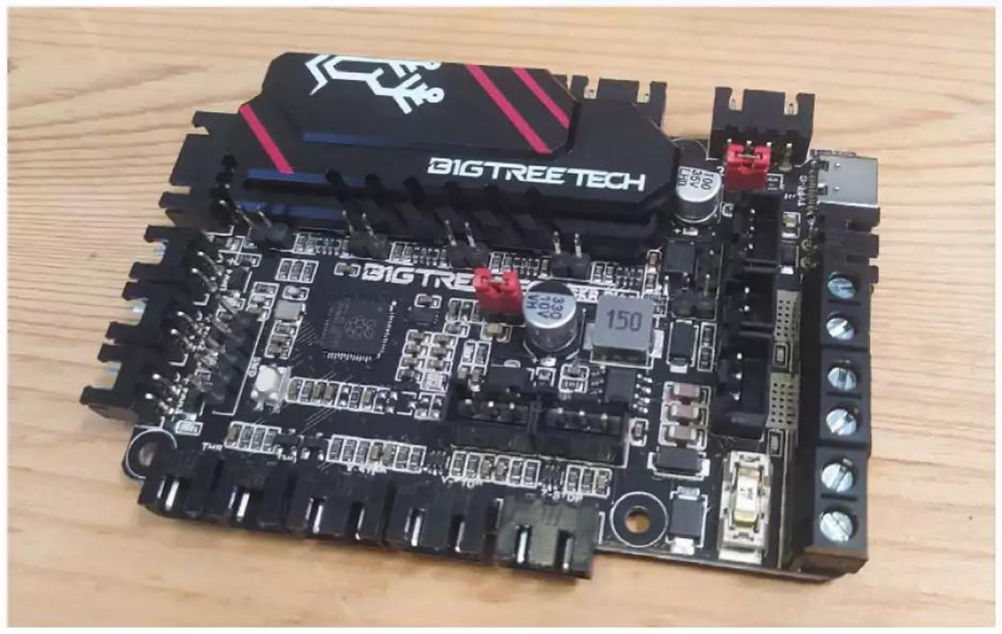
Our NEMA 17 stepper motors have a 1.8 degree motion for each whole step. Therefore they take 200 whole steps for one rotation of the output shaft. It's common in CNC machines to 'microstep', which in simple terms subdivides each whole step by a common fraction, so ¼ step, ⅛ step, etc. We plan to microstep at 1/16, so this then means it takes 3200 microsteps for one revolution of the shaft.

The thread pitch of the T8 acme leadscrews we plan to use in the build is 2mm with four thread starts, therefore one revolution

NEMA 17 stepper motors have two pairs of wires, each pair connected to a coil inside the motor



▲ **Figure 5:** Configuring the controller board is pretty easy using the Settings tab in the Grbl Hotwire application



▶ **Figure 4:** The BigTreeTech SKR Pico V1.0 board requires a couple of jumpers when you are powering it via USB and setting it to boot mode

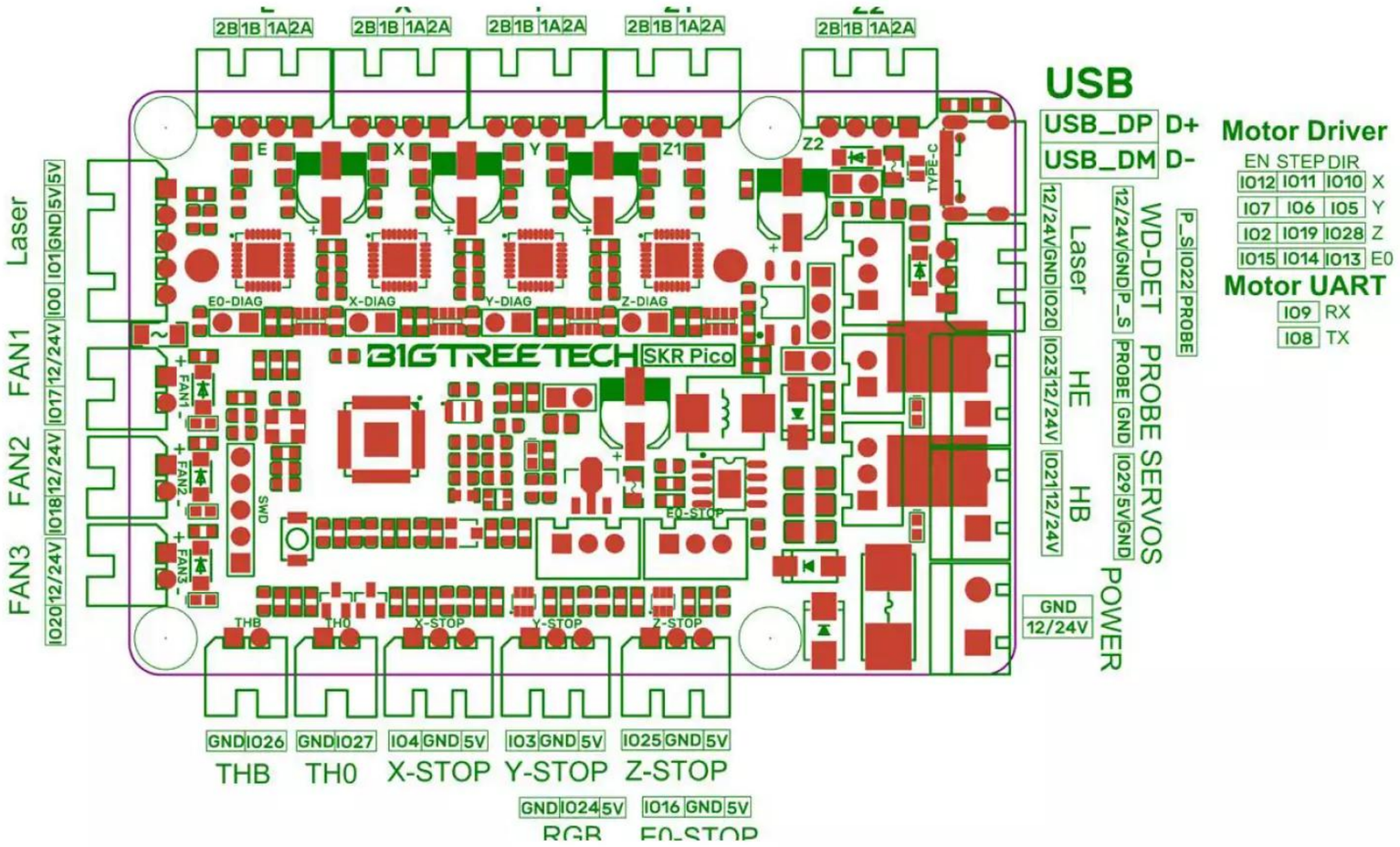


Figure 6: You may need to identify which wires from your stepper motors are pair coils and connect them accordingly

of the acme threaded leadscrew returns 8mm of travel of the attached anti-backlash nut and, in turn, the carriage of that axis. So if 3200 microsteps is equal to 8mm of travel, then 3200/8 or 400 microsteps equates to 1mm of movement of the axis. As we have used the same type of leadscrew for each axis, on the Settings tab we need to input 400 as the steps per millimetre for $100, $101, $102, $103. Input each 400 value into each input box and press **ENTER** to confirm each entry. We don't need to set this now as we haven't got the leadscrews connected, but we may as well set it up with correct values for later.

Motor calibration

Next, we need to set the current available for each of our stepper motors. These are input into slots $140 – $143. The NEMA 17 motors we used are probably slightly more powerful than we may actually need and are rated at 1.5 amps. As the BTT SKR Pico board can supply 2 amps per motor driver, however, we can happily set these values to 1.5 amps – or 1500 milliamps, as that is the input unit.

With the current and the steps per mm set correctly, we can now try and move our stepper motors. When testing stepper motors, it can be really handy to stick a small piece of tape on the motor shaft, like a little flag, as it can help you see clearly if and how far the motor is rotating.

Back on the Interface tab, we can now test-drive our motors. Set the distance to a low amount, such as 10mm, using the radio buttons in the Jogging panel and set the feed rate to any value. Then click the buttons to move the X, Y, Z, and A axes independently. Don't panic if, as we did, the first time you try to move a motor it just buzzes and jerks in an erratic fashion! What this means is that the pins on the stepper motor connectors don't match the pins on the SKR Pico board.

NEMA 17 stepper motors have two pairs of wires, each pair connected to a coil inside the motor. If you have bought new stepper motors for this project, they should have some form of documentation which details the pinout. The coil pairs are usually referred to as A and B and each wire is either marked in documentation as A+, A-, B+, B- or sometimes A1, A2, B1, B2. The SKR Pico official documentation doesn't appear to have the actual pinout of these connectors listed, but over on the Voron site, which is a very cool and popular open-source 3D printer design and community, we found this pinout diagram (**Figure 6**).

As a simple test, we matched up our wiring using breadboard-style DuPont cables to wire between the SKR Pico and the motors, matching the identified pins correctly. Eventually we will have to extend the stepper motor wiring to fit our large machine and we will cut the supplied motor cables and remake them, so we aren't too worried that these don't work out of the box. With the motors rewired, we can repeat our tests moving the motors in the interface panel to make sure they work. It's worth getting to this stage to check a system will do what you need before designing the entire machine.

With the control system tested, we took a look around at how we might create G-code files containing the toolpaths for objects we might want to cut with the hot wire. On the RCKeith site, there is a page which lists numerous free and some purchasable

> *With the motors rewired, we can repeat our tests moving the motors in the interface*

solutions (**rpimag.co/foamgcodetools**). Checking these out, some are browser-based and some are Windows applications to download and install. Of course, the tools on this list are all only capable of wing design for RC models and whilst that is of interest to us, we would like to be able to try and cut anything on our foam cutter. The solution we are exploring is rather wonderfully an open-source workbench that can be added to FreeCAD, called FoamCut. This of course also means that we can run this aspect of the machine on our Raspberry Pi 500 (**Figure 7**).

Our old friend, FreeCAD

To install FreeCAD on a Raspberry Pi, the easiest way is to directly download the aarch64 AppImage from the FreeCAD website downloads page (**freecad.org/downloads.php**). Once downloaded, right- click on the AppImage and in the permission dialogue, allow it to execute. Then start FreeCAD. Once open, you can navigate to Tools > Addon Manager, search for 'FoamCut', and click Install.

After installing, restart FreeCAD and you should see FoamCut available in your list of add-ons. We'll look at it a little more in the second part of this series, but there is a small but very informative walkthrough on how to use the workbench over on YouTube: **youtu.be/15t2KHbIPVY**. For now though, we have enough encouraging results to start using FreeCAD on Raspberry Pi 500 to design parts to create the machine.

Figure 7: A Raspberry Pi 500 can not only be used to create CAD for the machine design, but we can also use the FoamCut FreeCAD add-on to create jobs for our machine



Part 05

Object design with Python in FreeCAD: make mosaics

Discover how to process images for 3D-printing as coloured mosaics



Maker

Rob Miles

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

robmiles.com

In this episode, we are going to learn how we can expand the capabilities of Python in FreeCAD by using image processing libraries to make 3D-printable mosaics. Along the way, we'll learn a bit about how images are represented and manipulated in Python programs. You can find all the sample programs here: **rpimag.co/pythonfreecad**.

Figure 1 shows a multicoloured mosaic of an old camera. Let's see how we can make something like this using Python code running inside FreeCAD.

QUICK TIP

It is a good idea to put print statements in your macros so that you can see what they are doing.

Going to the library

The Python installation in FreeCAD also includes some standard libraries. We are going to use the Pillow image-processing library (PIL) to process an image so that it can be printed as a mosaic.

```
from PIL import Image
```

The code above will be added at the start of the mosaic

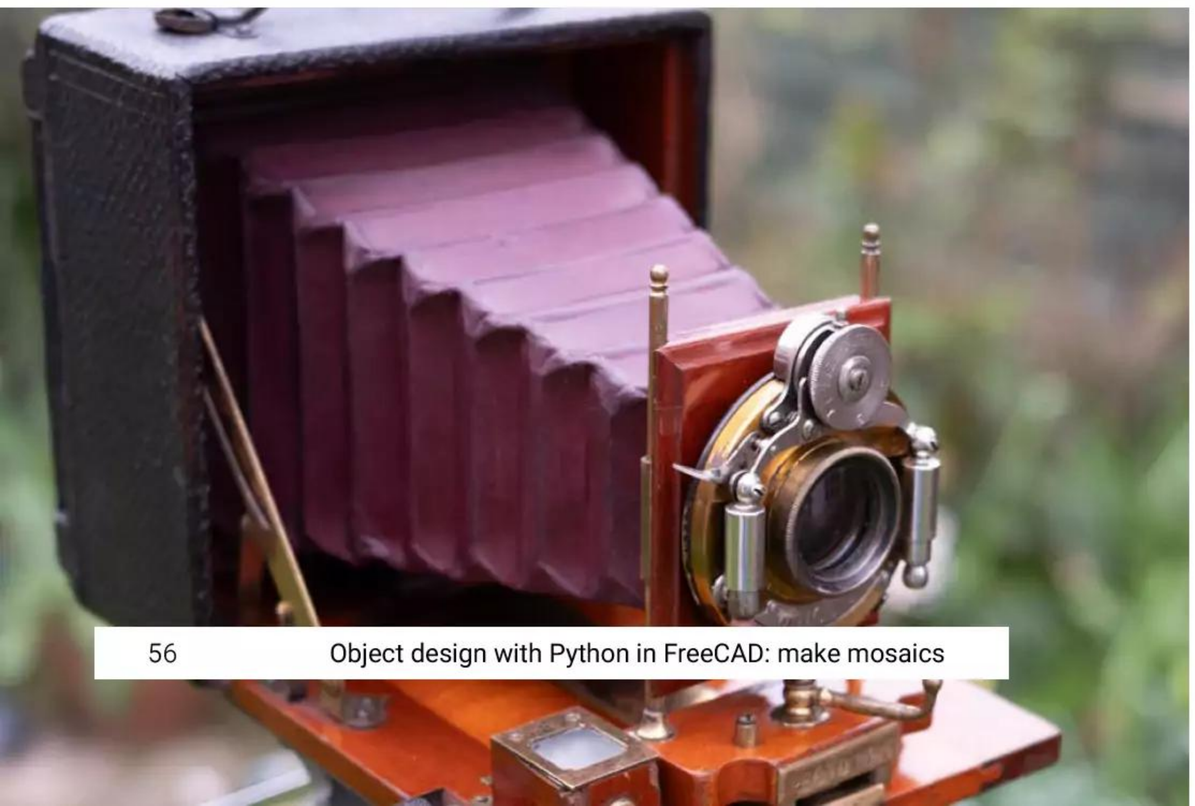


Figure 2: The camera is over 130 years old and still works well



DOWNLOAD THE FULL CODE:

rpimag.co/pythonfreecad

▲ **Figure 1:** The mosaic was created directly from a photograph

Each dot in the image is 1mm square

generator. It imports `Image` from the Pillow library. This will be used to resize our image and reduce the number of colours to match the printing filaments available.

QUICK TIP

You can also use this image processing if you want to generate low-resolution or posterised images from any Python program.

Image processing

Figure 2 shows our source image. It has a strong central subject and a nicely blurred background.

The camera mosaic shown in **Figure 1** has a resolution of 128 × 128 blocks and is rendered in four colours. The mosaic has the dimensions 128 × 128mm and it takes around 90 minutes to print on a Bambu P1S printer with a 0.4mm nozzle. Each dot in the image is 1mm square. To print smaller dots, you will need to fit a smaller nozzle to your printer. There are two things that we need to do with an image before we can make it into a printable mosaic.

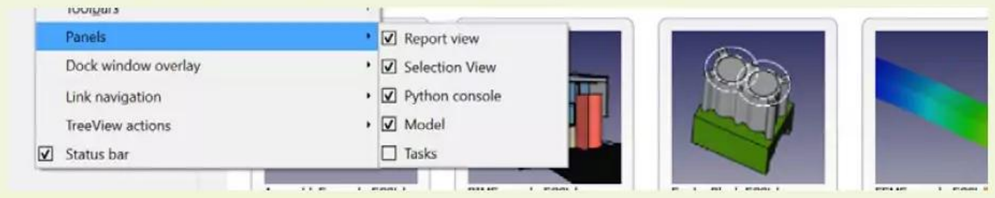
1. Reduce the number of pixels in the image to match the resolution of the mosaic dots we will print.
2. Reduce the number of colours to match the number of coloured filaments in the 3D printer.

QUICK TIP

You can improve the quality of your mosaics by simplifying source images and removing background distractions before turning them into mosaics.

FreeCAD is Python

Large parts of the FreeCAD program are built in Python. When you install FreeCAD, you also install a Python implementation for FreeCAD to use. The FreeCAD user interface includes a Python console.



▲ Select the 'Report view' option to see messages from FreeCAD including print output from your macros

The image above shows the bottom of the FreeCAD View menu. If 'Python console' is ticked, a Python console should be displayed at the bottom of the FreeCAD window.

```
Python 3.12.12 (main, Nov 10 2011, 15:00:00) [GCC 14.3.0] on linux
Type 'help', 'copyright', 'credits' or 'license' for more information.
>>> print("hello from FreeCAD")
hello from FreeCAD
>>>
```

▲ Code is written for a specific version of Python which is included in a FreeCAD installation

The image above shows the Python console in FreeCAD. You can enter Python commands at the terminal and run them. If you print something in the console, the output is displayed there.

QUICK TIP

Scaling an image before reducing the colour depth preserves the most image information.

We can use `Image` to perform both operations. The first thing we need to do is create an `Image` object containing the source image that we are converting into a mosaic.

```
INPUT_IMAGE = "/home/rob/Pictures/camera.jpg"
img = Image.open(INPUT_IMAGE).convert("RGB")
```

The first statement in the code above puts the path to the input image into a variable called `INPUT_IMAGE`. You will need to edit the path to match your picture source (unless you are called 'rob' and happen have an image called 'camera.jpg' in your Pictures folder). The second statement uses the `open` function in the `Image` module to open the image file and convert it into RGB format ready for processing. The `open` function returns a reference to the image it has created. The variable `img` refers to this object. Now we can make a rescaled version of this image.

Making resolutions

A digital image is made up of many picture elements, or 'pixels'. The more pixels, the better the image looks. A digital camera produces images that contain several million pixels (megapixels). We want to print an image that contains 128 × 128 pixels (or 0.0164 megapixels). The `img` we have just created provides a method called `resize` which changes the resolution of an image. You give it the width and height of the output, and it generates a new image with the specified dimensions.

However, we need to be careful not to change the shape of the original image. The ratio of width to height of an image is called the *aspect ratio*. This must be preserved during the scaling process otherwise things in the picture would look the wrong shape. We must calculate the aspect ratio of the original image and then multiply our desired width by this ratio to get the height of the scaled image.

```
src_w, src_h = img.size
```

The `size` property of `img` is a tuple (a Python data structure that can hold multiple values) containing the width and height of the image. The statement above extracts these values so that the program can work out the aspect ratio of the image.

```
aspect = float(src_h) / float(src_w)
```

The statement here calculates the aspect ratio from the width and height of the source image. Note that these are converted into floating-point values, which means that the value of aspect will also be a floating-point value – which is what we want. Now we multiply the requested width of our mosaic (128 dots) by the aspect ratio we have just calculated to give the height of the mosaic.

QUICK TIP

The camera image in **Figure 2** has an aspect ratio of 1.0 because it is square.

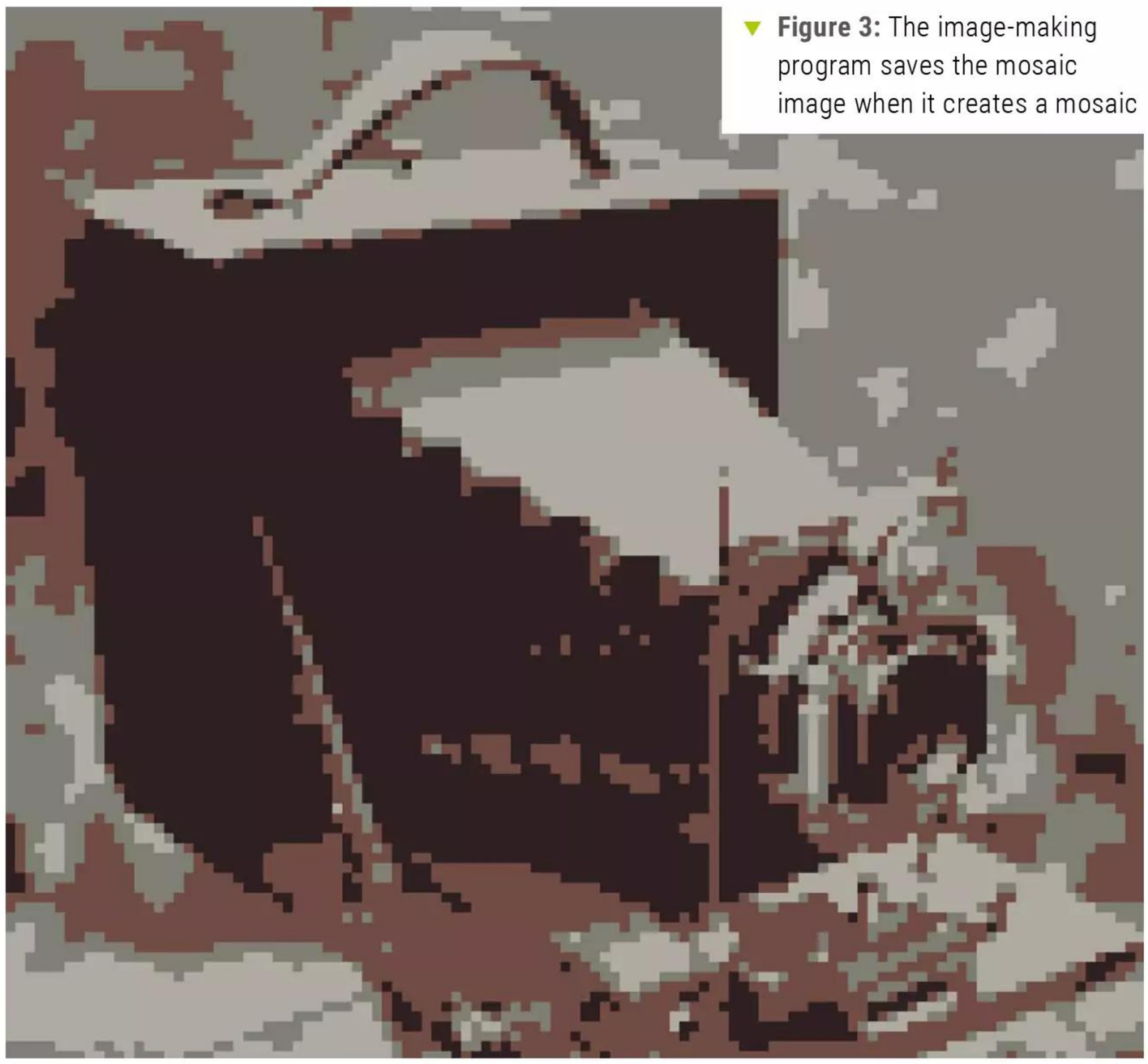
```
target_h = max(1, int(round(target_w * aspect)))
```

The statement above calculates the target height. It does a little extra work too. It converts the calculated height from a floating-point result into an integer (we can't have fractions of dots) and makes sure that we never have a height less than one. A one-pixel-high mosaic is not something that would be particularly artistically pleasing, but a height less than one would crash the program. Now that we have the width and height of the target, we can make a scaled image of the size we want.

```
scaled_img = img.resize((target_w, target_h),
Image.Resampling.LANCZOS)
```

The statement above calls the `resize` function on `img` to make a new image of the size that we want. The LANCZOS resampling method provides good-quality images quickly. Now that we have a low-resolution image, the next thing to do is reduce the number of colours in it.

The LANCZOS resampling method provides good-quality images quickly



▼ **Figure 3:** The image-making program saves the mosaic image when it creates a mosaic

Quantize mechanics

The `Image` module provides a `quantize` function that will reduce the number of colours in an image. This works on an image which contains red, green, and blue (RGB) values for each pixel. We need to make sure that our scaled image is in this format, and then call the `quantize` function on the image, telling `quantize` the number of colours that are required and how to process the image:

```
  quantized_image = scaled_img.convert("RGB").
quantize(
      colors=4,
      method=Image.MEDIANCUT,
      dither=Image.NONE)
```

The statement above reduces the number of different colours in the image to four. The `quantize` function is told to use the `Image.MEDIANCUT` algorithm, which works by splitting the colour space (the range of different colours in the image) into regions, picking the best colour for each region, and then changing the colours of the pixels in the image to the chosen values. The `quantize` function is also told not to dither the result. Dithering creates the appearance of additional colours by mixing pixels of different colours to simulate colour gradients. It only works well if the image is of much higher resolution than our mosaic, so we turn it off by selecting `Image.NONE`. The variable `quantized_image` is set to the result of the quantization process.

The quantize process does not create an image made up of RGB pixels. Instead, it produces a list of the colours in the image (which is called the *palette* or *colour table*) and an image of pixel values which map into the table. In the case of our quantized image, the table will contain four colours, and each pixel will have a colour value in the range 0 to 3. This can be a very compact way to store an image, but we want our program to be able to read the colour of each pixel and create pixels to be printed.

```
final_image = quantized_image.convert("RGB")
```

The above statement sets `final_image` to refer to an RGB version of the quantized one. Each pixel in the image now has an RGB colour value (although there are only four different colours in the image).

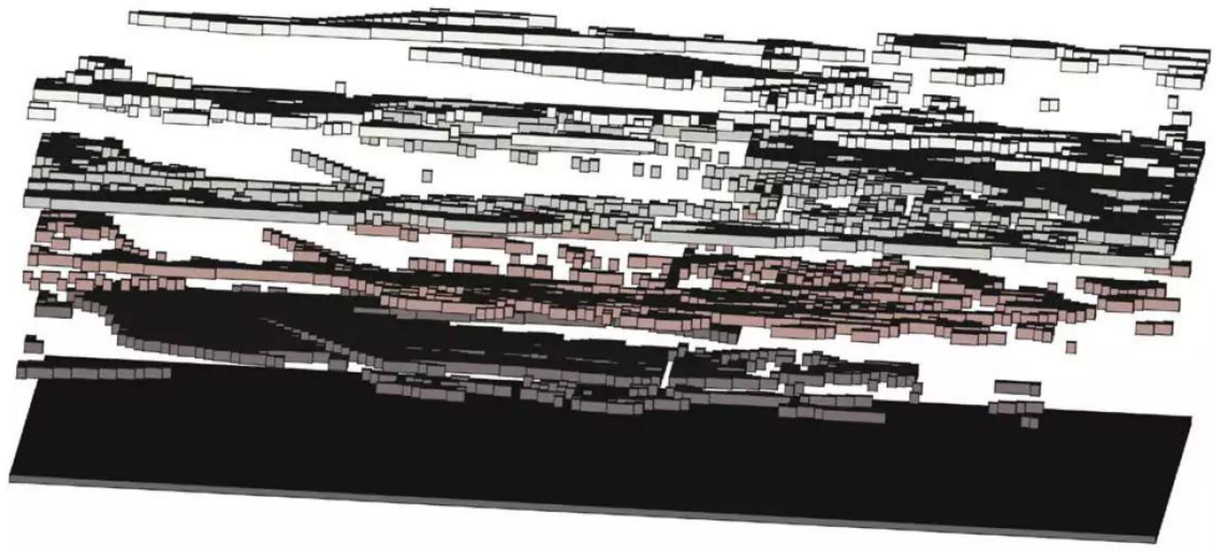
Figure 3 shows the result of applying these transformations to the original image. The background greens and browns have been converted to grey. The next thing to do is convert the image into printable layers.

QUICK TIP

You map the colours in the mosaic to the colours in your printer when you slice the design for printing.

Layer cake

Figure 4 shows an exploded view of what the mosaic program is going to create. At the bottom is the base layer, which will be printed in one of the four colours available. Above are four layers, one for each coloured filament. Each of these layers has been offset in the figure to make it visible. When the mosaic is printed all the layers are printed at the same level. The program is going to work through all the pixels and add a block to the colour layer that matches the colour of the pixel. To do this the program must first find out what colours are present in the image.



▲ **Figure 4:** The colours of the layers match the colours in the mosaic image

```
palette = final_image.getcolors(maxcolors=256)
```

The statement above sets the variable `palette` to the result of a call of `getcolors` on `final_image`. The `palette` is a list of data value tuples describing each of the colour values in the image. It might look like this:

```
[
    (1200, (255, 0, 0)),
    (800,  (0, 255, 0)),
    (800,  (0, 255, 255)),
    (400,  (0, 0, 255))
]
```

The first item in each data value is the number of times that colour appears in the image. The second item in the tuple is that palette colour in RGB form. The list above is telling us that the colour red appears 1200 times in the image. Our program doesn't care how often a colour is used in the picture, but it does want to know the colours. So, it does the following:

```
colours = [entry[1] for entry in palette]
```

The above statement works through the palette and builds a list of colour values extracted from each entry (remember that `entry[1]` refers to the second element in a palette value – which is the colour information). This is a neat way to use the `for` construction. You could write the statement like this:

```
colours = []
for entry in palette:
    colours.append(entry[1])
```

This code does the same job but is a bit longer. Now that we have our colours, we can use them to make a dictionary which will hold the mosaic layers we are building, indexed by colour:

```
solids_by_colour = {colour: [] for colour in
colours}
```

This looks a bit like the code we have just seen that built the `colours` list, but this time it builds a dictionary of lists, indexed by colour. What the program will now do is get the colour of a pixel, make a block to be 3D printed, and add that block to the list of solids for that colour. We do this by using a pair of nested `for` loops:

```
for y in range(depth):
    x = 0
    for x in range(width):
        pixel_colour = pixels[x, y]
        world_x = border + start_x * pixel_size
        world_y = border + (depth - 1 - y) *
pixel_size
        solid = make_box(
            world_x,
            world_y,
            z0,
            pixel_size,
            pixel_size,
            thickness
        )
        solids_by_colour[colour].append(solid)
```

The code above shows how this works. The program works through all the pixels, gets the colour of each pixel, works out where it is on the image, and then calls the function `make_box` to make a box which is added to the list of boxes for that colour layer. The final thing to do is make a single layer object out of all the boxes in the layer:

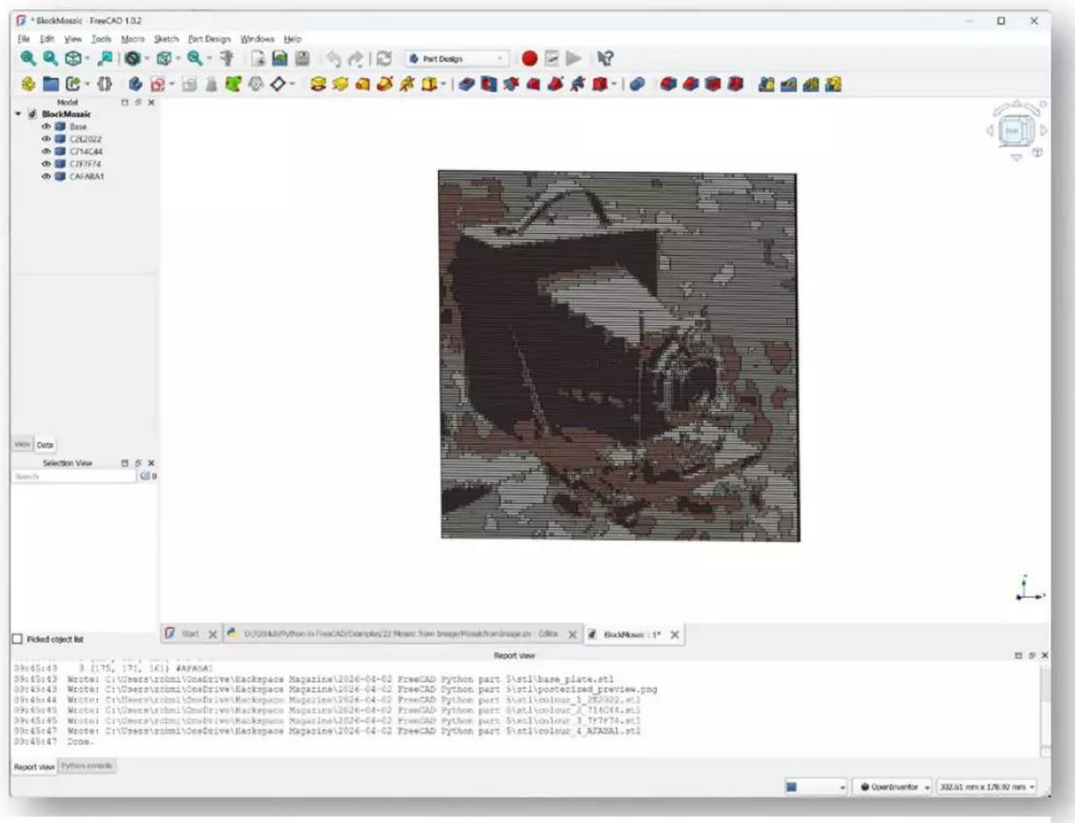


Figure 5: The program also outputs a file containing the posterised image

```
    colour_shapes = {}
    for colour in colours:
        solids = solids_by_colour[colour]
        colour_shapes[colour] =
Part.makeCompound(solids) if solids else None
```

The code above loops through all the colours, gets the solids for a colour and then uses `Part.makeCompound` to convert the list of solids into an object that can be printed.

Boxing clever

Figure 5 shows the finished mosaic in FreeCAD. The program generates five STL files. One is the base of the mosaic and the other four are the colour layers.

QUICK TIP

When the object is printed, all the different layers (and the base) must be imported into the slicer as a single object. There are instructions for how to do this in the sample code.

The code in the example program **MosaicFromImage.py**, which you can find in the GitHub repo for this article, is slightly more complex that the statements in the above explanation. It sorts the colours in order of brightness, so that when assigning filaments for a print, you can start with darker ones. It also runs together several pixels of the same colour on one line to make a long box, which makes the mosaic generation more efficient. The image resize and quantization operations are written as separate functions that you can reuse for other things. If you want to learn some neat Python tricks and tips, it is worth careful study.

With careful choice of filament colours, you can make impressive pictures

Puzzle time

Figure 6 shows an extension of the mosaic program which generates tile puzzles and a tray to hold them. You can change the size of the puzzle tiles to vary the difficulty of the puzzle. You can find this program in the sample code at **rpimag.co/pythonfreecad**. With careful choice of filament colours, you can make some very impressive pictures.

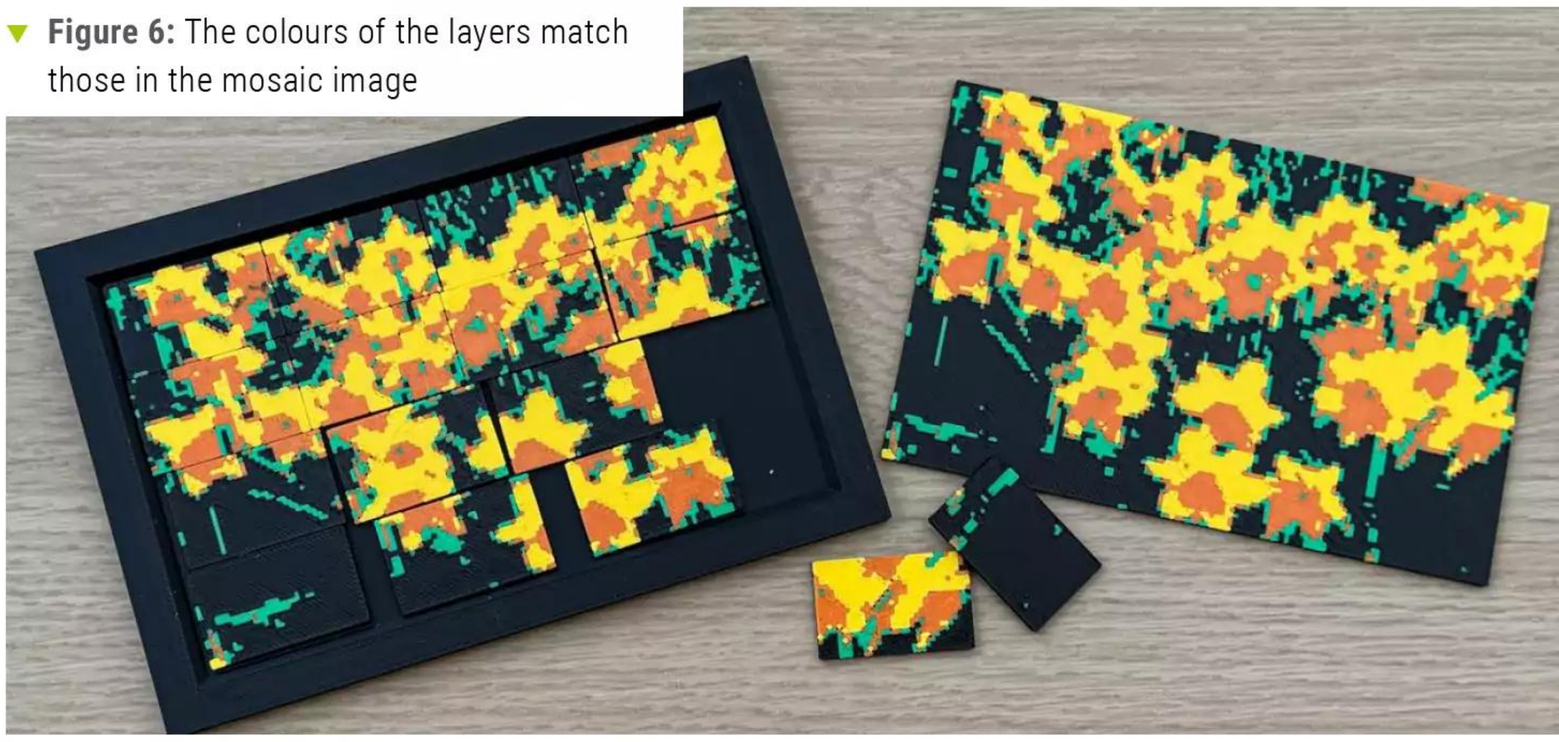


Figure 6: The colours of the layers match those in the mosaic image

Sew a sensory puppet

Make a puppet and then make it talk!



Maker

Nicola King

Nicola is a freelance writer and sub-editor. This month, she's gardening, sewing, and writing about puppets – just a lovely way to pass the time!

@holtonhandmade

QUICK TIP:

We found it much easier to hand-sew our fabric rather than machine it, as it was quite tricky to handle, and the machine would have been very unhappy!

Years ago, this author's sister made a very basic puppet out of an old sock and named him Pierre. She took him out on trips with our young son, and Pierre held his attention like no one else could. That may have been down to the French accent that she used, or ventriloquised, while Pierre was 'speaking' and this simple little puppet also, amusingly and memorably, mesmerised many other toddlers and babies when he was out and about. He was a really timeless little friend that our (now nearly grown-up) son remembers to this day.

The point of this little tale is that it really does not take a lot of effort, or indeed costly components to craft something that can have an impact, and puppets will arguably never lose their appeal to young and old alike. (Anyone else remember the TV show *Fingerbobs* from the 1970/80s with its very simple yet entertaining finger puppets?)

In this tutorial, we are going to make a rudimentary puppet and employ some basic tech that means it can speak... well, make some sounds anyway. So, you won't have to practise your ventriloquism techniques just yet – unless of course you want to do that too!

YOU'LL NEED

- Fabric for main body (e.g. faux fur)
- Felt for the mouth
- Cardboard or cuttable plastic for mouth baseplate
- A pattern/paper
- Scissors (paper and fabric)
- Pins
- Sewing thread
- Access to a sewing machine or a hand-sewing needle
- Googly eyes or safety eyes (lots of options here)
- Hot glue gun (and glue sticks)
- Yarn (for your puppet's hair)
- Ruler and fabric marker
- Any other embellishments (e.g. we used pom-poms for the eyes)
- Tiny FX (or FX W) microcontroller
- Mini speaker
- Push-button
- Battery pack or power bank



- Our puppet out in the wild, admiring our gardening skills

Types of puppet

Obviously, here we've made a simple hand puppet, where we use our hand to control the puppet's movements and, while we've made ours from fabric, others can be made from papier mâché, wood, plastic, or maybe rubber. However, there are other types of puppets, including but not limited to:

1. **Marionette puppets** – these are manipulated from above by strings/wires/threads that are attached to a control bar. This means the puppeteer (or marionettist) can move the puppet in more lifelike ways, although it is a very difficult skill to master. Often made of wood, the limbs are usually jointed to give more options for movement.

2. **Shadow puppets** – shadow puppetry is an ancient form of storytelling, using light and shadow, where intricately crafted puppets are used to project shadows or moving silhouettes onto a screen. The figures themselves are often made of leather or paper.

3. **Rod puppets** – these can often be very large and, as the name suggests, they are operated from underneath using a wooden or metal rod. A central rod is usually fixed to the head, and separate rods might be used for controlling arms, to give more movement possibilities.

Warning!
Sharp implements and glue
Children should be supervised.

We are going to make a rudimentary puppet and employ some basic tech that means it can speak

Puppet production

The first thing that you will need is some kind of pattern for the main body of your puppet, unless you plan on just repurposing an old sock as the body, in which case you can just adapt that to fit your purposes. You could design your own pattern, or find a free/paid one online: the choice is yours. We had a quick look online and found the perfect free pattern by Puppet Nerd that nicely met our needs: **rpimag.co/snoofpuppet**.

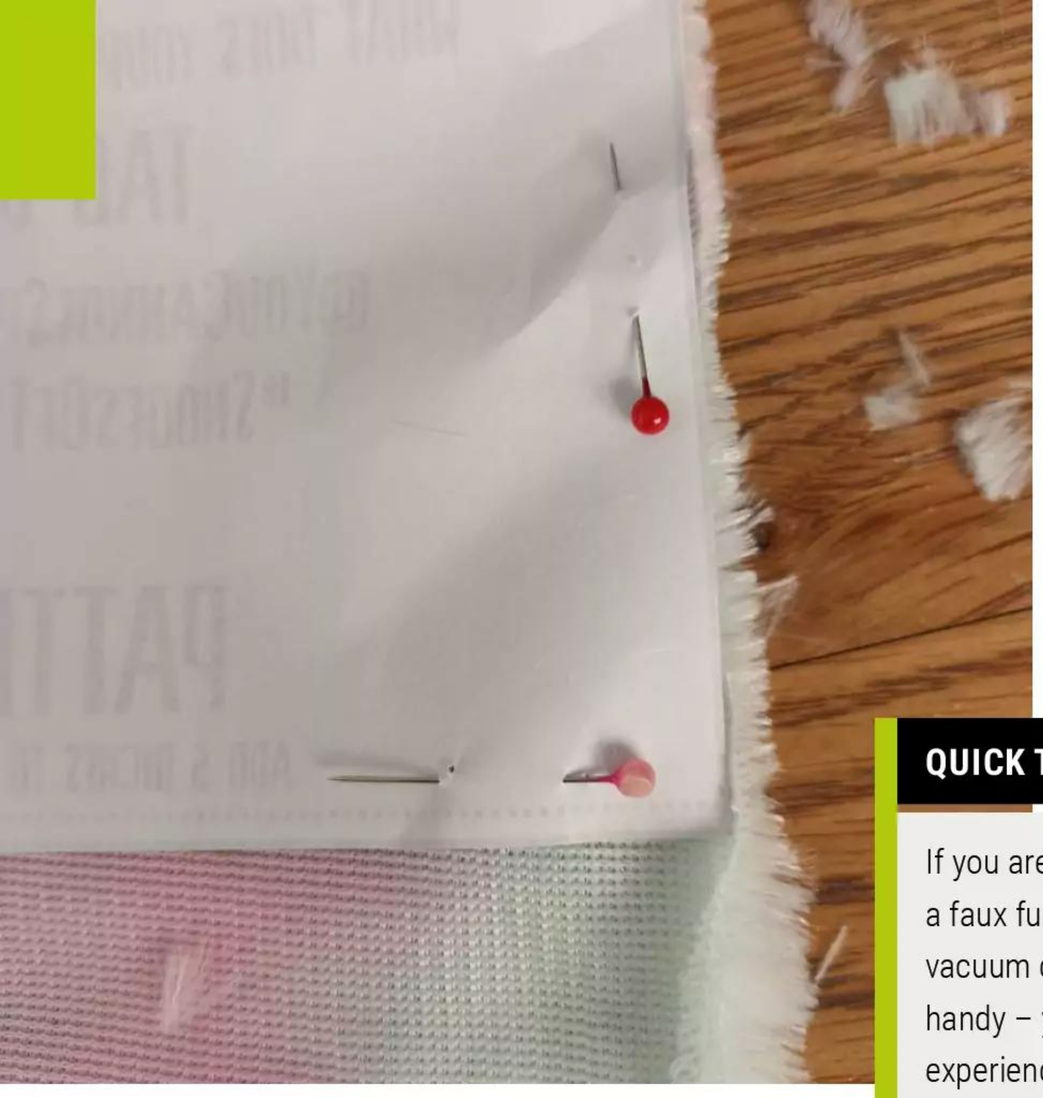
Paper pattern printed and ready to go, the next step is to turn your fabric wrong side up, and pin your pattern to it, cutting it out with fabric scissors. We also had to ensure that we flipped the pattern over when we cut out the other half of the body. We used a very fluffy faux fur which, while it looks great, was slightly more tricky to work with. For the mouth, we used a pink felt and cut out one piece. Then, right sides together, we sewed

QUICK TIP:

We're not aiming for any type of perfection here – just craft in some character!

- An inexpensive make, you'll probably be able to find most of your puppet parts in your stash





▲ Pinning the pattern to the material was quite important in this case because the material itself tended to shed

QUICK TIP:

If you are using a faux fur, have a vacuum cleaner handy – you may experience some shedding. We certainly did.

Premium puppet masters

If you'd like some further inspiration, take a look at these puppetry projects, some of which also incorporate a little electronic tinkering:

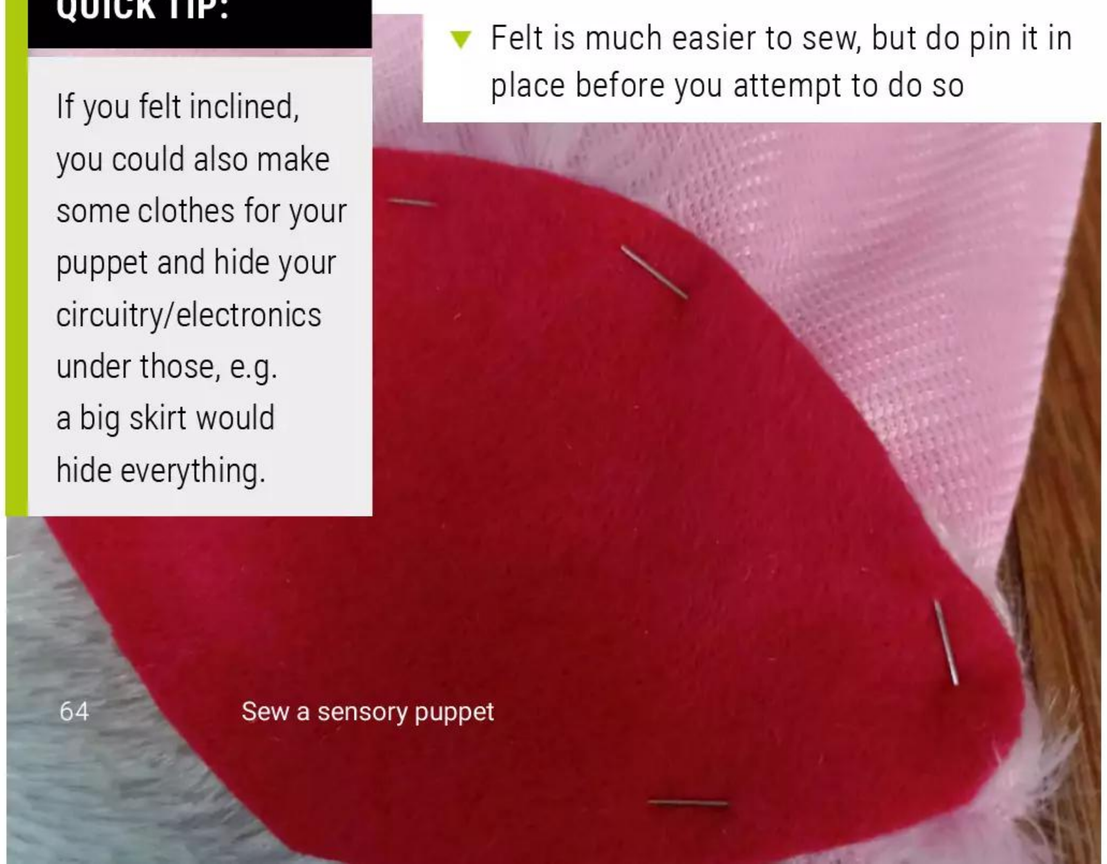
1. This maker really takes things to the next level, with a professional-looking and, we'd suggest, TV-ready-type puppet that has a real Muppet-like quality to it: **rpimag.co/qualitypuppet**. Clearly, this would take some time and a little more investment, but it might be worth considering if you want to take your puppet-making up a notch.

2. This Monk Makes puppet kit, using a Raspberry Pi, is now discontinued, but this does illustrate how you can use a traditional marionette puppet figure and tech together, controlling the puppet from your keyboard: **rpimag.co/monkmarionette**. Take a look and maybe think about how you could replicate it.

3. Puppetry projects are often going to appeal to younger makers, and this one is an e-textiles project that they will love: **rpimag.co/electropuppet**. Building a circuit couldn't be more fun.

QUICK TIP:

If you felt inclined, you could also make some clothes for your puppet and hide your circuitry/electronics under those, e.g. a big skirt would hide everything.



▼ Felt is much easier to sew, but do pin it in place before you attempt to do so

up each side using a basic backstitch, leaving the top open to add the mouth piece, and obviously the bottom was left open so that the user's hand can operate the puppet. You could seam the bottom for neatness if you wish.

It's also worth mentioning that, in addition, we cut out a rectangle of felt (around 8cm tall by 12cm wide), and hand-sewed this inside one side of the body as a pocket for the battery and electronics to sit in. This is optional, but will keep things neat and tidy. It's probably best to hand-sew this pocket in before you start sewing the body pieces together.

Next, sew the felt mouth in place while the body is still inside-out, again using a simple backstitch. We then cut out an additional mouthpiece in cardboard, cut it in half, and hot-glued each piece to the inside of the mouth to give more strength and structure. The pattern that we've adapted used a piece of plastic for this mouth-plate, but we found thick cardboard to be more than adequate.

Finally, embellish your puppet as you see fit. Really, the possibilities are endless here. See what you have lying around and just upcycle to add character. We used some super-chunky yarn for the 'hair' and hot-glued that on, and some pom-poms for the eyes.



◀ You don't need to worry about the sewing here looking pretty as the pocket is inside the puppet's body and your stitches won't be seen

Talking tech

Now let's turn our attention to the electronics. We need a way of detecting the mouth opening and closing. We opted for a simple push-button; when released, it triggers a sound. Since we needed a small controller board and speaker, we opted for an RP2040-based Tiny FX, as used in the singing bird tutorial in issue 159 (**rpimag.co/159**). This time, however, we connected an external button to its 'S' sensor port using a three-pin JST-SUR cable, with the sensor (GPIO 26) and ground pins wired to the metal legs on one side of the button. We powered the Tiny FX with a 3× AA battery pack, but you could use a USB power bank.

Using the tiny_fx library, the MicroPython program (as shown in **puppet_talk.py**) is similar to that for our singing bird.

puppet_talk.py

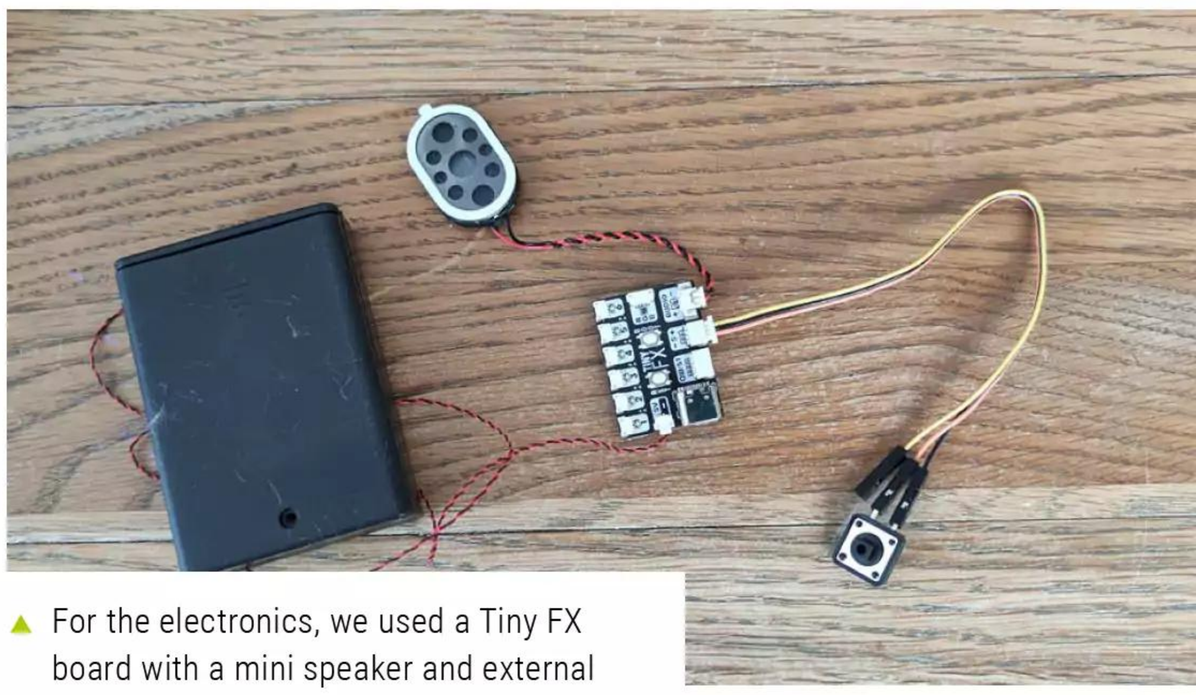
> Language: **MicroPython**

DOWNLOAD THE FULL CODE:

rpimag.co/github

```
001. import time
002. from tiny_fx import TinyFX
003. from machine import Pin
004.
005. # Create a new TinyFX object and point it to
     the files
006. tiny = TinyFX(wav_root="/Puppet")
007.
008. # Use a button connected to the 'S' sensor
     port
009. external_button = Pin(tiny.SENSOR_PIN,
     Pin.IN, Pin.PULL_UP)
010.
011. # Define a list of sounds
012. sounds = ["Puppet1.wav", "Puppet2.wav",
     "Puppet3.wav", "Puppet4.wav"]
013.
014. idx = -1 # Set list index number

015. try:
016.     last_state = False
017.     while True:
018.         # Read the pin and invert logic
     (Low/0 = pressed)
019.         pressed = not
     external_button.value()
020.         # Stop: triggered by button press
021.         if pressed and not last_state:
022.             if tiny.wav.is_playing():
023.                 tiny.wav.stop()
024.         # Play: triggered by button release
025.         if not pressed and last_state:
026.             idx = (idx + 1) % len(sounds)
027.             tiny.wav.play_wav(sounds[idx])
028.         # Update state for next loop
029.         last_state = pressed
030.         time.sleep(0.05)
031.
032. finally:
033.     tiny.shutdown()
```



▲ For the electronics, we used a Tiny FX board with a mini speaker and external push-button; you could use a different setup if you want, such as a Pico

This time the logic is inverted so that a sound plays only when the button is released (i.e. puppet's mouth opened), and stops when it's pressed again (closed). We placed some WAV audio files in a folder called **Puppet** on the Tiny FX, and added their file names to the `sounds` list in the code. Here the sounds are played in turn, although you could adapt the code to select them randomly. You can use any sounds for your puppet; we downloaded a 'burble' audio file from Freesound (**rpimag.co/puppetburble**) and then split it into smaller files with the Audacity audio editor (**audacity.org**). This worked well for our cute alien-like puppet.



▲ The Tiny FX board and mini speaker can be hidden away in the inner pocket

Final thoughts

Think of this project as a starting point and use it as inspiration to create your own characterful puppet with unique features. You can really go to town when it comes to facial characteristics, and upcycle things like ping-pong balls or marbles for eyes, or faux fur, feathers, fleece, or crocheted curls for hair, even add a zip to the mouth – really, you can be as creative as you like, and that's the fun of it. In the spirit of Pierre, who we referenced at the beginning, your puppet doesn't need to be particularly attractive either... and actually, that was a huge part of his charm.

QUICK TIP:

Instead of a push-button, you could use metal strips or conductive thread.

Make old cameras useful with 3D printing

Make light-tight film holders for use with 120-year-old cameras and then take pictures



Maker

Rob Miles

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

robmiles.com

YOU'LL NEED

- An ancient camera that takes 'quarter plate' sized images
- An old Fujifilm Instax Wide camera; it doesn't need to take pictures, so long as it can eject photographs
- Packs of Instax Wide film
- A 3D printer
- Some black 3D printing filament
- Some 4mm and 8mm M2 bolts
- A film changing bag where you can load and unload the film holders

Photography is surprisingly ancient. People have been taking photographs for nearly 200 years. Modern digital cameras can create superb images in the most challenging of conditions, but they are based on the same principles used to make the very first photographs. In this article we are going to take a "state of the art" camera from 150 years ago and discover how we can use 3D printing to make it usable today. You can find all the designs in the repository at **rpimag.co/RobMiles3DPrintCamera.**

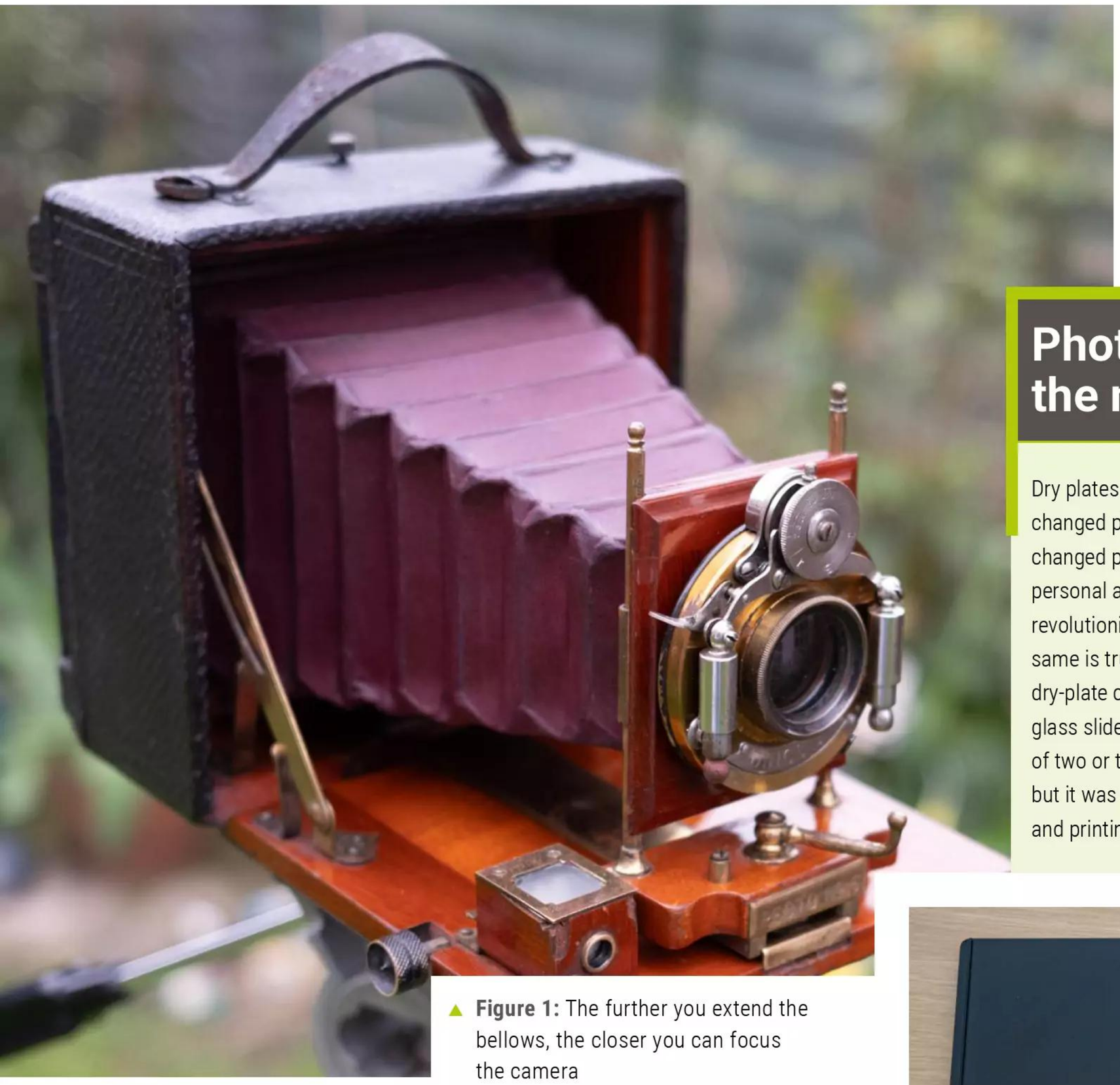
Figure 1 shows the Columbia 'Pecto No. 5' camera bought by the author at a camera fair. It is a quarter-plate camera made around 1897. It has light tight leather bellows, a shutter that goes 'click', and a clean lens. These are things to consider when choosing a vintage camera, along with whether your partner will let you buy it.

Wet and dry

The Pecto No. 5 exposes pictures onto 'dry plates'. These were a big improvement over the 'wet plates' used by very early photographers. As the name implies, wet plates were coated with liquid photographic emulsion just before use and then developed immediately after exposure. The wet plate coating and developing process used chemicals which required careful handling. Dry plates can be exposed and processed later, using much friendlier chemistry. The first dry plates were glass slides coated with emulsion. These were exposed and developed later to create negative images which were contact-printed onto photographic paper.

Photography for the masses

Dry plates like the ones used in the Pecto No. 5 camera changed photography in the same way that transistors changed portable electronics. The transistor radio was more personal and portable than its valve-based predecessors and revolutionised the way that we consume entertainment. The same is true of the dry-plate camera. Anyone could buy a dry-plate camera and start taking pictures. A single shot on a glass slide, developed and printed, would cost the equivalent of two or three pounds in modern money so it was not cheap, but it was affordable, particularly if you did the developing and printing yourself.



▲ **Figure 1:** The further you extend the bellows, the closer you can focus the camera

The 3D printed film holder is loaded with a sheet of Instax film

Nowadays it is hard to get hold of the dry plate slides, but the fundamental function of the camera – focus an image on a light-sensitive medium through a shutter – remains the same. So, we should be able to adapt the camera to work on other media. The author decided to turn the camera into one which could use instant films.

Instant photography has also been around for a long time. Polaroid released its first instant camera in the 1940s, around 50 years after the Pecto No. 5 was made. Then, around 50 years after that, at the end of the last century, Fujifilm started selling Instax instant film. Instax film is particularly suited to old cameras because it is exposed on one side and viewed from the other. The camera lens projects an image onto the film that is upside down and mirror-reversed. Because Instax is viewed from the opposite side to the one exposed, that reversal is reversed again, giving a correctly oriented result.

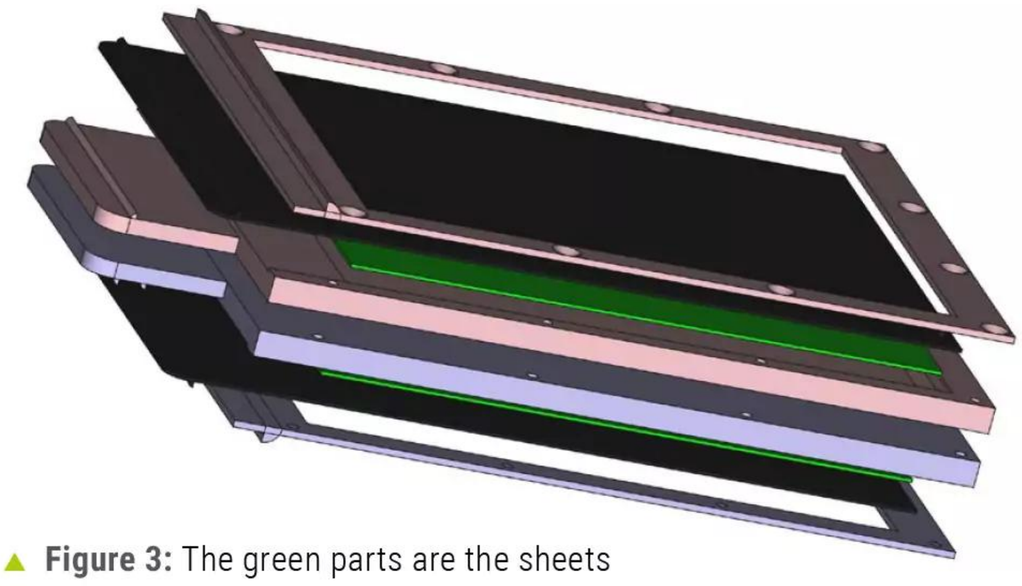


▲ **Figure 2:** The original film holder still has a glass negative slide in it

Getting holder

The camera was sold with a set of wooden film holders. A photographer of the time would go into their darkroom and load film slides into the holders, one glass slide on each side of the holder. Then, when they wanted to take a picture, they would put the film holder into a slot in the back of the camera, pull out the 'dark slide' which covers the film, and take the picture. They would then replace the dark slide, flip the film holder over, and take one more shot. Then they would take the film holder back home to develop the images.

Figure 2 shows the 3D printed film holder at the top and an original one below. The dark slides have been opened to show the film inside. The 3D printed film holder is loaded with a sheet of Instax film.



▲ **Figure 3:** The green parts are the sheets of Instax film; the black parts are the dark slides

QUICK TIP

You should only remove the dark slide when the film is in the camera or a dark room, otherwise the film will be exposed to light and ruined.

The author was hoping that they would be able to load Instax film directly into the original film holders, but it turns out that Instax film is slightly too deep to fit in the holder. So, a 3D printable film holder was designed.

Figure 3 shows the components of the film holder. It is made up of two single film holders (pink and lilac) bolted back-to-back using M2 bolts. The bottom set of bolts is longer and goes up through the bottom film holder and tightens into the top holder, holding them together. The design means that there are no overhangs in any of the components, making it much easier to print. The dark slides are sandwiched between the two parts of each holder and held in place by friction.

To start taking pictures, you first need to load the film holders with Instax film. Instax film is sold in plastic magazines for use in Instax cameras. Fortunately, it is easy to extract sheets of film from a magazine and load them into our film holders. Unfortunately, you must do this in the dark, but you can use a dark bag for this. You load everything in the dark bag and put your hands into the bag through a pair of light-tight cuffs. The author finds it easier if he closes his eyes while working in the dark bag.

Self-developing pictures

A sheet of Instax film contains all the chemicals required to develop the picture. When a sheet of film is ejected from an Instax camera, it passes between two metal rollers that squash open pods down one edge of the print that contain chemicals and dyes. These spread through layers in the print from the negative layer, which captures the camera image to the dye layers that form the picture we see. It is important that unexposed Instax film is handled very carefully to avoid bursting the pods.

Aiming the camera

Figure 4 shows how you aim the Pecto No. 5 camera at a subject. The camera contains a sheet of ground glass in the place where the film will be when the picture is taken. The camera lens projects an image onto the ground glass, and you use this image to focus and compose your picture. You do all your composition with the lens aperture (the size of the hole that lets in light) set to show the brightest possible view, but once you are happy with the picture, you need to adjust this and the shutter speed to make sure that the film will receive the correct exposure.



◀ **Figure 4:** The viewfinder view is upside down and swapped left to right, which takes some getting used to

Setting exposure

The Instax Wide film we are using has a film speed of 800 ASA. The bigger the film speed number, the more sensitive the film is to light. Too much light and the picture will be too bright. Too little light and it will be too dark. You can get exposure values by pointing a light meter at the scene. You tell the meter the speed of the film you are using, and it will tell you the shutter speed (the time the film should be exposed to light) and the aperture (the size of the hole that the light comes through). The shutter speed is expressed in fractions of a second. The aperture is expressed as a mysterious value called the 'f-stop'. Once you have these values, you can use them to set up the camera.

QUICK TIP

There are lots of free light meter apps available for your phone.



▲ **Figure 5:** The shutter button is on the far left of the controls



▲ **Figure 6:** Forgetting to slide open the dark slide is both embarrassing and expensive

If you have used analogue cameras before, you should find the controls (**Figure 5**) hold no surprises. The dial at the top sets the shutter speed from one second to 100th of a second. The scale at the bottom sets the f-stop. The bigger the aperture value, the smaller the hole. Each successive aperture lets through half as much light as the previous one. In modern cameras, the aperture number sequence is 8,16,22,32, but for this old camera the aperture value doubles for each successive step.

QUICK TIP

Tell your light meter the shutter speed you are using and it will suggest an aperture for you, and vice versa.

Once you have composed and focused your picture and set the exposure, you are ready to take your shot.

Taking the shot

When you want to take a shot, you close the shutter, slide a film holder in front of the ground glass, pull out the dark slide as shown in **Figure 6**, fire the shutter, and close the dark slide. Then everything goes back into the dark bag, the film is removed from the holder and loaded back into the Instax magazine. The Instax magazine is then loaded into an Instax camera and a 'photo' taken with the lens covered. This causes the film to be ejected and processed, producing the picture. It is best to practise these steps before using actual film. It is also a good idea to develop one shot first and check the exposure before taking any more pictures. Instax film is amazing, but it is also very fussy about exposure and so you should take care that your exposure is sorted before taking too many shots.

Figure 7 shows the results of the photograph. The image is nice and sharp left to right and shows very good separation from the background. The author was surprised just how good these images look.

Say cheeeeeeeeese

The process of taking pictures with these cameras is a lot more long-winded and prone to error than whipping out a smartphone and grabbing a shot. But it really forces you to think hard about the kind of picture you want to end up with, which can lead to much more considered photographs. The author really enjoys using the camera and is considering buying a Victorian frock coat and top hat to wear when taking pictures with it.



▲ **Figure 7:** Black and white film can be used interchangeably with colour

QUICK TIP

Keep a log of meter readings and camera settings for each shot so you can learn whether your camera shutter has accurate timings and come up with compensation factors for future pictures.



Bulb mode

To remote-control the camera, you use a rubber tube with a bulb on the end. The author used 6.5mm laboratory tube and an air blower bulb, with a cable tie to help it fit on the camera. This is attached to a tiny piston on the shutter. Squeezing the bulb sends a pulse of air that pushes the piston up and triggers a photograph. This is the origin of the 'Bulb' mode setting on cameras.

▲ You have to give the bulb a fairly hefty squeeze because the piston leaks a bit

Part 10

EDVAC, UNIVAC, & Princeton IAS

The computer that never was, and its offspring



Maker

Tim Danton

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

dantonmedia.com



Robert Oppenheimer and John von Neumann in front of the Princeton IAS computer, 1952
Image: Public Domain

This story of the EDVAC, UNIVAC, and Princeton IAS continues from issue 165.

Both Eckert and Mauchly had had an easy option, especially if they wished to stay working together, as they had both been approached by IBM about setting up a computing department there, and they also knew they had built enough of a reputation to start their own company. Eckert was in favour of playing it safe and taking the IBM job, but a combination of Mauchly and Eckert's first wife (who didn't think he would be happy at IBM) meant that they decided to go their own way.

Eckert, however, always had his reservations. "I didn't think we'd have enough money for this development [of new computers] and as it turned out I was right," he told Stern. "I also think that it was probably a mistake [to go it alone] and it wasn't a mistake. It's healthier for the United States to have a UNIVAC." In 1980, at the time of the interview, UNIVAC computers were still selling in volume and giving IBM healthy competition. Even today, it echoes in the name of global services company Unisys.[1]

The pair's first contract came, indirectly, through the Census Bureau. Eckert and Mauchly had visited them several times over the previous months, but on behalf of the Moore School. The bureau were interested in the idea of an EDVAC-style digital computer, but couldn't commission it directly themselves – they were only allowed to order finished products, not those in development – and so the contract went via the Army Ordnance Department. But even with funds approved, things weren't straightforward: as always with such contracts, the army used

1 Unisys was the name given to the organisation formed when the Burroughs Corporation and the Sperry Corporation merged in 1986, with Sperry having previously been Sperry Rand. The official reason given for the name Unisys is that it shortens the words 'united, information, and systems', and was suggested by Christian Lee Machen. But we know the real reason.





▲ A complete BINAC system; Albert Auerbach, who ran the first test routine on it, is seated at the rear
Image: courtesy of the Computer History Museum, CC BY-NC-SA

an independent expert to assess the proposal. In this case, George Stibitz from Bell Labs.

His response was lukewarm. "There are so many things undecided [about the proposal] that I do not think a contract should be let for the whole job," he wrote to John Curtiss, assistant director of the National Bureau of Standards (which was itself helping the Census Bureau assess the proposed computer's feasibility, such were the layers within layers of government contracts). But Stibitz offered the lifeline that "their suggestion seems promising enough to let a contract to study the problem, leading to a solid proposal and schematic".[2]

The mercury line

Curtiss decided to largely follow Stibitz's advice, opting to award the young-ish entrepreneurs – Mauchly was 39, Eckert a fresh-faced 27 – an initial "study contract" of $75,000 in June 1946. This was meant to cover the creation of a scale model computer with two mercury delay lines "complete with associated pulse shaping and regenerative circuits" plus a tape transport mechanism.

The contract was for an "EDVAC-type machine", with the name of the new computer formally becoming UNIVAC, standing for Universal Automatic Computer, in May the following year.[3] On successfully completing this phase, the Census Bureau would award a second fixed fee of $169,600 to complete the full computer. That brought the total to roughly half the minimum Eckert and Mauchly estimated the first UNIVAC would cost to build: between $413,000 and $671,000.[4] Their plan, 'hope' may be a better word, was to absorb the initial loss in the belief that more orders would follow. Plus, they would hold all patents stemming from the work.

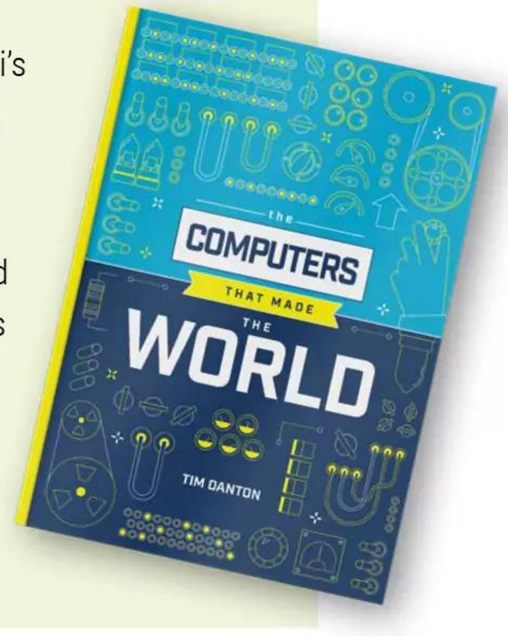
UNIVAC, standing for Universal Automatic Computer

2 Nancy Stern, *From ENIAC to UNIVAC*, p105
3 All quotes in this paragraph are from p106 of *From ENIAC to UNIVAC* by Nancy Stern, p106
4 Arthur L Norbert, 'New Engineering Companies and the Evolution of The United States Computer Industry', in Business and Economic History, Vol 22, No 1, Fall 1993, Business History Conference, ISSN 0849-6825, p186

The Computers that Made the World

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

rpimag.co/tctmtw



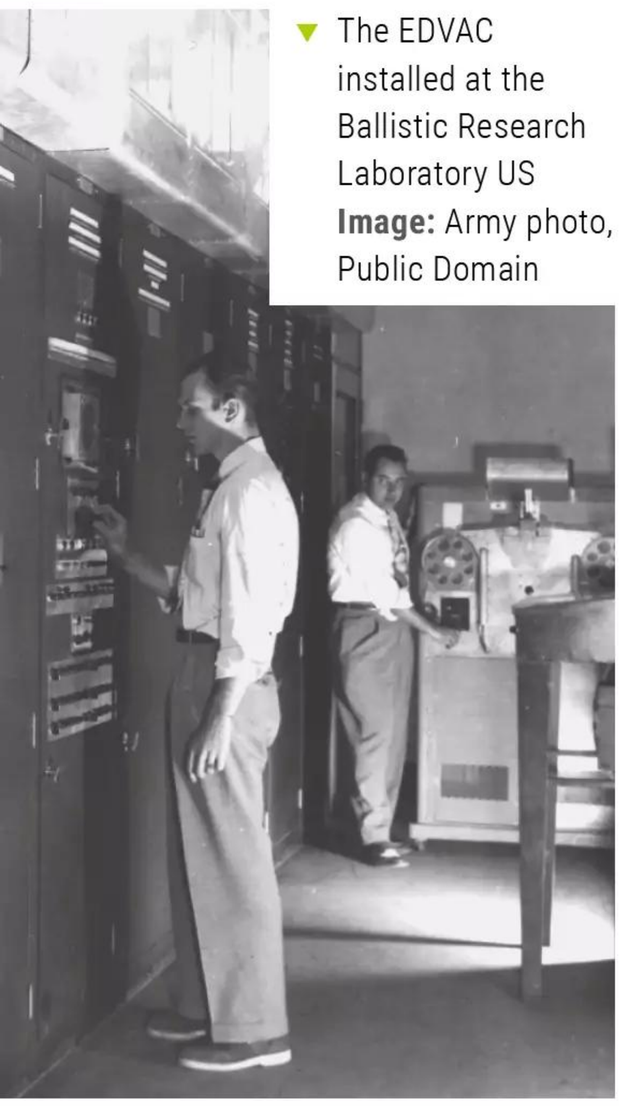
With $75,000 in the bank – although the contract only officially came into effect in October – Mauchly and Eckert created their business, initially titled the Electronic Control Company. They hired two floors of a building in downtown Philadelphia and started hiring engineers and programmers.

In today's money, $75,000 translates to roughly $1.25 million. Without more contracts, the company was going to run out of funds long before they built the proof-of-concept computer, so clearly they needed to find money from elsewhere. The answer, of a kind, came from Northrop Aircraft. It was developing a long-range guided missile for the US Air Force and needed a computer to guide its in-flight navigation.

Northrop ultimately wanted "a compact, airborne computer", with one of Northrop's stipulations being that the computer must be compact enough (less than 20 cubic feet, 566 litres) to fit through bomb bay doors.

The price is wrong

The price Northrop was willing to pay for this ambitious machine? $100,000, with $80,000 upfront and $20,000 on delivery. It would ultimately cost almost $300,000 to build,[5] and there's every chance that Eckert and Mauchly suspected as much when they agreed the deal in October 1947. But it gave them an injection of funds – and, they hoped, could also act as completion of the study for the Census Bureau, releasing the much-needed balance of $169,600. Whether they believed they could complete the system by the stated end date of 15 May 1948 is a different question altogether.



▼ The EDVAC installed at the Ballistic Research Laboratory US
Image: Army photo, Public Domain

If you're beginning to have doubts over Eckert and Mauchly's business acumen, you're not alone. Isaac Auerbach was the Electronic Control Company's first employee who hadn't worked on the ENIAC, joining the engineering team right at the start. "Neither Eckert or Mauchly in my opinion were competent managers, competent leaders, or competent executives, or understood business at all," he said.[6] "They were visionaries, and they were brilliant technically, and they would not let somebody else run the side of the company in which they were inept."

The contract with Northrop, which would eventually produce the BINAC (short for Binary Automatic Computer), is compelling proof for Auerbach's argument. On one hand, it was a triumph: the first stored-program electronic digital computer to operate in the USA. On the other, a loss-making failure, as it never performed to the satisfaction of Northrop.

The BINAC gamble

This wasn't the only ambitious part of BINAC. It was actually two computers in one, with calculations being simultaneously run through two separate arithmetic units: that way, if there was a difference in the answers, something had gone wrong. That meant two separate storage systems, each with a 512-word capacity, plus separate power supplies. The only thing they shared was a decimal to binary converter and their input/output mechanism, which could be either electronic typewriter or magnetic tape.

The latter was another first for a computer and an important one in historical terms: magnetic tape remains important as a form of data storage even today (most often as a backup medium). As is often the case with first attempts, though, it did not work well, and would be eventually replaced by "metallic tape plated with nickel", according to historian Nancy Stern.[7]

The BINAC was a small computer by ENIAC standards. Each processor included 700 valves, giving a total of 1400 versus 18,000 for the ENIAC, and stood "five feet high, four feet long, and one foot wide" according to the press release[8] that celebrated its official arrival on 22 August 1949.

This was also the day Northrop formally accepted the machine, more than a year later than the contract stated. So not only had Eckert and Mauchly grossly underpriced the BINAC, they had also underestimated how long it would take to build. Then again, this is hardly unusual: even today, IT projects are legendary for costing far more and taking far longer than originally estimated.

Despite its tardiness, the BINAC scored an important first by being the first stored-program electronic computer to run successfully in the United States. On 7 February 1949, Albert Auerbach[9]

5 To be precise, $278,000 according to Nancy Stern, *From ENIAC to UNIVAC*, p123

6 'Oral history interview with Isaac L Auerbach' by Bruce H Bruemmer, 2-3 October 1992, Charles Babbage Institute, University of Minnesota, **rpimag.co/auerbachinterview**, p5

7 Nancy Stern, From ENIAC to UNIVAC, p120

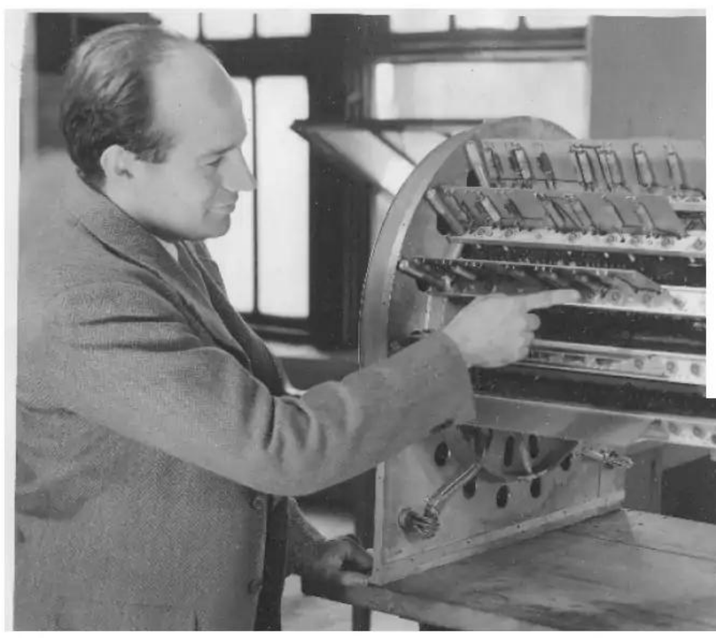
8 Press release as reprinted in a collection of papers ('The Albert A Auerbach Collection'), **rpimag.co/binaccollec**

9 No relation to Isaac Auerbach

ran a test routine on the computer,[10] following that up with a fuller 23-line program to compute squares in March.

It isn't mentioned often enough, but much of the programming work for the early computers – including the BINAC and UNIVAC – was done by women. Betty Holberton was one of the first Electronic Control Computer employees, even working for free at weekends before Eckert and Mauchly had money to pay her. That was in early 1946, before the money from the first government contract arrived. Her enthusiasm hadn't dimmed by the time the BINAC was created: "I didn't leave the company. I was on the machine 16 hours and eight hours off and I slept in the ladies' room," for two straight weeks, she said.[11] "I was working on, essentially, the simulation of the sorting system that was to be used on the UNIVAC I in calculating [tape buffer delays]."

> *Much of the programming was done by women*



◀ Presper Eckert demonstrating a BINAC memory unit, c.1948-9 **Image:** courtesy of the Computer History Museum, CC BY-NC-SA

Eckert and Mauchly had always hoped that by being the first stored-program computer in the USA they would attract more business, and the press release wasn't afraid to shout of its achievements. "The new computer showed its mettle by solving 'Poisson's Equation,' a typical engineering problem," it stated.[12] "BINAC spent more than two hours of actual computation to obtain 26 solutions. For each solution the computer did 500,000 additions, 200,000 multiplications, and 300,000 transfers of control, all in the space of 5 minutes. A man with an adding machine would have [needed] years to complete the same job."

A train departs

The computer was ready for shipping to Northrop, a journey of several hundred miles by train. As per the terms of the contract, Northrop had responsibility for packing it up securely, and this is where – perhaps – things started to go wrong. Whether the quality of construction was too low, it was packaged poorly or a combination of the two, it took several months of work before the Northrop engineers would obtain usable results from their new computer.

Rumours that it never worked at all were contradicted by two former Northrop employees, who wrote in to the IEEE Annals of the History of Computing in reaction to an article by Nancy Stern. Florence Anderson pointed out that at that early "stage of computer development ... the amount of time required for maintenance, the quality of components, and the physical environment were all greatly underestimated. For example, BINAC seemed to operate well on sunny days, but poorly on rainy days".[13] Jerry Mendelson was even more definite: "I can state categorically... the BINAC did run successfully after it was installed in Hawthorne, California. I know this to be a fact because a colleague of mine named Bob Douthitt and I made two absolutely successful uninterrupted runs, each in excess of 25-minute duration."

To the outside world, the BINAC was a success, and it also served as an important landmark in the company's contract with the Census Bureau. Finally, that $169,600 was theirs, even if it didn't cover Eckert and Mauchly's outgoings to date – not to mention the fact they still had to build the UNIVAC itself.

For now, we will leave Eckert, Mauchly, and the rest of the team, and travel back to April 1946 and the Moore School. The brains trust that created the ENIAC and designed the EDVAC – not just Eckert and Mauchly, but also Burks, Goldstine, and von Neumann – have all left. What's left is a contract and the outline of a design. Fortunately for Irven Travis, he not only had a handful of ENIAC old-timers, most notably Kite Sharpless and Carl Chambers,[14] but also Moore School alumni who were returning from the war.

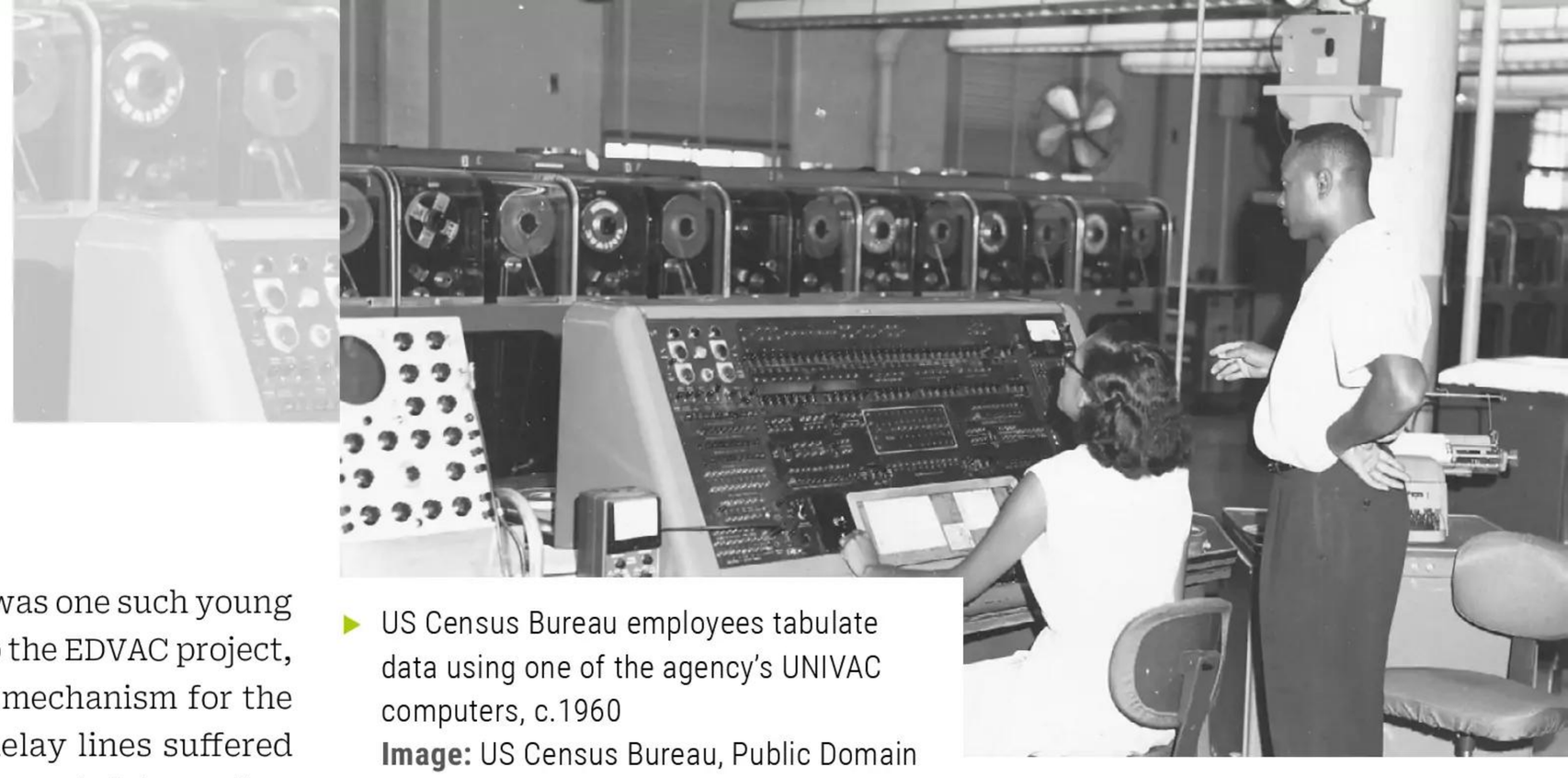
10 Herman Lukoff, *From Dits to Bits: A Personal History of the Electronic Computer* (Robotics Press, 1979, ISBN 978-0896610026 – the 'Dits' refers to his early passion of transmitting the dits and dahs of Morse code), p84

11 'Oral history with Jean J Bartik and Frances E (Betty) Snyder Holberton' by Henry S Tropp, 27 April 1973, Smithsonian National Museum of American History, **rpimag.co/bartikholbertoninterview**, p110

12 Press release as reprinted in a collection of papers ('The Albert A. Auerbach Collection'), **rpimag.co/binaccollec**

13 IEEE Annals of the History of Computing, Vol 2, No 1, Jan 1980, p83

14 While Harry Huskey was still working at the university, at this point he was an assistant professor in the Maths department. In June 1946, he was offered the chance to head the EDVAC project and would have taken it, but no-one had consulted with the chair of the department who demanded they withdraw the offer. They did and Huskey resigned. As we'll cover in the next article on the Pilot ACE, he spent a year in Manchester after being invited by Douglas Hartree.



US Census Bureau employees tabulate data using one of the agency's UNIVAC computers, c.1960
Image: US Census Bureau, Public Domain

Herman Lukoff, who had just turned 23, was one such young veteran. Travis immediately assigned him to the EDVAC project, giving Lukoff the job of devising a control mechanism for the mercury delay line. Most early mercury-delay lines suffered terribly due to the way their delay properties varied depending on the temperature, with the common solution being to control the temperature. Lukoff took a different approach, varying the gaps between pulses so "that they could be spread or condensed to give the exact number of pulses required to fill the column".[15]

With the help of Steve Chen and Joe Chedaker, over the space of a few months Lukoff solved one of the biggest problems affecting mercury delay line storage: that the timings changed with temperature. Their creations were even ready for public demonstration, so with the help of a test assembly and an oscilloscope to show the pulses, Chen and Lukoff headed to New York for the Institute of Radio Engineers show in March 1947.

They set it up the night before the show started and it worked as planned. Then came the public demonstration. "No sooner was the equipment turned on than unusual things started to happen. The mercury memory would suddenly fill up with all kinds of extraneous pulses," Lukoff recalled.[16] The crowd wanted to know what was going wrong, and it was only when he noticed that the pulses were changing "rhythmically every two seconds" that he caught sight of "a rotating Army Signal Corps radar antenna halfway across the hall". Lukoff convinced the operator to switch off the power and the demonstration worked perfectly once more.

Logical problems

Unfortunately, the rest of the Moore School's EDVAC project did not run so smoothly. Without Burks, Mauchly, or von Neumann, they were relearning the fundamentals of computer logic design. Even the gifted Lukoff was struggling to get to grips with the problem. "Many months were spent on the paper design of the logic," he wrote.[17] "Each time we reviewed the drawings, we would invariably find some illogical condition we had not thought of before that required a redesign." Nor, according to Carl Chambers, was von Neumann's First Draft of any use, describing it as coming "from the point of view of a neurologist" and "of not much practical help".[18]

Amidst all this, one amazing, almost miraculous event took place: a special course entitled 'Theory and Techniques for Design of Electronic Digital Computers'. But it became universally known as the Moore School Lectures, and has reached legendary status for not only the wealth of information shared but the influence it went on to have. Over the course of 48 lectures, taking place at the Moore School from 8 July to 30 August 1946, it equipped the students with the information they needed to start building their own computers. Even if there were still many problems they would need to solve themselves – such as building mercury delay lines.

The lecturers read like a Who's Who of computing in America at the time. George Stibitz kicked things off with an introduction to the course where he explained why electronic digital computers were already so important, and were set to become more important in the future. Other early computer luminaries giving talks included Howard Aiken, John von Neumann, Herman Goldstine, Douglas Hartree, and Jan Rajchman. Perhaps surprisingly, given their abrupt departure mere weeks earlier, the bulk of the lectures were given by Mauchly and Eckert. Perhaps it helped that they were both given $1200 each for doing so, and at that point no money had yet arrived from the Census Bureau contract.

The list of attendees was no less impressive. There were senior people from the National Bureau of Standards, MIT, the Naval Research Laboratory, General Electric, not to mention Maurice Wilkes (who was so inspired by the conference he built the EDSAC) and Claude Shannon of Bell Labs. Officially the total came to 28, with invitations strictly limited, but we know that gate-crashers included Cuthbert Hurd of IBM and the MIT's Jay Forrester.[19]

The main reason for holding the event was that after unveiling the ENIAC to the world, the Moore School had been beset by calls from industry and government departments alike to find out more. This, Pender thought, was the most efficient way to share their learnings. Even better that the army and navy would jointly pay for it.

They recorded the lectures, with this unenviable job shared between Lukoff and his Moore School colleague Dick Merwin. "Wire recordings were not very foolproof; if you think rewinding tape on a reel that has spewed on the floor is a pain, try rewinding

15 Herman Lukoff, *From Dits to Bits*, p59
16 As above, pp61-62
17 As above, p62
18 Thomas Haigh, Mark Priestley, and Crispin Rope, *ENIAC in Action*, p147

19 Martin Campbell-Kelly and Michael R Williams (eds.), *The Moore School Lectures* (The MIT Press and Tomash Publishers, 1985, ISBN 978-0262031097), pxvii

fine wire without getting a kink in it," he wrote.[20] The results were painstakingly transcribed by Moore School secretaries before being turned into a published set of lectures in late 1947, with supplementary material and notes supplied by the lecturers where available.

Unfortunately for the Moore School's EDVAC project, this was a lonely bright spot in a difficult period of transition. Lukoff was eventually lured away by Eckert and Mauchly with the promise of more pay and better prospects, and it's telling that the same day he joined their company – 1 September 1947 – another Moore School colleague joined with him. Kite Sharpless and Carl Chambers had also departed in recent months, leaving the project without a clear technical director until Richard Snyder appeared on the scene. He not only saw the project to its conclusion, but supervised its running in Aberdeen at the BRL.

Unlike the BINAC, this was a big computer

By this point, they were no longer technically building the EDVAC but the EDVAC 1.5. This name came about after a meeting between the Moore School and the BRL in October 1946 on exactly what they were going to build, which still hadn't been formally defined. The Moore School proposed three possible computers of increasing complexity, with EDVAC 1 a binary computer that could add, subtract and multiply, and with a 1000-word memory. The 1.5 came about because during the meeting they decided to add division in hardware (rather than being programmable) and adding internal checking for all arithmetic operations.

Then, seven months later, they reconvened and decided that a 1.5B version could be created with floating-point operations and an extended instruction set. The floating-point hardware was later dropped as it proved too complicated. Like the BINAC, they also decided that to detect errors they would need to build two identical arithmetic units to check against one another. And now, finally, in May 1947, they finalised the design, almost three years after the computer was first conceived.

Rather than attempt to build it at the Moore School, construction was subcontracted to the Reeves Instrument Corporation. In Michael Williams's 1993 article, 'The Origins, Uses, and Fate of the EDVAC', a footnote explains that the overconfident company thought it could build several copies and promised their first one to the Rand Corporation in return for $100,000.[21] They confidently predicted delivery in May 1948, but when a Rand representative arrived in April in the full expectation of seeing "EDVACs rolling off the production line" he discovered, well, nothing at all. Reeves quickly announced that it was exiting the computer business before it had even started.

It turned out that creating computers in the late 1940s was a lot tougher than first draft reports made it look, so it must have been with some relief that the Moore School – which did eventually build the EDVAC 1.5B for an estimated total cost of $467,000[22] – delivered its machine to the Aberdeen Proving Ground in late 1949. Unlike the BINAC, this was a big computer, consisting of a 2700lb (1.2-tonne) memory unit and eight further units (such as arithmetic units and controllers) weighing 850lbs (385kg). Not to mention a 4000lb (1.8-tonne) power supply.

At the time of delivery, it included around 3563[23] vacuum tubes and 50,000 feet of wiring, measuring 30 feet long by 14 feet wide. That's a little over nine metres and four metres respectively. There wasn't much room left in the room that housed it for people or the various pieces of add-on equipment that appeared over the years. We should also feel sympathy for the operator of the paper-tape input system – a late substitute for a magnetic wire system that never worked properly – who had to pull it by hand past the read head.

Williams-Kilburn to the rescue

But there was more bad news for the Army scientists who wished to operate the EDVAC: it wouldn't work smoothly for years. It ran its first proper program, as opposed to test routines, on 28 October 1951, some two years after it was delivered. "Even then," wrote Michael Williams,[24] "it took a further three months before it was considered reliable enough to run a large calculation – to find the eighteen values of a 12×12 matrix in January 1952."

Later that year the faster and more reliable ORDVAC, based on the Princeton IAS, was delivered, pushing the EDVAC into third position behind the still operational ENIAC. It was only



▲ James Pomerene working on the Princeton IAS machine, holding a Williams tube
Image: Wikimedia Commons, CC BY-SA 4.0

20 Herman Lukoff, *From Dits to Bits*, p60
21 Michael R Williams, 'The Origins, Uses, and Fate of the EDVAC', in IEEE Annals of the History of Computing, Vol 15, No 1, 1993, p29
22 As before, p30
23 The physical measurements and weights come from Michael Williams's article; the number of vacuum tubes from *The Computer as von Neumann Planned It* by Michael Godfrey and DF Hendry.
24 Michael R Williams, 'The Origins, Uses, and Fate of the EDVAC', p30

after that computer blew its metaphorical fuse during an electrical storm in October 1955 that the BRL focused their time and effort into improving the reliability and utility of the EDVAC. Which they did, and for the next five years it turned into something of a workhorse, operating productively for over 130 hours per week on average.

As the arrival of the ORDVAC gives away, the Princeton IAS computer had enjoyed a much smoother journey to completion. If the Moore School was a magnet pushing away its most talented staff, the IAS was its polar opposite dragging talented people in. By the middle of 1946 it had hired top engineers such as James Pomerene, Ralph Slutz, Willis Ware and, at the head of them all, Julian Bigelow. Not all of them had computing experience – how could they – but were all highly respected engineers.

Then, on the logic side, the IAS project boasted an all-star team of Burks, Goldstine, and von Neumann. Even before Burks and Goldstine formally set to work in March, von Neumann had thought in great detail about the principles and specifications of a next-generation computer. Keeping in mind the difficulty of the problems he wanted to solve, von Neumann specified the use of binary numbers with 40-digit accuracy and a capacity of 4000 words. He also specified the need for automatic checking facilities. He believed it could all be done with "1000 to 2000 off-the-shelf vacuum tubes" plus "50 to 100 special tubes", according to historian William Aspray.[25]

This was the outline von Neumann presented to Burks and Goldstine when they joined him at the IAS in March 1946. They couldn't have asked for a stronger intellectual head start, and this aura was only added to by the presence of pre-eminent scientists such as Albert Einstein and Kurt Gödel.[26] "Kurt Gödel didn't have a secretary, didn't want one, I assume," said Burks. "So for that summer, when of course we didn't yet have a building for the computer, Herman and I occupied the secretary's office next to Gödel's office. It had a blackboard on the wall. We spent most of our time the first few months planning this new machine, working out the structure and the instructions, and we would consult periodically with von Neumann."

The IAS project boasted an all-star team of Burks, Goldstine, and von Neumann

The first fruit of their labours came in the form of a report, published in late June 1946, entitled 'Preliminary Discussion of the Logical Design of an Electronic Computer'. Unlike the First Draft, which presented an abstracted, idealised version of how a computer might work (and was full of errors), this paper goes into prosaic detail. For example, it explains how a computer might use Selectron tubes for storage. Still, Neumann's love of anthropomorphism remains evident in the section titles of 'The Memory Organ' and 'The Arithmetic Organ'.

One point is worth labouring here. The trio wanted to share their findings, as is common in scientific literature; von Neumann also felt it was only right that, as government agencies were funding a big proportion of the work, there was no place for patent chasing by individuals. 175 copies of the report were printed and freely distributed to interested readers, with the promise of a second part of the report to follow (this would in fact follow as three separate reports).

Burks left at the end of August, as had always been planned, to take up a position as assistant professor of Philosophy at his alma mater, the University of Michigan (he became a full professor in 1954)[27]. But not before leaving 50 pages of notes on how one might program the computer, a subject covered by one of the later reports.

By this time, the engineering team was starting its work, but under Julian Bigelow's diligent direction they didn't rush things. "The period from June 1946 through June 1947 can fairly be described as one of engineering and organisation rather than actual design," he wrote.[28] Having decided to use readily available components whenever they could, they set out to test the options available (although they had to build the test equipment themselves). Another reason for caution is that as soon as the team crystallised around a decision, they would share that with five other groups around the country[29] in the form of engineering drawings. These groups could then build their own versions of the computer to the IAS design. Any mistakes would be multiplied, and at great cost.

25 William Aspray, *John von Neumann and the Origins of Modern Computing*, p63

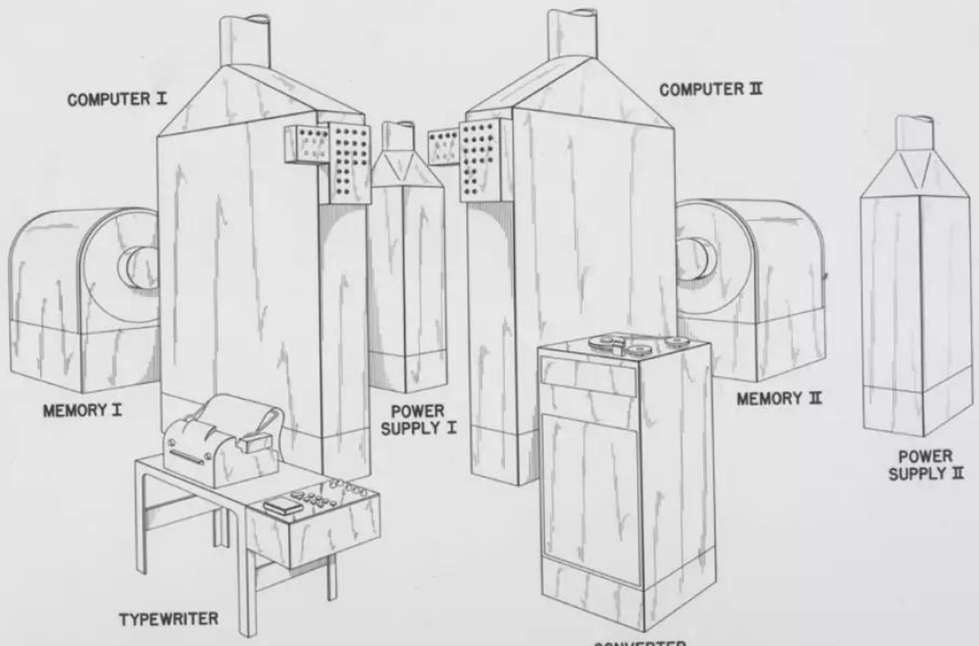
26 To paraphrase his Encyclopedia Britannica entry, Austrian-born mathematician, logician and philosopher Kurt Gödel is considered one of the greatest logicians since Aristotle "due to his incompleteness theorem which states that within any axiomatic mathematical system there are propositions that cannot be proved or disproved on the basis of the axioms within that system".

27 For an accurate potted biography of Arthur Burks, head to the University of Michigan's own description at **rpimag.co/burksbio**

28 Julian Bigelow, 'Computer Development at IAS Princeton', as printed in *A History of Computing in the Twentieth Century*, p294

29 These were the Los Alamos Laboratory, the University of Illinois, Oak Ridge National Laboratory, Argonne National Laboratory, and the Rand Corporation.





▲ A line drawing of a complete BINAC system, 1949
Image: courtesy of the Computer History Museum, CC BY-NC-SA

The engineers moved into their newly built lab in January 1947, and by mid-1947 they started to design the IAS computer's arithmetic unit. It's worth reading Bigelow's detailed – but not impenetrable – account of this work, which not only lists successes but also the blind alleys they went down and mistakes he personally made. It's not merely a technical delight, but a guide for project managers everywhere.

By spring 1948 they had finally assembled and tested the arithmetic unit to the point that it could be fully demonstrated to von Neumann. Unlike most mathematicians, he was a talented mental arithmetician, so enjoyed trying to outpace the arithmetic unit when given increasingly difficult sums. "The first few times he was right, but then we put in more complicated [numbers] with also a 'left end' carry, and it eventually happened that what he called out as the answer disagreed with the result shown by the adder," wrote Bigelow. "Upon rechecking, Johnny found he had made an error, and acknowledged the victory of matter over mind."[30]

But there was a problem: the RCA was struggling to make the Selectron, and this was key to the Princeton IAS design. For the computer to function as planned, it needed to have instant access to data, which was the key advantage of CRT storage technology over mercury delay lines (where you had to wait for each stored bit of data to emerge). The RCA made bold claims that it could store 4096 digits per tube, but no samples were forthcoming. The chances of it delivering 40 such tubes were looking slim to impossible.

The Princeton IAS team needed a backup plan, so – using the small workshop that formed part of their engineering lab – they started work on a magnetic drum that would be far slower than the Selectron but would at least act as a working data store. They also experimented with their own home-built CRT storage mechanism.

Perhaps both plans might ultimately have worked, but fortune smiled upon them in the form of the Williams-Kilburn tube being developed in Manchester. In June, they received a copy of Williams and Kilburn's report, and so Bigelow flew over to see the work for himself while his deputy Jim Pomerene worked on their own replica of the British design. "I can remember [Williams] explaining it to me," wrote Bigelow,[31] "when there was a flash and a puff of smoke and everything went dead, but Williams was unperturbed, turned off the power, and with a handy soldering iron, replaced a few dangling wires and resistors so that everything was working again in a few minutes."

When Pomerene reported that they had a trial tube working in the IAS lab, storing 16 digits, Bigelow headed back. Their challenge now was to not only devise a way to up that number from 16 to 1024, but to do so reliably across 40 CRTs. These were all 'off the shelf' monitors, but not all models worked as the phosphor layers needed to behave perfectly – in terms of electron emission – if they were to act as storage. The engineering team tested hundreds of potential CRTs in search of potential models, and by the summer of 1949 had their 40-strong collection of working storage tubes.

Now for the small matter of building a working computer. There was the physical component, with assembly performed by the workshop team, but also highly technical issues around timing. Only by January 1950 were they ready to start testing a fully assembled unit, with Bigelow describing the year as one of "extreme pressure for the IAS engineering group; our laboratory building was overflowing with applied scientists of all sorts".[32] Using genuine programs from nuclear physicists and meteorologists (the project had a whole section dedicated to meteorology), they honed their machine over the course of that year.

Over the following spring it was pushed into wider use, reaching the point where errors were more likely to come from the programmers than from the machine. And then in the summer of 1951 its biggest test: Los Alamos scientists ran a huge thermonuclear calculation that ran for 24 hours per day and for 60 days (with brief respites for them to check for errors, of which only six were disclosed).

"The engineering group split up into teams and was in full-time attendance and ran diagnostic and test routines a few times per day, but had little else to do," wrote Bigelow.[33] "So it had come alive."

And it wasn't only the Princeton IAS computer that sprang into being. Between 1951 and 1953, the Los Alamos National Laboratory built the MANIAC based on the IAS designs, the University of

30 Julian Bigelow, 'Computer Development at IAS Princeton', p302
31 As above, p304
32 As before, p307
33 As before, p308



▲ Vacuum tubes in a UNIVAC I computer
Image: Diomidis Spinellis, CC BY-SA 4.0

Illinois the ORDVAC (for the US Army) and the ILLIAC (for its own use), while the Argonne National Lab built the AVIDAC and the ORACLE. Even other countries got in on the action, with Australia, Denmark, Israel, Japan, and Sweden all creating computers closely based on the Princeton IAS computer.

Commercial companies weren't shy of taking advantage either. 1953's 'Defense Calculator', the IBM 701, was based on an adapted IAS design, while the Rand Corporation created the JOHNNIAC – in honour of von Neumann – the following year.

All these machines were notable for their longevity, with some carrying on their work into the mid-1960s. This meant they outlasted both the Princeton IAS computer, decommissioned in 1960, and most sadly of all John von Neumann himself. He died of bone cancer at the age of 53 in 1957, having left an indelible print on the world through his computing achievements – the computers we use today are still based on what is known as the von Neumann architecture – and an incredible body of mathematical work.

The legacy of the computing lab at Princeton IAS proved far shorter-lived. There had always been a tension between the purely academic leanings of the Institute for Advanced Study and the dirtier, applied work of computing. In 1958, the computing laboratory was closed, control of the computer passed to the main university, and in 1960 the IAS computer was donated to the Smithsonian National Museum of American History. Sadly, there are no plans to put it back on display.

We have one final story to complete in the EDVAC's rich and complex history: what happened next at Eckert and Mauchly's company. Lukoff, who left the Moore School for the Electronic Control Company in September 1947, was immediately impressed by Eckert. "Pres Eckert's genius was very apparent to everyone as he multiplexed from project to project, asking penetrating questions and offering ingenious new approaches," he wrote,[34] adding that "Pres was the engineer's engineer".

Although others, including Isaac Auerbach, looked at this approach slightly differently. "Every time you would design something, [Eckert] had a change," he said.[35] "[You] never could finish the design before he had a different design; and then that one had to be rebuilt and checked and made to work and the tolerances worked out. I mean, it was every day with something new."

Many people also found him a daunting character to work for, in ways both good and bad. He certainly wasn't a natural people person – unlike the jovial John Mauchly – but inspired fierce loyalty. "His intellect is overpowering, particularly for a person that hasn't worked with him and just meets him casually," said Jean Bartik, one of the earliest programmers hired by Eckert and Mauchly.[36]

But, she added, "working with him is very exciting, because even the slightest thing you say that's new, he recognises it as new. And he immediately builds on it and sees how it can be used in many different ways. So that you increase the sense of your own worth working with this kind of person instead of being intimidated."

It's clear, then, that Eckert and Mauchly's business problems had nothing to do with intellect or ingenuity; it was their inability to structure deals that wouldn't leave them hundreds of thousands of dollars out of pocket. Part of this is understandable: they needed funds and they also needed contracts to give other companies confidence in them. After all, the Electronic Control Company was effectively a startup.

Even from that perspective, it's hard to defend the deal they struck with Prudential. On 8 December 1948, ECC agreed to build a UNIVAC by 15 September 1950 for $150,000. Along with converters for the IBM punch cards on which its data processing system relied: one card-to-tape converter, two tape-to-card converters. Plus twelve tape drives, five line printers and two key-to-tape encoders. As Nancy Stern put it: "Even by 1948 standards, $150,000 was a paltry sum for such an array of revolutionary equipment."[37]

This deal coincided with the Electronic Control Company incorporating as the Eckert-Mauchly Computer Corporation, EMCC. This restructure meant outside companies could invest, and the market research company AC Nielsen made an offer to take a controlling interest in what it saw as an innovative but poorly run company. Eckert and Mauchly rejected the offer, but Nielsen still placed an order for a UNIVAC, six tape drives, six key-to-tape encoders, and a line printer.

The deal again made little financial sense for Eckert and Mauchly at $151,400, especially as the founders realised that they needed $500,000 of working capital or they would be unable to fulfil their current order book.[38] They needed a big surge of investment and they needed it fast, or their odds of survival were becoming slim.

34 Herman Lukoff, *From Dits to Bits*, p74
35 Oral history with Nancy Stern, p9
36 'Oral history with Jean J Bartik and Frances E (Betty) Snyder Holberton' by Henry S Tropp, 27 April 1973, Smithsonian National Museum of American History, **rpimag.co/bartikholbertoninterview**, p110
37 Nancy Stern, *From ENIAC to UNIVAC*, p142
38 As above, p144

Which is where a betting company came to its aid. On 8 August 1948, Henry Straus, the vice president of American Totalisator became chairman, with the company owning 40% of EMCC's shares. In return, it paid $438,000 and lent EMCC a total of $112,000, with the loan to mature in January 1950.

It's easy to see why Straus backed the young entrepreneurs. An electrical engineer himself, he had invented the electric totalisator but struggled to make money from it. He only earned his fortune after forming a partnership with the American Totalisator Company and, intrigued by this new era of electronic digital computers, recognised the potential of EMCC.

It was a cash injection at exactly the right time for Eckert and Mauchly, not only meaning they could deliver the BINAC, which was in the final testing stages at this point in time, but also invest heavily in more employees and a new office building. It had been a bumpy start, but Eckert and Mauchly were now thriving and their company booming. It was the American dream writ large.

> *The computer industry was born*

But as quickly as they'd been rescued, their world collapsed: in October 1949, Straus's private plane crashed, killing him, the pilot, and two passengers. Without the backing of Straus, the American Totalisator Company had no interest in EMCC but great interest in reclaiming its loans. Which were set to mature in February 1950.

With banks refusing their advances, bankruptcy loomed unless EMCC could find a new backer. IBM was one obvious buyer, but wary of anti-trust litigation. Other interested parties, which included rivals Burroughs and the National Cash Register, couldn't summon up deals in time. This left only one realistic bidder: Remington Rand. It would pay the American Totalisator $438,000 for its shares, plus $100,000 to share among all EMCC employees who held stock.

The computer industry is born

The benefits to Remington Rand are clear: for a cash investment of $538,000, it instantly became the second biggest player in the nascent market for electronic computers. IBM being the other. And if you're thinking how odd that a typewriter company would wish to do so, it already sold automated office machines so this takeover wasn't entirely out of keeping. Indeed, we know from Lukoff's account that the company attempted to head-hunt him and fellow EMCC engineers the previous year; they had all turned the offer down. "My first thought [on hearing of the takeover] was that Remington Rand had to buy the company because it wasn't successful in hiring any of its engineers," wrote Lukoff.[39] While Eckert and Mauchly had not become the millionaires they might have aspired to be, there was the comfort of their share of the $100,000 – plus the guarantee of a job for eight years at $18,000 per year. A generous salary in 1950, roughly $250,000 in today's money, and they would also get a minimum annual dividend of $5,000 from profits deriving from the patents.

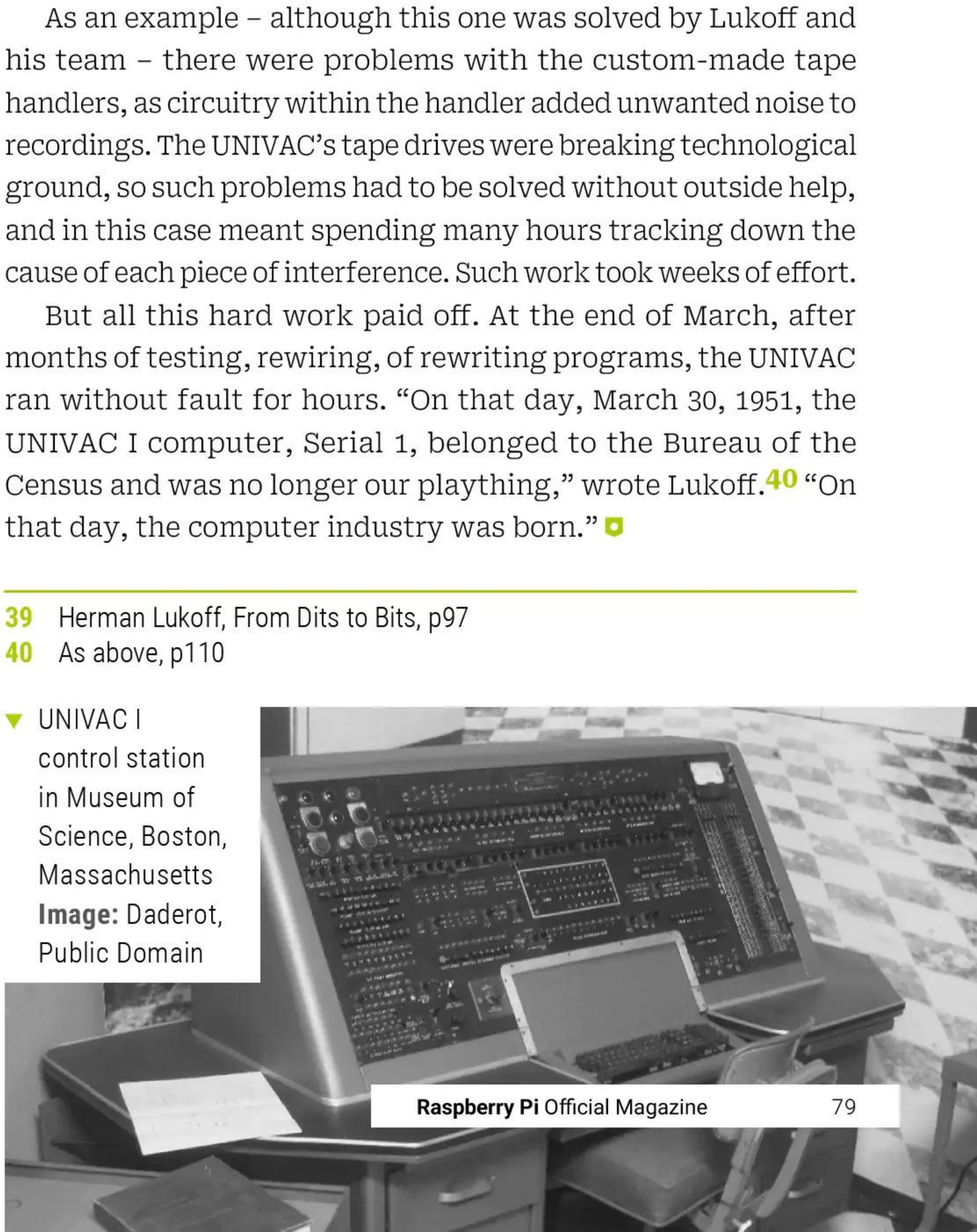
But they lost ultimate control of their company, becoming a division of Remington Rand. Straus had been a hands-off chairman, but VP in charge of research General Leslie Groves – the military head of the Manhattan Project – had his own ideas. Having realised that the true cost of building each UNIVAC was around half a million dollars, his first challenge was to either renegotiate the existing contracts or to somehow get them cancelled. After threats – renegotiate or else – from Remington Rand's aggressive lawyers, Nielsen and Prudential took the easy route out and agreed to cancel their contracts in return for refunds. A similar tactic against the government contractors proved less successful, with all its computers delivered at the agreed prices.

By summer, the first UNIVAC was entering its final testing stage. Unlike the BINAC, here they had few major problems to fight, but instead many minor problems. Many of these fell to Eckert who used whoever was in earshot to bounce problems and solutions off, although Lukoff said that only the similarly gifted Frazer Welsh was quick enough to respond in kind.

As an example – although this one was solved by Lukoff and his team – there were problems with the custom-made tape handlers, as circuitry within the handler added unwanted noise to recordings. The UNIVAC's tape drives were breaking technological ground, so such problems had to be solved without outside help, and in this case meant spending many hours tracking down the cause of each piece of interference. Such work took weeks of effort.

But all this hard work paid off. At the end of March, after months of testing, rewiring, of rewriting programs, the UNIVAC ran without fault for hours. "On that day, March 30, 1951, the UNIVAC I computer, Serial 1, belonged to the Bureau of the Census and was no longer our plaything," wrote Lukoff.[40] "On that day, the computer industry was born."

39 Herman Lukoff, From Dits to Bits, p97
40 As above, p110



▼ UNIVAC I control station in Museum of Science, Boston, Massachusetts
Image: Daderot, Public Domain

Part 11

Conquer the command line: start and stop

We take a look at scripts to manage the way Raspberry Pi OS starts and shuts down



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

Raspberry Pi OS, much like every other GNU/Linux distribution, uses a very specific software suite (systemd) on startup that makes sure all the relevant software you need is started and loaded before you log in.

The first process the kernel starts is **/lib/systemd/systemd** (with a PID of 1), and it is the parent process of everything that happens in user space once the Linux kernel has finished initialising devices and drivers, and mounted the file system. If you run `ps auwx`, you'll see `init` listed as PID 1, but take a look under the hood with `ls -l /sbin/init` and you'll see that init is a symbolic link (see 'Linked In' later in the article) to systemd:

```
  $ ls -hgo /sbin/init
  lrwxrwxrwx 1 20 Mar  6 09:56 /sbin/init ->
/lib/systemd/systemd
```

So, systemd, masquerading as init, gets everything else started, and usually ends with the prompt inviting you to log in to your Raspberry Pi. Raspberry Pi OS hides most of the messages that this startup process generates, but you can see them by typing `dmesg`, as seen in **Figure 1**.

Systemd

Systemd starts services in parallel, and can defer service starts until they are needed. Rather than many scripts for individual components, a target is set, and systemd resolves the dependencies until it reaches that target, avoiding any fixed startup sequence along the way.

`systemctl` is the service that can restart, or query status, of server software

```
pi@raspberrypi:/etc/systemd $ ls system/*
system/dbus-fi.w1.wpa_supplicant1.service          system/dbus-org.freedesktop.timesync1.service
system/dbus-org.bluez.service                      system/default.target
system/dbus-org.freedesktop.Avahi.service          system/display-manager.service
system/dbus-org.freedesktop.ModemManager1.service  system/sshd.service
system/dbus-org.freedesktop.nm-dispatcher.service

system/bluetooth.target.wants:
bluetooth.service

system/dev-serial1.device.wants:
hciuart.service

system/getty.target.wants:
getty@tty1.service
```

```
pi@raspberrypi:~ $ ls -l /etc/systemd/system/multi-user.target.wants/
total 0
lrwxrwxrwx 1 root root 40 Nov 19 08:33 avahi-daemon.service -> /lib/systemd/system/avahi-daemon.se
rvice
lrwxrwxrwx 1 root root 41 Nov 19 08:33 console-setup.service -> /lib/systemd/system/console-setup.
service
lrwxrwxrwx 1 root root 32 Nov 19 08:31 cron.service -> /lib/systemd/system/cron.service
lrwxrwxrwx 1 root root 40 Nov 19 08:35 cups-browsed.service -> /lib/systemd/system/cups-browsed.se
rvice
lrwxrwxrwx 1 root root 29 Nov 19 08:35 cups.path -> /lib/systemd/system/cups.path
lrwxrwxrwx 1 root root 32 Nov 19 08:35 cups.service -> /lib/systemd/system/cups.service
lrwxrwxrwx 1 root root 42 Nov 19 08:33 dphys-swapfile.service -> /lib/systemd/system/dphys-swapfil
e.service
lrwxrwxrwx 1 root root 40 Nov 19 08:31 e2scrub_reap.service -> /lib/systemd/system/e2scrub_reap.se
rvice
lrwxrwxrwx 1 root root 39 Nov 19 08:34 glamor-test.service -> /lib/systemd/system/glamor-test.serv
```

Adding a service to `systemd` creates a symlink to its real location as part of the process — don't do it manually

Systemd starts services in parallel, and can defer service starts until they are needed

▶ **Figure 1:** `dmesg` gives you the startup messages from the kernel — nuggets of useful info buried in plenty of legacy boot information

```
[  489.751880] drm_atomic_commit+0xbc/0x100 [drm]
[  489.752090] drm_mode_atomic_ioctl+0xa38/0xc98 [drm]
[  489.752164] drm_ioctl_kernel+0xc4/0x140 [drm]
[  489.752240] drm_ioctl+0x218/0x4e0 [drm]
[  489.752310] __arm64_sys_ioctl+0xb4/0x100
[  489.752330] invoke_syscall+0x50/0x120
[  489.752346] el0_svc_common.constprop.0+0x48/0xf0
[  489.752353] do_el0_svc+0x24/0x38
[  489.752359] el0_svc+0x30/0xd0
[  489.752370] el0t_64_sync_handler+0x100/0x130
[  489.752376] el0t_64_sync+0x190/0x198
[  489.752383] ---[ end trace 0000000000000000 ]---
[ 8899.234727] usb 1-1.4: new high-speed USB device number 5 using xhci_hcd
[ 8899.325243] usb 1-1.4: New USB device found, idVendor=2222, idProduct=1111, bcdDevice=
0.01
[ 8899.325271] usb 1-1.4: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 8899.325285] usb 1-1.4: Product: Mass storage
[ 8899.325296] usb 1-1.4: Manufacturer: Generic
[ 8899.325305] usb 1-1.4: SerialNumber: 0008906484
[ 8899.330165] usb-storage 1-1.4:1.0: USB Mass Storage device detected
[ 8899.335463] scsi host0: usb-storage 1-1.4:1.0
[ 8900.343870] scsi 0:0:0:0: Direct-Access     Generic  Flash-Disk       1.09 PQ: 0 ANSI:
2
[ 8900.349010] sd 0:0:0:0: [sda] 1968128 512-byte logical blocks: (1.01 GB/961 MiB)
[ 8900.351055] sd 0:0:0:0: [sda] Write Protect is off
[ 8900.351076] sd 0:0:0:0: [sda] Mode Sense: 03 00 00 00
[ 8900.352839] sd 0:0:0:0: [sda] No Caching mode page found
[ 8900.352854] sd 0:0:0:0: [sda] Assuming drive cache: write through
[ 8900.384025]  sda: sda1
[ 8900.384385] sd 0:0:0:0: [sda] Attached SCSI removable disk
[ 8900.396504] sd 0:0:0:0: Attached scsi generic sg0 type 0
pi@raspberrypi:/etc/systemd $ dmesg
```

Conquer the Command Line 3rd Edition out now

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



- Backwards compatibility with much older init.d scripts — and even run levels — is maintained by systemd

Files are found under **/etc/systemd/system** and there's a lot to learn. The one thing to remember is that systemd and its service manager are controlled with the `systemctl` command.

```
$ sudo systemctl restart ssh
```

That command will restart the SSH server — something you'll need to do if you edit **/etc/ssh/sshd_config** to change the port it listens on, for example. For backwards compatibility, as well as **/etc/init.d** scripts to start and stop services, the system of service commands that worked on older versions of Raspberry Pi OS, such as:

```
$ sudo service apache2 reload
```

...still works (here we ask a running Apache2 Web server to reread its configuration files).

In **/etc/systemd/system/multi-user.target.wants**, you will find files like **cron.service** which, when examined closely with `ls -l`, you'll see are links to files of the same name in **/lib/systemd/system** (other distributions may place the files under **/usr/lib**).

systemd dot

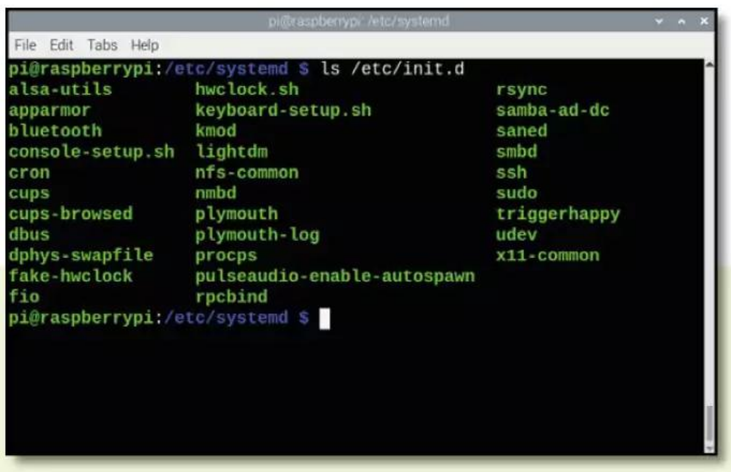
Systemd is a complex system, with many elements — from **systemd.unit** to **systemd.slice** — but they all have their own man pages, which are worth reading to help get the overall concept.

Don't worry if what's inside these files looks confusing; there's a logic to them with their conditional dependencies, but you can safely forget about them until you need to get some software working automatically on every system restart for your Raspberry Pi project. Even then, we'll show you another way with crontab — otherwise you'll need to be aware of the following, as those links aren't created manually:

```
$ sudo systemctl enable postgresql.service
```

...will create the link, enabling PostgreSQL to run on startup (assuming it's installed). To make systemd aware of configuration changes (such as when editing **/etc/fstab**) before the next restart, enter:

```
$ sudo systemctl daemon-reload
```



Linked ln

Systemd makes links between files automatically, but there will be times you'll want a file to appear to be in a local directory when it is elsewhere, with a handy little command we have not so far had a chance to show you: `ln`.

The `ln` command makes a link which allows a file to effectively exist in two places at once. In the following example:

```
$ ln -s /usr/share/code-the-classics/README.md
README-ctc.txt
```

...a file will appear in your current working directory. But `ls -l` and you'll see that it's a special type of file, a symbolic link pointing to the actual file. Edit **README-ctc.txt** and you'll find that **README.md** in the linked directory will be edited.

Soft, or 'symbolic' links, are created with the `-s` switch — you don't even need the file to which you're linking to actually exist, which makes it handy if you're linking across a network, or to a removable drive.

It's called a symbolic link because it works by linking to the name of the target file, rather than to the file data itself. Create a new file and a hard link:

```
$ ls > listing.txt
$ ln listing.txt listing2.txt
```

...and you have two names (and locations) for the same file — sounds like the same thing? Not exactly: if you delete the original file in the first example, you can replace it with a new file of the same name, but different contents, and the symbolic link will point to the new file. Remove the original file in the hard link case, and the link still points to the original data.

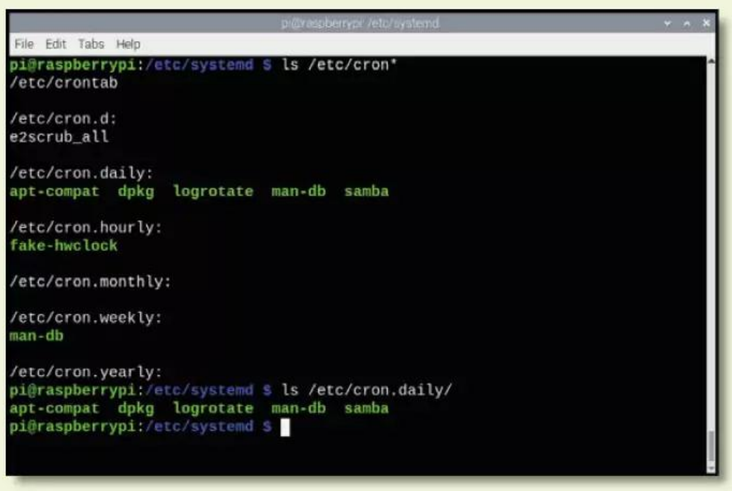
Location, location

The startup scripts are generally for daemons, processes you want running all of the time, like web servers and databases. There are plenty of programs which do housekeeping that need to be run periodically — hourly, daily, weekly. For this purpose, the cron software utility is ideal for scheduling the running of such programs and tasks. Cron searches its configuration

directories and runs the scripts it finds there — have a look at the different folders in **/etc** with names beginning with 'cron'.

The easiest way to learn where such things are on your system is to search with `locate`. First, enable the database update service with `sudo systemctl enable plocate-updatedb.timer`, which will update `locate`'s database daily. You can update it immediately with `sudo updatedb`, then try something like `locate cron` — which will find you every file or directory with cron as its name or as part of its name.

The built-in alternative is `find`, a powerful utility which enables you to search particular directories — or the whole file system — by name or name fragment, size of file, how long ago they were modified, or whether they're bigger than another



▲ Examining some of the cron directories

file — enough to deserve a whole tutorial of its own. Because it searches the file system, rather than a cached listing, it takes longer than `locate`, but it is always up to date, and has search options not found in `locate` (see 'Find and Regular Expressions'). To replicate our `locate cron` command with the `find` tool:

```
$ find / -name '*cron*'
```

If you were looking for cron or crontab, but not anacron, you could search for 'cron*' instead. There will be more output than you want, so pipe it through a pager, or perhaps a grep. Back to using cron — the easiest way is via crontab, which maintains a table where each row specifies a command, and how often it is to run.

You edit the crontab file not directly, but with `crontab -e`, which calls up the default editor to do it. Here's an example that helps keep track of how much free disk space you have on your primary disk:

```
45 05 * * 1-5 df -h / >> diskspace.txt
```

... which grabs the output from `df -h` and appends it to the file **diskspace.txt** in your home directory every weekday at 5:45am. The first five crontab fields cover minute, hour, day of month, month, day of the week, and can all be replaced with a single special value, like `@daily` or `@hourly`. While `man crontab` tells you a little about crontab, `man 5 crontab` is far more useful as it covers the layout of the file, with examples. Run `man man` for more on the numbered sections available with `man` commands.

A fresh startup

Using the value `@reboot`, we can easily run our own scripts on startup, without messing about with system startup scripts. There are times when a full systemd startup script will be more appropriate, but for quickly getting something tested, put the script into crontab.

There are two things that may catch you out. Firstly, you might be running scripts out of a directory that you have in your `$PATH`, which defines where Bash looks for commands. As `$PATH` is only set once you log in and your personalised **.bashrc** file is read, scripts running from crontab which are run immediately upon startup will not be aware of your `$PATH` setting. So, you will need to express some commands by their full paths, such as **/home/pi/bin/test.sh** — but in general, cron should be able to find commands in **/bin** and **/usr/bin**. You can replace the line you added previously with one that runs the command each time you reboot:

```
@reboot sleep 10; df -h / >> diskspace.txt
```

Systemd's parallel service startups also mean that some services, such as the network, may not be ready when your `@reboot` commands in the crontab entry are called, but you could use the smallest time delay that consistently works on testing. You can also use the automation of cron to run your own backup scripts, so we'll discuss backup options in the next part.

Find and regular expressions

The `find` tool can search by regular expressions, as well as (part) name. They're a whole tutorial topic in themselves, but well worth investigating once you've got command-line basics under your belt, as regexps can be used with many commands.

Part 07

Make a range finder

Link an ultrasonic distance sensor and seven-segment display to measure distances



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
- HC-SR04 ultrasonic distance sensor
- Broadcom 5082-7650 seven-segment display
- 9× resistors: 7× 220Ω, 1× 510Ω, 1 or 2× 1kΩ

The HC-SR04 ultrasonic distance sensor is a favourite with Raspberry Pi robot makers. It works by bouncing ultrasonic sound off an object and timing how long it takes for the echo to return. This time is then converted into a distance, which can be displayed on a single-digit, seven-segment display. Using this sensor, you'll improve your skills in working with inputs and outputs. You also get to use seven-segment displays, which are quite cool in a retro kind of way.

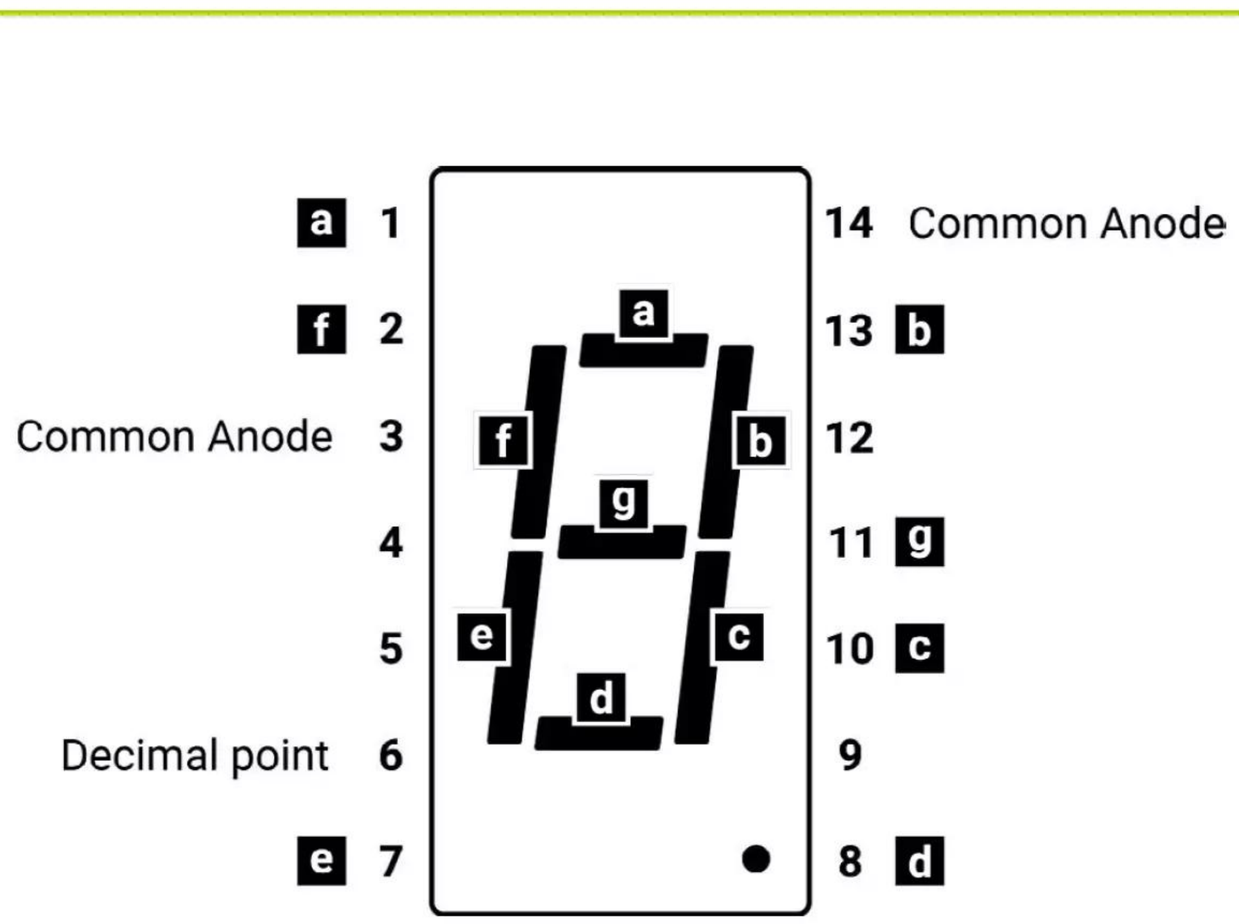
Because the HC-SR04 uses a 5V power supply, it transmits a 5V pulse. We must use a voltage divider to take it down to the 3.3V tolerance of Raspberry Pi's GPIO pins. The maths for a voltage divider is simple:

```
(source voltage * R2) / (R1 + R2) =
output voltage
```

Resistor 1 (R1) is connected to the source voltage, and Resistor 2 (R2) is connected to ground. With a 510Ω resistor for R1 and a 1kΩ for R2, the formula is:

```
(5 * 1000) / (510 + 1000) = 3.31
```

Although the GPIO pins can tolerate no more than 3.3V, they recognize 1.8V or higher as a HIGH signal. So you could use 1kΩ for both R1 and R2, which will give you a (very safe) 2.5V signal.



▶ **Figure 1:** LED segment display pins

DOWNLOAD THE FULL CODE:

rpimag.co/gpiobookgit

Lighting the display

The seven-segment display is a collection of LEDs, with one LED corresponding to one of the segments. All the anodes (positive ends) are connected; this should be connected to the 3V3 supply. Each cathode (negative end) should be connected to a resistor to limit the LED current, and the other end of the resistor to a GPIO pin. To turn the LED on, all you have to do is set the GPIO output to be LOW (0V) and it will complete the circuit for the current to flow. This is true for common-anode displays; we'll explain what changes you need to make if you're using a common-cathode display.

Generating a seven-segment pattern

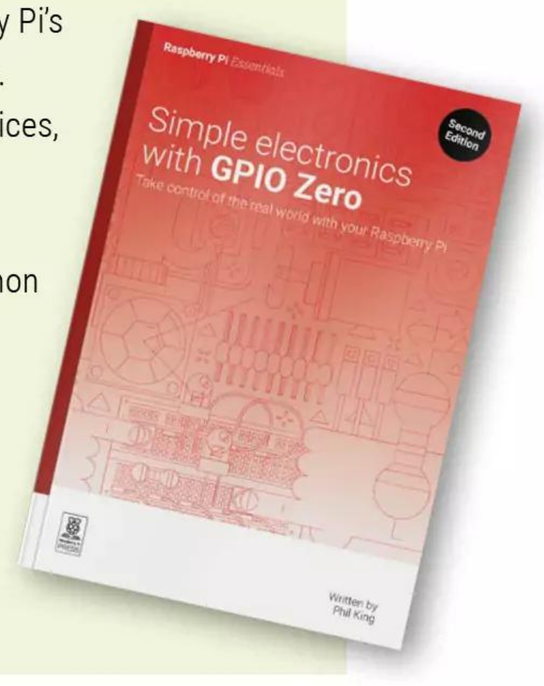
The display consists of four bars or segments that can be lit. By choosing the segments to light up, you can display a number from 0 to 15, although you have to resort to letters A–F (hexadecimal) for this. There are, in fact, 128 different patterns you can make, but most are meaningless. Each pattern is a list of seven values, each one corresponding to an LED segment. By convention, the LED segments are identified by the letters **a–g** (not to be confused with the hexadecimal digits we are displaying on the LED), starting with the topmost horizontal segment, and continuing clockwise as shown in **Figure 1**; **g** identifies the middle horizontal segment.

Using this sensor, you'll improve your skills in working with inputs and outputs

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, range finder, laser-powered tripwire, and Raspberry Pi robot.

rpimag.co/gpiozerobook



The patterns identify which segments should be turned on. The first one, with index 0, is (1, 1, 1, 1, 1, 1, 0) and illuminates all the segments except the middle segment, displaying the digit 0. The last one, with index 15, is (1, 0, 0, 0, 1, 1, 1), and illuminates the topmost segment (**a**), the two leftmost segments (**e** and **f**), and the middle segment (**g**), which displays the letter F. **Figure 2** shows the breadboard layout.

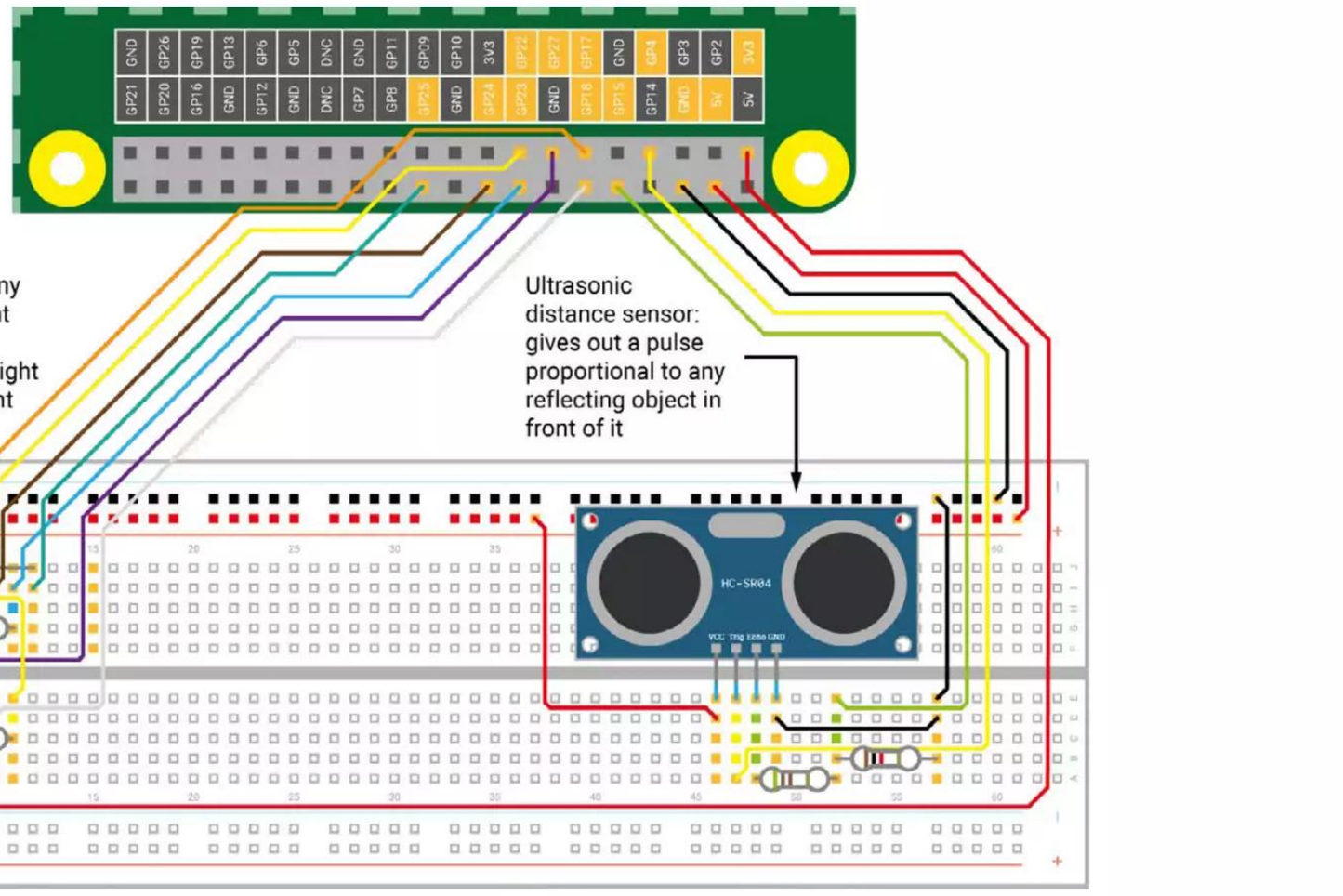
If you have a different seven-segment display, consult its datasheet to find out whether it's common anode or common cathode. We used a common-anode display in **Figure 2**, hence the connection between common anode and 3V3. If you're using a common-cathode display, you'll need to connect its common cathode pin to the GND pin and not connect any of its pins to 3V3 (you'll also need to make a change to the code, explained later, to take the GPIO pins HIGH instead of LOW when illuminating a segment). See **Figure 3** for a common-cathode version (SKU 103527 from The Pi Hut).

You'll also need the datasheet to confirm which pins correspond to which segment and adjust your wiring as needed. If you need help figuring out which LED pin goes with which segment, you can use the following code (**test_segments.py**) to light each segment in sequence, from **a** to **g**:

```
# Cycle through each segment in order
from gpiozero import LEDBoard
from time import sleep

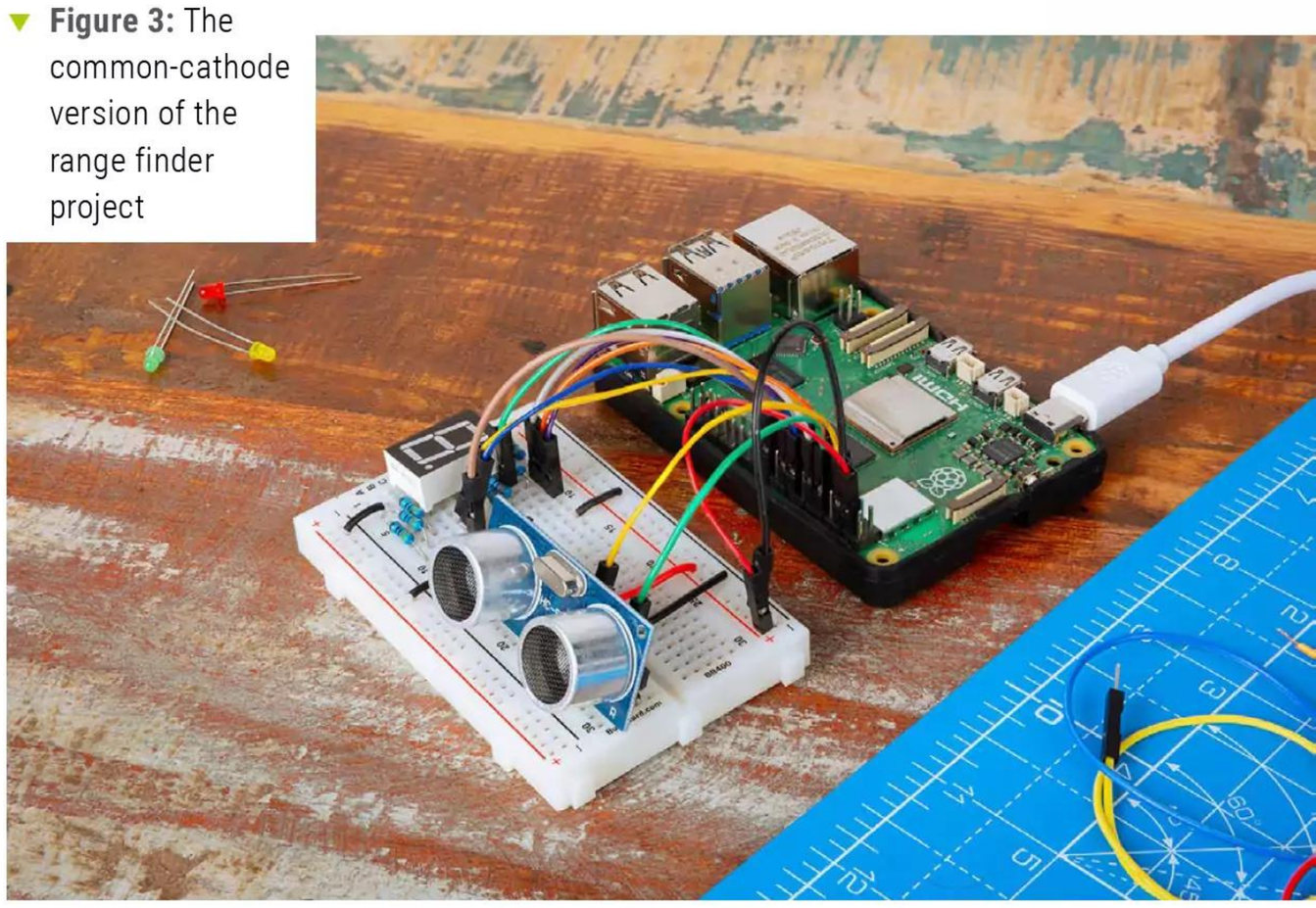
#              a   b   c   d   e   f   g
seg = LEDBoard(17, 27, 24, 23, 22, 18, 25,
active_high=False)

while True:
    for i in range(0, 7):
        print("abcdefg"[i])
        seg.on(i)
        sleep(1)
        seg.off(i)
```



▲ **Figure 2:** Building the range finder

▼ **Figure 3:** The common-cathode version of the range finder project



Now you're ready to create the range finder code. Create a new file, add the following code, and save it as **range_finder.py**.

```
  # displays the distance in decimetres on a
7-segment display
  from gpiozero import LEDBoard, DistanceSensor
  from time import sleep

  seg = LEDBoard(17, 27, 24, 23, 22, 18, 25,
active_high=False)
  sensor = DistanceSensor(echo=15, trigger=4)

  seg_patterns = [
      (1, 1, 1, 1, 1, 1, 0),
      (0, 1, 1, 0, 0, 0, 0),
      (1, 1, 0, 1, 1, 0, 1),
      (1, 1, 1, 1, 0, 0, 1),
      (0, 1, 1, 0, 0, 1, 1),
      (1, 0, 1, 1, 0, 1, 1),
      (1, 0, 1, 1, 1, 1, 1),
      (1, 1, 1, 0, 0, 0, 0),
      (1, 1, 1, 1, 1, 1, 1),
      (1, 1, 1, 0, 0, 1, 1),
      (1, 1, 1, 0, 1, 1, 1),
      (0, 0, 1, 1, 1, 1, 1),
      (1, 0, 0, 1, 1, 1, 0),
      (0, 1, 1, 1, 1, 0, 1),
      (1, 0, 0, 1, 1, 1, 1),
      (1, 0, 0, 0, 1, 1, 1),
  ]

  print("Display distance on a 7-seg display")
  while True:
      distance = sensor.distance * 10 # distance
in decimetres
      print("distance", distance)
      if distance > 15:
          distance = 15
      seg.value = seg_patterns[int(distance)]
      sleep(0.8)
```

In our code, we create an `LEDBoard` object (a collection of LEDs in one object), defining which pins are connected to which segments, and a list called `seg_patterns` defines the LED pattern for each number.

If you are using a common-cathode LED segment display, you'll need to remove the part that sets `active_high` to `False`, changing:

```
seg = LEDBoard(17, 27, 24, 23, 22, 18, 25,
active_high=False)
```

to:

```
seg = LEDBoard(17, 27, 24, 23, 22, 18, 25)
```

Run the code and watch the distance values appear on the display. You can stop the program with **CTRL+C.**

Displaying numbers

The way to set the desired LED configuration to make a number is to set the `LEDBoard`'s value to a 7-tuple of the states of the LEDs (each element of the `seg_patterns` list is a tuple — an immutable sequence; because each tuple has seven values, we call them 7-tuples). The `seg_patterns` list contains all the `LEDBoard` values required to display numbers 0-9 and letters A-F. `seg_patterns[0]` gives the pattern for 0 and `seg_patterns[15]` gives the pattern for F. This pattern is given to the `LEDBoard` by setting its value with `seg.value = seg_patterns[number]`.

GPIO Zero offers an easier way of working with segment LEDs, but if we'd used it in this tutorial, you would not have learned as much about how these displays work. You can use `LEDCharDisplay` to display more characters than we used in this example, though you're still subject to the limits of what characters a seven-segment display can represent. See **rpimag.co/ledchardisp** for details.

The distance sensor

The HC-SR04 distance sensor reports its reading by producing an output pulse that our Raspberry Pi tries to measure. The GPIO Zero library measures this pulse and converts it into a distance by returning a floating-point number that maxes out at 1 metre. We then multiply this number by 10 to give decimetres. Next, we convert it to an integer to get rid of the fractional part of the measurement, so we can show it on our single-digit display.

Using the range finder

The distance to the reflective object is updated every 0.8 seconds. A display of 0 indicates that the object is less than 10cm away. Don't touch the sensor, otherwise its readings will be wrong. Also, as it has quite a wide beam, you can get reflections from the side. If several objects are in the field of view, then the distance to the closest one is returned.

DOS & DON'TS

Get the best from your Raspberry Pi computer with this clear, friendly advice

By **Sean McManus**

You've joined the world of Raspberry Pi, perhaps enticed by the opportunity to build awesome projects and learn to code. Maybe you just need an affordable desktop machine or media centre. Whatever your motivation, you're welcome here.

It might be your first time tinkering with electronics or using a Linux-based computer. So, how can you get the best from Raspberry Pi and avoid the common mistakes and misunderstandings?

In this article, we'll share advice on setting up your microSD card, preventing problems, using the operating system (OS), and building your first electronics circuits.

So, DON'T delay and DO dive in!

DON'T ENABLE REMOTE CONNECTIONS UNTIL NEEDED

SSH and Raspberry Pi Connect enable you to manage your Raspberry Pi from another computer. They're great for projects like a wildlife camera, where you don't need a screen and keyboard on Raspberry Pi. Unless you need remote connections, keep them off to improve security. You can enable them later.

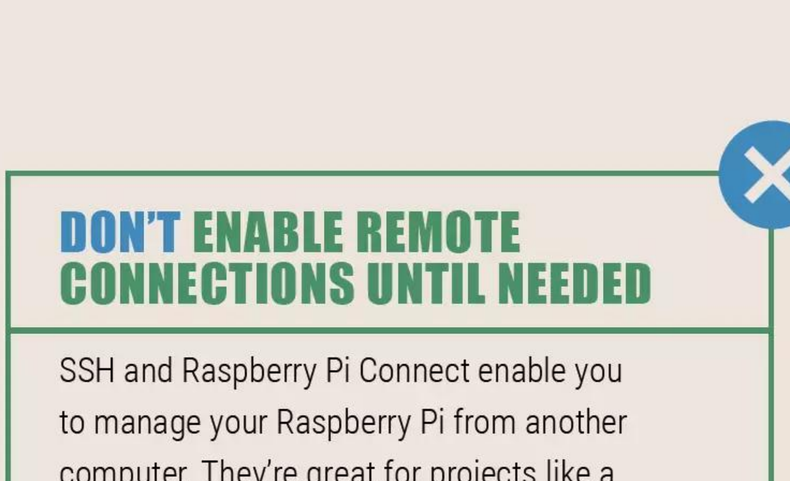
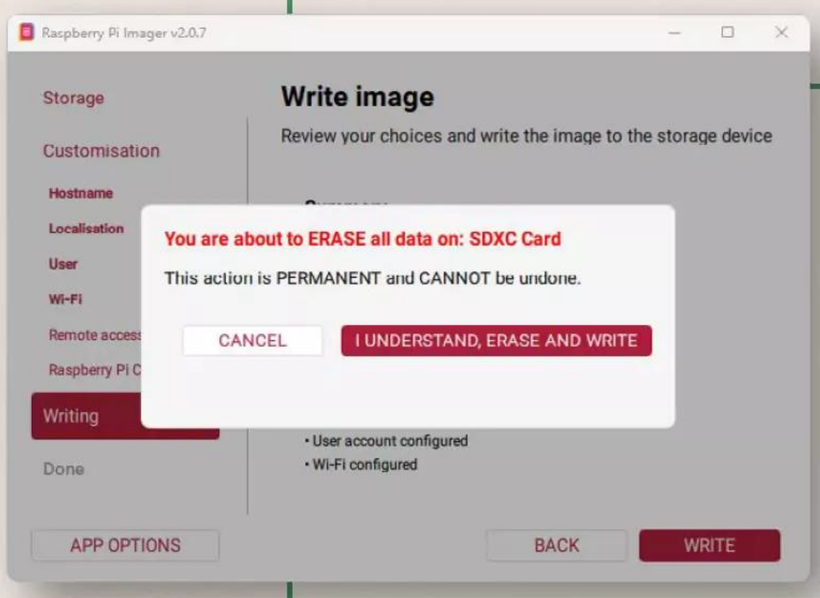


IMAGE A MICROSD CARD

The free Imager software (**rpimag.co/imager**) has a simple step-by-step process for installing Raspberry Pi OS on your microSD card

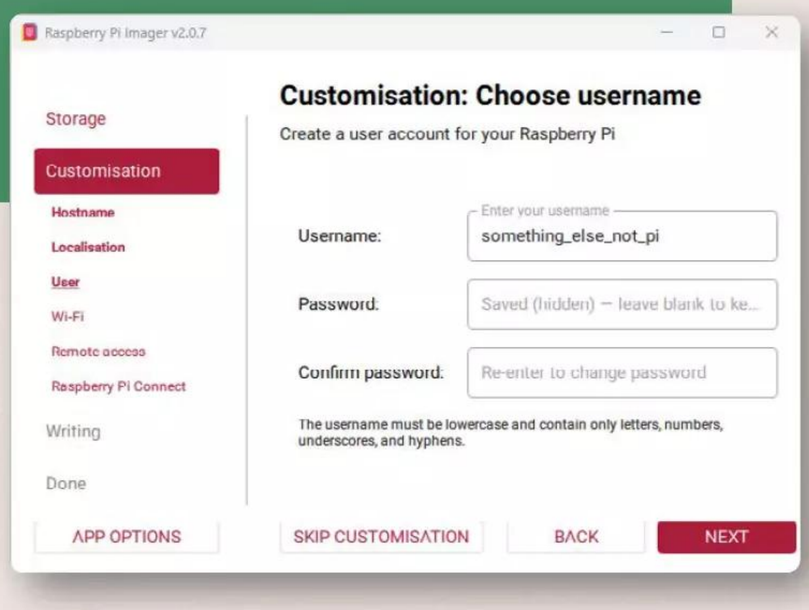


DON'T FORMAT THE WRONG DRIVE

You'll need another computer (Windows, macOS, or Linux) to set up your microSD card, at least your first one. Raspberry Pi can run its operating system from a USB flash drive or SSD, so Imager also enables you to install on them. So – and we really can't stress this enough – make sure you choose the right drive. It'll get wiped in the process.

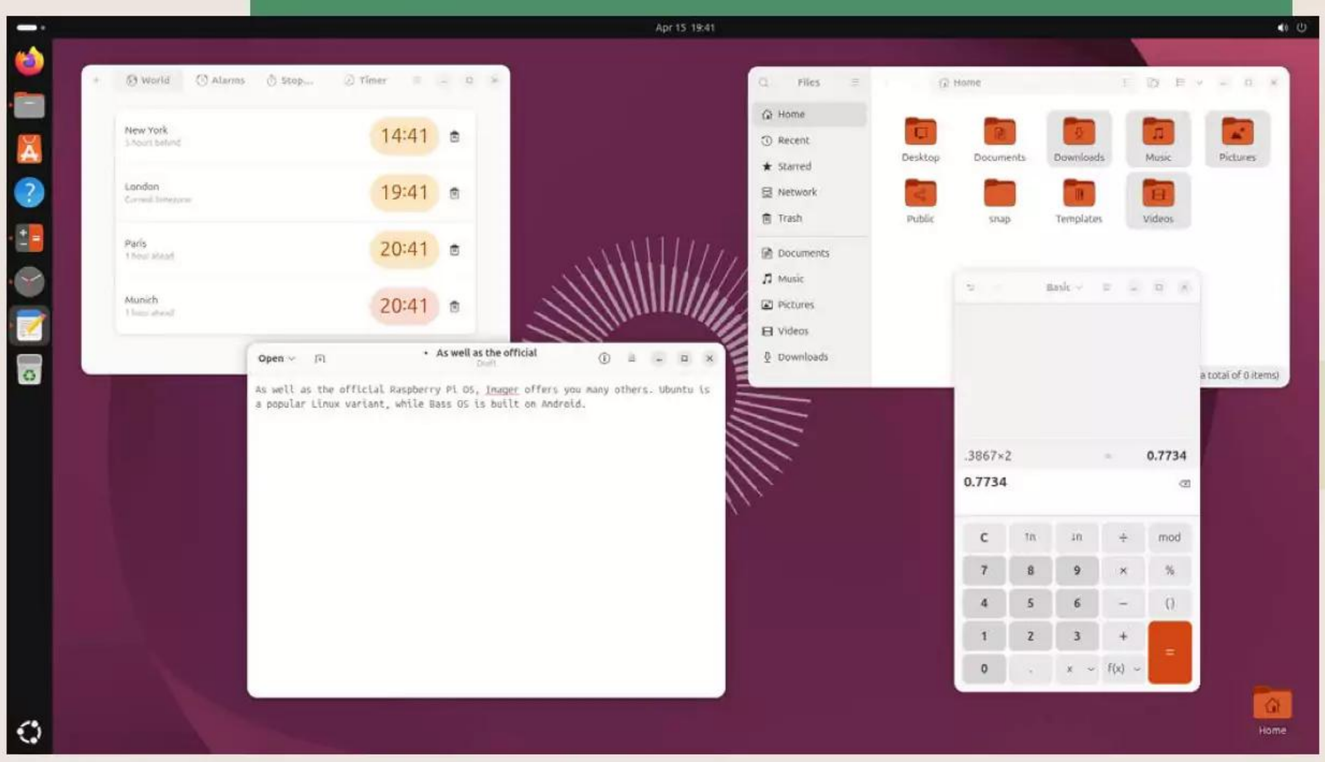
DO PICK A GOOD PASSWORD

Previously, the default username was 'pi' and the password was 'raspberry'. If you set up your Raspberry Pi for remote access over the internet, it'll be vulnerable with either of these widely known credentials, so choose new ones.



DO EXPERIMENT WITH DIFFERENT OPERATING SYSTEMS

As well as the official Raspberry Pi OS, Imager offers you many others. Ubuntu (pictured) is a popular Linux variant, while Bass OS is built on Android. There are five media players to choose from, and two platforms for playing old computer and video games. Other operating systems support 3D printers, ad-blocking, and home automation. Explore!

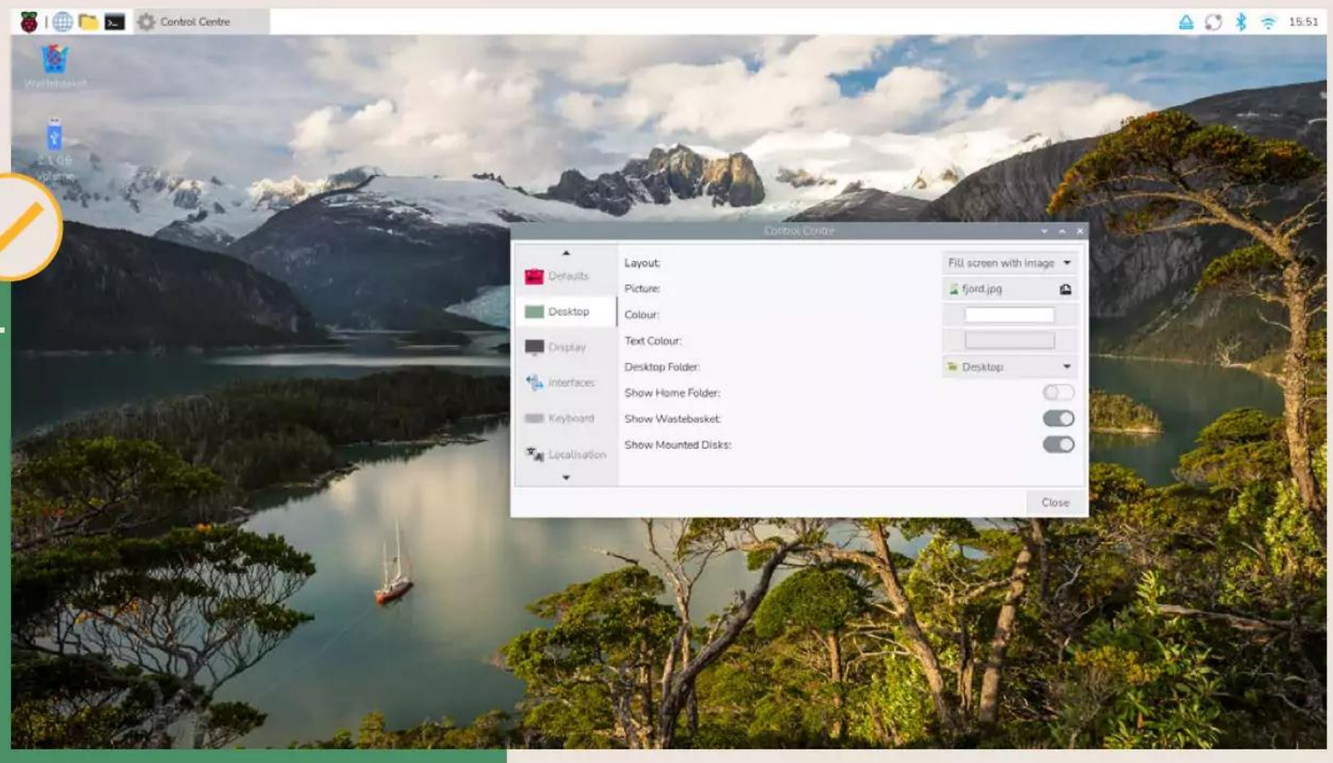


RASPBERRY PI OS

Familiarise yourself with the desktop and adjust the settings to make it your own

DO CUSTOMISE YOUR SETTINGS

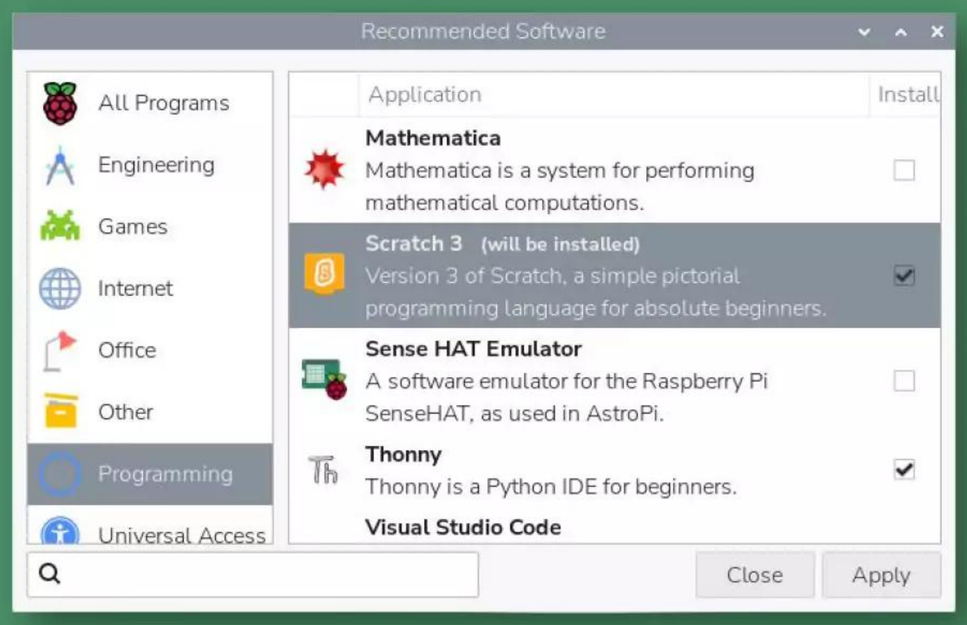
Do you prefer your taskbar at the bottom? Want the text to be bigger? Are the keyboard and mouse too sensitive? You'll find all the settings you need in Control Centre, under Preferences in the menu. You can also set your Raspberry Pi to boot to the command line, change your password, and enable interfaces like SSH for remote connection. Check out the fabulous wallpaper photos, too!

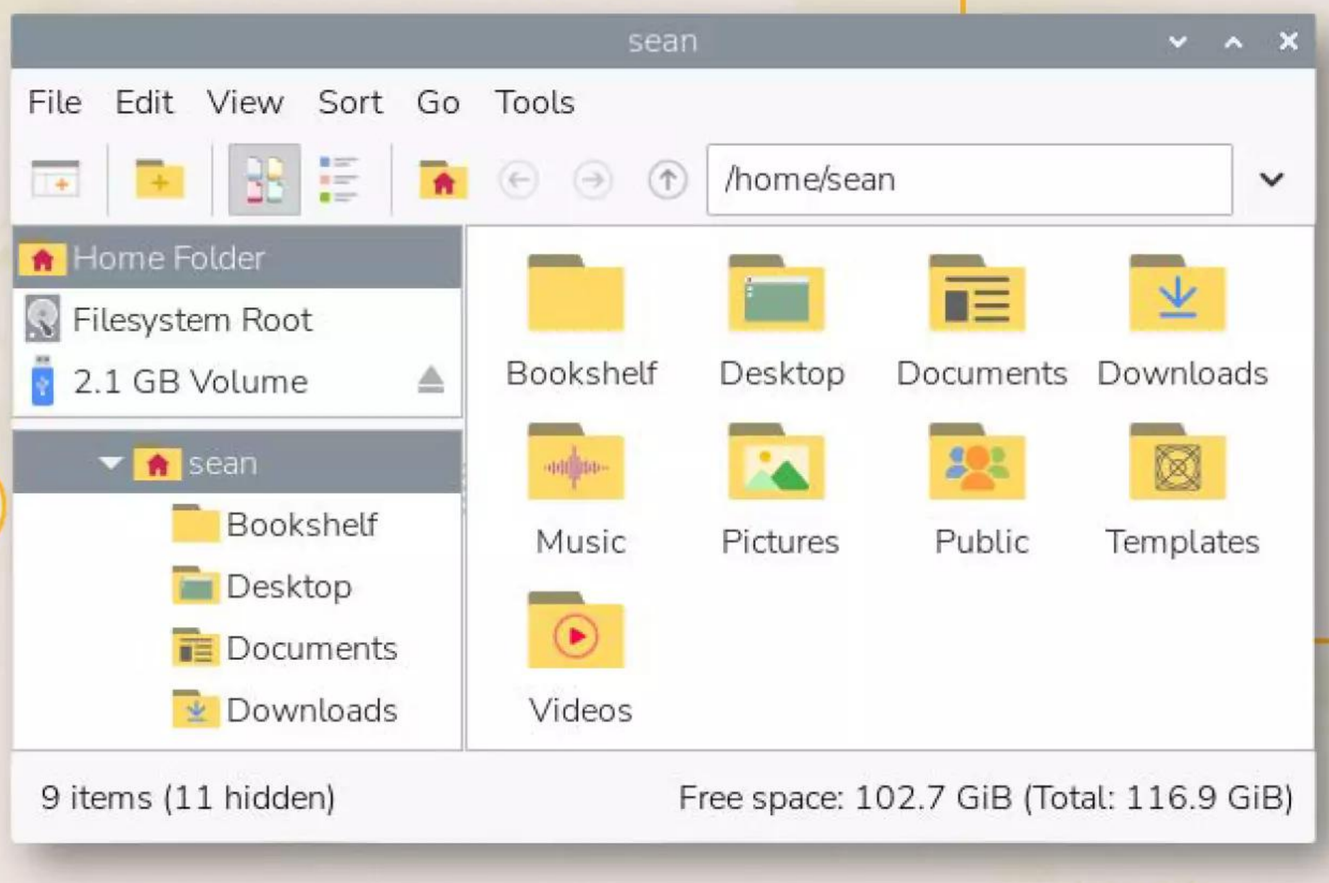


"FIND ALL THE SETTINGS IN CONTROL CENTRE"

DO CUSTOMISE YOUR SOFTWARE

Use the Recommended Software or Add/Remove Software tools to find and install free software for Raspberry Pi. You'll find both tools under Preferences in the menu. There's a version of Raspberry Pi OS that includes all the recommended software, so you can save time by putting that on your microSD card in one go.



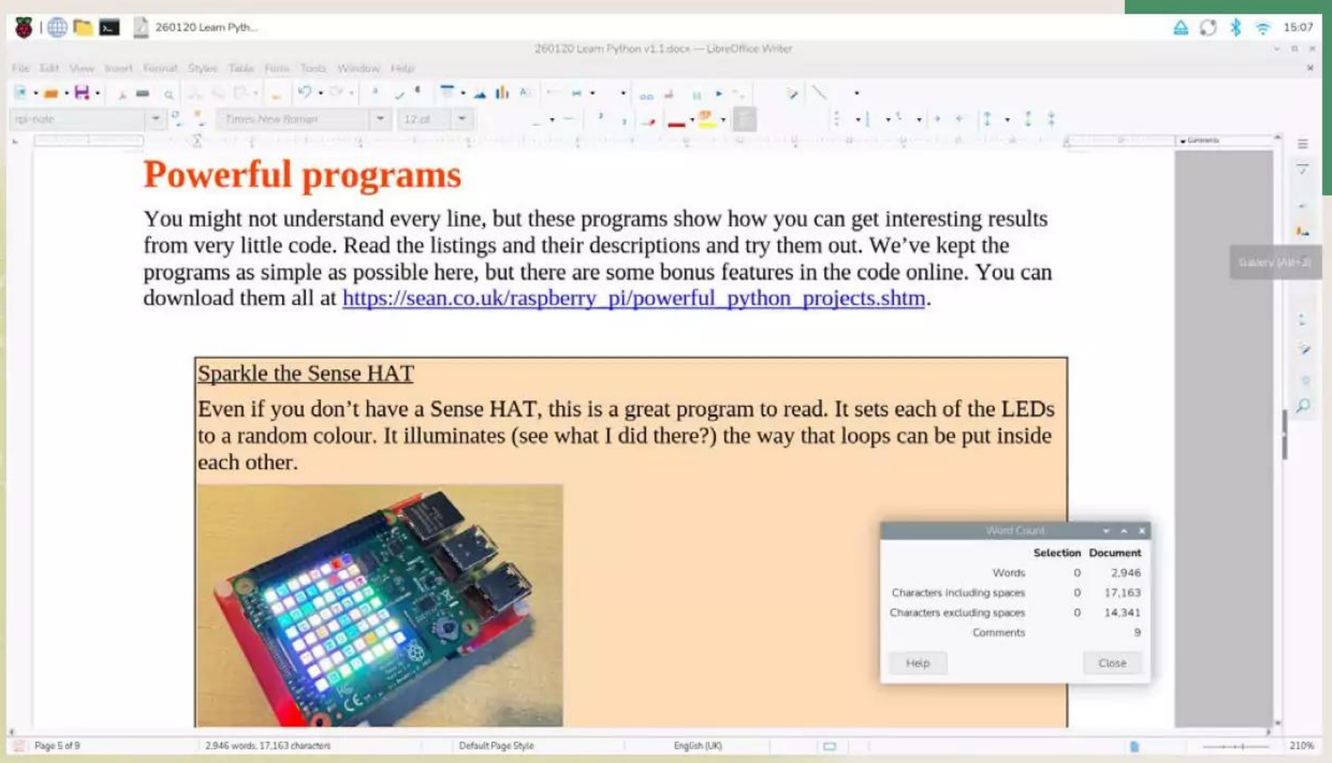


DO STORE FILES IN YOUR HOME FOLDER

Linux is strict about permissions, so you can't just put files anywhere. Store your files in your user folder, which has the same name as your username. In File Manager, click the Home icon to go straight there. It comes prepared with folders for downloads, pictures, and documents, among other things. You can find the File Manager icon on the taskbar.

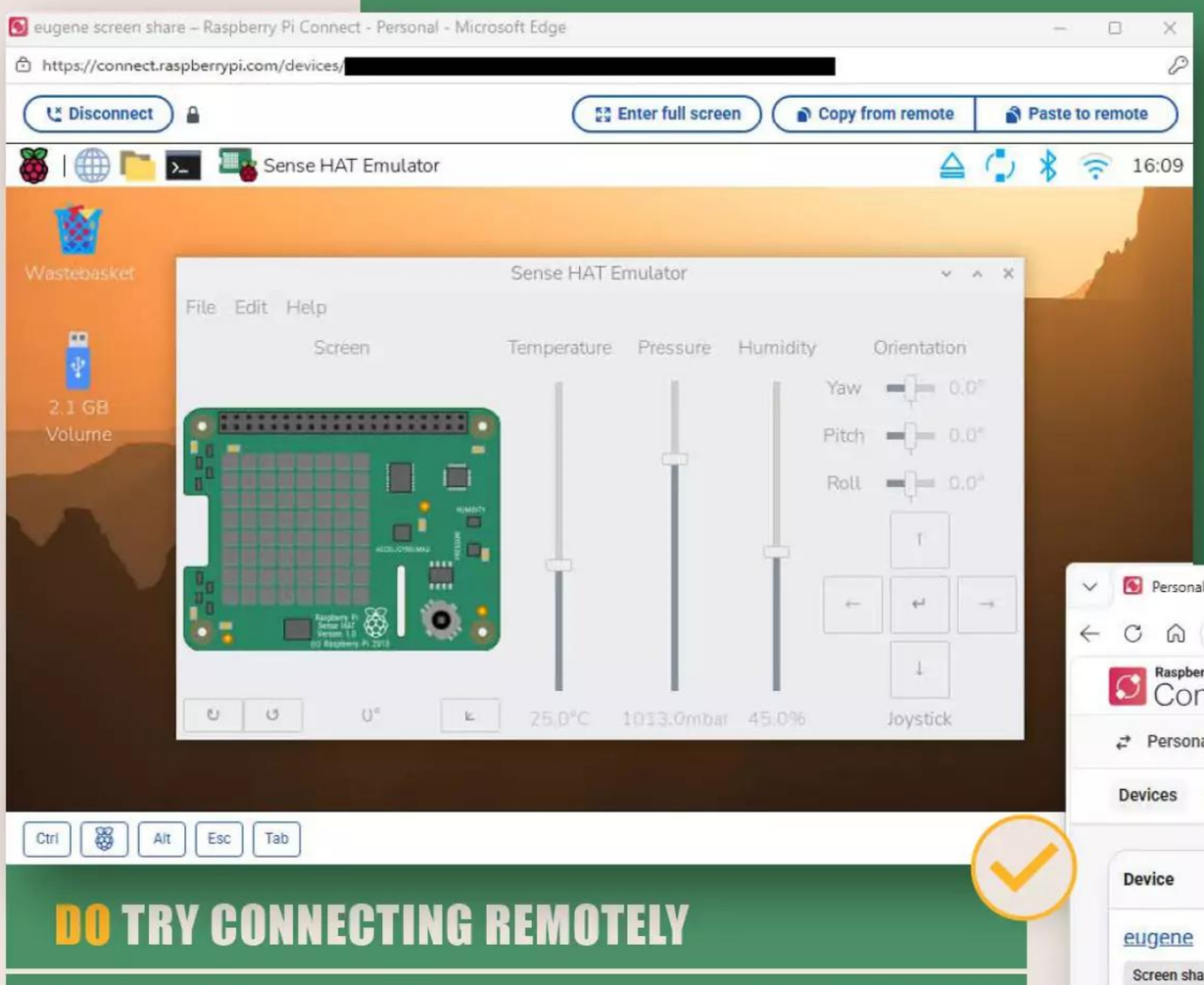
DO GET SOME WORK DONE!

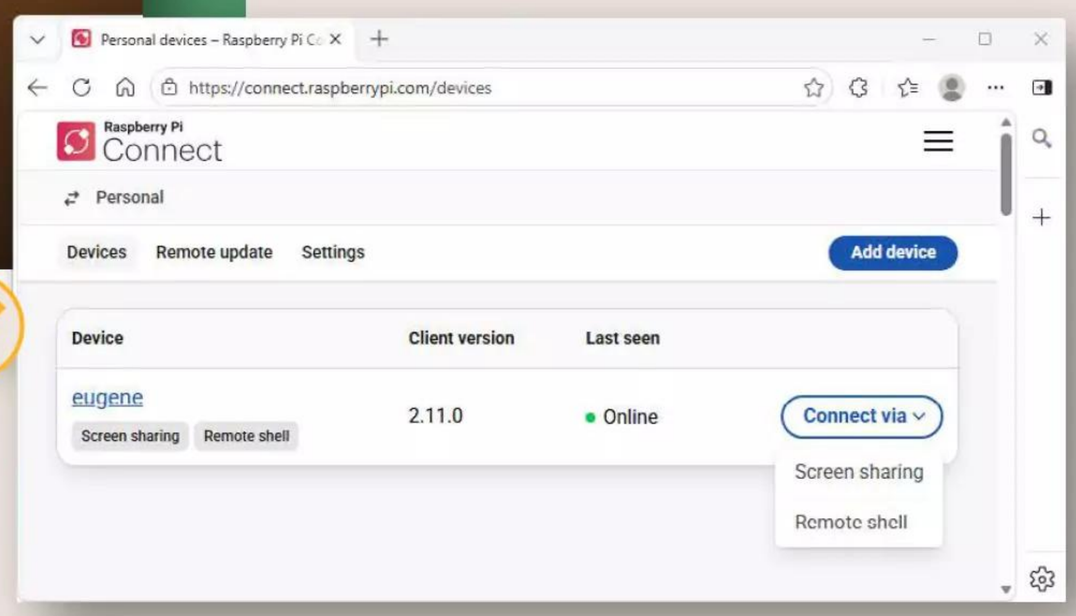
Raspberry Pi is famous for its hackability, but it's not only for things like robots and weather stations. When there's work to be done, it helps with that too. LibreOffice (in Recommended Software) includes word processor, spreadsheet, and presentation applications that are compatible with Microsoft Office.



DO SET A KEYRING PASSWORD

When you first use the web browser, it'll prompt you to set a password for the keyring. This is used to encrypt saved passwords in the browser, so make sure you pick a strong keyring password.



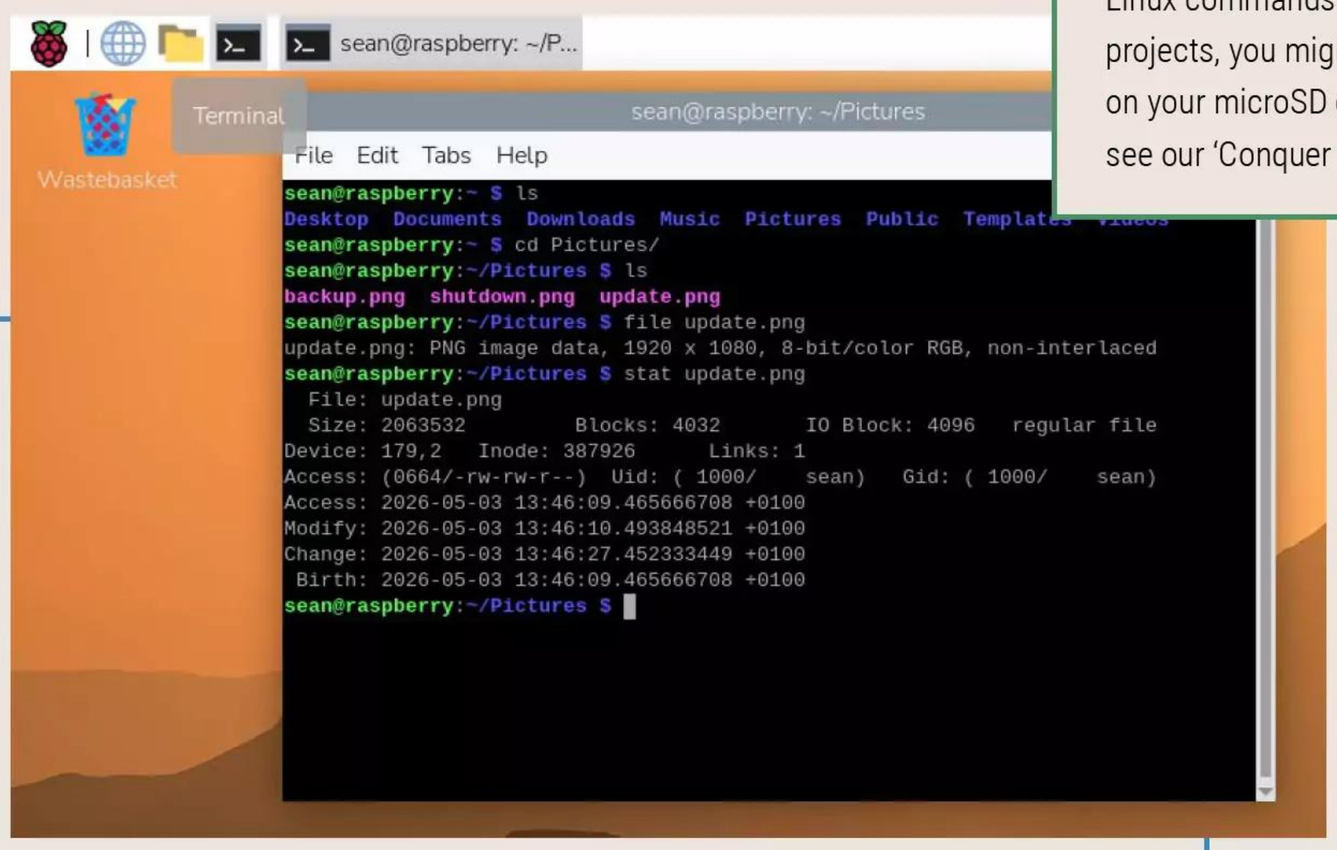


DO TRY CONNECTING REMOTELY

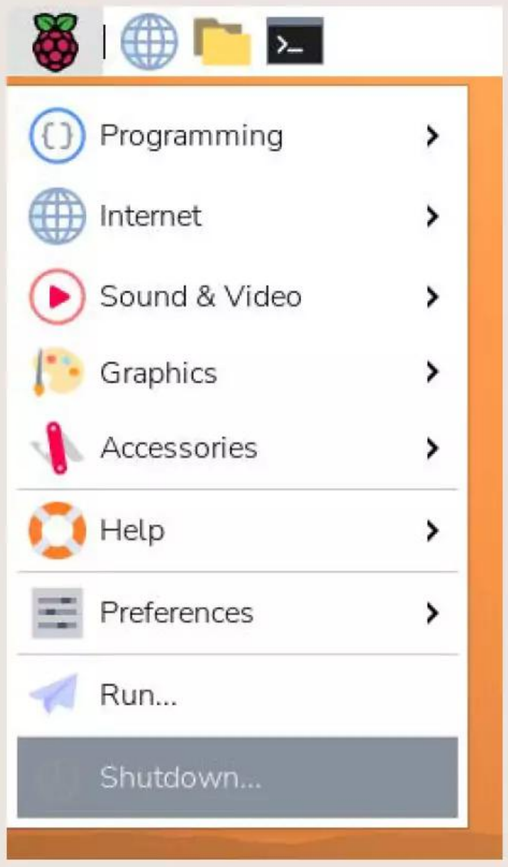
Raspberry Pi Connect enables you to connect to your Raspberry Pi computer over the internet. It's simple: you log in to your Raspberry Pi account using the icon in the top right and give your device a name. Using a web browser on another computer (including a Windows or macOS machine), you can then log in to manage your Raspberry Pi remotely. You can use the Raspberry Pi desktop or the command line (the remote shell).

DON'T FEAR THE TERMINAL

We all love the simplicity of click, drag, and drop, but for some processes it's quicker to type in instructions. Use the terminal to enter Linux commands for managing your files and software. For some projects, you might not even need to install the desktop, saving space on your microSD card. Open the terminal using the taskbar icon and see our 'Conquer the command line' series and book to get started.



"TO TURN OFF RASPBERRY PI SAFELY, USE THE SHUTDOWN OPTION IN THE MENU"

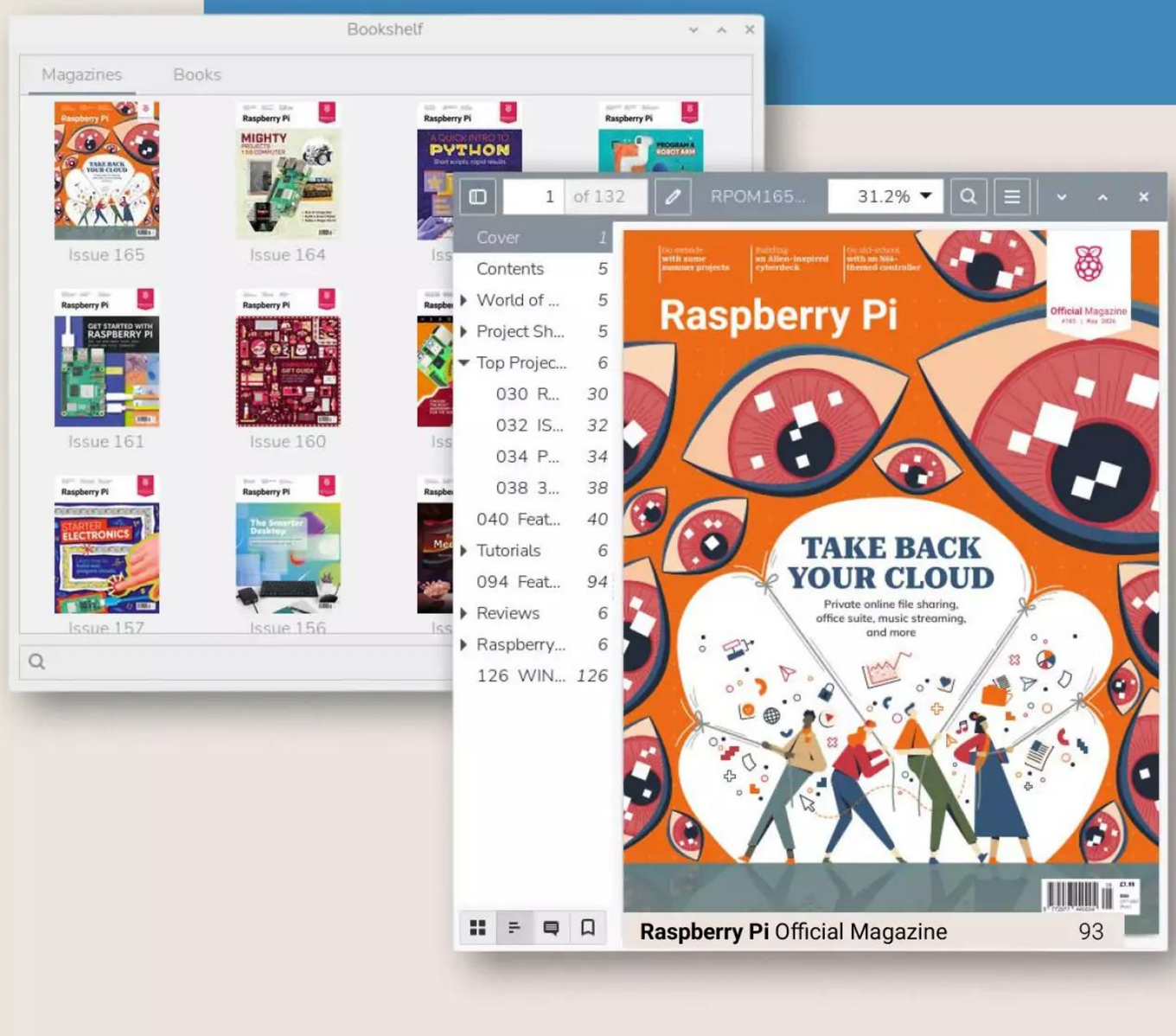


DO SHUT DOWN PROPERLY

Don't just pull the power, because there's a risk of corrupting your microSD card. To turn off Raspberry Pi safely, use the Shutdown option in the menu. If you're using the terminal, you can shut down immediately by entering `sudo poweroff`. You'll be prompted to enter your password.

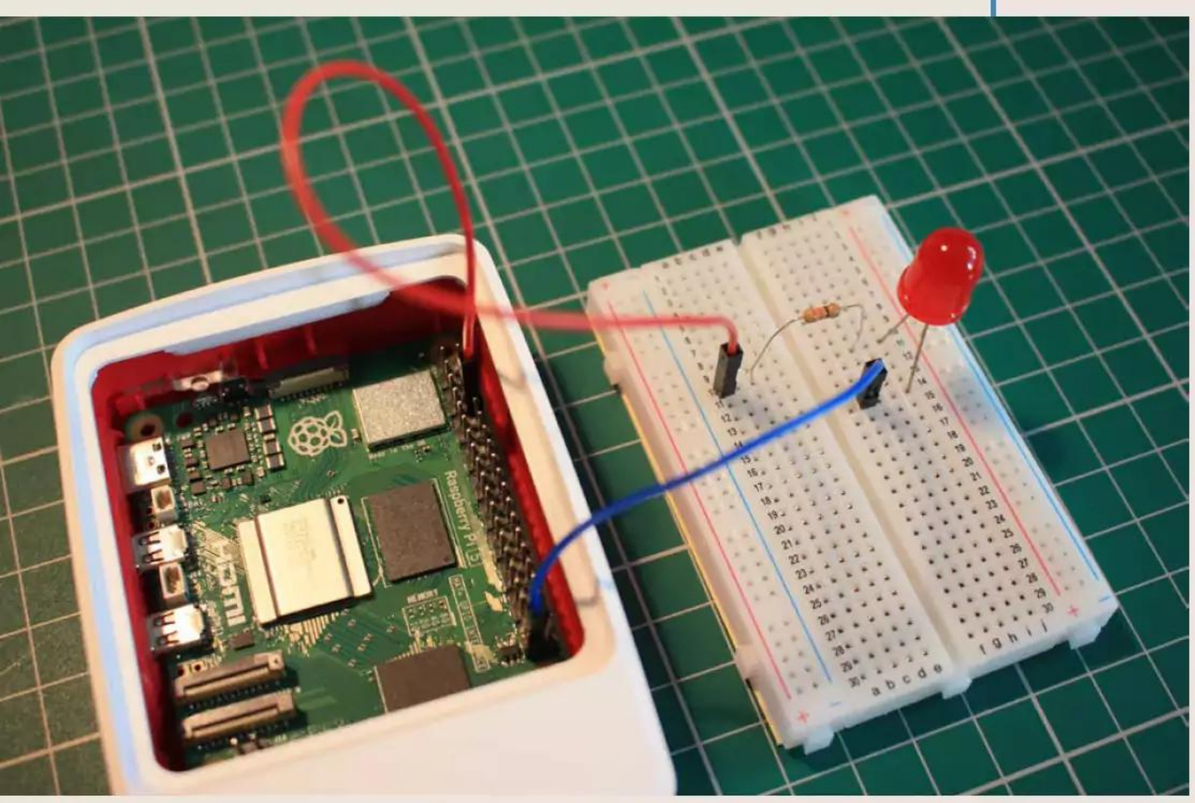
READ ALL ABOUT IT

There's lots of documentation built in to Raspberry Pi OS in the Help section of the menu, including the user guide. Back issues of this magazine are available to download in Bookshelf, together with official Raspberry Pi books. Some titles are restricted to people who subscribe or donate to the magazine. Your PDFs are stored in the Bookshelf folder, in your home folder.



ELECTRONICS

You only need a few components to build your first computer-controlled circuit

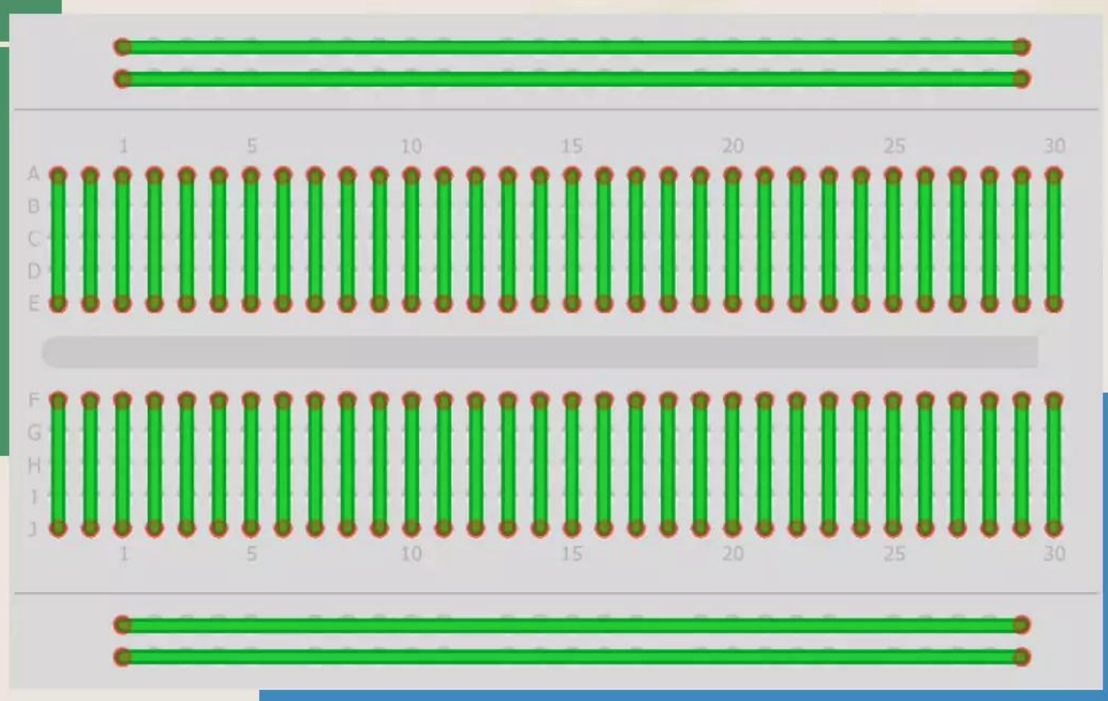


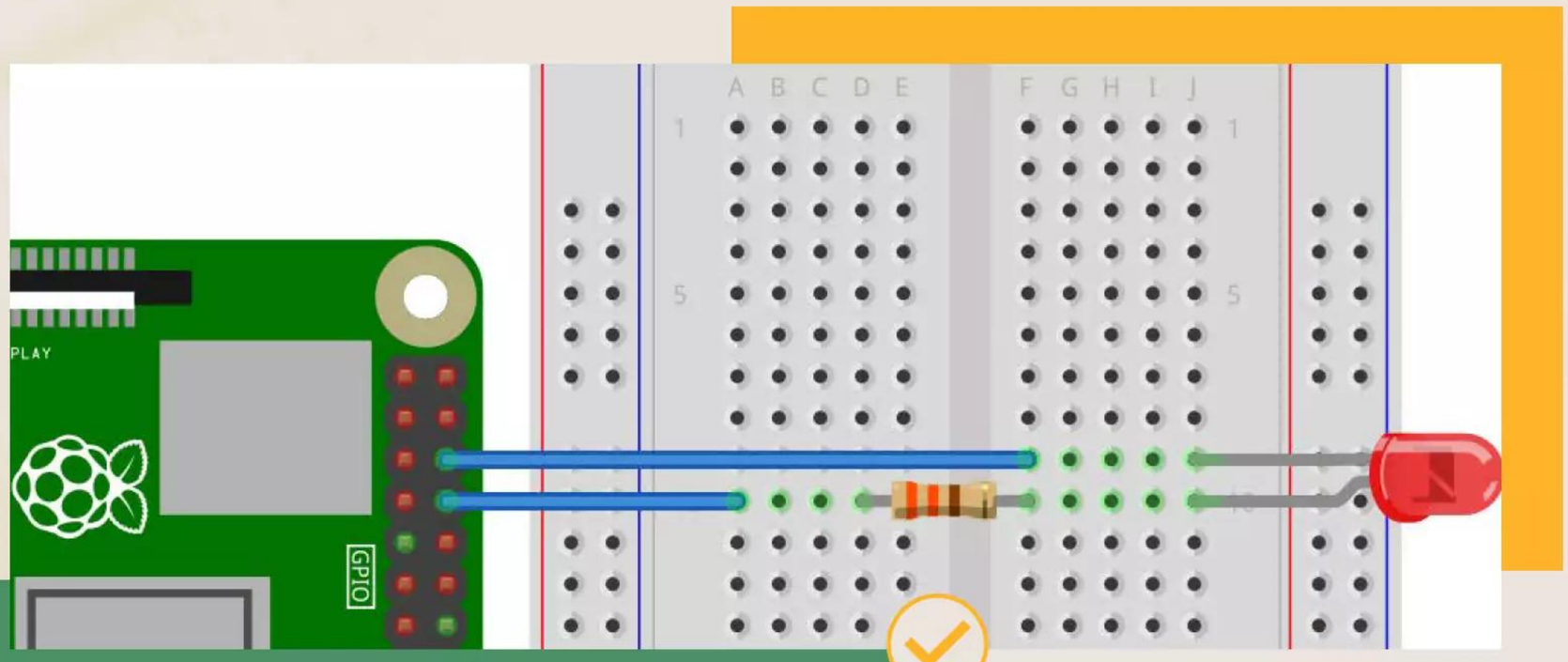
DON'T SOLDER TO YOUR GPIO PINS

To ensure you can easily reuse your Raspberry Pi computer for other projects, don't solder them directly to its GPIO pins. Instead, use socket-to-pin jumper wires to connect the GPIO pins to a breadboard. Breadboards enable you to press components into them without soldering, so you can easily build and test circuits.

DO UNDERSTAND THE FLOW THROUGH A BREADBOARD

Learn how the holes in a breadboard are connected to each other, so that you don't create a short circuit. The wires of a resistor need to be plugged into different rows, for example, otherwise the current will flow under the resistor, largely bypassing it. This picture shows how the holes are connected.





DO USE A RESISTOR WITH AN LED

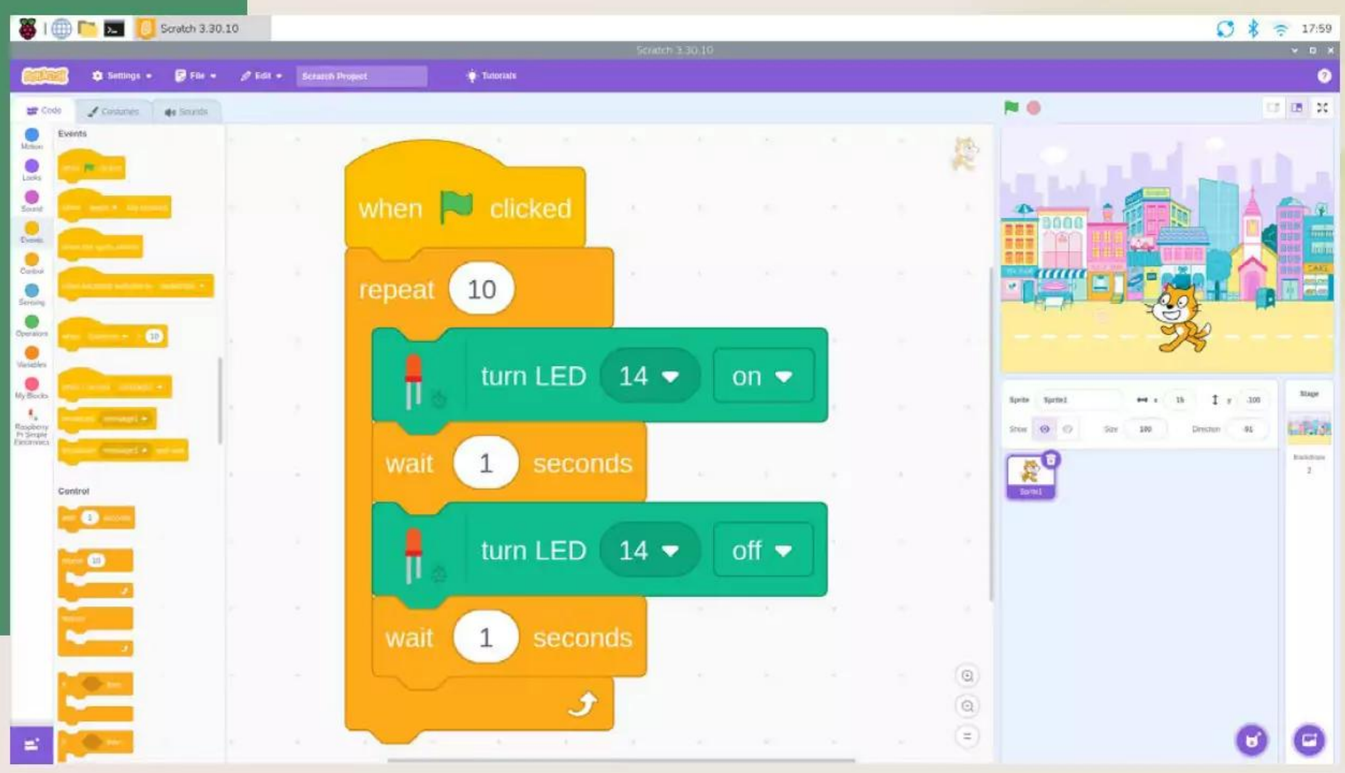
A light emitting diode (LED) is a fun and useful component. To protect your Raspberry Pi and stop your LED burning out, make sure you have a resistor in series in the circuit. That means the current is forced to go through the resistor and doesn't have another path around it. We recommend a 330 ohm resistor, which protects Raspberry Pi without dimming the LED too much.

DON'T GET YOUR LED THE WRONG WAY AROUND

Electricity only flows one way through an LED. Make sure your current goes through the long leg (the anode or positive end) before it goes through the cathode (the negative end). The flat edge of the LED indicates the cathode.

DO TRY SCRATCH SIMPLE ELECTRONICS

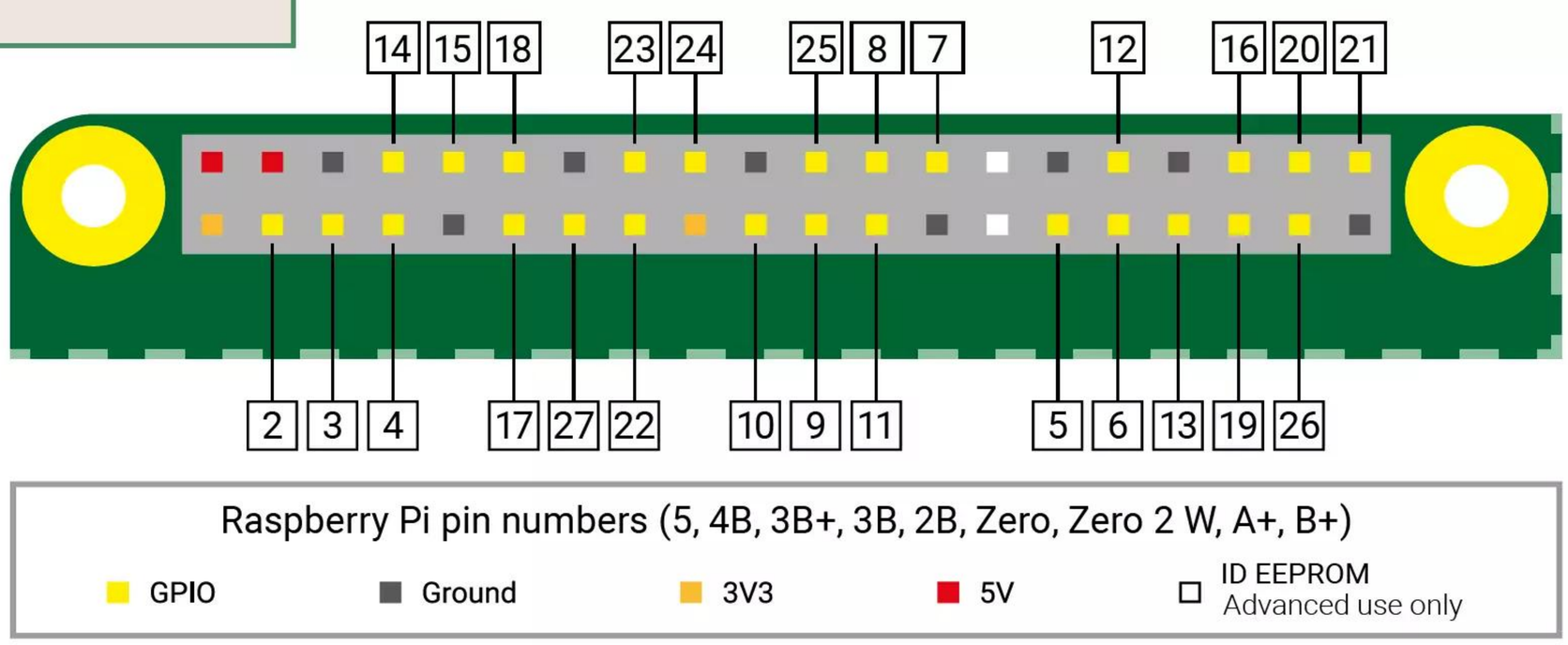
Most Raspberry Pi projects are programmed using the Python language. If you haven't done any programming before or you have a child helping you, Scratch is a recommended application that provides a friendly introduction. The Simple Electronics extension (add it with the icon in the bottom left) gives you drag-and-drop instructions to turn LEDs on and off and respond to button presses. This code works with the circuit diagram on this page. For similar code in Python, see page 42 of issue 161.



DON'T USE THE 5V PINS

Usually, you build a circuit where electricity flows from one of the numbered GPIO pins, through your components, and back to one of your Raspberry Pi's ground pins. To provide continuous power to a circuit, you could use a 3.3V pin instead. Some advanced circuits do need a continuous 5V, but unless you know what you're doing you risk damaging Raspberry Pi. Beginners should familiarise themselves with the pins and only use the GPIO and 3.3V pins, flowing to a ground pin.

"FAMILIARISE YOURSELF WITH THE PINS"



DO TURN OFF TO CONNECT UP

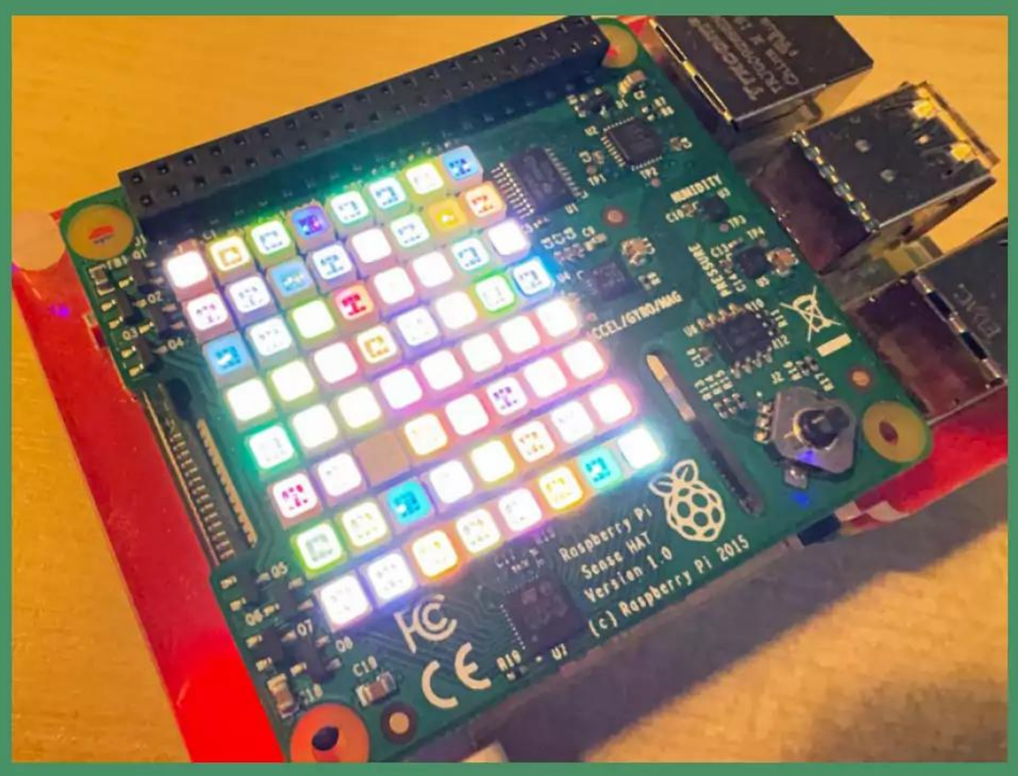
It's safe to connect simple circuits like an LED while the Raspberry Pi is switched on. However, connecting the wrong pins could immediately damage your Raspberry Pi. So, why not buy some extra thinking time? Shut down Raspberry Pi , connect your circuit at your leisure, and double-check the wiring before you power on.

DON'T DIRECTLY CONNECT A MOTOR

Motors need more current than Raspberry Pi can give them and connecting them directly risks damage. Instead, use a motor driver that receives control signals from Raspberry Pi and powers the motor from a separate battery.

DO EXPERIMENT WITH THE SENSE HAT

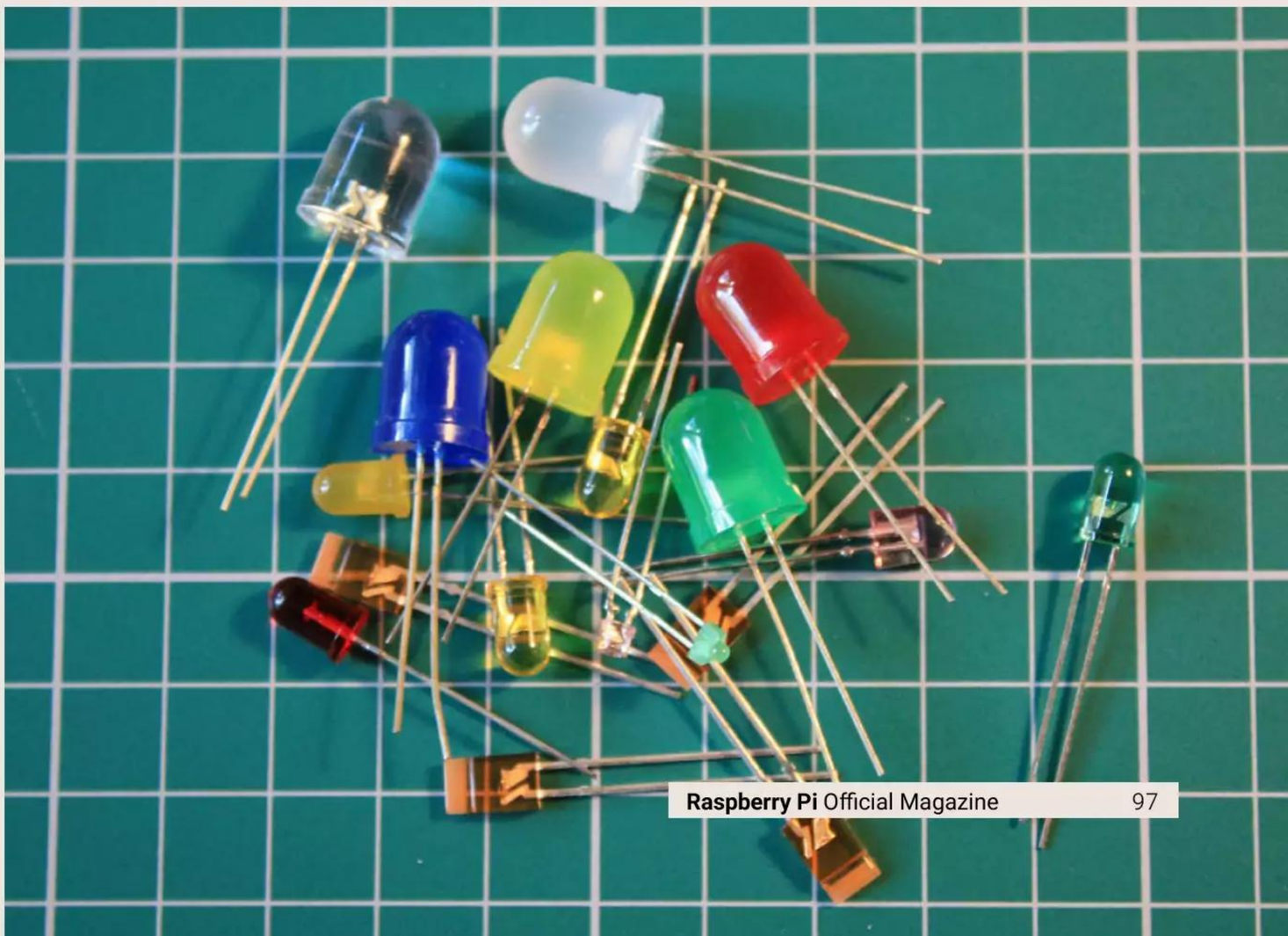
The Sense HAT is an add-on that features sensors for pressure, humidity, temperature, colour, orientation, and movement, as well as a joystick and a grid of 64 LEDs. Using it, you can quickly and confidently build physical computing projects, without any knowledge of electronics.



DISCOVER YOUR NEXT ELECTRONICS PROJECT

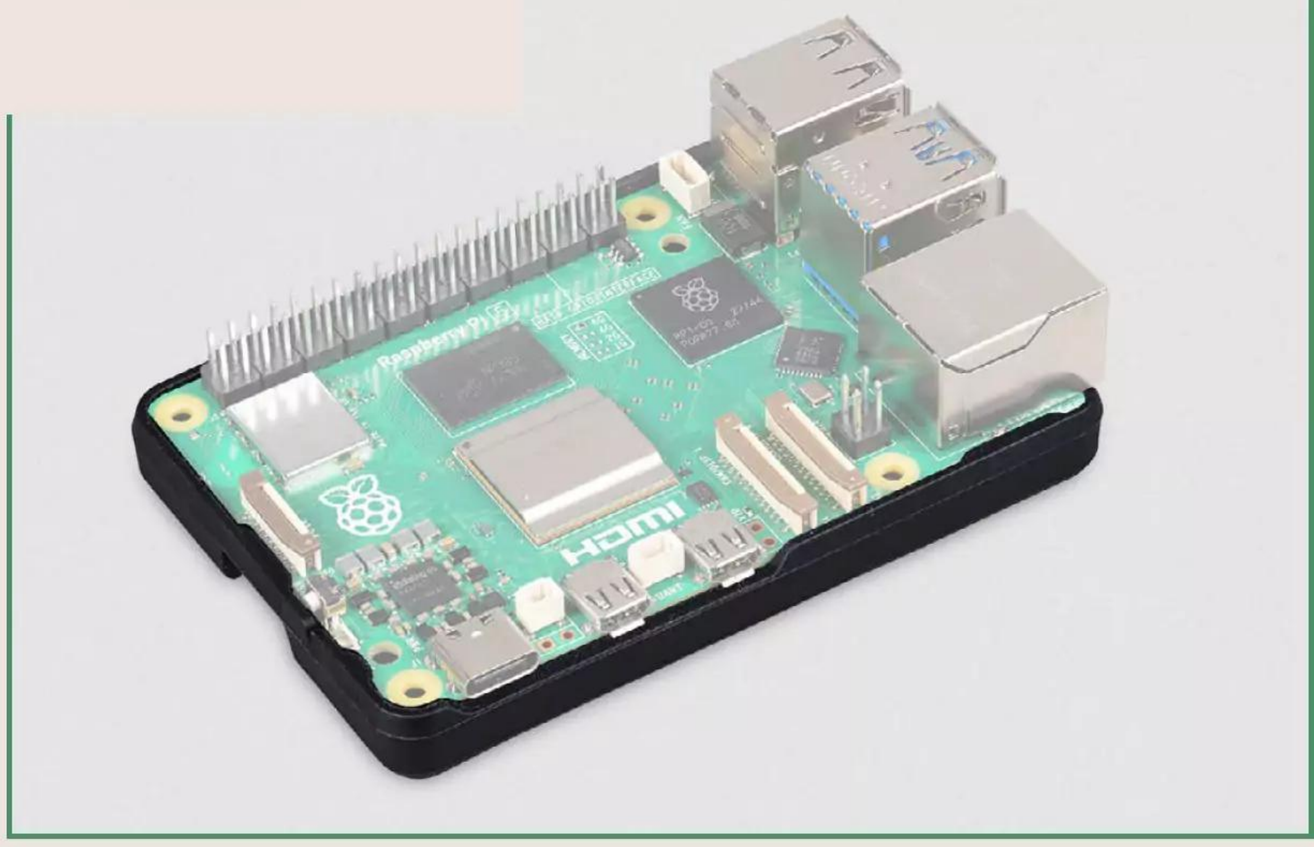
There are lots of small circuits you can make with a handful of inexpensive components. To start, buy some jumper wires (pin-to-pin, socket-to-pin), a breadboard, LEDs (pictured), resistors, buttons, and a buzzer. The CamJam EduKit and Simon Monk's Project Box are affordable starter kits with instructions (see reviews in issue 165).

Code Club has a collection of fun physical computing projects, including a reusable party popper, a music box, and a whoopee cushion. Find instructions and inspiration at **rpimag.co/projects**.



PROBLEM PREVENTION

To ensure a trouble-free experience using Raspberry Pi, follow this advice for your setup



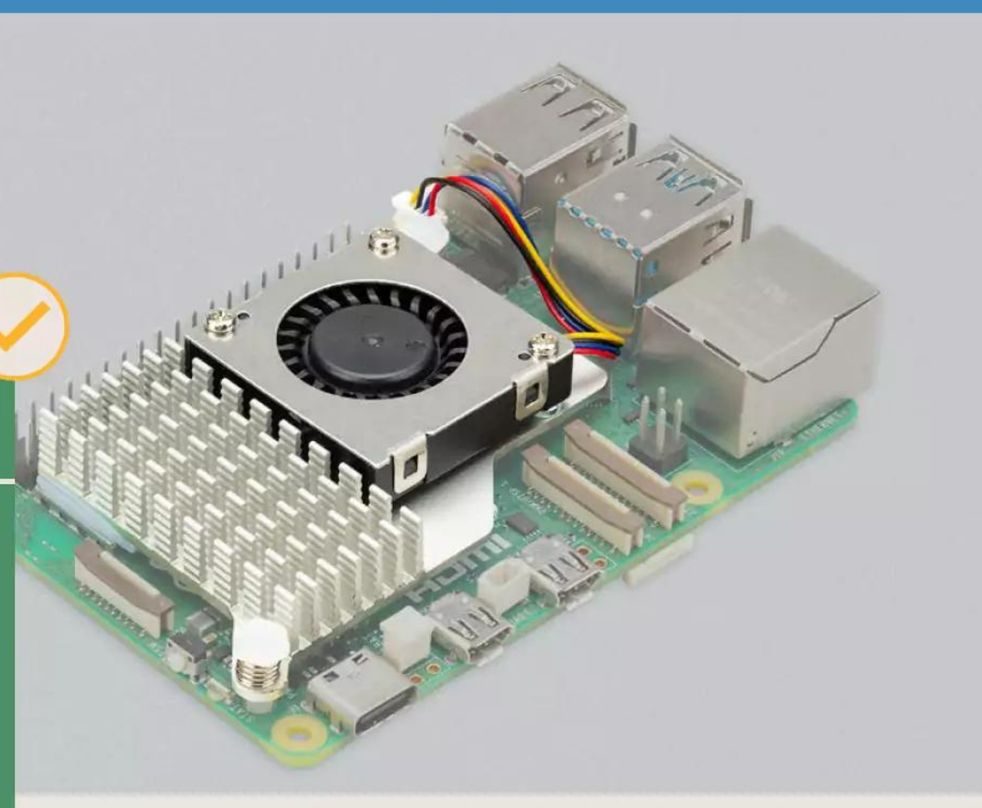
DO USE A CASE

A dropped paper clip on the GPIO pins could be fatal for Raspberry Pi, so a case is a wise investment. Many designs are available. The official Raspberry Pi case has a lid that you can remove to access the GPIO pins for your projects. There's also an official 'bumper': a silicon base that gives you access to all sides.

DON'T USE A MOBILE PHONE CHARGER

Your mobile phone charger might plug into Raspberry Pi, but it probably won't deliver the stable power required. Newer Raspberry Pi models need more power than some older ones, too, so you sometimes need to upgrade the power supply alongside Raspberry Pi. Raspberry Pi makes USB-C power supplies at 27W for Raspberry Pi 5 and peripherals, 15W for Raspberry Pi 4 and 400, and 12.5W for other models.





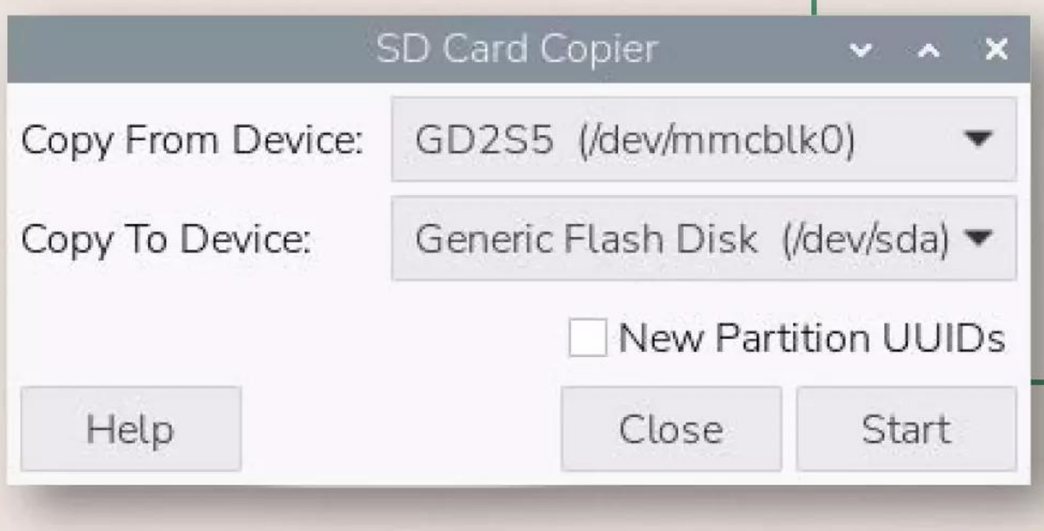
DO USE A FAN AND HEAT SINK

If you're pushing the performance of Raspberry Pi, adding a heatsink and a fan can avoid any problems caused by overheating. The Raspberry Pi Active Cooler for Raspberry Pi 5 is software-controlled, so the fan only activates when required. You can still access and use the GPIO pins with it.

"BACK UP YOUR DATA"

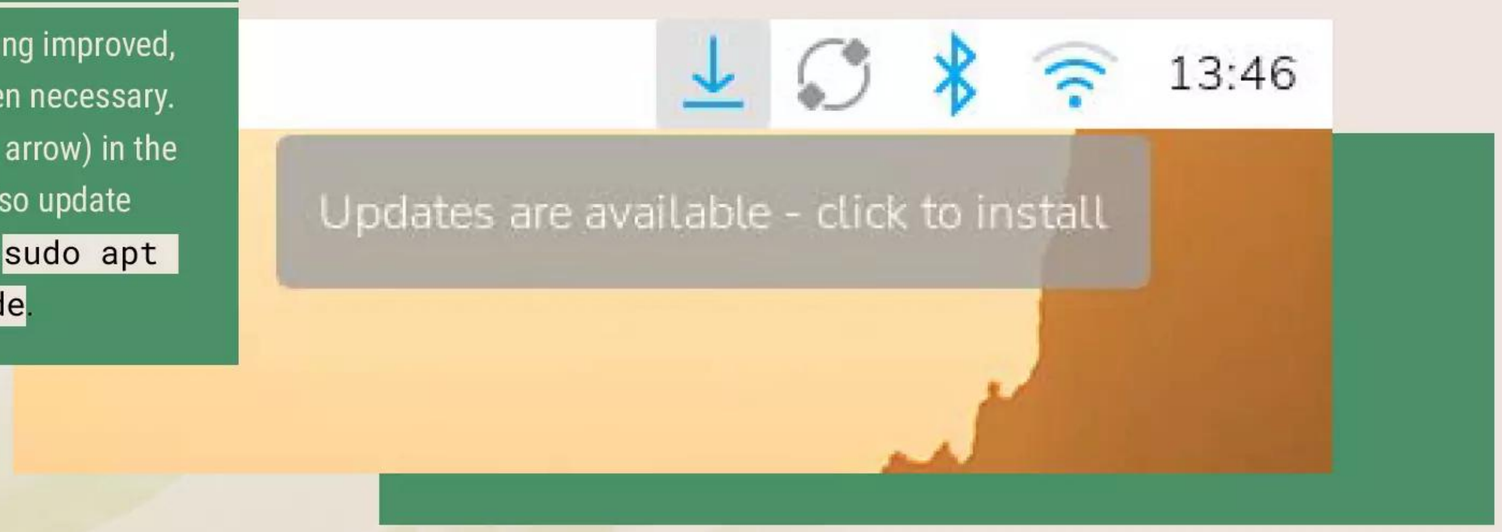
DON'T LOSE YOUR DATA

No matter what computer you use, you should back up to protect your data. You can insert a USB flash drive and use File Manager to copy your files to it. To keep a safe copy of everything, including your software and settings, use SD Card Copier. It's in the Accessories section of the Raspberry Pi menu. You can back up to a USB flash drive, or to another microSD card if you have a USB card reader.



DO KEEP THE OS UPDATED

Raspberry Pi OS is continuously being improved, including with security patches when necessary. Look out for the update icon (down arrow) in the top right of the desktop. You can also update from the command line by entering `sudo apt update && sudo apt upgrade`.



DO CHECK YOUR CABLES

If something's not working, check your cables and connections. Try swapping out the HDMI cable if the screen's not working. It's rare, but not unheard of, for incompatible peripherals such as keyboards to cause problems, so try swapping them too if you're getting unpredictable results.

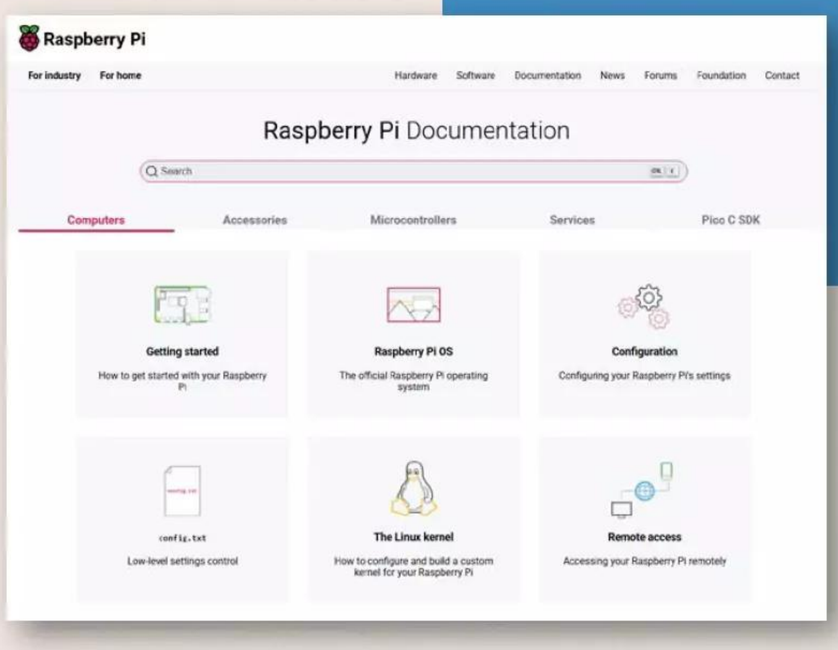


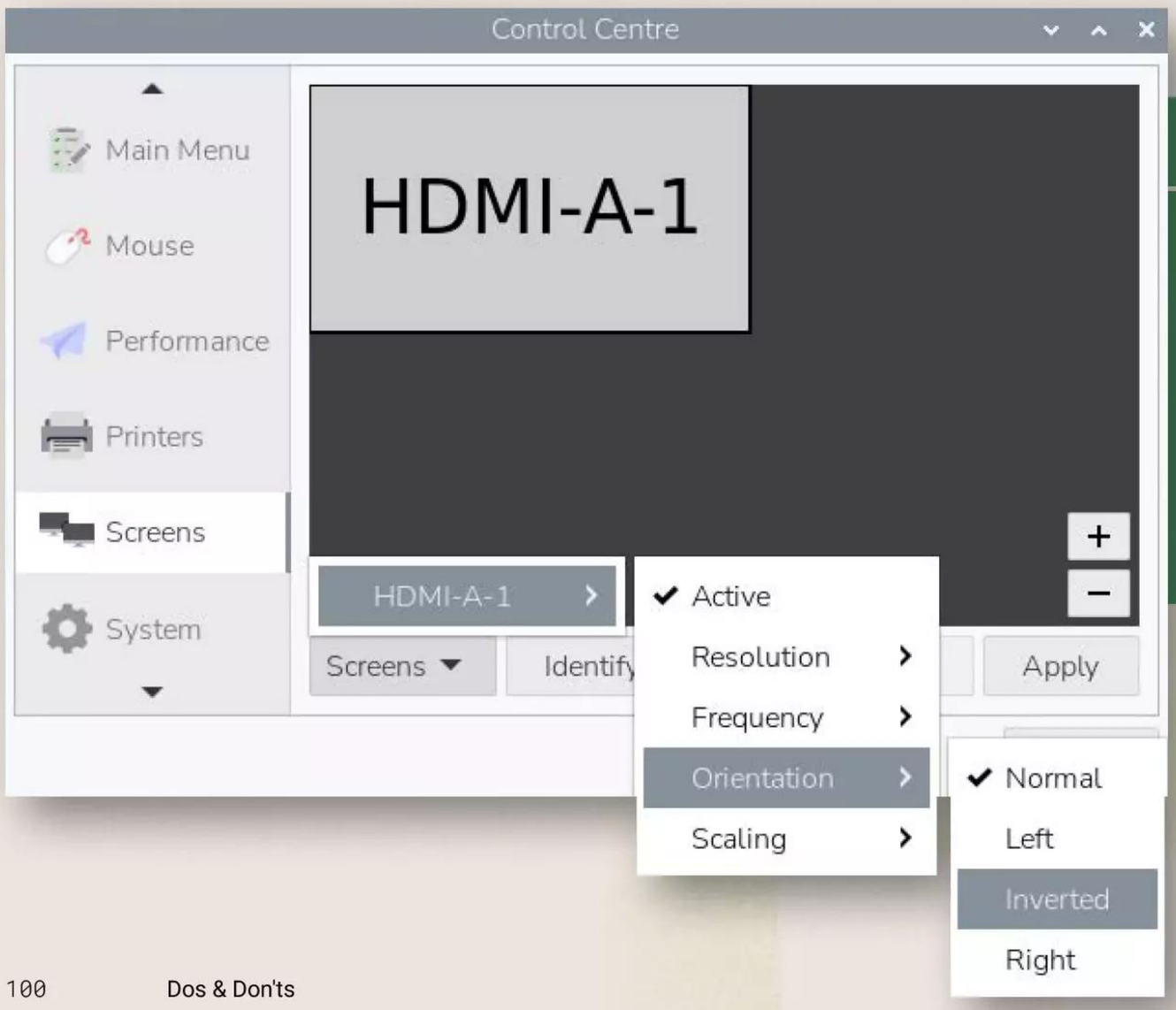
DO USE A RELIABLE MICROSD CARD

Your microSD card will store your data and software, so pick a brand you trust. For the best performance, get a Class A2 microSD card, such as the official Raspberry Pi one. We recommend 32GB to install the desktop and have enough room for your files and software.

HELP!

For most people's needs, Raspberry Pi is easy to set up. You insert the microSD card, plug in a USB keyboard and mouse, connect a HDMI monitor, and power it up. If you encounter difficulties or want to use a less standard configuration, see the online documentation at **rpimag.co/docs**. It includes headless setup, changing the boot order to try SSD first, and advanced settings in the configuration file (which now lives at **/boot/firmware/config.txt**).





DO OPTIMISE MONITOR SETTINGS

It's a good idea to turn your monitor on before you turn on Raspberry Pi so it can be detected as Raspberry Pi boots. HDMI monitors usually work without needing additional configuration, but if you encounter problems or have special requirements, go to the Screens section in Control Centre. There, you can change the resolution, orientation (for rotated displays), and scaling.









- ***Get game design tips and tricks from the masters***
- *Download and play game examples inspired by classics*
- ***Learn how to code your own games with Pygame Zero***
- *Explore the code listings and find out how they work*

Code the Classics Volume II not only tells the stories of some of the seminal videogames of the 1980s, but shows you how to create your own games inspired by them using Python and Pygame Zero, following examples co-programmed by Andrew Gillett and Raspberry Pi founder Eben Upton.

Available now: **rpimag.co/store**

ONLY THE **BEST**

LED lighting

Shining a light on the best LED lighting controllers and displays

By **Phil King**

We last did a roundup on LED devices back in issue 147. Since then, a few new products have appeared on the market while other existing ones have been upgraded with more powerful processors and/or the addition of wireless connectivity.

The most common type of addressable RGB LEDs are NeoPixels (aka WS2812B LEDs). We start off by looking at a kit with a string of them placed inside a glass bottle, controlled by a Raspberry Pi RP2350-based driver board.

We also try out a couple of RGB LED matrix displays. These are ideal for digital signage and animations. Then there are generic HUB75 matrix panels that can be daisy-chained together to create a larger display. For these you'll need a driver board, and we look at a couple: one powered by RP2350 and the other for use with a Raspberry Pi computer.

Finally, we have some miniature fun with the Tiny FX W, ideal for lighting Lego scenes and the like.

Wireless Plasma Kit

Pimoroni | £30 / $33 | pimoroni.com



◀ Lighting in a bottle that you can control via the internet

This cool-looking IoT mood light in a bottle is available in three shapes: cube, star, or spooky skull (the one we tested). Whichever you choose, it comes with a cork and a 5m string of 50 WS2812/NeoPixel-compatible addressable RGB LEDs to put inside it – you may need a pencil to poke them into the nooks and crannies.

To drive the LEDs, the kit contains a Plasma 2350 W. As featured in our roundup of RP2350-based boards (issue 164, **rpimag.co/164**), this is easy to program in MicroPython or CircuitPython. You just need to connect the three wires from the LED string to the Plasma's 5V, ground, and data screw terminals. You can then stick the board to the outside of the bottle with a couple of Velcro dots.

The Plasma 2350 W also benefits from wireless connectivity (Wi-Fi and Bluetooth), which means you can control your LEDs' colour with the Cheerlights API, for instance. Some other code examples are provided in the GitHub repo, including lighting that's dependent on the current weather or time, plus some spectacular rainbow, sparkle, and fire effects.

Verdict

It looks great on your shelf and you can program it with your own effects.

RGB LED Matrix for Raspberry Pi Pico (64×32)

Waveshare / The Pi Hut | £41 / $56 | waveshare.com / thepihut.com



This nice-looking RGB LED matrix has a few bonus features up its sleeve, including five side buttons; an RTC; temperature, light, and IR sensors; and even an IR remote. Note that you can't use Pico 2 with it, however, due to the lack of firmware support.

You plug a PCB into the back of the matrix display, then connect a couple of wires (which need cutting and stripping) from the latter to screw terminals on the PCB. You can then insert your Pico into the female headers, with pogo pins connecting to its test pads to extend its micro USB port to the PCB edge for power and programming. An acrylic rear panel is attached with four magnetic screws. There's also an optional diffuser panel you can stick to the front of the matrix.

Waveshare has provided some demo software with code examples in C/C++ and CircuitPython, including a clock, fruit machine reels animation, and scrolling text. You can learn from those – the online documentation is a little unclear.

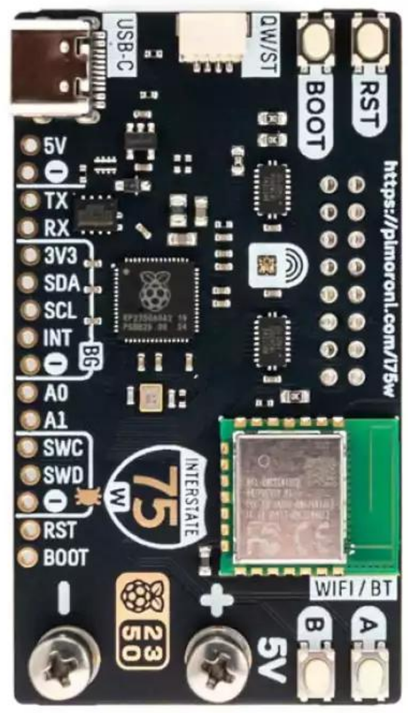
Verdict

While billed as a digital clock, it's a versatile RGB LED matrix for your Pico.

▲ The RGB LED matrix has several sensors, a real-time clock, and even an IR remote

Interstate 75 W

Pimoroni | £20 / $22 | pimoroni.com



▲ The board also features a breakout header and Qwiic/STEMMA QT port

Powered by RP2350, this is an all-in-one driver board for HUB75-style RGB LED matrices. HUB75 is a standard 16-pin interface protocol used to control modular panels, so you can chain them together to create larger displays, such as for scrolling signage or even a video wall.

The Interstate 75 W supersedes the original RP2040-based model and adds wireless capability (Wi-Fi and Bluetooth), useful for fetching data. Just plug it into the HUB75 data-in connector on the rear of a panel. With a suitable 5V power supply connected to its USB-C port, you should be able to power a couple of smallish (e.g. 128 × 64) panels via a power cable connected to the board's screw terminals. Or you could power your panels independently.

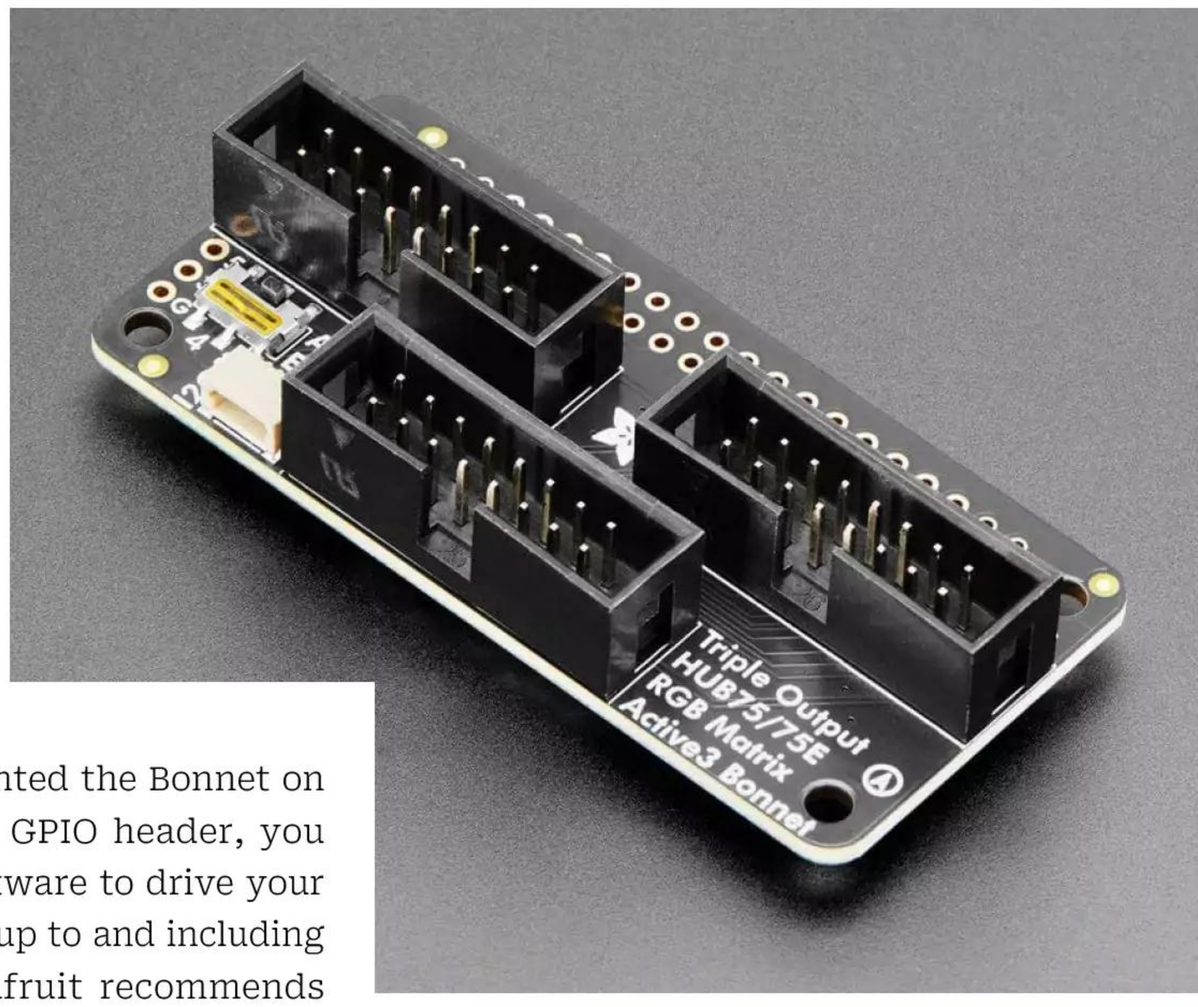
With the MicroPython firmware installed, you can connect via USB to a computer and try out some of the code examples. These include polygon/vector animations, album cover art, rainbow and fire effects, GIF playback, a maze game, and analogue-style vector clock.

Verdict

A fairly easy way to get started controlling HUB75 LED matrix panels.

Triple LED Matrix Bonnet for Raspberry Pi

Adafruit / The Pi Hut | £10 / $13 | adafruit.com / thepihut.com



A 'Bonnet' for Raspberry Pi, it lets you control up to three HUB75 panels in parallel (aka an 'active3' pinout) – or even three strings of daisy-chained panels. It supports panel sizes of 16 × 32, 32 × 32, 32 × 64, and 64 × 64. Note that you'll need to power the panels independently, however, as the board has no power output: a 5V 10A+ supply is recommended (for about 4A per panel), as are power distributions bars to make for easier wiring for multiple panels.

Once you've mounted the Bonnet on your Raspberry Pi's GPIO header, you have a choice of software to drive your panel(s). For models up to and including Raspberry Pi 4, Adafruit recommends Henner Zeller's rpi-rgb-led-matrix driver (**rpimag.co/rpirgbledmatrix**), which has Python bindings and code examples (as well as C/C++). For Raspberry Pi 5, Adafruit's own Blinka-based Piomatter library can take advantage of its PIO functionality (**rpimag.co/adarpi5matrix**).

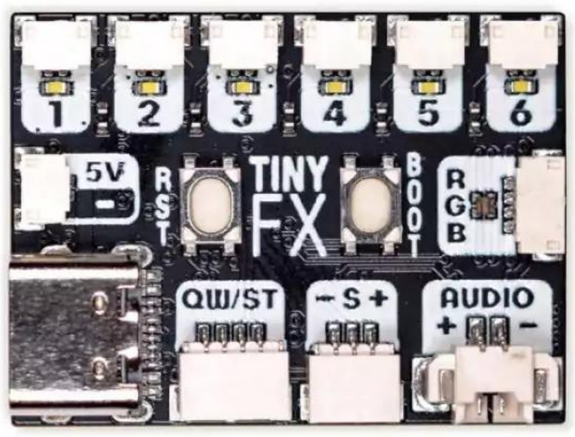
▲ The board has three 16-pin output connectors for HUB75 RGB LED matrix panels

Verdict

Control three RGB LED matrix panels in parallel with this Bonnet.

Tiny FX W

Pimoroni | From £20 / $22 | pimoroni.com



This multifunctional RP2040-based board is indeed tiny, measuring just 31.2 × 23.2 × 6.7mm. It's aimed mainly at Lego (or other construction set or model-making) fans, to supply miniature lighting via mini single-colour LED dots, noodles, and/or RGB LEDs. Note that it mainly uses itty-bitty JST-SUR connectors with a 0.8mm pin pitch.

In addition, it has connections for a battery, mini speaker (there's an on-board 3.2W I2S mono amplifier), digital or analogue sensor, and Qwiic/STEMMA QT breakout. If you buy the Tiny FX W starter kit version (£40 / $44), you get a speaker, six LED dots, five RGB LEDs (with expansion board), 3× AAA battery holder, PIR stick, and cables.

A standard Tiny FX board was used to light a miniature room scene in our issue 158 tutorial (**rpimag.co/158**). As the name suggests, the Tiny FX W adds wireless functionality, opening up even more possibilities. Code examples include Cheerlights and random colours/patterns via the internet.

Verdict

More than just an LED lighting controller, and very small.

▲ Now with added wireless functionality, this tiny board can control mini LEDs and more

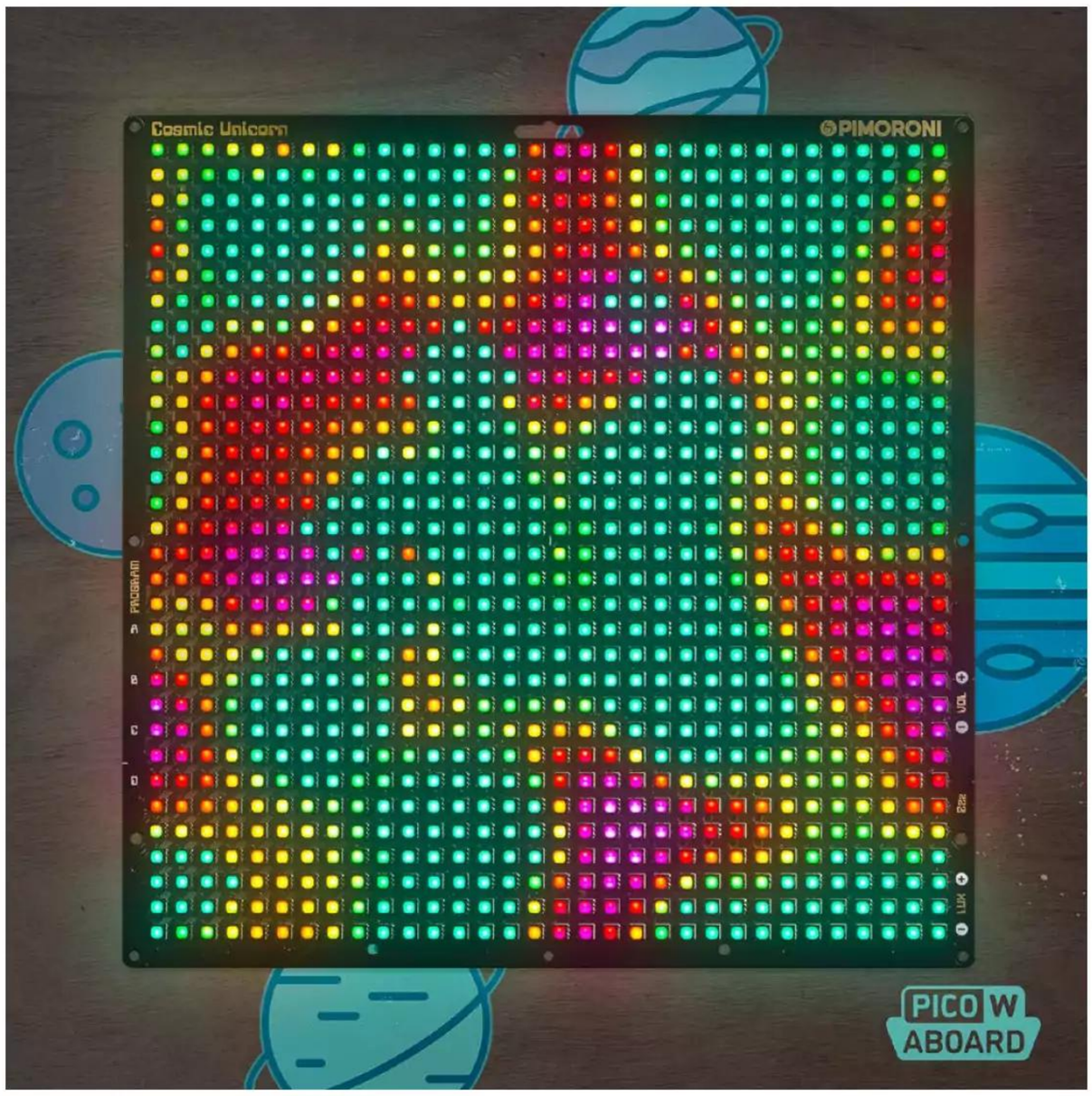
Pico 2 W Unicorn

Pimoroni | From £39 / $43 | pimoroni.com

This standalone RGB LED matrix comes in three sizes/shapes: Stellar (16 × 16 LEDs), Galactic (53 × 11), and Cosmic (32 × 32). They're like the previous Pico W Unicorn models, but now equipped with a Pico 2 W – pre-soldered to the rear – which provides extra processing power and double the on-chip SRAM (520kB) and flash storage (4MB). It's programmable with MicroPython or C/C++.

The LEDs are 'squircular' (squares with rounded corners) and very bright. So you might want to get the optional diffuser kit to shield your eyes while giving a neat bokeh-style blur effect. As on the original models, the LEDs update at a super-fast rate – 300fps at 14-bit precision – so there's no visible flickering, even when you record them on video.

You can also connect an optional LiPo battery pack for portable power. There's even a mini speaker powered by an on-board amplifier. So you can combine light and sound, as we did in our issue 129 tutorial (**rpimag.co/129**), based on the original Cosmic Unicorn, for a Bluetooth music visualiser.



▲ The Unicorn comes in three sizes. This is the larger Cosmic (32 × 32 pixel) version

Verdict

Powered by Pico 2 W, the Unicorn is an easy-to-use LED matrix.

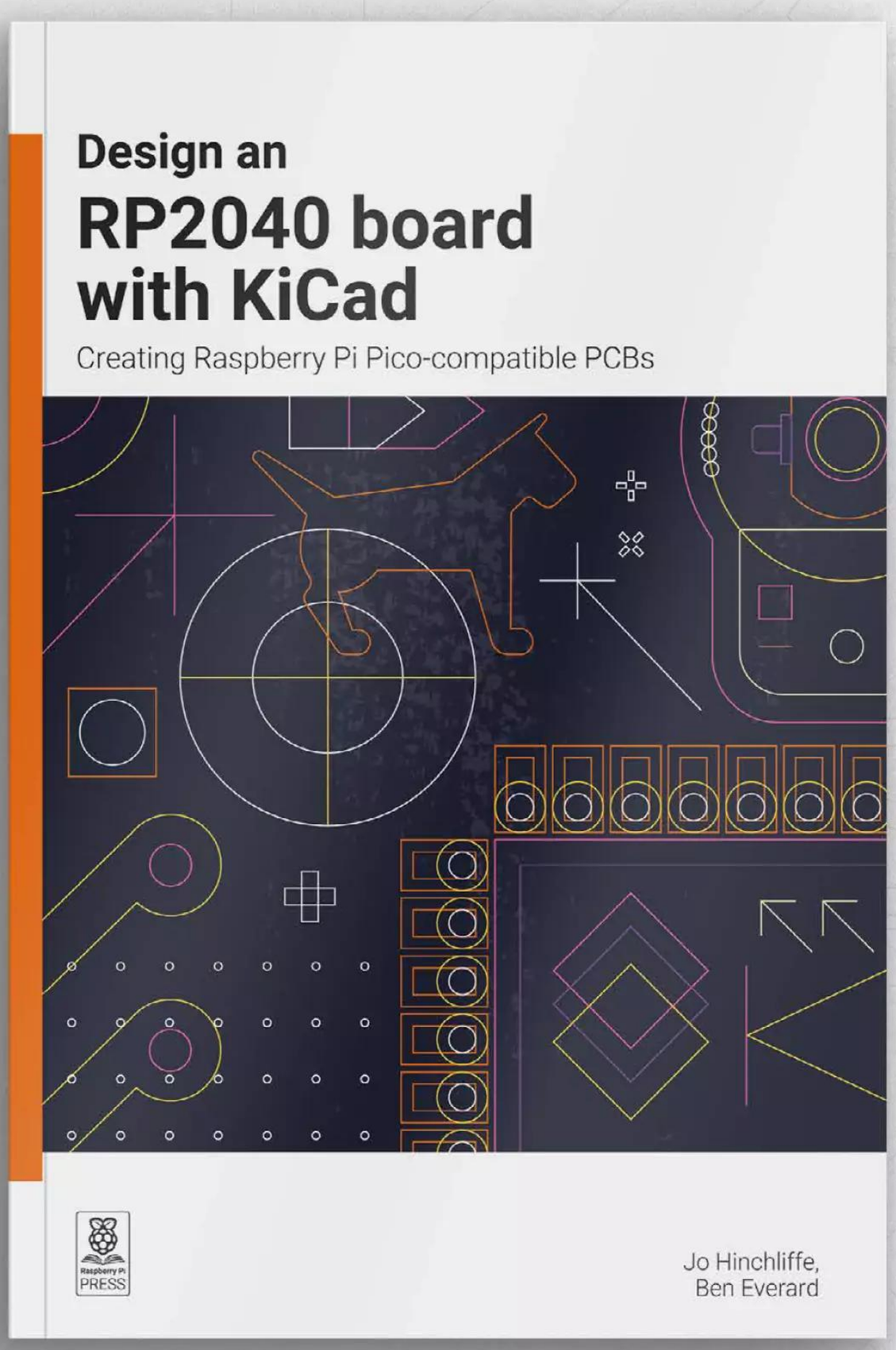
NEOPIXEL RING

Adafruit / The Pi Hut | £16 / $22 | adafruit.com / thepihut.com

As well as on reels or strings, you can get addressable LEDs in other form factors – including straight strips, curves, and rings. Adafruit makes NeoPixel rings in several sizes, controllable with CircuitPython. They're great for wearables and decorations.



▲ This NeoPixel ring has 24 RGB LEDs



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

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

Buy online: rpimag.co/kicad2040

Pironman 5 Pro Max

This ultra-cool case adds a touchscreen, speakers, and twin NVMe slots. By **Phil King**

SunFounder **rpimag.co/pironman5promax** £126 / $146

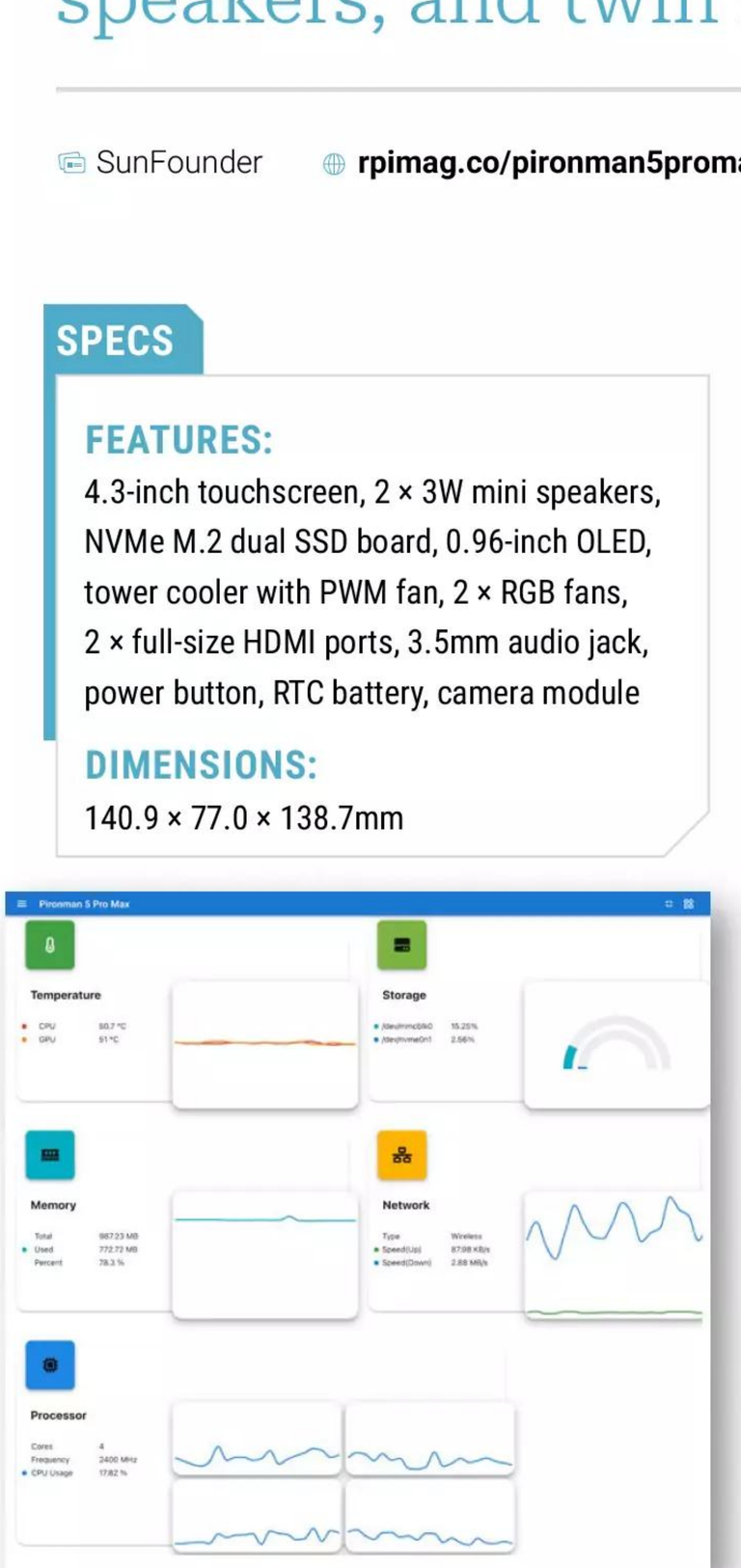
SPECS

FEATURES:

4.3-inch touchscreen, 2 × 3W mini speakers, NVMe M.2 dual SSD board, 0.96-inch OLED, tower cooler with PWM fan, 2 × RGB fans, 2 × full-size HDMI ports, 3.5mm audio jack, power button, RTC battery, camera module

DIMENSIONS:

140.9 × 77.0 × 138.7mm



▲ A web dashboard shows all the vital system stats and gives access to various settings



◀ The RGB fans light up the case interior while the 4.3-inch touchscreen proves useful

The Pironman 5 Pro Max is the latest addition to the line-up of mini tower PC-style cases from SunFounder – all of which feature RGB lighting and multiple fans, so they look cool while keeping your Raspberry Pi 5 cool.

We reviewed the original Pironman 5 back in issue 145 (**rpimag.co/145**) and were impressed by its looks and cooling performance. In addition, we checked out the smaller Mini model in issue 159 (**rpimag.co/159**). The Pro Max is the same size as the standard Pironman 5, but adds a few extra features. The most obvious is a 4.3-inch touchscreen attached to the translucent acrylic panel. There are also two mini speakers and a dual-slot M.2 NVMe board to add two SSDs (not supplied), or one SSD and an AI module (e.g. Hailo-8). You even get an own-brand camera module and USB mic thrown in.



Audio Amplifier Board



Dual NVMe PIP

◀ The USB extender board includes an amplifier for the two 3W mini speakers

Assembly line

The kit assembly process seems a little daunting at first, with a box full of components – including a bewildering array of different-size standoffs and screws – and you'll need a spare two or three hours. An illustrated flatpack-style guide takes you through the 37 steps. Sadly, there was no online video at the time of writing, although the process is similar to the Max model but with the addition of the touchscreen.

Supplied in separate plastic packets, the numerous parts are well labelled. The first few steps involve fitting your Raspberry Pi 5 to the metal base plate, along with extender boards to reroute ports and the microSD card slot to the case exterior. As with the standard Pironman 5, you get full-size HDMI ports, but also a 3.5mm audio jack.

You'll then add the mini speakers, and place a jumper on the board to determine whether they're always active or only when the HDMI0 port or audio jack aren't being used – the speakers can't be used with them simultaneously. A coin cell battery is supplied for the real-time clock.

Next, you connect FPC ribbon cables to Raspberry Pi 5 for the camera and touchscreen, so you need to ensure they're inserted securely. Thermal pads are then added before placing the tower cooler fan on top of Raspberry Pi 5. Along with the two RGB fans added later to the side of the case, this provides highly effective cooling.

The NVMe board features two M.2 slots for your SSD(s) and/or AI module. One downside here is that due to the ASMedia ASM1182e switch used the PCIe connection is limited to Gen2, which is roughly half the speed of Gen3.



◀ Raspberry Pi 5's ports are extended to the case exterior, including two full-size HDMI ports

> *The desktop soon appears on the touchscreen, which proves responsive and large enough to be practical*

Final steps

The final assembly steps involve fitting the GPIO extender, two RGB fans, mini OLED, power switch, and touchscreen. The camera fits into a tiltable holder on the top of the case, which does spoil the aesthetic somewhat, especially with the trailing cable on the outside of the case.

Upon booting up Raspberry Pi 5, the desktop soon appears on the touchscreen, which proves responsive and large enough to be practical. An on-screen keyboard appears automatically or when you tap its icon. You can long-press the screen to emulate a right click.

To get the RGB lighting working, you just need to install the Pironman 5 software. This also enables a handy web dashboard to monitor system performance and change settings.


▲ The assembly process: here, we've slotted a Hailo-8L module and SSD (not supplied) into the dual NVMe board

Verdict

A superb-looking cooling case with the bonus of a touchscreen and dual NVMe slots.

9/10



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

From aerial display specialists to prototyping tools, Raspberry Pi-accredited hardware enables it all. **Rosie Hattersley** highlights what's in the Powered By Pi roster

As summer looms into view here in Europe, weekend skies are filled with birds, kites, and buzzing craft directed aloft by eager pilots. Where once drones were heavy, clunky and chewed up batteries in mere minutes, modern UAVs (unmanned aerial vehicles) are elegant, agile and can remain in the sky for long periods. There are now drones for everything from underwater research to aerial filming, as well as simply a modern take on flying a remote-control plane. Paired with a lightweight camera module, they can be used to take stunning photos and capture incredible video footage. HighGreat, which we feature in this issue of Powered by Pi, uses them in coordinated light shows, not least at the Beijing Olympics in 2022. With a global audience to admire its engineering prowess, it's no wonder the company also runs a commercially successful education and engineering scheme, as well as offering unmanned aerial craft that customers can build themselves.

Getting a start in all things electronic is the very essence of specialist stores such as our very own Pimoroni, which continues to entrance us with HATs, accessories, and bespoke boards at pocket-money prices. Once you've mastered the essentials of Raspberry Pi and Python/MicroPython, it's immediately rewarding to experiment with sensors, connectors, lights and LCDs and simply see what your newly acquired coding skills can make them do. Pimoroni's Explorer board based around the RP2350 chip offers just this, functioning as an all-in-one electronics kit.

While sky-spanning spectacles featuring hundreds of drones show off the possibility of harnessing Raspberry Pi as a central controller, our microcontrollers can operate alongside the tiniest items. Read on to find out all about DigiCue and its role in helping amateur and professional snooker and billiards players achieve pinpoint accuracy. It's every bit as wonderfully nerdy as it sounds.

Find detailed information about the compliance regulations and testing procedures for every Raspberry Pi product at **pip.raspberrypi.com**.

Freya Vivariums

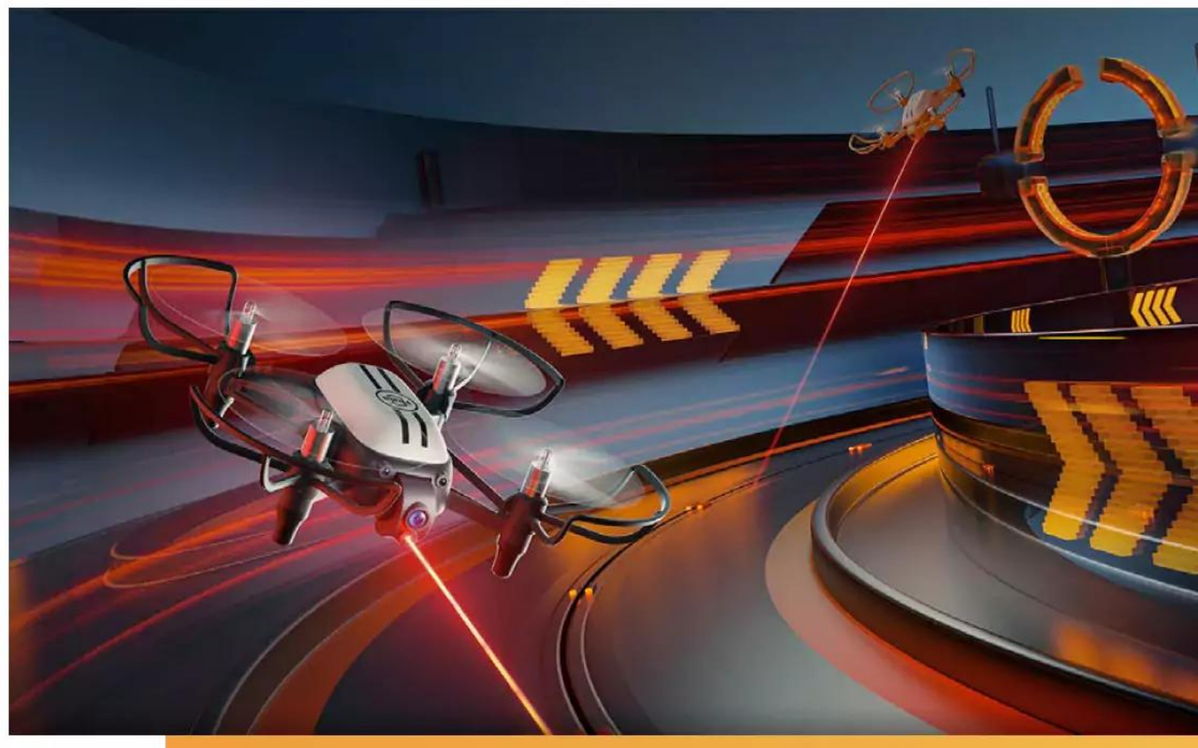
Belgium | freyavivariums.com



We have featured a fair few pets and reptiles over the years, including a specially constructed hibernation chamber for a much-loved tortoise. This reptile-focused setup extends the idea of the ideal living conditions for a reptile to coexist with humans. Freya Vivariums offers a whole slew of open-source equipment that can be purchased or 3D-printed. The resulting environment can be temperature, light, and humidity controlled via Raspberry Pi's lab-grade sensors, while the Perspex enclosure allows curious owners of all ages to observe the reptile sunning itself, strutting around its own domain, or hiding among lush and tasty foliage.

HighGreat Hula

China | rpimag.co/highgreat



HighGreat specialises in impressive drone formation light shows and spectacular performances that wow and inspire its audiences, including at the 2022 Beijing Winter Olympics. There is therefore a real interest among young tech fans in how to go about creating their own flying vehicles. The educational arm of the company features design competitions and AI coding courses, as well as its Fylo Edu+ flying machines and Hula drone complete with its own app. All this comes with full learning support and adheres to RoboAlliance standards, meaning Hula and Fylo are open source and can be controlled by Raspberry Pi. The educational programme for both vehicle types aims to promote a deep curricular understanding of artificial intelligence systems to equip students with the skills needed for their future in an intelligent society.

Pimoroni Explorer

UK | rpimag.co/pimoexplorer



Pimoroni's self-described 'electronic adventure playground' is a fantastic showcase for Raspberry Pi's RP2350 chip. The kit features a mini breadboard on which to wire up components, two servos and servo headers to power the pair of 60mm wheels, analogue inputs and GPIO pins galore, leads, jumpers, and I2C breakouts. While these take care of connectivity and motors, there is also a healthy provision of buzzers, lights, and a built-in speaker, plus a movement, light and environment sensor and 2.8in LCD to see what's going on. With all this in a £33 kit, it's no wonder the Pimoroni Explorer routinely features in our must-have Raspberry Pi accessories guides.

BitMechanics Pixel Pump

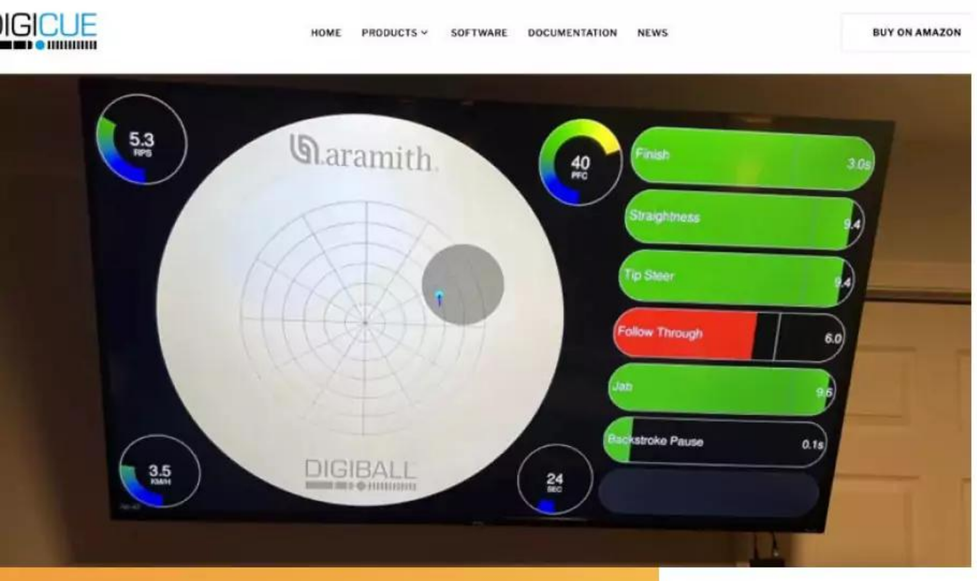
Germany | robins-tools.com



There is a lot to like about the Pixel Pump. As well as being based around Raspberry Pi RP2040 (so we thoroughly approve), it can be bought as a finished product or as a 3D-printable kit version, potentially saving on shipping costs as well as money, and meaning it can be produced in any colour for which you can get filament. If you're wondering what on earth it does, it's a vacuum pump for rapid prototyping and small batch production, suctioning up tiny components with ease, while avoiding issues of greasy or clumsy fingers. Enthusiastic owners of a Pixel Pump at Nottingham Hackspace use theirs as a manual pick-and-place aid akin to a fancy pair of tweezers. The Pixel Pump has modes to lift and lower 'on trigger', as well as using a foot pedal, and can optionally be controlled at a distance via USB.

DigiCue DigiCast TV Kit

USA | digicue.net/digicast.php



This Raspberry Pi-based TV Kit is specially designed to work with DigiBall and DigiCue stroke trainers, displaying the performance of multiple players on a large screen at 1920 × 1080p. Once connected via HDMI, the DigiCast software pre-installed on the SD card in the TV Kit loads automatically. The DigiCue trainer fits over the butt of a snooker or billiards cue and is recognised by the Raspberry Pi module in the TV Kit. Details are displayed on screen so the keen-to-improve player can see what they are doing well and areas to work on.

Angles and straightness are accurately plotted and represented, allowing ardent snooker and pool players to gain real insights into their cueing and improve their accuracy. It can even be used with carom billiards and is able to recognise additional cue balls and provide insights into the stroke. The DigiCue system provides a visual display of the amount of spin and curve applied to the cue ball and simulates its rotations per second and where on the ball the cue struck it. During play, the cue subtly vibrates to hint at a flaw in the player's stroke.

Presearch Node

Canada | presearch.com

Privacy and data storage have been hot topics for years, with cybersecurity experts warning users not just about tracking and hacking, but where you store your digital files. Private browsing modes and privacy-focused search engines such as DuckDuckGo can help avoid cookies sharing your every online move, but decentralised search offers another good option. Presearch promises unfiltered and untracked online searches, via nodes that operate as community servers. The software runs on a 64-bit Raspberry Pi computer. You can buy a system already set up in a custom case from US-based Coinmonster: **rpimag.co/presearchnode**). This 4 × 4in Presearch Node hardware is based on an 8GB Raspberry Pi 4 and comes with the Presearch Node software pre-installed on a 32GB microSD card as well as Raspberry Pi OS.

Given the Raspberry Pi 4 inside, it's little surprise to learn that the Presearch Node sports a Broadcom BCM2711 SoC with Arm Cortex-A72 cores alongside Gigabit Ethernet, 2.4GHZ and 5GHz wireless connectivity and Bluetooth 5.0, has two USB 3.0 and two USB 2.0 ports, plus two micro HDMI ports for a display. You will need a separate IP address for each Presearch node you run.



10 amazing:

game controllers

Up your game(play) with these Raspberry Pi compatible joypads and sticks

We know a lot of you love to use Raspberry Pi to play games – retro or 'technically retro but admitting it will make me feel too old' – and how you play them is also very important if you want to capture some of the feelings of yoof. Here are ten great ways to play your old faves.



01. 8BitDo Arcade Stick

Gates and switches

rpimag.co/8bdarcade

£75 / $67

This big, beefy arcade stick is perfect for fighting games. It's got storage for a cable and wireless dongle, and can be upgraded with official arcade parts to suit your needs.

02. Qw/ST pad

DIY flexibility

rpimag.co/qwstpad

£9 / $12

More for your custom builds than regular gaming, these Qw/ST pads were good enough to prototype playing the recent, official SNES re-release of DOOM.

03. 8BitDo 64

Now two prongs

rpimag.co/8bd64

£35 / $40

We reviewed this charming take on the classic and much misunderstood N64 controller last issue and it really captures the feel while upgrading it for modern sensibilities.

04. iCode Arcade Trackball

Command your missiles

rpimag.co/icodetrack

£110 / $150

A rare control scheme now, but one that was popular in arcades in the 1980s. This is one of the cheaper ways to get one in the 2020s.

05. Wired Xbox 360 controller

The old standard

Various

For 20 years the wired Xbox 360 controller has been the standard for plug-and-play with retro projects.

06. Regular controller

Hook it up

Various

Raspberry Pi operating systems are Linux distributions, meaning they're all built on the core Linux kernel which supports a large number of modern controllers.

07. Cheap USB controller

Affordable classic

rpimag.co/usbpad

£6 / $8

If you just want something that will let you jump through a (retro) game for the price of a nice (modern) coffee, a USB controller like this will gladly do the job.

08. 8BitDo Ultimate 2

High-tech perfection

rpimag.co/8bdult

£50 / $52

A very customisable controller that is extremely comfy and will never experience drift.

09. Retro-Bit Mega Drive pad

Relicensed classic

rpimag.co/rbmd

£13 / $18

For retroheads, the difference between the floaty D-pad of the Mega Drive and the rigid cross of the competition means these licensed (eight-button) recreations are very welcome.

10. 8BitDo Micro

Pocket sized

rpimag.co/8bdmicro

£23 / $25

This itty-bitty Bluetooth controller will connect to your portable Raspberry Pi gaming system as well as a few other devices.


03





02



06


04



07


05



09


10



08



Steve Davies

Meet the new member of the documentation team at Raspberry Pi

- **Name** Steve Davies
- **Occupation** Documentation Specialist
- **Community role** Documentation
- **URL** rpimag.co/docs

The official documentation for Raspberry Pi is really something to behold – incredibly detailed, in-house created info for everything Raspberry Pi and full of examples to help you too. We often find ourselves there when we have a little issue (usually with a couple tabs from the forum open too). Recently, a new member, Steve Davies, joined to help maintain and add to the existing documentation.

"I've lived in Cambridge most of my life, and Raspberry Pi is a household name here, so I've known about it for a long time," Steve tells us. "But I wanted to work here because my wife told me that it was a fantastic place to be – she contracted at the company and absolutely loved the experience."

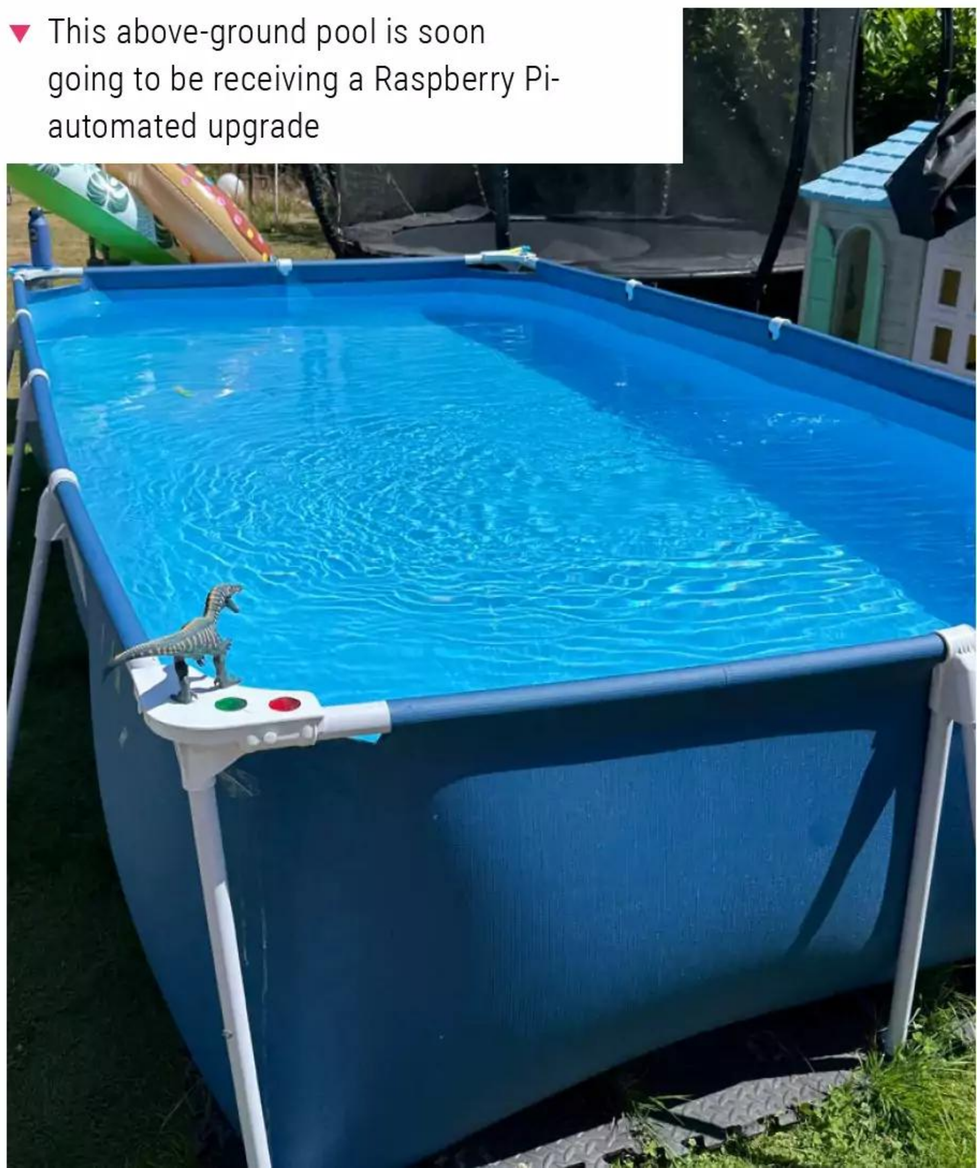
What is your history with making?

I'm not actually a techie. I suppose it's no surprise then that I love the simplicity and accessibility of Raspberry Pi!

What did you do before joining Raspberry Pi?

I worked for various soulless enterprise software development houses in London, creating documentation for financial services products that banks use to monitor their IT systems and maintain compliance.

▼ This above-ground pool is soon going to be receiving a Raspberry Pi-automated upgrade





▲ We find a lot of people enjoy retro gaming with Raspberry Pi

What have you made with a Raspberry Pi?

It's almost summer here, so the above ground swimming pool will soon go up! Each year, I have to manually check the pool temperature and turn on the heater, otherwise the kids complain it's too cold. This year, I'm building a Raspberry Pi configuration that will automatically take temperature readings and feed them into my AI agent. The agent checks the weather forecasts, then uses a couple of years' worth of collected weather and pool temperature readings to suggest the most cost-efficient heating schedule. I also want it to perform the daily and weekly chemical analyses – chlorine, pH, and CYA levels – but that might be a stretch. If anyone has a blueprint for this, please let me know!

What are some of your favourite things you've seen people make with a Raspberry Pi?

So many to choose from... but as an old-school gamer, the games emulators that take me back to my childhood are just brilliant.

Anything else you'd like to add?

My wife was right – this is a great company to work for.

> *I'm building a Raspberry Pi configuration that will automatically take temperature readings and feed them into my AI agent*

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. This is a great DIY way to make Raspberry Pi Pico compatible with anything you'd like

02. Oh, this is a very cool space use of Raspberry Pi computers

03. Game Gear Sonic 1 came out around the same time as the main Mega Drive version

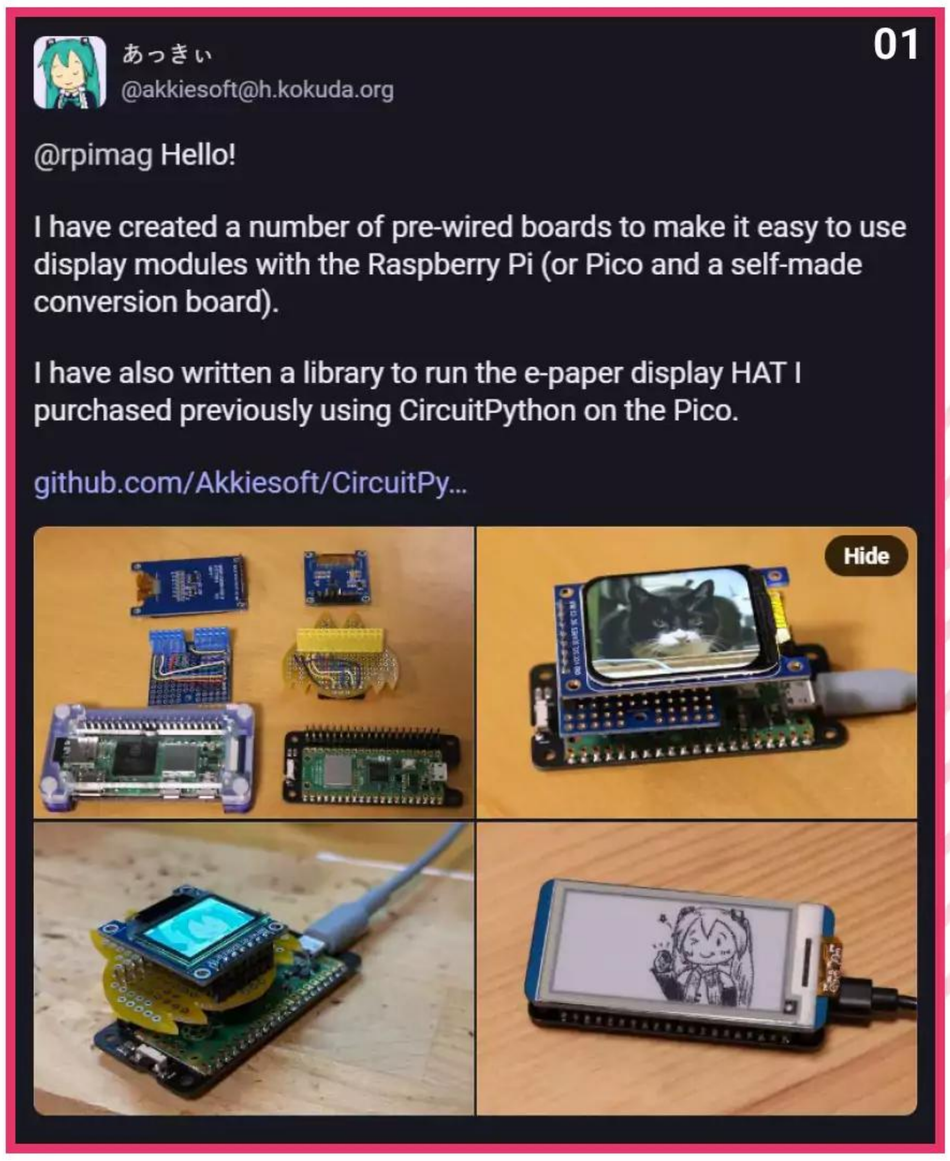
04. An excellent way to upcycle your old laptop – by making a maker paradise

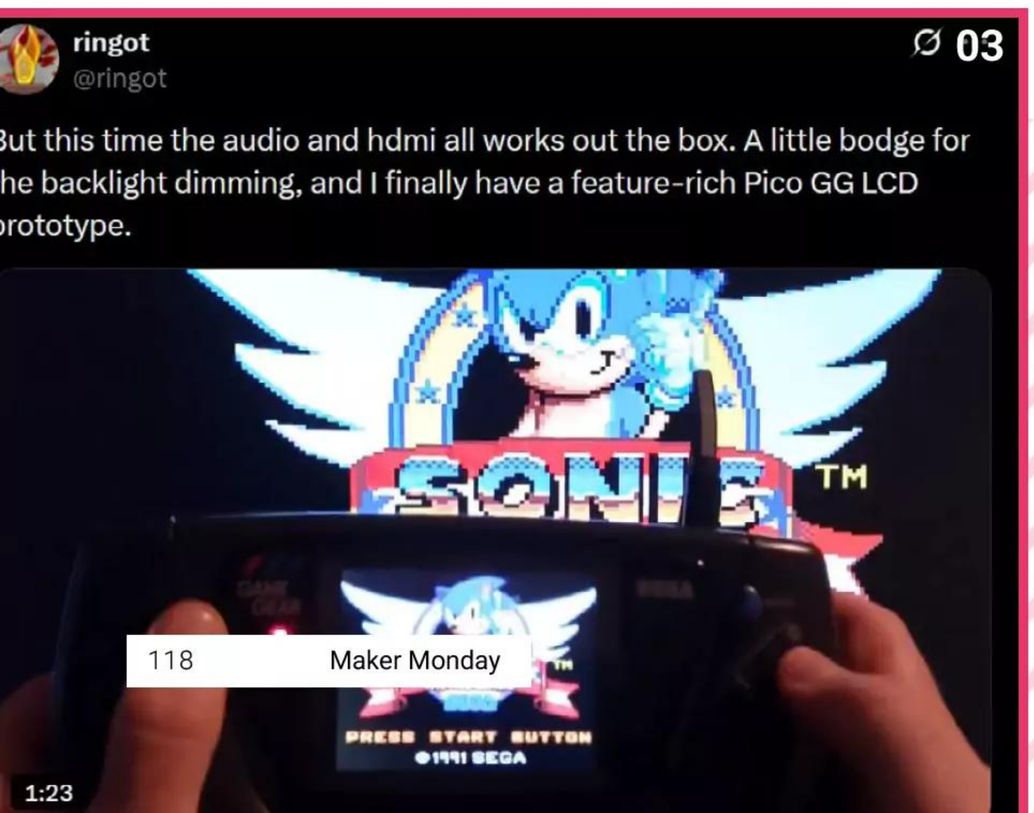
05. A great simple project if you have a smallish display lying around

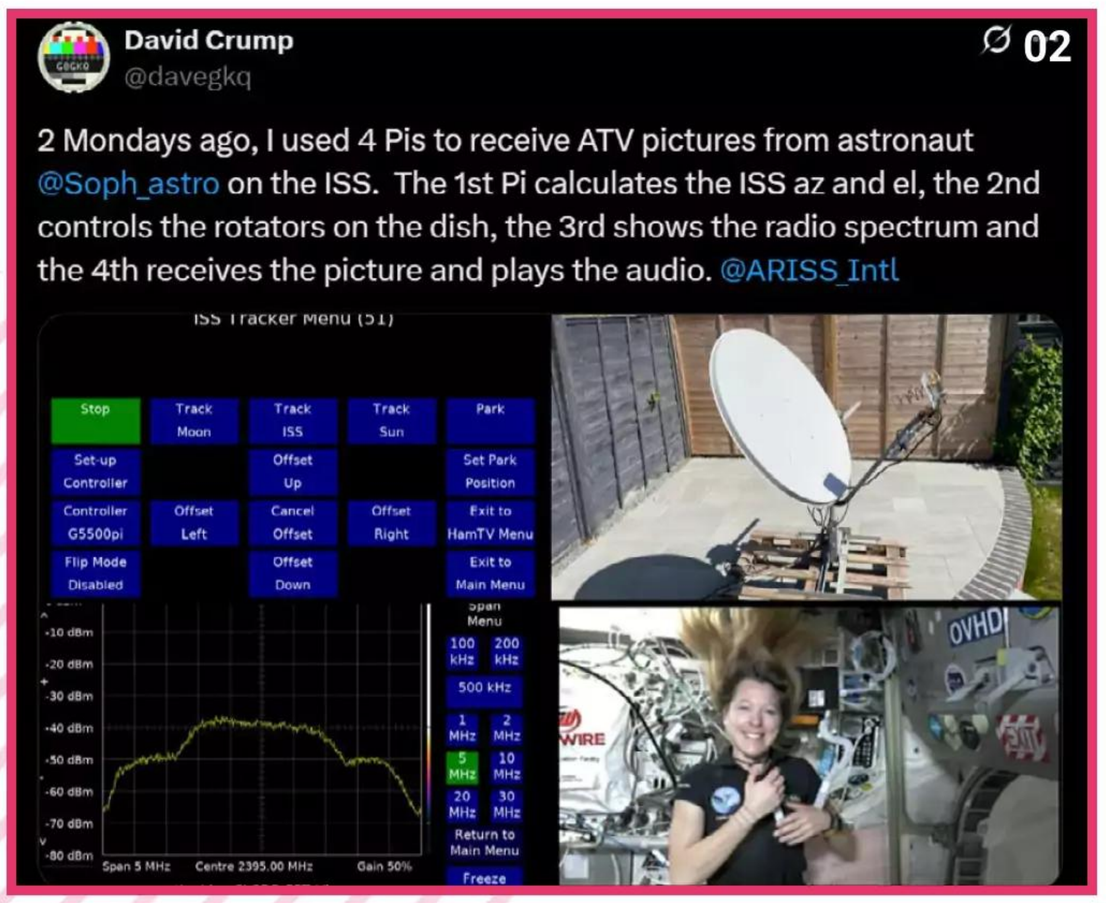
06. Nature conservation with Raspberry Pi? We love it

07. Maintain the illusion with a smooth cross gate opener

08. A safer way for kids to browse the internet







thecarolinedunn 28/04/2026

05

I made a DIY RPi Photo Frame. Here's the tutorial - youtu.be/2yzrW...



RickP_maker
@RickP04482661

04

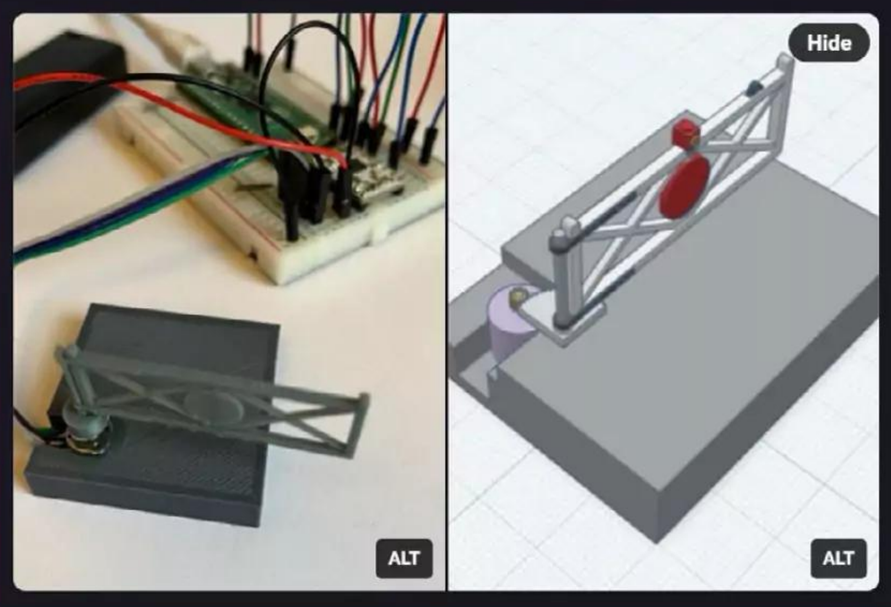
Old PC laptop is now a laptop with a Pi Zero, the breadboard is for projects



Pater Practicus
@paterpracticus@mastodon.world

07

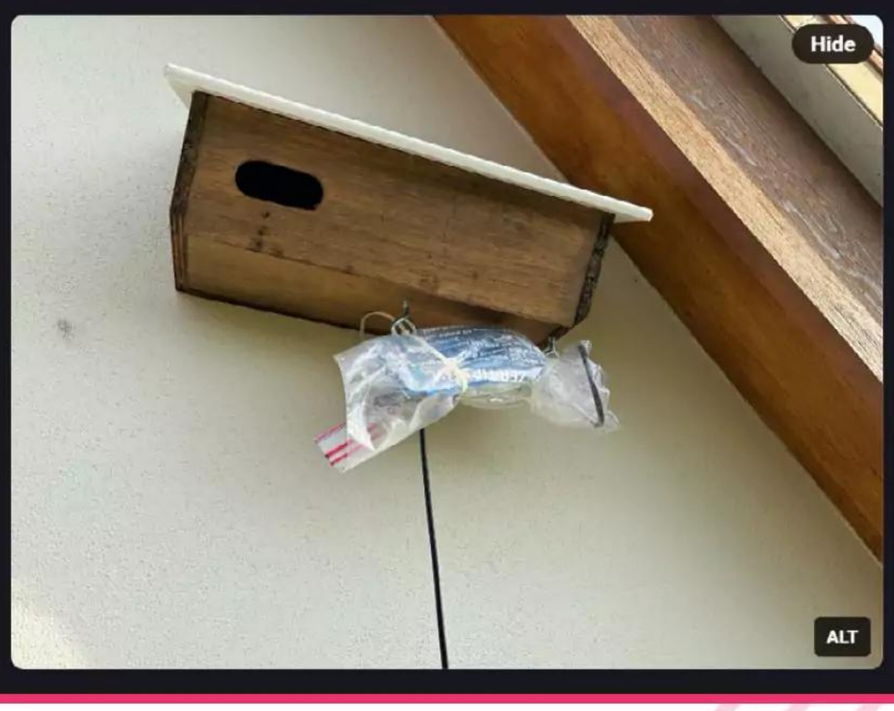
@rpimag The stepper motor option for my crossing gate automation is proving a bit jerky using direct drive, so I may have to sort out some 3D printed gears. #MakerMonday



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

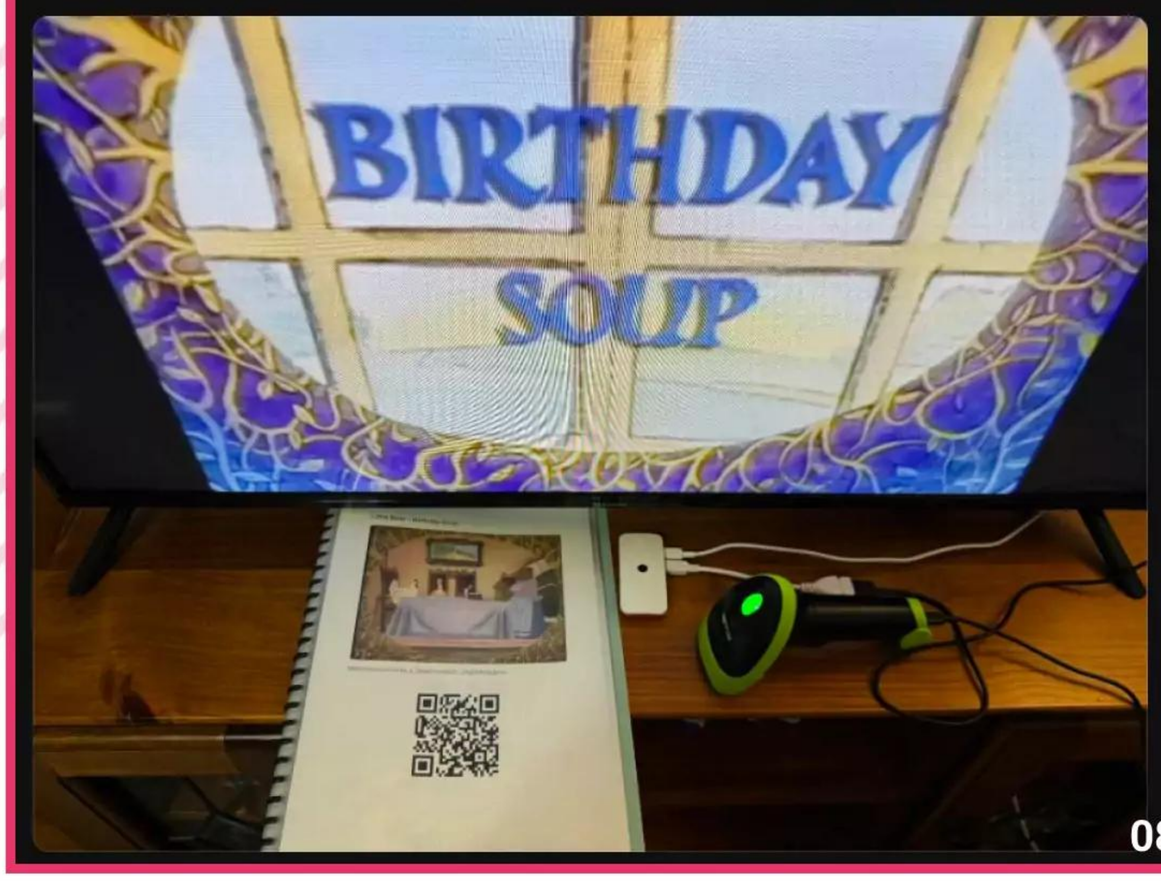
06

@rpimag I installed a Raspberry Pi media player below a Swift nest box on the side of my house. It plays recordings of Swifts nesting at dawn and dusk to try and attract first year nesters to the new site as part of a Swifts conservation project. #MakerMonday



tracyannebruce 2

A scanner on a Pi Zero reads from a printed booklet of QR-codes, sends the URL to a Pi 400 running Jellyfin, and plays on the TV. Based on Guy Dupont's project for kids to control videos without browsing YouTube. Codex/Python tracybruce.net/tools



08

Flight Watch

Using Raspberry Pi Zero to figure out what's flying overhead

As we often mention in these pages, folks often email us with information on projects they've made. This one tickled us – bored with checking a website to figure out what planes were flying overhead, maker Robin Shand made a watch to check the flights of course.

"Since starting work in the defence sector, I've taken more of an interest in aircraft, especially what's been flying overhead from the nearby MOD bases from my house," Robin says in his email.



▼ It reminds us of the communicators from classic sci-fi TV shows

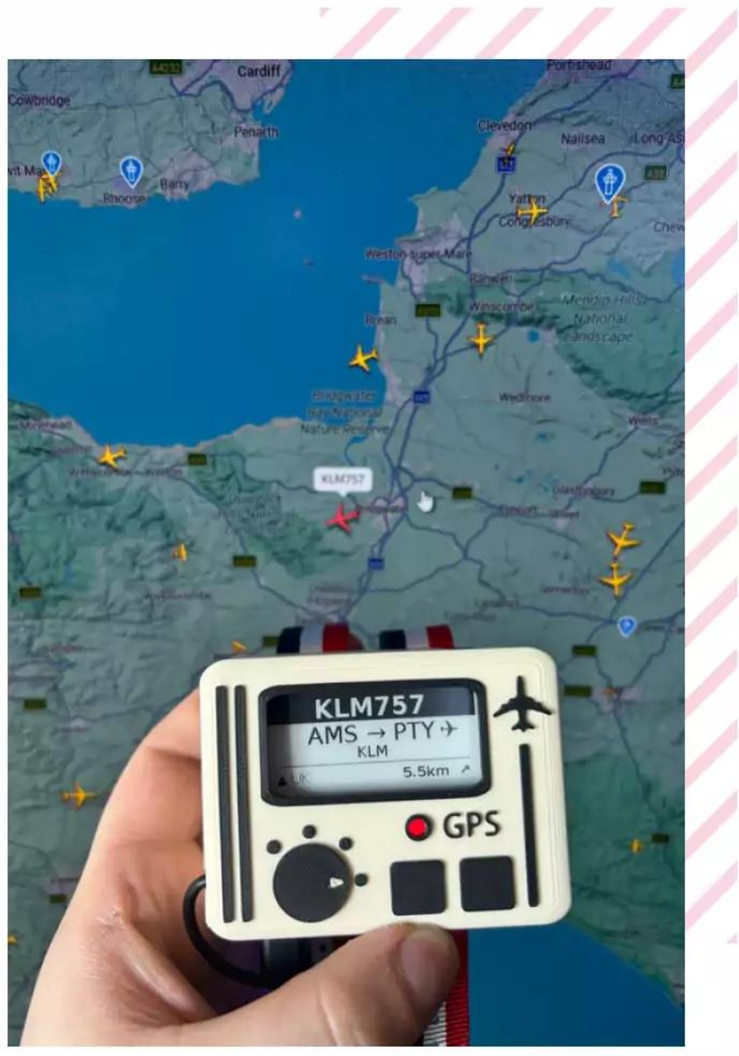
"I was always checking flightradar24 out of curiosity and this project was an attempt to bring that capability into a simple, wearable device."

Basically, the watch locates the wearer and shows them the nearest aircraft on the e-paper display. Robin explains it works thus:

- Built on a Raspberry Pi Zero running a Python script
- Uses an on-board GPS module to determine location
- Sends coordinates via a 4G LTE connection to the OpenSky API to identify nearby aircraft
- Enhances flight data (origin/destination) using AeroAPI
- Displays key flight details on an e-paper screen, refreshing every minute to balance performance, API usage, and battery life

"To make it fully portable, I designed and 3D printed the case from scratch and packed in a 5000mAh battery," he adds. "This gives it at least 12 hours of runtime (all I've tested so far)."

It is extra funny to us that it also does not tell the time. Beautiful, an excellent project, we love it anyway.



▲ Showing off that the watch works as advertised

Crowdfund this

Crowdfunding campaigns to keep an eye on

M5 CardputerZero



This 'computer for hackers' will run on the new Raspberry Pi Compute Module Zero and promises to be a pocket powerhouse, with the ability to have its functionality modified with M5 modules. It's not up on Kickstarter at the time of writing, but should be by the time you read this.

▶ **rpimag.co/m5zero**

Lini



Another mini computer albeit this time one built on Raspberry Pi 5. As well as a decent-sized touchscreen, it has a little IBM-style mouse nubbin and a built-in keyboard that looks quite swish. You can sign up to be notified when it goes live on Kickstarter.

▶ **rpimag.co/liniks**

Pironman 5 Pro Max: Your All-in-One Raspberry Pi 5 Desktop for AI OpenClaw, Entertainment, Development & NAS

Dual NVMe PIP | 4.3" IPS Touchscreen | Microphone
5MP Camera | Speaker | PWM Tower Cooler





Exclusively for Raspberry Pi Official Magazine Readers:
Use code RPOM15 for 15% OFF.

Your Letters

Autofinance

I found your [issue 161] article very interesting on installing Ollama to run LLMs. I've accomplished this and am delighted by your easy-to-follow talk-through instructions which made the exercise an afternoon job.

Next up is the article in [issue 163] about running a Chatbot GPT.

It looks like I will need to buy an AI HAT, so I want to be sure about my expectations before the purchase. As I am interested in stocks and shares, Ollama3 suggested I use FinGPT; in this way, I would like to supersede the market analysis sites such as ADVFN and Stockopedia which I am currently subscribing to – but they provide live market data.

I understand that your setup is running locally, which is the reason I like it, but how may I access up-to-date or live data? Would regularly uploading a new llama3 dataset be a workaround? Obviously it's still not live.

Can I achieve this or are my expectations too high?

Second query: the board gets rather hot running this application (it's in one of those mini tower kits with a fan on a huge radiator). What would be the best way of cooling it if I did buy an AI HAT?

Steve, via email

That is very interesting. One of the advantages of using AI HAT+ 2 is that you can offload the GPT model to the HAT and leave your Raspberry Pi running other tasks. Which could be accessing live data on stocks.

However, with AI HAT+ 2 you are limited to the models supplied by Hailo and we don't believe FinGPT is one of them. One of the other models may be a viable option though. This is a bit beyond where we have explored so far, so I suggest you post a question on the Hailo Community forum (**community.hailo.ai**) and get answers from the source. We don't, personally, recommend leaving your finances to a homespun GPT. Do let us know if you get something up and running.



▲ AI is moving so quickly, it's worth having a separate computer – such as a Raspberry Pi 5 – as a sandbox so you can experiment without breaking anything important

April Fool

You made a right fool out of me last month. What was the thought process behind releasing the 3GB Raspberry Pi 4 on April Fool's Day? It doesn't make any sense – the product or the timing. Everyone knows RAM doesn't move in odd numbers – it goes 128, 256, 512, and then 1, 2, 4, 8 etc. By introducing a 3GB variant, Raspberry Pi has broken reality. It's really not on.

Mark, via email

We're not just used to RAM going up in binary increments; we're used to it going up, full stop. Every new laptop, desktop or other machine had to be faster and more powerful, mostly to cope with whatever technical demands the latest version of Windows placed on it. That's not exactly the case now; we've been fully functional with cloud-based apps for a while, and programming on a command line uses only the same amount of RAM as it ever did: minimal. As well as that, there's the sad fact that RAM has shot up in price over the last year or so, so I think it's likely that we'll see more odd numbers in products, simply because the usual incumbents are now so expensive. Think again: you'll find that for a lot of projects, 3GB (or 1GB, or 2GB) is just right.

▼ 3GB sounds daft, but for loads of projects, it's just right





▲ An heirloom computer that your grandkids will admire

Paper console

I love the paper console that you published in last month's magazine. Information overload is frying my tiny brain, but still I pick my phone up first thing and want the headlines, football results, weather etc.; and that leads on to all sorts of stuff that I don't want to know but can't look away from, like social media. Having a machine that only gets data from the important bits of the internet is a brilliant idea, and the fact that it looks so good is a bonus.

Emma, via email

There's something about ubiquitous digital technology that lends itself to cheap plastic and touchscreens. I guess it's the knowledge that there's always going to be something better or faster around the corner, so your computer is on a countdown to replacement as soon as you get it. But with computers taking a vital part in our homes, maybe it's time to start treating them like furniture: buy something well made and tasteful once, and keep it for the next 50 years? That's why we like the paper console. It makes us rethink the role of the computer in the house. Maybe we should pay more attention to our machines, but less often?

Contact us!

- @rpimagazine
- @rpimag
- rpimag.co/facebook
- magazine@raspberrypi.com
- forums.raspberrypi.com

USA SPECIAL!

6 ISSUES FOR $43



Subscribe online:

rpimag.co/subscribe

Continuous credit card orders will auto-renew at the same price unless cancelled.
A choice of free Pico 2 W or Pico 2 is included with all subscriptions. This is a limited offer.
Not included with renewals. Offer subject to change or withdrawal at any time.

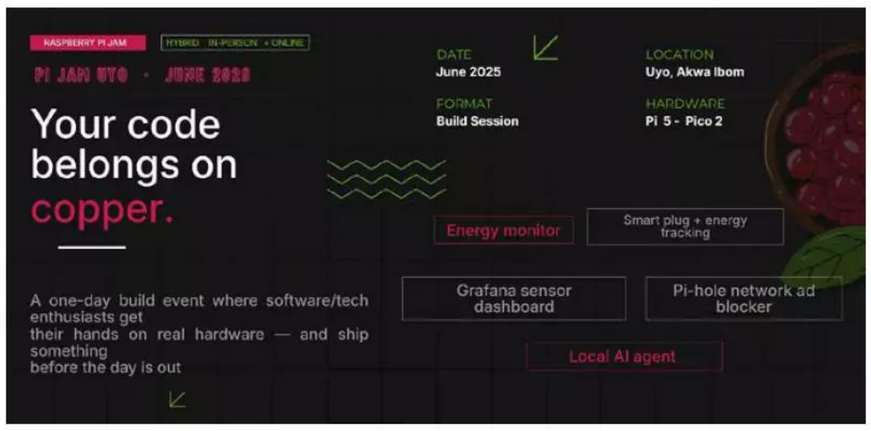
Community Events Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

01. Pi Jam Uyo

- **Saturday 6 June**
- **Online; Uyo, Nigeria**
- **rpimag.co/pijamuyo**

Most software engineers have never held a circuit board with their own code running on it. Pi Jam Uyo is built to change that — in a single afternoon. This is a hybrid hackathon jam held in Uyo, Akwa Ibom, open to developers of all levels who want to experience the physical world their software could be touching.



02. Riverside Raspberry Pi Meetup

- **Monday 8 June**
- **3600 Lime Street, Riverside, CA, USA**
- **rpimag.co/rrpm166**

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. As well as Raspberry Pi, the group also covers topics about all kinds of maker technology and electronics.

03. CoderDojo Kingston University

- **Saturday 13 June**
- **Kingston University, Kingston upon Thames, UK**
- **rpimag.co/cdku166**

A free computer coding and digital skill session for youngsters aged between 7 and 17. Activities can include programming with Scratch, Python, or Unity, Coolest Projects AI Toolkit, robotics with Raspberry Pi RP2040, along with web page creation (HTML/CSS/JS) and micro:bit digital electronics.



04. Leicester Hackspace Pi Jam

- **Saturday 20 June**
- **Leicester Hackspace, Leicester, UK**
- **rpimag.co/lhpj166**

Leicester Hackspace will have a Raspberry Pi 5, Raspberry Pi 500, AI HAT, and AI Camera Module, plus Raspberry Pi Picos for attendees to try. If you want advice for a problem, come along and their members will be happy to help. There will also be robot events such as line following, maze navigation, and straight-line speed. If you have one, bring it along.



FULL CALENDAR

Get a full list of upcoming community events here:

rpimag.co/events

05. Automate 2026

- **Monday 22 June to Thursday 25 June**
- **McCormick Place, Chicago, IL, USA**
- **rpimag.co/auto26**

Raspberry Pi will have a stand at Automate 2026, Chicago, from 22–25 June. There, you'll be able to meet the team and experience a wide array of Raspberry Pi products, including Raspberry Pi Pico 2, the AI product range, RP2350-based solutions, and the latest industrial device: Compute Module 5.

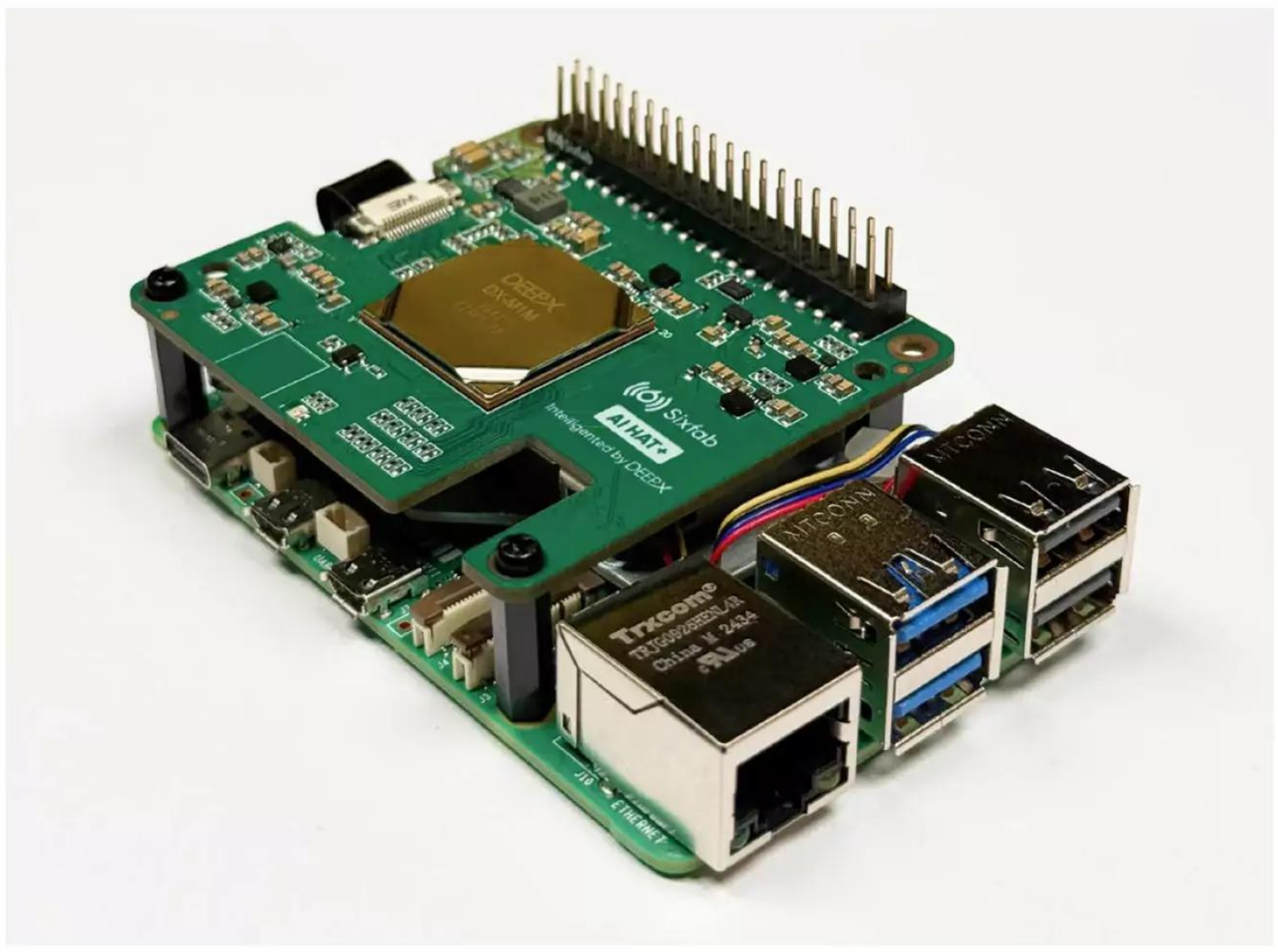


Official Raspberry Pi Event

Competition

Win 1 of 10 Sixfab AI HAT+

The new Sixfab AI HAT+ brings accessible on-device AI acceleration to Raspberry Pi, helping users to create faster, cooler AI projects. This new HAT was developed in conjunction with the Raspberry Pi engineering team.



Head here to enter:

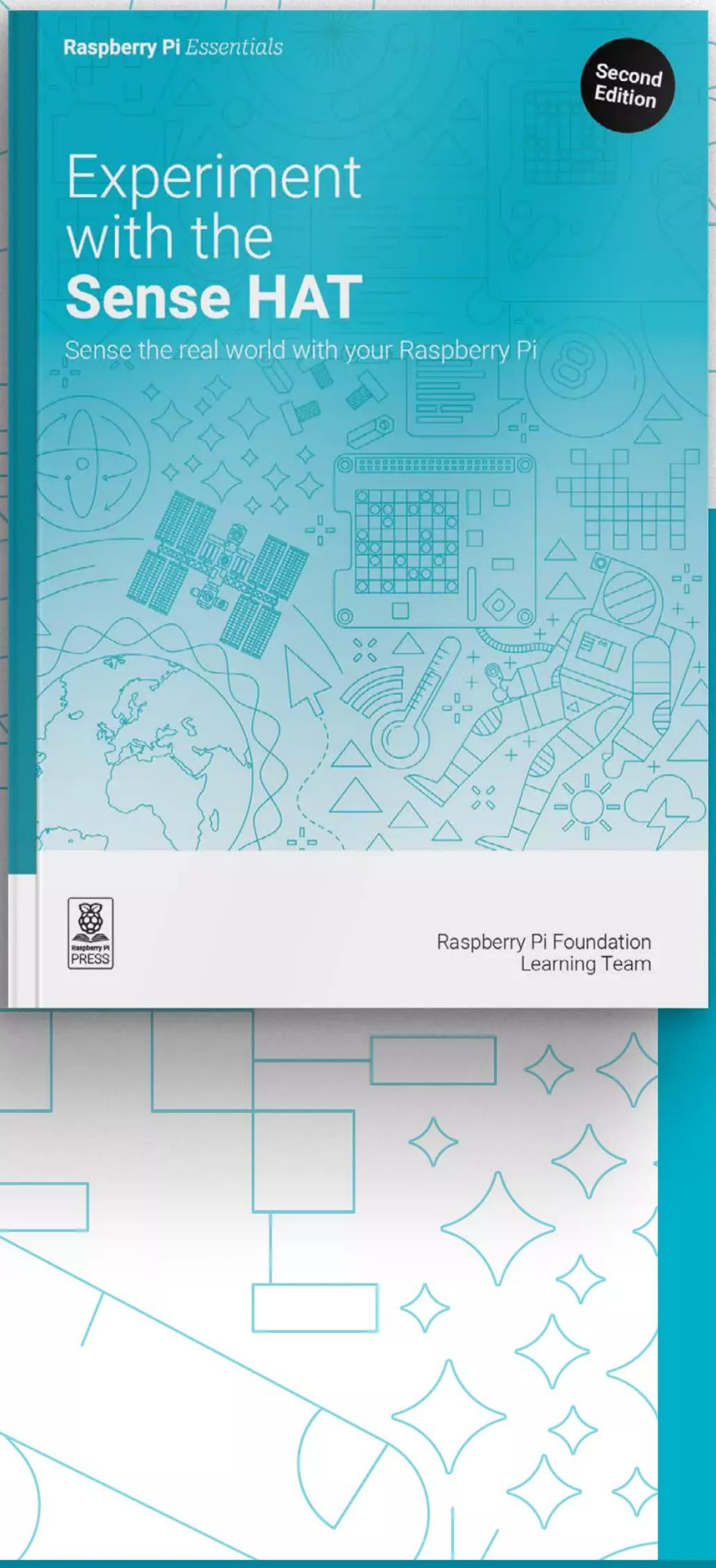
rpimag.co/win

Learn more:

rpimag.co/sixfabaihat

Terms & Conditions

Competition opens on **27 May 2026** and closes on **25 June 2026**. Prize is offered to participants worldwide aged 13 or over, except employees of Raspberry Pi Ltd, the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from Raspberry Pi Official magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram, Facebook, Twitter (X) or any other companies used to promote the service.



The Sense HAT is an incredibly versatile and flexible bit of kit with plenty of obvious uses, along with a huge number of less obvious ones, that you'll love to make and share. Updated for the latest Raspberry Pi devices and hardware, this book has everything you need to get started.

- ***Getting started with Sense HAT***
- ***Learn by building:***
  - *A digital twist on the Magic 8 Ball*
  - *Your own interactive pixel pet*
  - *A sparkly light show*
  - *An environmental data logger*
  - *Flappy Astronaut, a low-res, high-fun video game*

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

REACH A GLOBAL COMMUNITY

Advertise in **Raspberry Pi Official Magazine**

Our readers are passionate about technology and the Raspberry Pi ecosystem. From DIY enthusiasts to professional engineers.

Your advertisement can sit alongside our cutting-edge tutorials, features, and reviews. And we have flexible advertising options for a range of different businesses.

Take the next step –
advertise with us today!



Email Charlie Milligan at: **charlotte.milligan@raspberrypi.com**

Official Magazine #167

Next Month

Home Automation Special

Build a smarter home with Raspberry Pi



Editorial

Editor
Lucy Hattersley
lucy@raspberrypi.com

Features Editor
Andrew Gregory
andrew.gregory@raspberrypi.com

Features Editor
Rob Zwetsloot
rob@raspberrypi.com

Sub Editor
Phil King

Advertising
Charlotte Milligan
charlotte.milligan@raspberrypi.com
+44 (0)7725 368887

Design

Head of Design
Jack Willis

Designers
Sara Parodi, Natalie Turner

Illustrator
Sam Alder

Brand Manager
Brian O Halloran

Contributors

David Crookes, Tim Danton, Rosemary Hattersley, Jo Hinchliffe, Nicola King, Phil King, Simon Long, Rob Miles, KG Orphanides, Sean McManus, Paul Mucur, Richard Smedley

Publishing

Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com

Director of Communications
Helen Lynn

CEO
Eben Upton

Distribution

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

Subscriptions

Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
rpimag.co/subscribe
rpipress@subscriptionhelpline.co.uk

recycle

CC BY NC SA

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.
Raspberry Pi Official Magazine is published by Raspberry Pi Ltd, 194 Cambridge Science Park, Milton Road, Cambridge, England, CB4 0AB. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products, or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).

Raspberry Pi PRESS

ISSN: 2977-4403 (Print)
ISSN: 2977-4411 (Digital)



Versatility

Scaling your Raspberry Pi, and your expectations, when making a fun project with **Rob Zwetsloot**

When you do this kind of job, friends love to come to you with project ideas. "I was thinking of doing this thing and realised I knew someone who could help with it!" pops up in my inbox not too infrequently, and I never find it a bother. I love working with folks to make their fun creative dream a reality. The cosplay eyes from… ah, ten years ago, wow – well, that was a friend reaching out about their cosplay to me (**rpimag.co/cosplayeyes**). Raspberry Pi Zero was very new when we started working on it, and I immediately knew that it would be better than a full Raspberry Pi for the task. Not only is it easier to slip into a pocket, it would allow the NeoPixel lights to run longer on the same batteries.

From arcade machines to smart screens to, unfortunately, pie-in-the-sky concepts that weren't technically possible, I've been asked about many project ideas over the years and I love how my brain tries to work out the minimum viable solution to them. An internal flow chart where the requirements get processed until my brain spits out 'oh yeah, a Raspberry Pi Pico W is best for this one'. In fact, if Pico had existed back in 2016, I would have used it for the NeoPixel eyes too.

Current challenges

The current project I'm working on with a friend involves a ⅙ size doll (about 25cm tall) that has a head in the shape of a CRT TV. My friend wanted to replace the plastic screen with a display that could run custom animations she created, controlled over a remote interface. No alterations to the head (holes, 3D printed parts) if possible. And the front is attached by small magnets so comes right off.

Here's how my brain processed this. A full-size Raspberry Pi could handle this, obviously. It would be a bit overkill though, and the head is definitely too small for a Raspberry Pi anyway. Raspberry Pi Zero could do this easily, doesn't even need to be a Zero 2 W. It's the right size and can play videos – there's also a ton of displays available that it works on. However, that's the same with Pico, and you'd need to do a lot less work to get it playing the animations. No cronjobs, no custom Raspberry Pi OS install, just a script and some animations to play on the screen. We've seen folks with web-controlled Raspberry Pi Pico projects for years now, so that part is sorted.

As well as powering it with a small rechargeable battery (fairly simple), all it needs is the right size screen. That's the tricky part, especially with a little hook inside the TV head meaning I'm currently having to connect Pico to the display using jumper cables instead of attaching it directly to the purpose-built headers.

> *I love working with folks to make their fun creative dream a reality*

I'm getting there though! Making truly is about working around limitations, and thanks to all the various Raspberry Pi versions, it's never been easier for this stuff.

Rob Zwetsloot – Author

Rob is the features editor for *Raspberry Pi Official Magazine* and is regretting not taking a versatile Raspberry Pi 'smart TV' plug-in while in deepest darkest Cornwall recently.

rpimag.co

HiPi.io

HighPi 5S

The new case from the HiPi team



Reliable case for complex projects:

- Built for Raspberry Pi 5
- Rapid tool-free assembly and disassembly
- Large internal volume for HATs even with the active cooler installed
- Camera cable slot to connect to external cameras
- Power button and external status LED
- Passive & active cooling options
- Secure microSD cover
- VESA mount support

Customization options for volume orders:

- Output ports molded to your exact specs
- Logo printing for a branded finish

Available at these great Pi stores:

PiShop.us PiShop.ca ThePiHut Pi-Shop.ch Welectron.

Contact your favorite Pi store if it's not listed here

PiKVM

Remote control **redefined**

Manage your servers or PCs remotely!

Powered by Raspberry Pi

PiKVM V4 Mini

Small, cost-effective, and powerful!

- Power consumption in idle mode: just 2.67 Watts!
- Transfer your mouse and keyboard actions
- Access to all configuration settings like UEFI/BIOS
- Capture video signal up to 1920x1200@60 Hz
- Take full control of a remote PC's power

PiKVM V4 Plus

The most feature-rich edition

- More connectivity
- Extra storage via internal USB3.0
- Upgraded powering options
- More physical security features
- Extra HDMI output
- Advanced cooling solution



A cost-effective solution for data-centers, IT departments or remote machines!

Available at the main Raspberry Pi resellers

core ELECTRONICS · PiShop.ca · RPishop.cz · Welectron. · elektorstore · PIMORONI

Pi-Shop.ch · The PiHut · PiShop.us · CloudFree · KUBii · Cytron Marketplace

ROBU.IN Your ideas, Our Parts · NETTOP COMPUTERS & ELECTRONICS · netsket mall · DIREKTRONIK · RaspberryPi.dk

HiPi.io

No reseller in your country?
Check shop.hipi.io (import fees might apply).

List of official resellers by country: