The MagPi

Issue 127 | March 2023 | magpi.cc

The official Raspberry Pi magazine

50 HACKS & HINTS

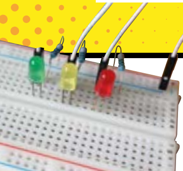
Know your Raspberry Pi better!

#MONTHOFMAKING

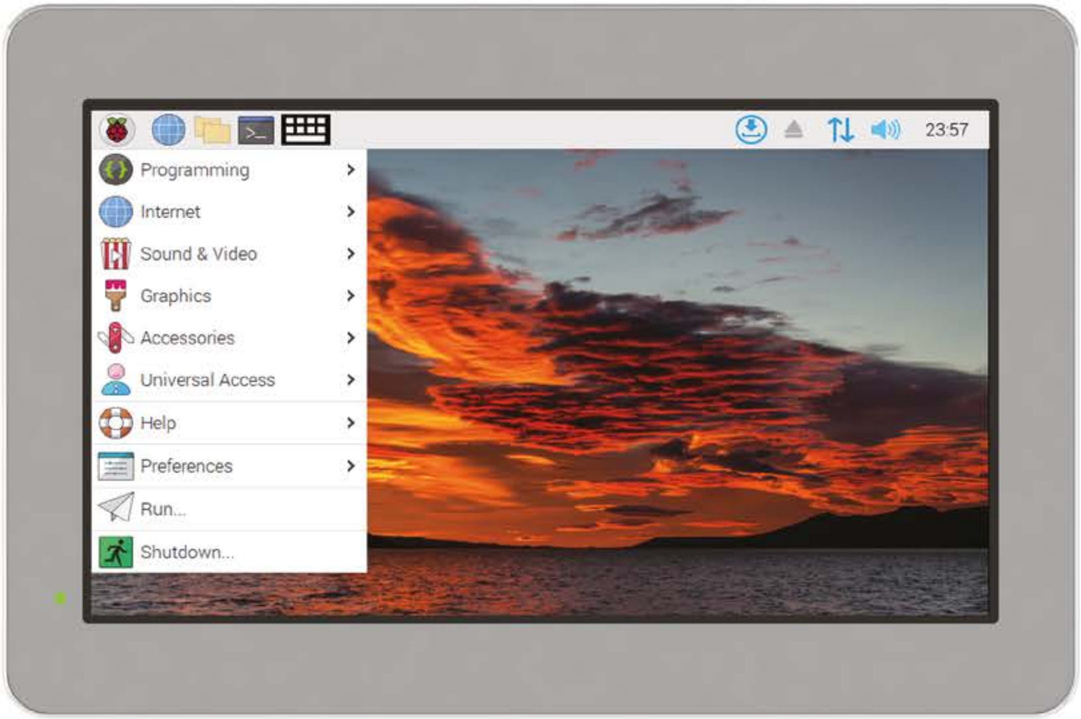
JOIN OUR COMMUNITY BUILD-A-LONG!

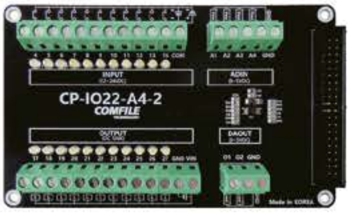
+ BUILD A **GEIGER COUNTER**

+ RECREATE A **BBC MICRO**



GET STARTED WITH **ELECTRONICS & CIRCUIT BUILDING**


23:57
Programming
Internet
Sound & Video
Graphics
Accessories
Universal Access
Help
Preferences
Run...
Shutdown...


INPUT
ADIN
CP-IO22-A4-2
COMFILE
OUTPUT
DAOUT

WELCOME
to The MagPi 127

Every March we ask the Raspberry Pi community to join us in our #MonthOfMaking (page 68). This year, Rob has asked some of the biggest names in the maker community for advice on building incredible things with computers. The result is a wonderful feature that is packed with advice.

Armed with a little knowledge, your Raspberry Pi can become lots of big things! This month, the team also shared 50 Hacks & Hints (**page 34**). These are our favourite tips for making with small computers, and they will make a huge difference to your builds.

This issue is packed with the best builds, expert advice, and enough inspiration to get you determined to make that project. Don't forget to show us what you make.

Lucy Hattersley Editor

EDITOR

Lucy Hattersley

Lucy is the editor of *The MagPi* and is making keto bread this evening. There must be some way to involve Raspberry Pi!

magpi.cc

WIZnet

TOE

TCP/IP Offload Ethernet

Design Contest

- Supporting up to 80 Mbps Speed
- Targeting High Performance Applications

USE W5300

- 8/16 bit Data Bus Interface (Direct & Indirect Address Mode)
- High Network Performance : Max 80Mbps (by DMA)
- 8 Independent hardware sockets

Buy

W5300($5) or W5300 TOE Shield($10)

at WIZnet e-shop

W5300

W5300 TOE Shield

WIN A CHANCE FOR $ 30,000 IN TOTAL

1st PRIZE	$ 5,000	x 2
2nd PRIZE	$ 2,500	x 4
3rd PRIZE	$ 1,000	x 10

Contest begins	Mar 16
Special price at WIZnet eshop	Mar 16 ~ April 30
Submissions close	Jun 30
Winners announced by	July 18

at maker.wiznet.io

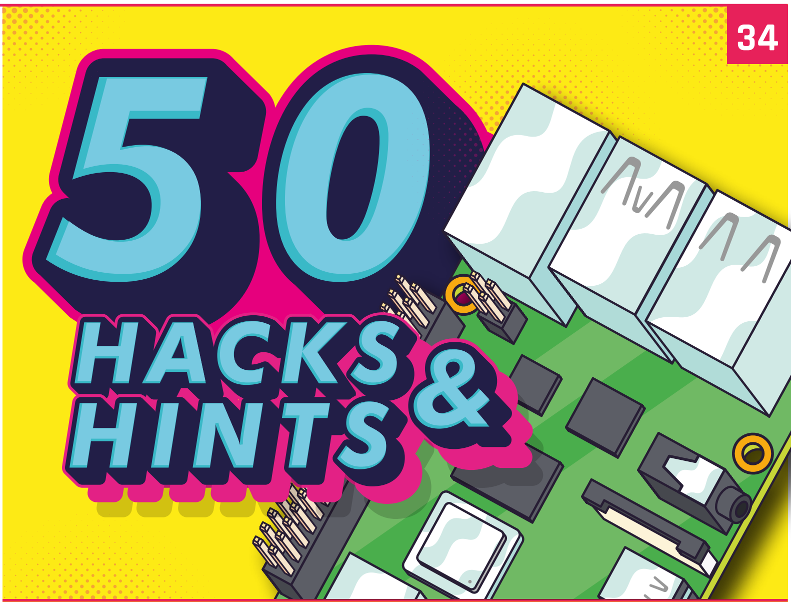
Contents

Issue 127 March 2023

Cover Feature

Regulars

Project Showcases



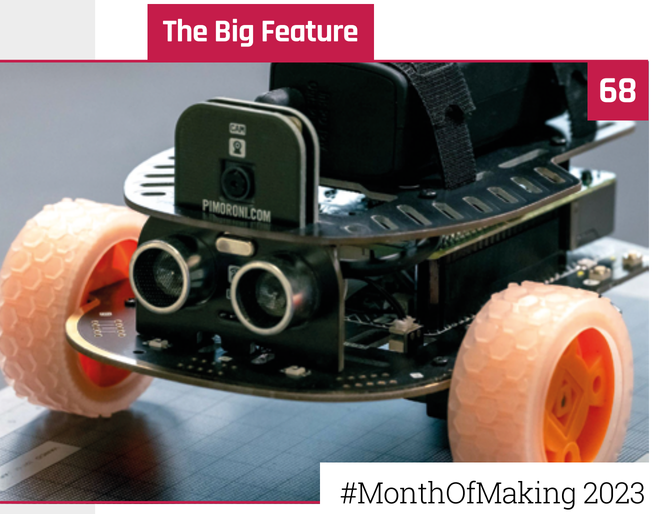
Desert Eye 2.0

Mini PC

The MagPi magazine is published monthly by Raspberry Pi Ltd. Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701, is the mailing agent for copies distributed in the US. Application to mail at Periodicals prices is pending at Williamsport, PA. POSTMASTER: Send address changes to The MagPi magazine c/o Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701.

Tutorials

The Big Feature



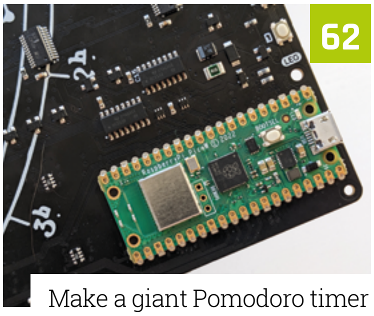
#MonthOfMaking 2023

Reviews

Community

48

Emulate a BBC Micro on Raspberry Pi 400



62

Make a giant Pomodoro timer

76

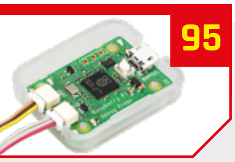
PiBoy XRS

86

VEEB interview

WIN 1 OF 10 RASPBERRY PI DEBUG PROBE

95



DISCLAIMER: Some of the tools and techniques shown in The MagPi 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 The MagPi magazine. Laws and regulations covering many of the topics in The MagPi 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 The MagPi magazine may go beyond. It is your responsibility to understand the manufacturer’s limits.

Raspberry Pi Debug Probe

Bridging the gap between non-Raspberry Pi computers and Raspberry Pi Pico

Raspberry Pi Debug Probe is a new USB Device designed to enable debugging of Raspberry Pi Pico, and other Arm-based microcontrollers, with a non-Raspberry Pi computer (that lacks GPIO pins).

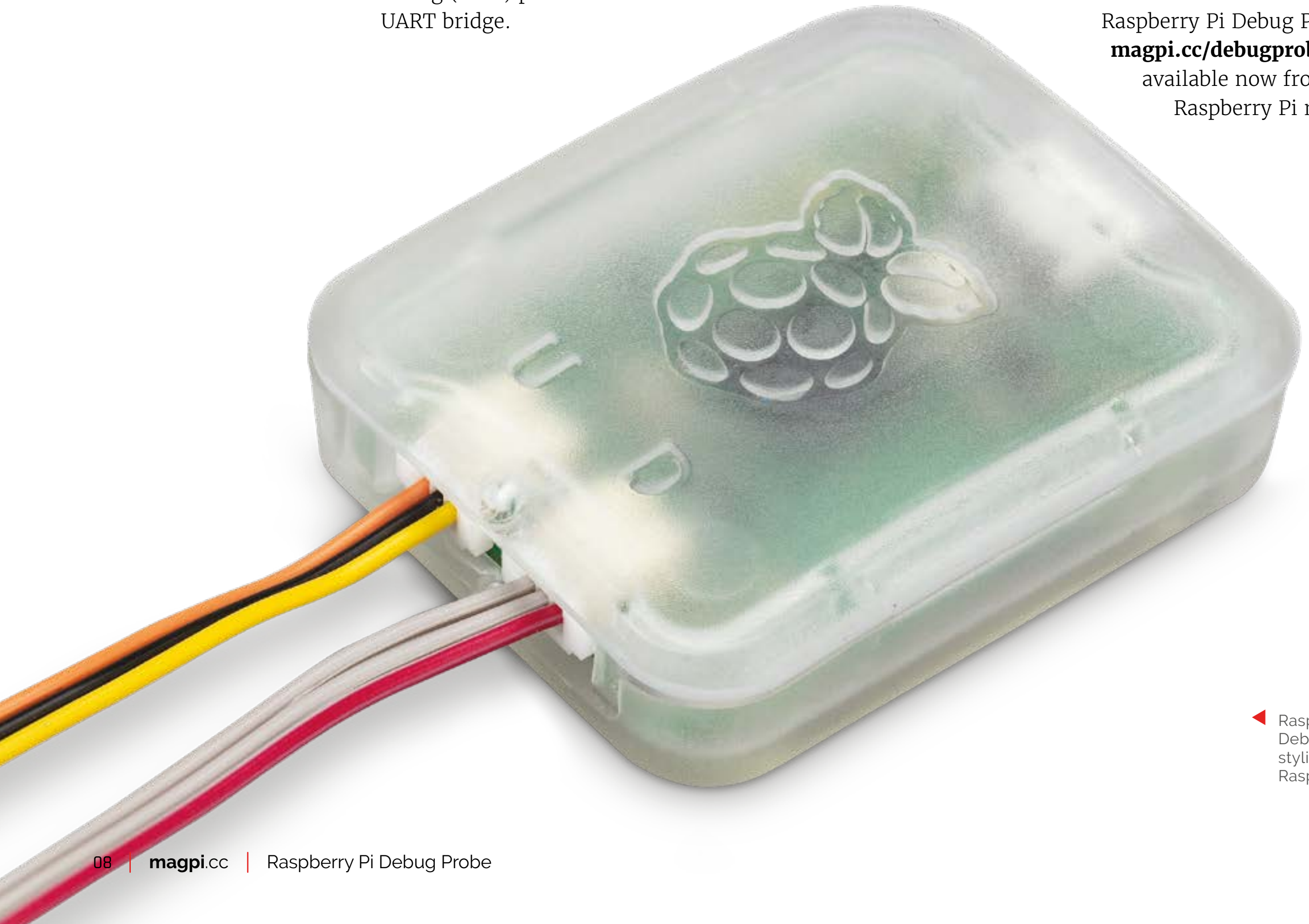
Resembling a tiny Raspberry Pi inside a clear plastic case, the probe has a USB connection to your computer on one side and two JST connections on the other. The probe is designed for easy, solderless, plug-and-play debugging. It features a USB to Arm Serial Wire Debug (SWD) port and USB to UART bridge.

Increasing compatibility

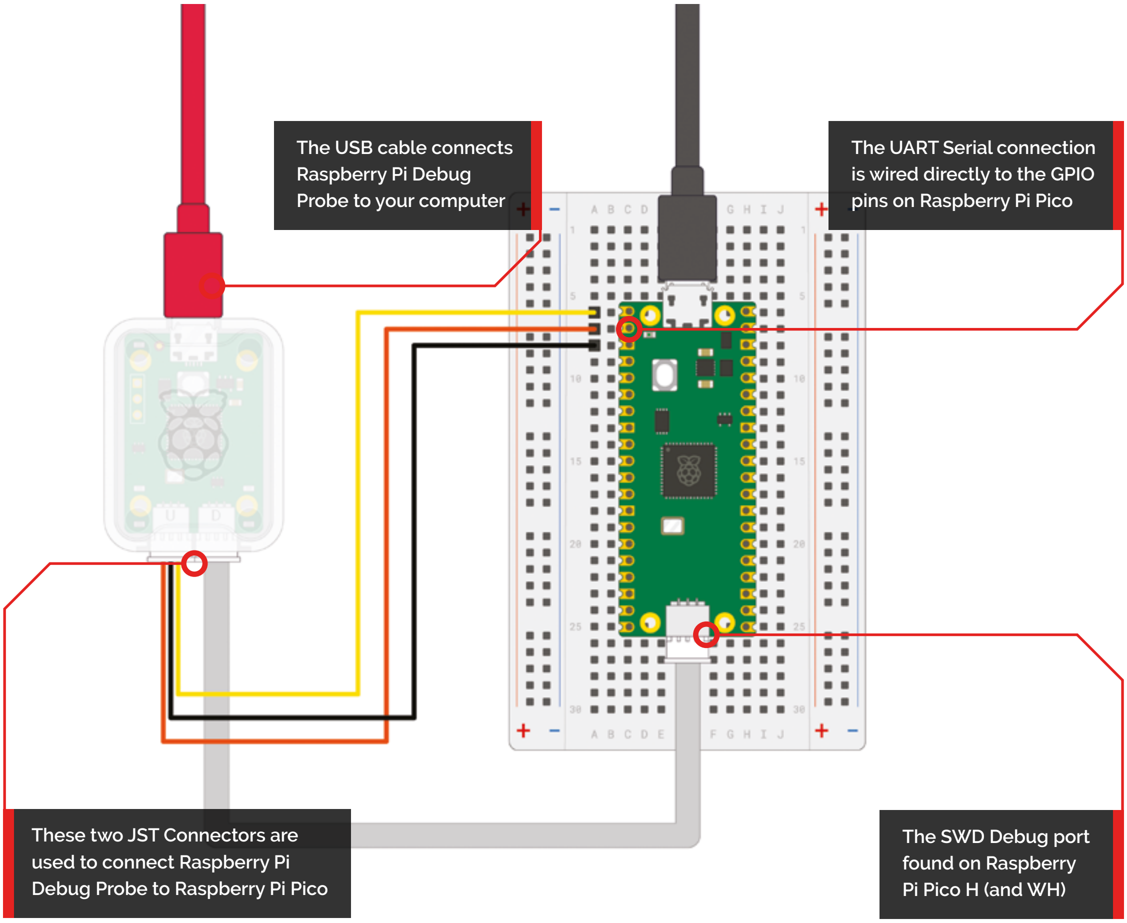
Debug Probe is compatible with the CMSIS-DAP standard (**magpi.cc/cmsisdap**) and works with OpenOCD (**openocd.org**) and other tools that support CMSIS-DAP.

Raspberry Pi Debug Probe is designed to make it easy to use a Raspberry Pi Pico on platforms such as Windows, Mac, and Linux-based computers. On these machines, you are unlikely to have a GPIO header to connect to Pico's serial UART or SWD port. Debug Probe acts as a bridge between the two devices.

Raspberry Pi Debug Probe ($12, **magpi.cc/debugprobe**) is available now from official Raspberry Pi resellers.



Raspberry Pi Debug Probe looks stylish in its small Raspberry Pi case



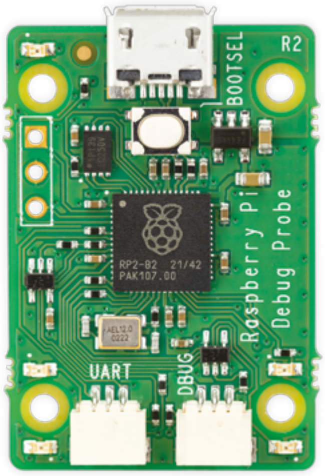
The USB cable connects Raspberry Pi Debug Probe to your computer

The UART Serial connection is wired directly to the GPIO pins on Raspberry Pi Pico

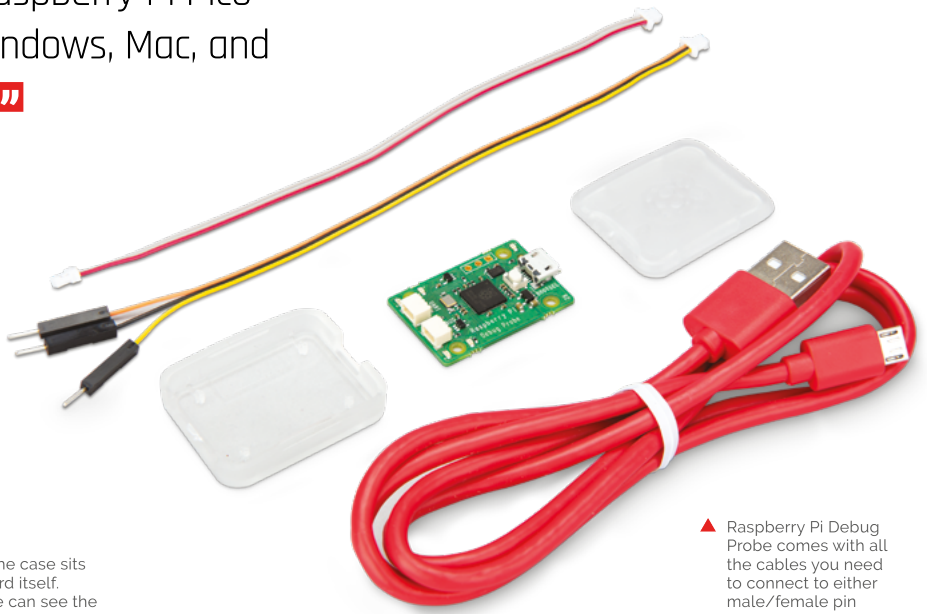
These two JST Connectors are used to connect Raspberry Pi Debug Probe to Raspberry Pi Pico

The SWD Debug port found on Raspberry Pi Pico H (and WH)

> “Raspberry Pi Debug Probe is designed to make it easy to use a Raspberry Pi Pico on platforms such as Windows, Mac, and Linux-based computers”



Inside the case sits the board itself. Here we can see the UART and Debug JST connections



Raspberry Pi Debug Probe comes with all the cables you need to connect to either male/female pin headers or the SWD Debug port

Desert Eye 2.0

A sophisticated Raspberry Pi 4 robot that observes terrain for danger could save human lives, maker Aviv tells **Rosie Hattersley**

MAKER

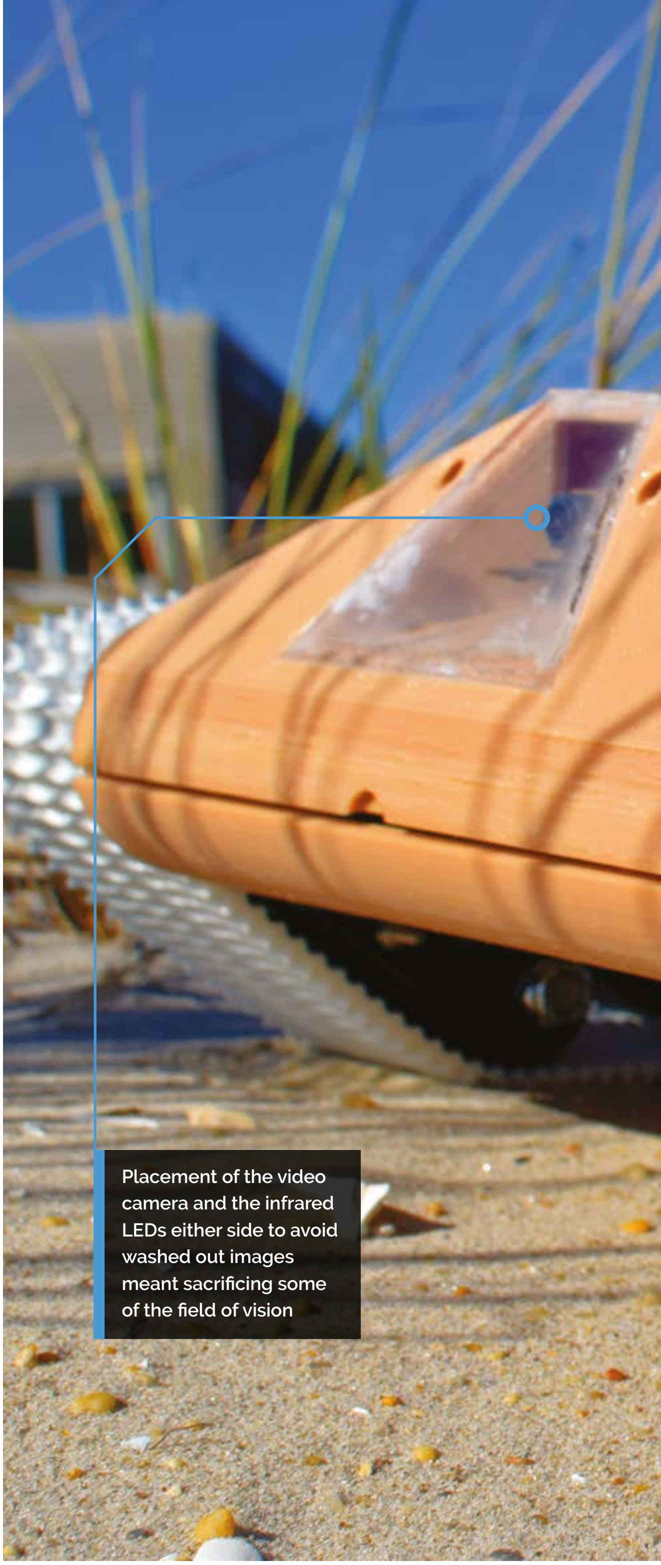
Aviv Butvinik

New York-based Aviv loves designing robots and has turned this passion into a career as an industrial designer and engineer.

magpi.cc/deserteyerobotics

Imagine a robot that could traverse the unforgiving sands of the desert in the darkness of night, covering many miles at a time and stopping at given waypoints to surveil. It could collect video at these stop-points from a front-mounted controllable zoom camera equipped with infrared LEDs invisible to the naked eye, but illuminating for camera visibility. Then, it could beam this video data over an encrypted wirelessly connection to a distant location. Possible dangers surveilled, soldiers or explorers would be briefed on whether it was safe to proceed, potentially saving lives as well as time and money. This is the concept robot enthusiast and engineer Aviv Butvinik has been working on over the past two years. In December, he unveiled the second iteration of his Desert Eye surveillance robot with a YouTube video that shows off its ability to cope with harsh environments replete with sandstorms: **magpi.cc/deserteyeyt**.

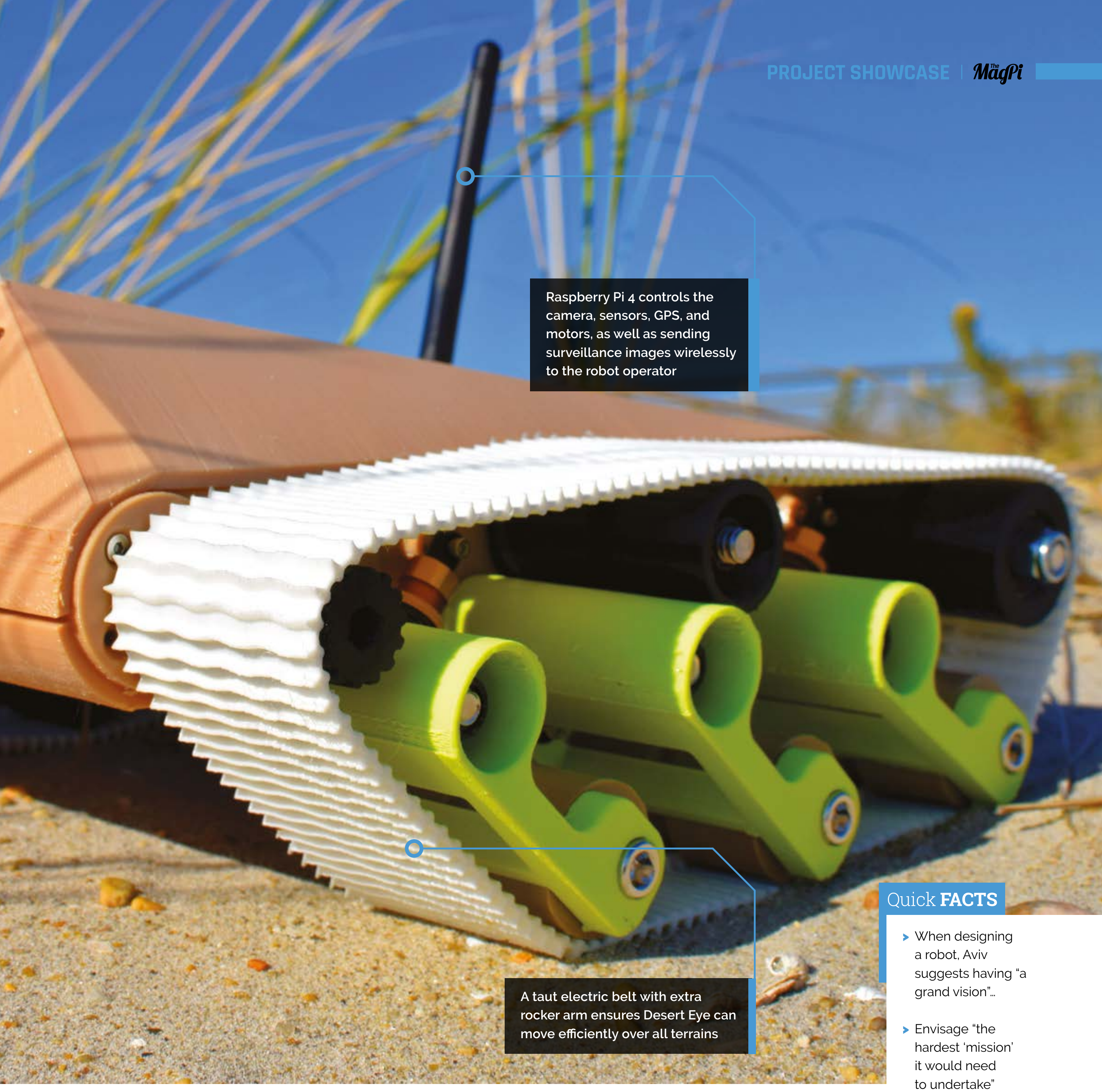
Desert Eye is also a great showcase for Raspberry Pi 4, which Aviv chose for its video processing capabilities – "a large consideration in this robot" – as well as controlling its motion and communications. Aside from the cameras, Raspberry Pi 4 controls the two main DC motors, a stepper motor so the camera can rotate while being stabilised, the GPS and wireless connectivity, and a three-axis sensor and fan.



Placement of the video camera and the infrared LEDs either side to avoid washed out images meant sacrificing some of the field of vision

Grand designs

Aviv had the sort of early school encouragement most of us can only dream of, getting to build model rockets, foam boats, and motorised balsa wood cars with his elementary school science teacher. "Those years instilled in me a love for science," he says. Aviv's father suggested he apply for the industrial design degree at the local university, where he eventually found his niche with a proposal for an engineering project that bagged him first place at the Design for Safety Symposium. This success led to Aviv taking up



Raspberry Pi 4 controls the camera, sensors, GPS, and motors, as well as sending surveillance images wirelessly to the robot operator

A taut electric belt with extra rocker arm ensures Desert Eye can move efficiently over all terrains

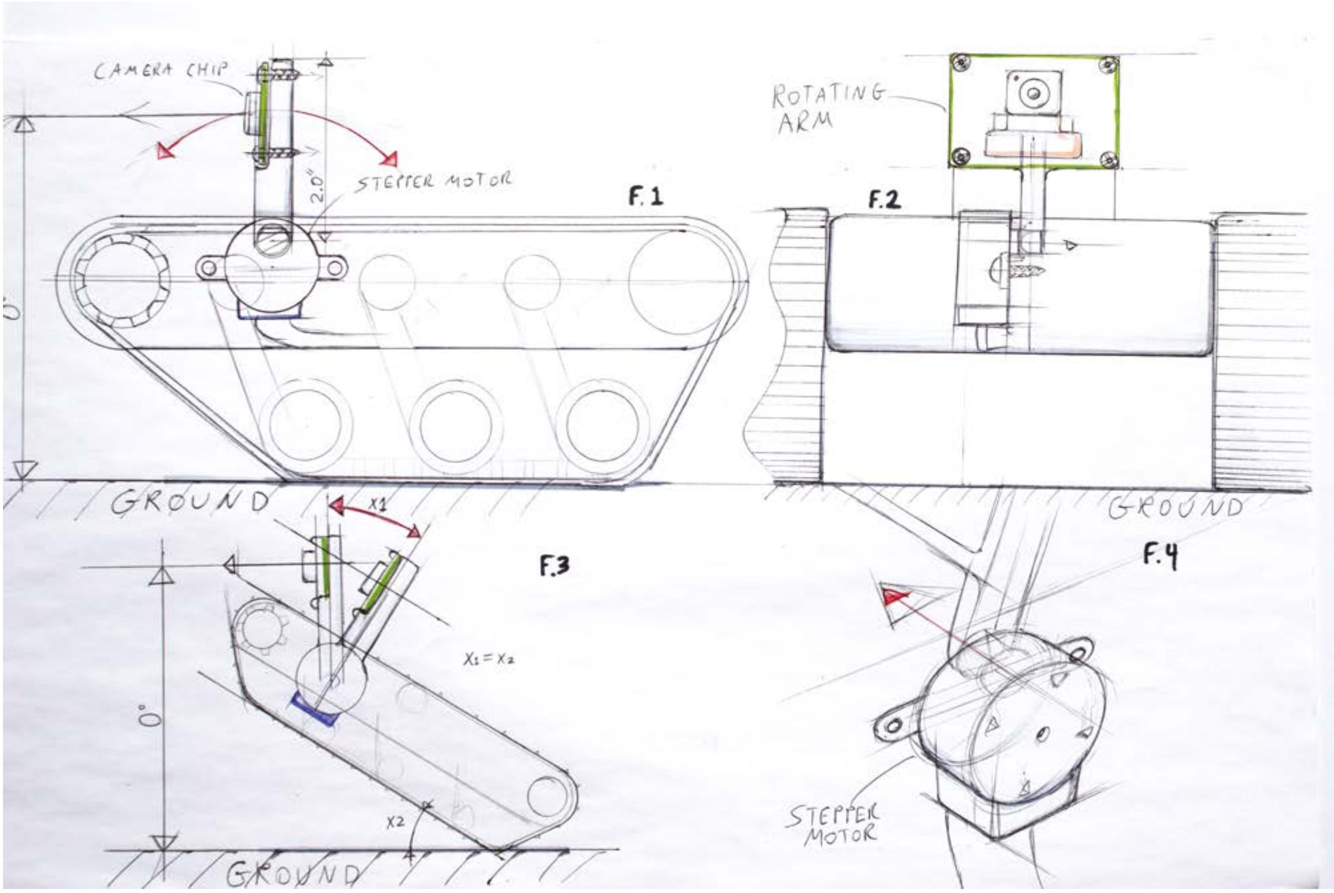
> “The VIAM team spent many hours helping Aviv get Desert Eye's programming just right”

a post as a mechanical engineer and industrial designer creating automotive safety products. He says the project challenges, and solutions he’s exposed to make him think about “the physical forces, electrical engineering, and kinematics involved in designs.”

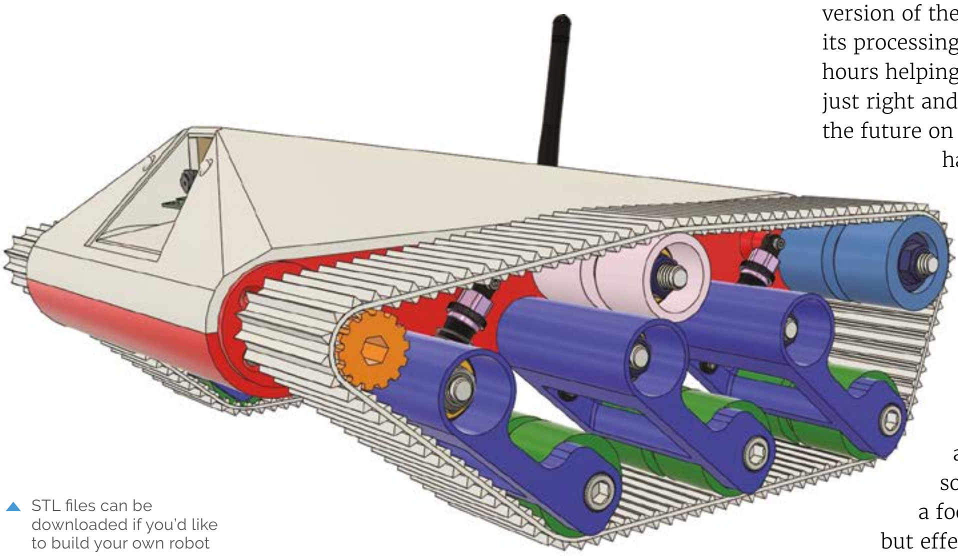
Aviv’s passion for robotics saw him design almost every aspect of Desert Eye himself, including the suspension dynamics, which he carried over from the first design to the current version. In an ideal world, he would have designed his own fasteners, motors, and even his own take on Raspberry Pi 4. Aviv sourced components from a range of online sources and used a 3D printing shop, but his best tip is to use VIAM open-source robot control software (**viam.com**), which he

Quick FACTS

- When designing a robot, Aviv suggests having “a grand vision”…
- Envisage “the hardest ‘mission’ it would need to undertake”
- Desert Eye originally had two bike chain-like loops
- These were connected with 100 3D-printed links to create a belt…
- But were incredibly time-consuming to assemble



Aviv starts with a "grand vision" of what he wants his robots to be able to do – and sketches out how it will work

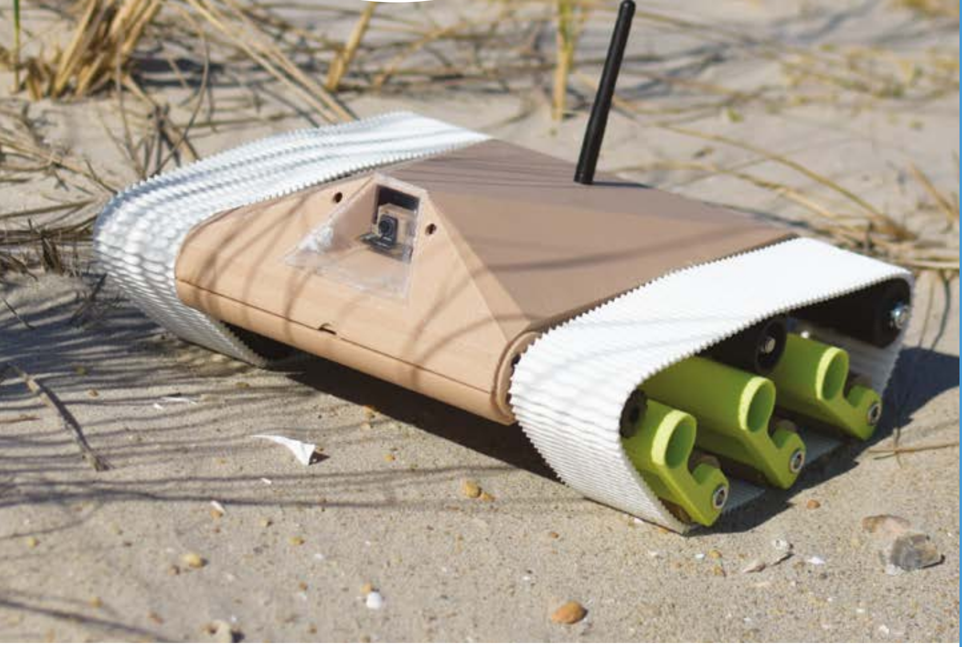


STL files can be downloaded if you'd like to build your own robot

favours for the online platform's ease of use, and because Raspberry Pi can be flashed with a Lite version of the OS for the VIAM server, increasing its processing speed. "The VIAM team spent many hours helping me get Desert Eye's programming just right and I hope to work with them again in the future on some of the more advanced visions I have for the robot."

While Aviv was confident about Raspberry Pi 4 being powerful enough to process the video camera footage his surveillance robot captured, the infrared LEDs either side of the camera module had a tendency to wash out the footage. Moving them further away from the camera and angling them away from the ground so the light beams were narrower and a foot in front of the robot was a simple but effective fix.

Aviv shows off his completed Desert Eye 2.0 robot, as featured on the cover of Servo magazine



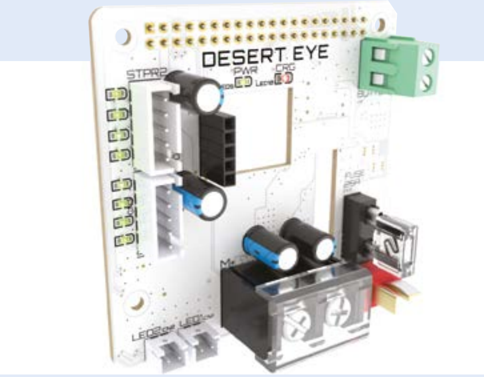
Desert Eye 2.0 spotted on manoeuvres in the wild

Iterate and improve

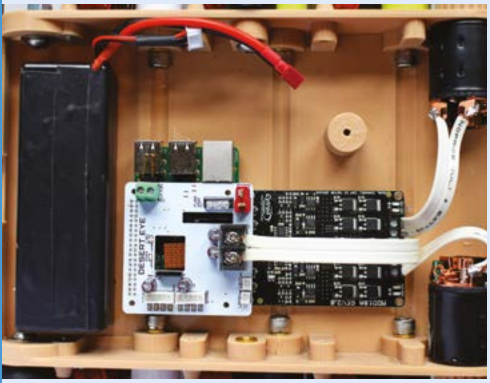
Many of the challenges for Desert Eye's design relate to its locomotion. The first version of the robot had an overly complex belt loop and dual sprocket system that drove the tank forward. "It looked and worked a lot like the suspension on a standard tank," says Aviv, but the belts would sometimes fall off their sprockets due to road vibration. Aviv has replaced these with a lighter, 3D-printed flexible belt loop made from elastic that can tolerate much higher degrees of vibration and misalignment than chains.

Aviv also found that when the robot started up, there was "a huge drop in belt tension (extra slack), as the rockers all moved inwards." He fixed this by adding a second rocker arm to the back rocker to compensate for the loss of tension.

Go go robot!



01 **DesertEyeRobotics.com** details the hardware needed to create your own surveillance robot, along with CAD STL files. You need a Raspberry Pi 4, two DC motors, stepper motor, jumper wires, and a motor controller – and, optionally, a Desert Eye PCB.



02 Complete construction and house it in a suitable chassis, then download and install the VIAM control software: **magpi.cc/viamsetup**. The VIAM app (**app.viam.com**) is used to connect and control the robot. It also has links to the documentation for more detailed instructions.

03 Once you've connected everything and installed VIAM, press the button on the PCB to turn the robot on. It will connect automatically to the VIAM server, ready for you to control your robot.

Mini PC

It's time to make some noise as Carter Hurd strips a Bluetooth speaker and turns it into a miniature computer, as **David Crookes** explains

MAKER

Carter Hurd

Carter is an artist, inventor, and automotive sensing engineer who runs a small YouTube channel documenting PC and robotics projects.

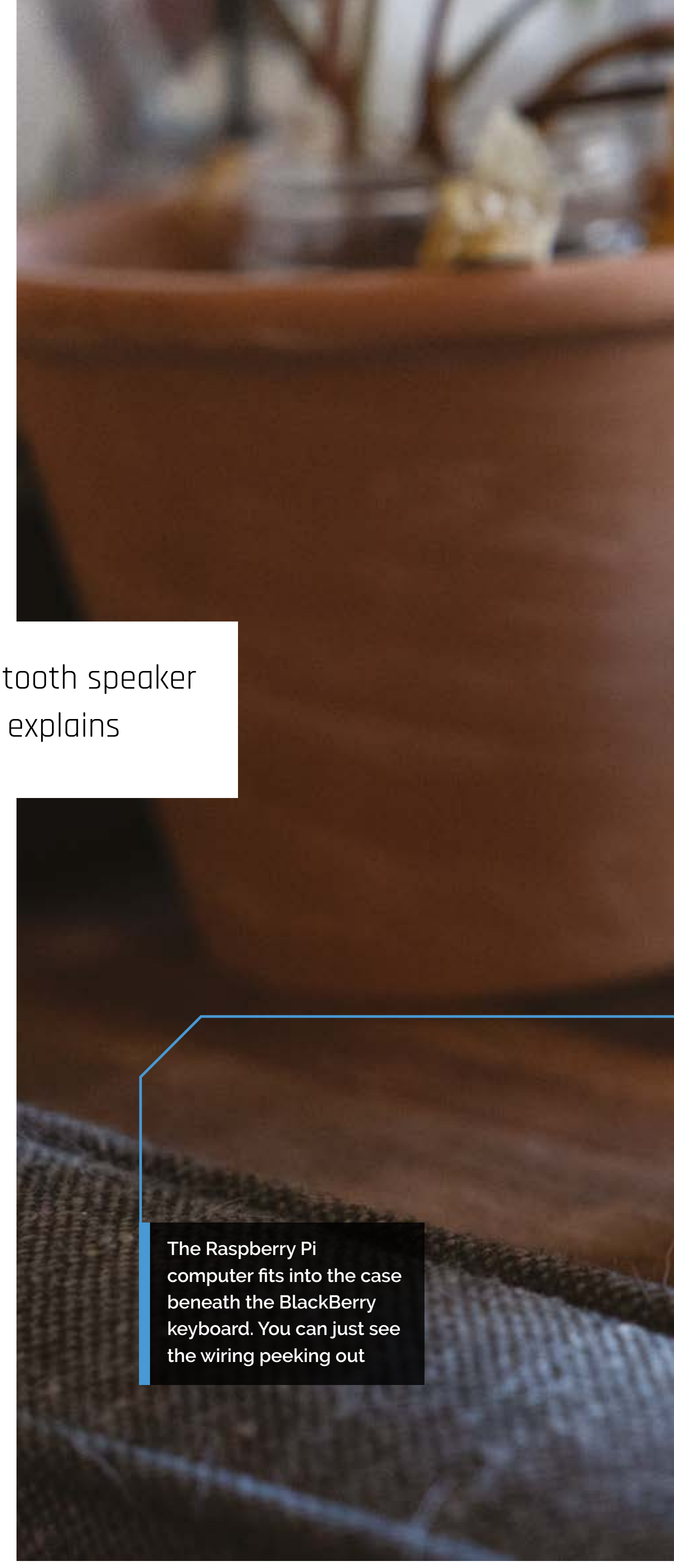
carterhurd.com

When Carter Hurd was approached by a company and asked to review one of its products, he accepted on one condition: that he wouldn't actually review it, but instead pull it apart. Given Carter is an engineer and inventor, this was perhaps to be expected.

The result was a Bluetooth speaker that ended up being turned into an attractive miniature PC, powered by a Raspberry Pi computer (**magpi.cc/minipc**).

"I have a YouTube channel (**magpi.cc/carteryt**), and a company called Divoom sent me a Bluetooth speaker to review," Carter explains. "Unfortunately, I'm not a product reviewer, so I decided to make a project with it. Initially, I was hoping to make a standalone music and media player for my kitchen but, as I got further along, I realised that I'd need to remove the speaker if I wanted to fit in a display."

The item Carter received was the Divoom Ditoo Plus Retro Pixel Art Game Bluetooth Speaker. It's a curious little thing that's styled to look like a desktop PC and, as well as six buttons and a tiny stick to navigate its many features, it has a screen which allows users to display pixel art, play games, or view information.



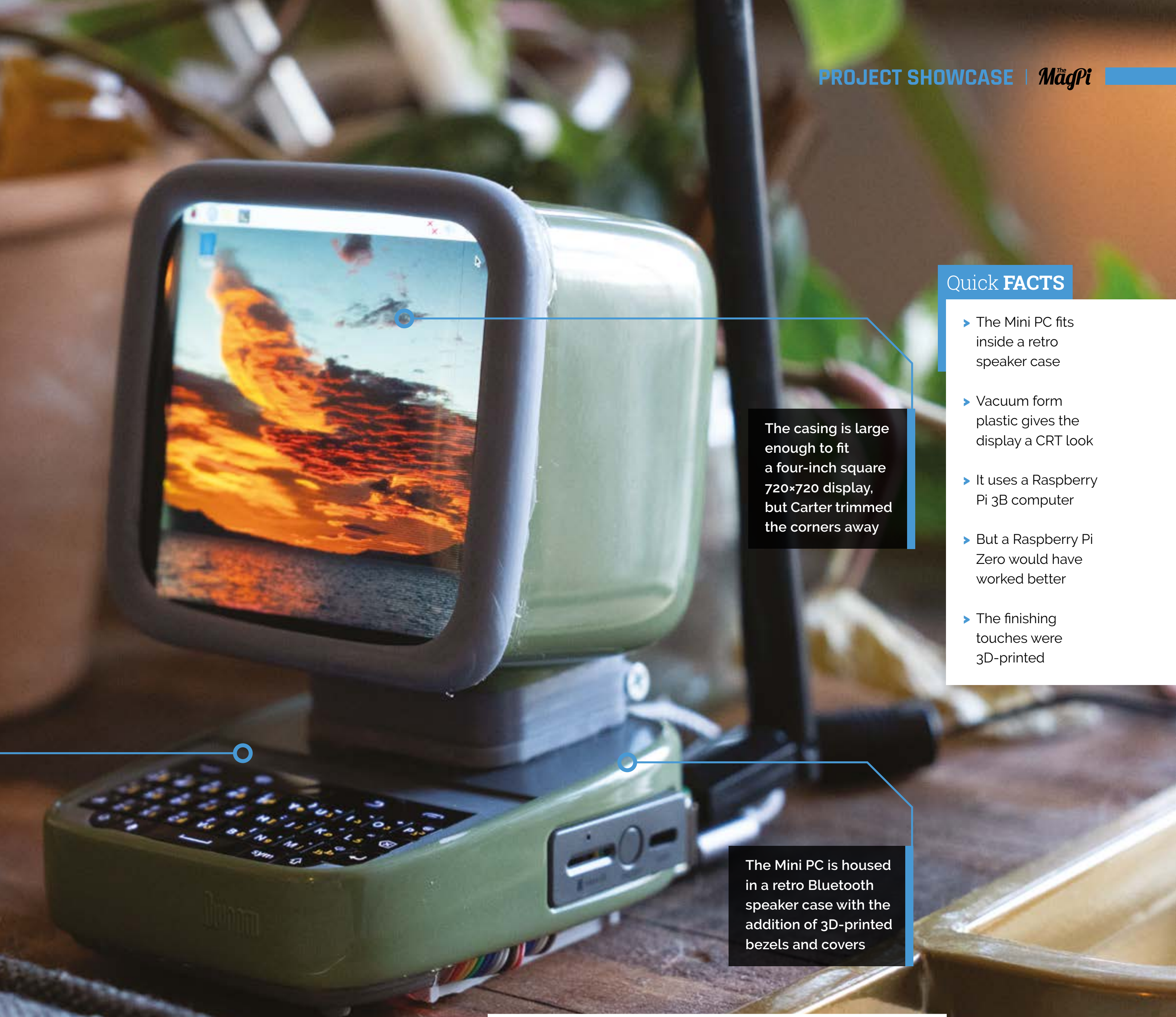
The Raspberry Pi computer fits into the case beneath the BlackBerry keyboard. You can just see the wiring peeking out

Key to success

To turn this device into a PC, however, the buttons, the screen, and the speaker needed to be stripped away. Carter grabbed cutting tools and a screwdriver and began disassembling, eventually leaving just the case. He then began inserting new hardware, starting with a replacement keyboard. He happened to have a USB BlackBerry Q20 keyboard with trackpad to hand, and this fitted neatly once he trimmed the corners.

"It was simply luck that the keyboard I had on hand was such a good fit," he says, explaining that he hid the rough edges and the surrounding board by 3D-printing a bezel. Unfortunately, the display proved trickier. "The screen I had was not a good fit," he laments. "Although being square was a good start, the glass was too large to fit within the speaker housing.

"I cut the glass with shears and covered the edge shards with a 3D-printed bezel. But cutting electronics with shears is a bit of an inside joke on my YouTube channel. It's dumb and it's not a method I recommend. To make the display look more like a CRT, I also shaped a piece of thermal



The casing is large enough to fit a four-inch square 720×720 display, but Carter trimmed the corners away

The Mini PC is housed in a retro Bluetooth speaker case with the addition of 3D-printed bezels and covers

Quick FACTS

- The Mini PC fits inside a retro speaker case
- Vacuum form plastic gives the display a CRT look
- It uses a Raspberry Pi 3B computer
- But a Raspberry Pi Zero would have worked better
- The finishing touches were 3D-printed

> “It was simply luck that the keyboard I had on hand was such a good fit”

plastic into a dome with a hair-dryer and placed it over the LCD.”

Making sacrifices

The final step was to fit a Raspberry Pi 3B computer. Again, it needed to squeeze into the

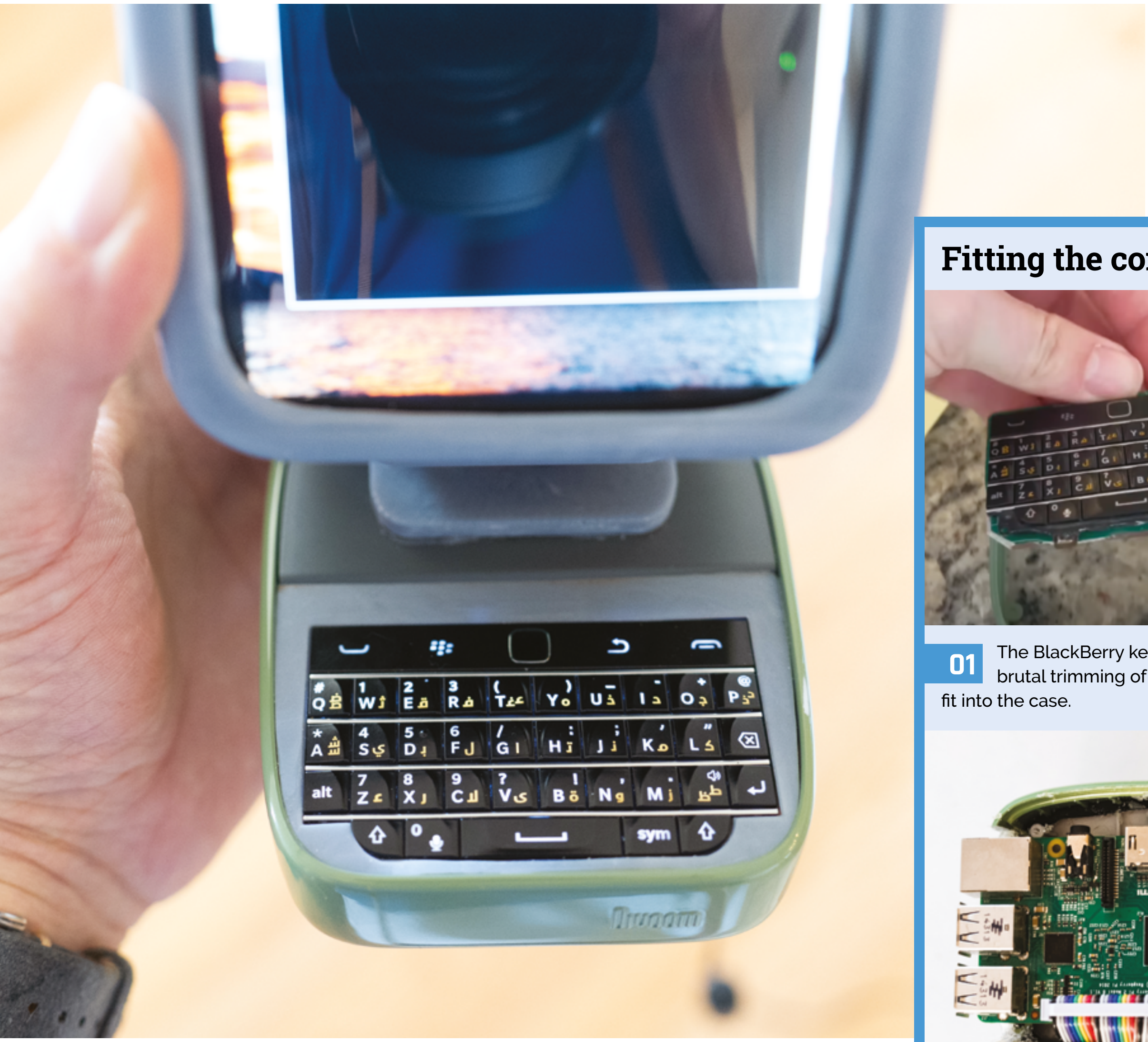
The Mini PC is so small, it can sit on the palm of your hand, but it meant the internal space was tight

Raspberry Pi OS is capable of being displayed on the square screen and it can be controlled by the keyboard and trackpad

The display got in the way of the newly fitted keyboard, so Carter created a new hinge to allow the screen to sit higher and tilt back more

Carter says he could have connected Raspberry Pi to the USB-C port and on/off button on the right-hand side of the speaker housing

Here is a rear view of the device. Carter says he would like to make the project battery-powered



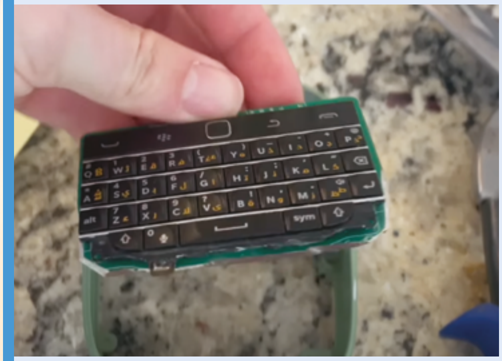
case and it required removing the GPIO pins to fix on a right-angle header. This proved tricky and Carter still couldn't create a sufficiently low profile, so he ended up cutting the bottom of the housing instead. "If I was to buy the parts, I would have used a Raspberry Pi Zero computer which would have fit better," he says.

Even so, once assembled, the Mini PC has worked a treat. It boots up, displays the familiar Raspberry Pi OS screen, and it's controllable via the keyboard. But if Carter worked on it again, he says he might do things differently.

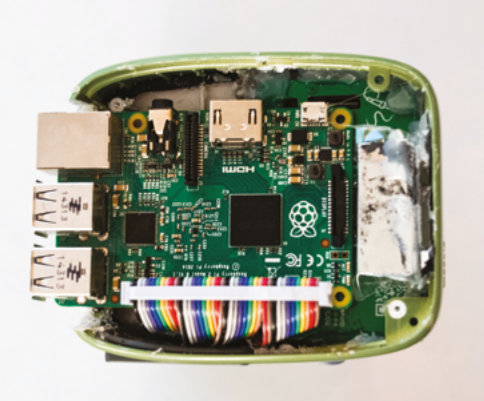
"The bottom cover could be designed better to fit over the Raspberry Pi computer, and the upper housing could be redesigned to include a speaker," he says. "I could find some way of preserving the LCD's touch sensitivity while still achieving the CRT domed look. To be honest, though, I haven't really used it much. I kinda just built it to see if I could."

▲ The BlackBerry keyboard fits smartly into the bottom of the casing, surrounded by a snug bezel

Fitting the components



01 The BlackBerry keyboard needed some brutal trimming of the PCB to ensure it could fit into the case.



02 The Raspberry Pi computer fits well inside the bottom casing, but it still sits rather too high.

03 The display's corners could be seen beneath the bezel, so they were trimmed, but this did crack the screen.

Proto Shield with Robot Arm and Snake Game

Proto Shields usually work with Arduinos, but one maker created a Raspberry Pi version for some sneaky gameplay, he tells **Rosie Hattersley**

MAKER

Ronin

Ronin is an embedded systems engineer who enjoys making robots and microprocessor-based projects.

github.com/byronin

The first indication that Ronin (aka byronin) would grow up to be an enthusiastic maker and technical engineer came at the age of 10 when he bought his first soldering iron; at 15 he built his first mini sumo robot, winning first prize in the very first competition he entered. An embedded systems engineer in his native Turkey, Ronin loves designing his own projects. Although his Raspberry Pi Proto Shield and Snake Game is his first appearance in *The MagPi* (and his first Pico project), his prodigious output means it's unlikely to be his last.

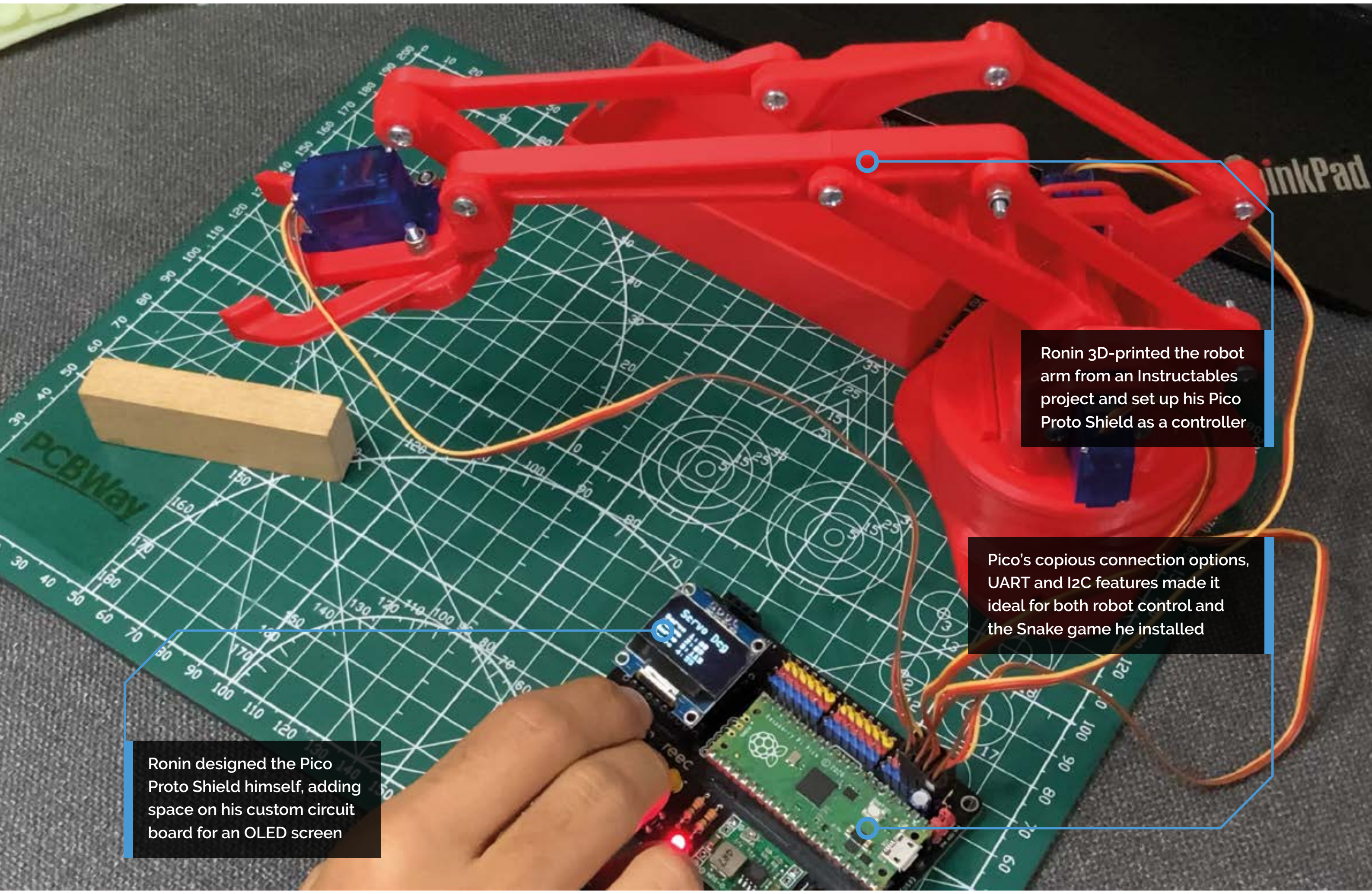
Ronin enjoys designing and manufacturing his own boards

Ronin built his first prototype circuit board six years ago and discovered Raspberry Pi a year later. He usually programs microcontrollers using embedded C, but learning MicroPython led him to Raspberry Pi and, more specifically, the Pico. After realising the scope of what Pico can do, Ronin says he was "mad at myself" for not buying a Raspberry Pi Pico for his workshop before, given that every project he works on has buttons, an OLED screen, and motors.

> Ronin built his first prototype circuit board six years ago and discovered Raspberry Pi a year later

Protection measures

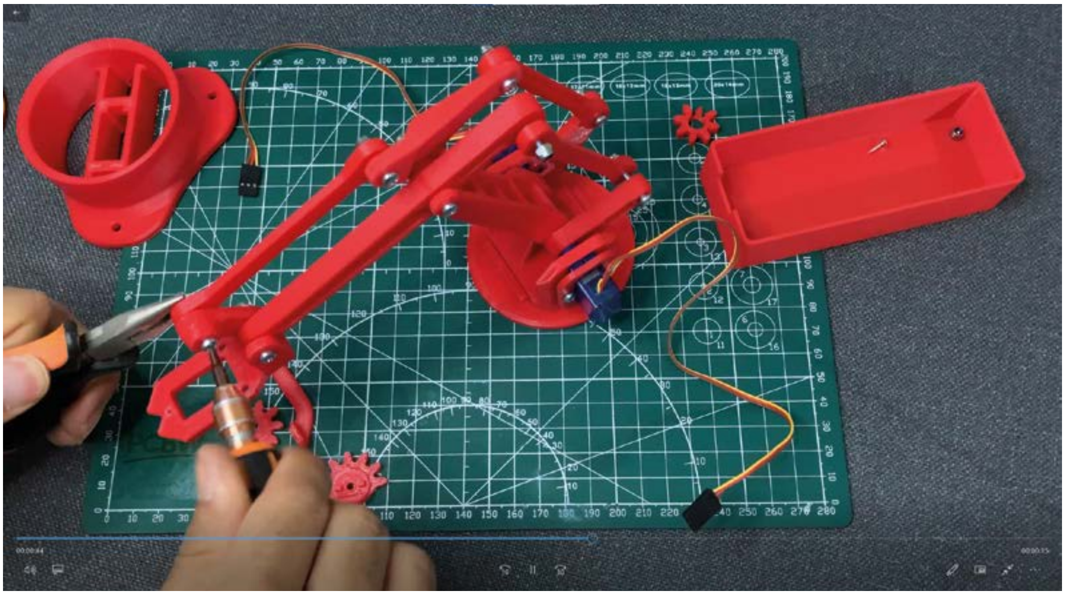
Ronin likes to design and manufacture a bespoke shield whenever he buys a new development board or a microcontroller, and is now a regular customer of custom circuit board printing company PCBWay. Most of his project builds are based on off-the-shelf or easy-to-acquire parts, but PCB production is "a difficult process", so he prefers to rely on a paid-for service. Although Raspberry Pi has its own system of HATs as



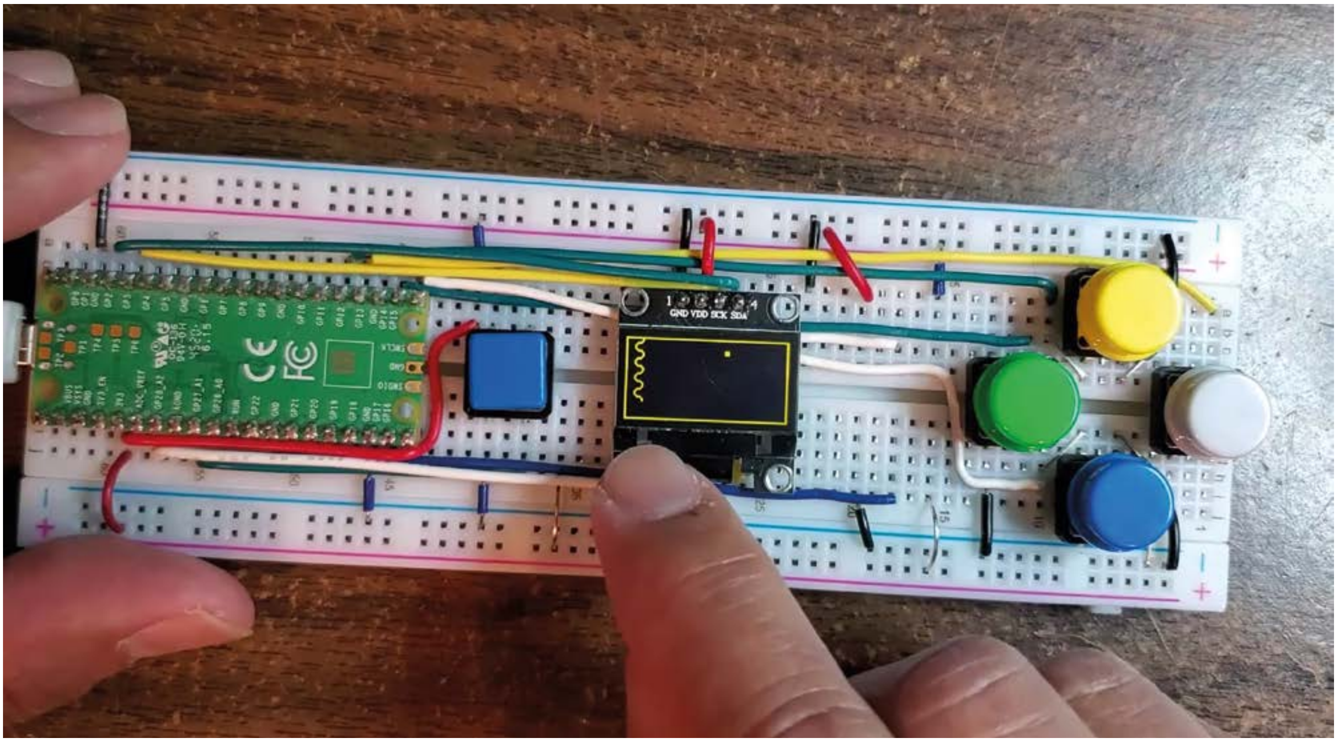
Ronin 3D-printed the robot arm from an Instructables project and set up his Pico Proto Shield as a controller

Pico's copious connection options, UART and I2C features made it ideal for both robot control and the Snake game he installed

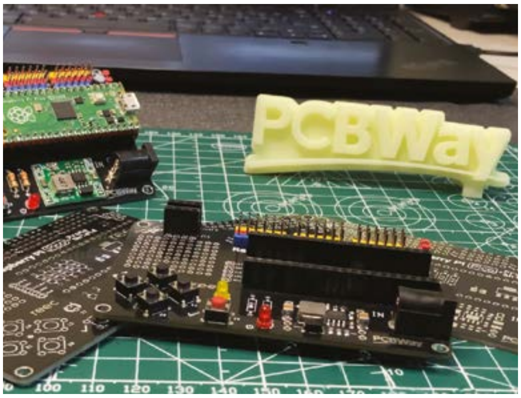
Ronin designed the Pico Proto Shield himself, adding space on his custom circuit board for an OLED screen



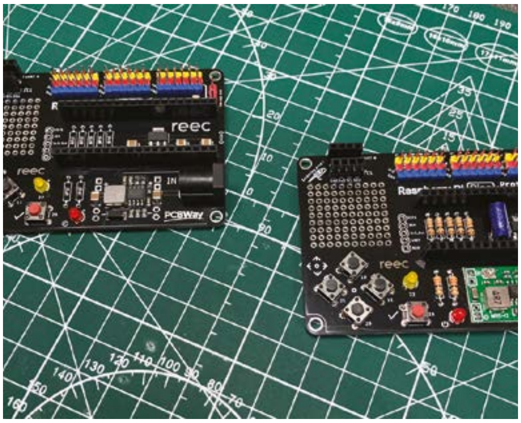
Assembling the 3D-printed the parts for the robot arm, ready to use with Pico



Ronin adapted Hari Wiguna's Pico Snake Game to display on the OLED on his Proto Shield



PCBWay helped with creating the PCB (printed circuit board) production and delivery



Most project builds are based on off-the-shelf parts, but PCB production is a complex process

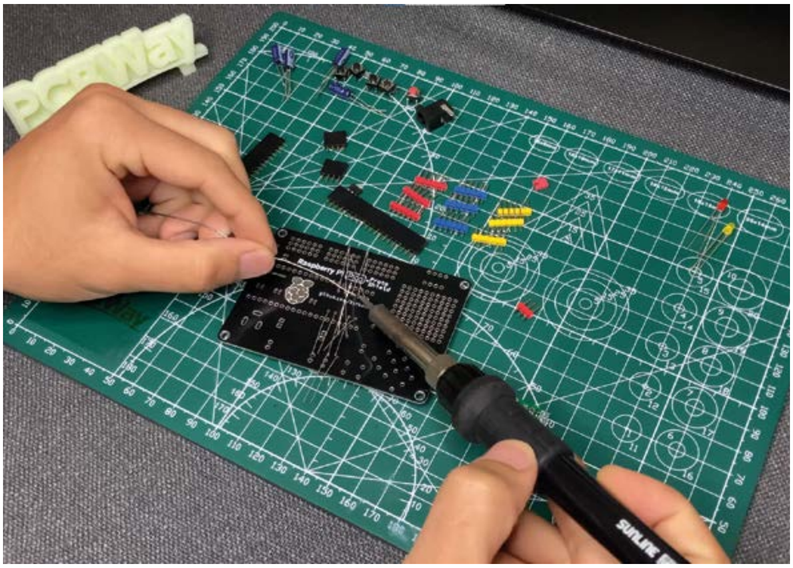
well as plenty of ports, a Proto Shield mimics the Arduino approach he was used to. Further research led Ronin to the idea of learning about Pico by making a robot arm – a popular Raspberry Pi project, for which he followed GHIZmo's Instructable: **magpi.cc/eezybotarm**. Ronin had access to a 3D printer to print parts for this.

He also decided to incorporate an OLED screen into his custom Proto Shield, turning his Pico into an interactive Snake game as well as a robot arm controller.

> "He had little experience of MicroPython before working on his inaugural Pico project"

Gaming the system

Quite sensibly, Ronin focused on the elements he could do well, namely developing a Proto Shield and designing a circuit board, reporting that Raspberry Pi's liberal provision of GPIO pins and PWM, UART, and I2C features "provided a lot of convenience in the design process of the Proto Shield." He chose Hari Wiguna's enticing Pico Snake game (**magpi.cc/picosnakeyt**) for its visual engagement, but had to write image-to-byte array instructions: **magpi.cc/imagetobyte**. Aside from

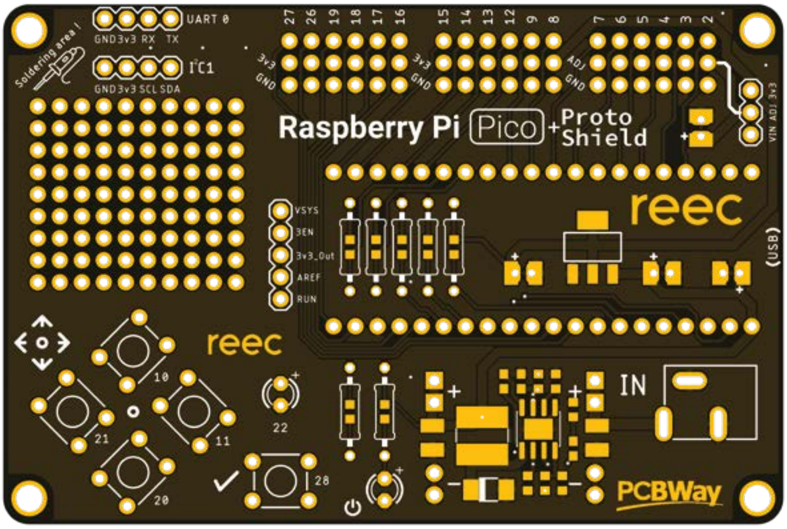


the time taken to order components and print parts, the whole thing took less than two days' work to assemble and get working. He cheerfully admits he had little experience of MicroPython before working on his inaugural Pico project, but says any coding errors were because he wrote code that would have worked in C rather than typing in Python commands.

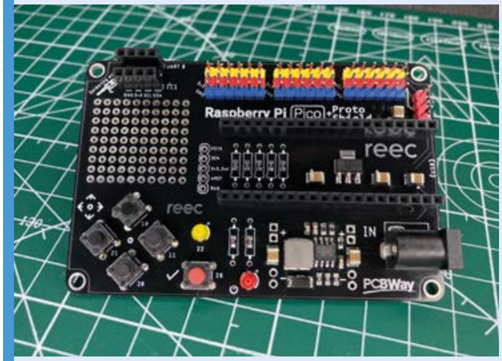
Responses to the project have been overwhelmingly positive, with so many email alerts he could barely keep up. Buoyed up by the feedback, Ronin is already hard at work on a similar project for a mini sumo robot, neatly recalling his first steps into robotics as a teenager.

▲ Soldering the components into place

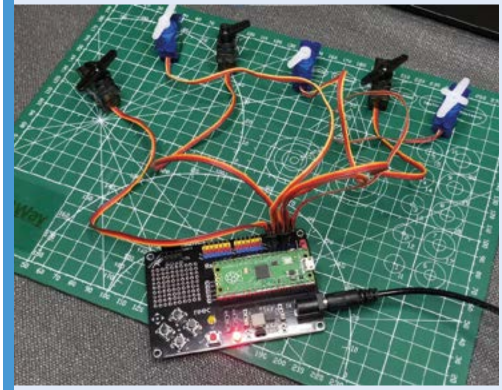
▼ Ronin has designed his own Proto Shield, which allows devices to be connected to Raspberry Pi Pico in a similar fashion to a HAT



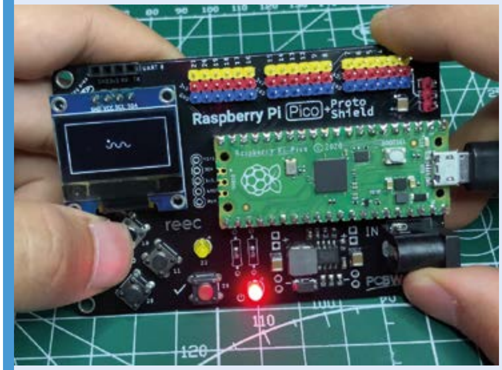
Proto Shield setup



01 Ronin details the Proto Shield design process at **magpi.cc/picoprotoshield**. For simplicity, you can send the PCB files to an online service such as PCBWay to print it.



02 The Proto Shield provides lots of extra options for attaching components such as motors and screens to Raspberry Pi Pico. Test the setup by trying out lights and other attachments.



03 Install the Snake game code from HariFun's GitHub page. The tiny keypad beneath the OLED makes for plenty of diverting fun. Next up, design your own case, perhaps?

Workshop Rudolph the Red Nose Reindeer

How do you get people to start learning programming around Christmas time? With Rudolph, of course. **Rob Zwetsloot** saw it, and even says it glows

MAKER

Bas op ten Berg

An expert in data analytics who now consults on electro-mechanical prototyping and new technologies, all while still making.

botberg.eu

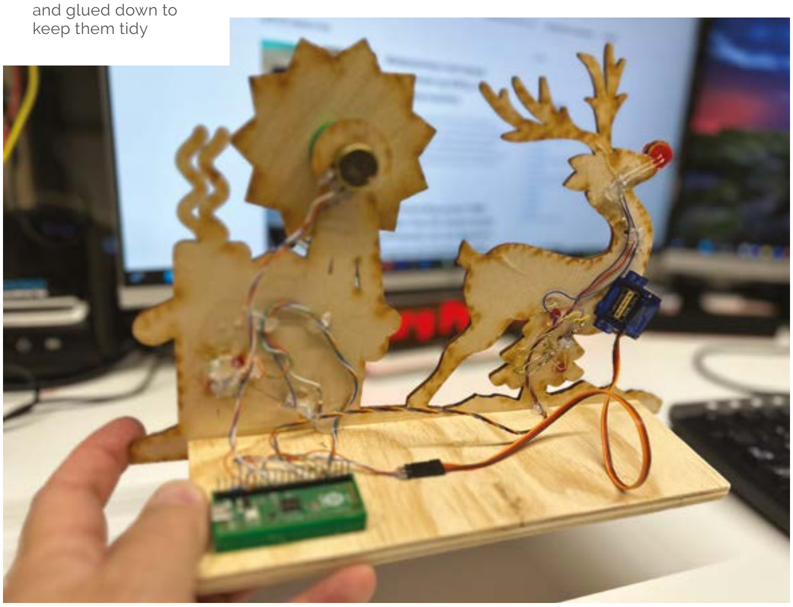
How to start with Python or programming in general is a question that the Raspberry Pi Foundation has been asking – and answering very well! – for over a decade now. When Bas op ten Berg was faced with this question, he went a slightly different route.

"The Workshop Rudolph The Red Nose Reindeer is a low-entry approach in starting learning to program Python with Raspberry Pi Pico," Bas explains. "I created this workshop when I was requested for opportunities to entertain a team of non-technical colleges, and get them enthusiastic with a positive programming experience."

Specifically, Rudolph uses MicroPython. Bas was "astonished" by how quick and simple it was to create scripts for Pico and get them running, which seemingly served as inspiration for his programming experience.

"I foresaw a big blinking LED as a red nose, an animated leg (actuated by a simple servo), a turning star connected to a potentiometer, and lots of blinking lights to support the Christmas theme," Bas says. "Simple, versatile to support users in programming their individual scenes, simple to connect by USB, and easy to update and modify with Thonny. Besides all of this, Pico was available at an attractive price."

▼ The parts are wired up in an easy to understand way – and glued down to keep them tidy



Very shiny nose

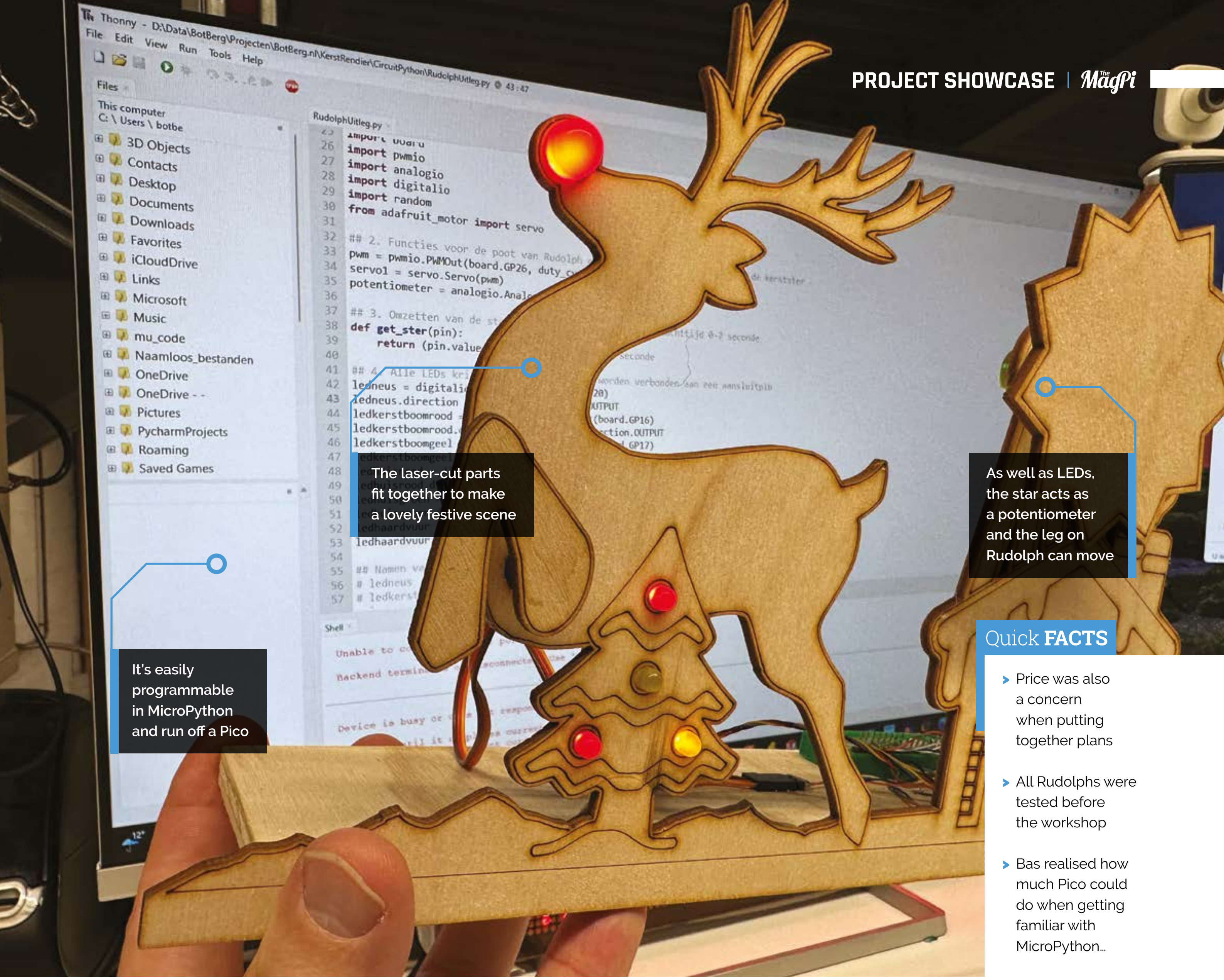
While drafting up the project, Bas made several examples using LEDs, servos, speakers, illustrations, etc., with the final version also using laser-cut and 3D-printed parts for people to construct.

"Rudolph, the burning wood-stove, the house, and the star are all laser-cut and glued together," Bas tells us. "There is a cut-out mounting hole to keep the micro servo in place. The front leg of Rudolph is glued to the servo stick. The potentiometer is mounted at the top of the chimney of the house. LEDs are mounted in the Christmas tree, Rudolph's nose, and in the house. Pico is clicked in a 3D-printed bracket to keep it in place, with the USB connector aligned to right. I created a full-option sketch where Rudolph's nose is blinking, his front leg is twisting, and all LEDs have their own rhythm. By turning the star knob, the analogue value changes, which then changes several parameters in the [script] to make it even more active."

Down in history

The workshop was quite successful, according to Bas, with attendees having the planned 'positive programming experience'.

"They were enthusiastic and eager to get started," Bas mentions. "At first, they were very surprised to have an electronic Christmas-themed Rudolph which they could connect to their own laptop. After a starting introduction of 30 minutes, they could get the first Christmas lights



The laser-cut parts fit together to make a lovely festive scene

As well as LEDs, the star acts as a potentiometer and the leg on Rudolph can move

It's easily programmable in MicroPython and run off a Pico

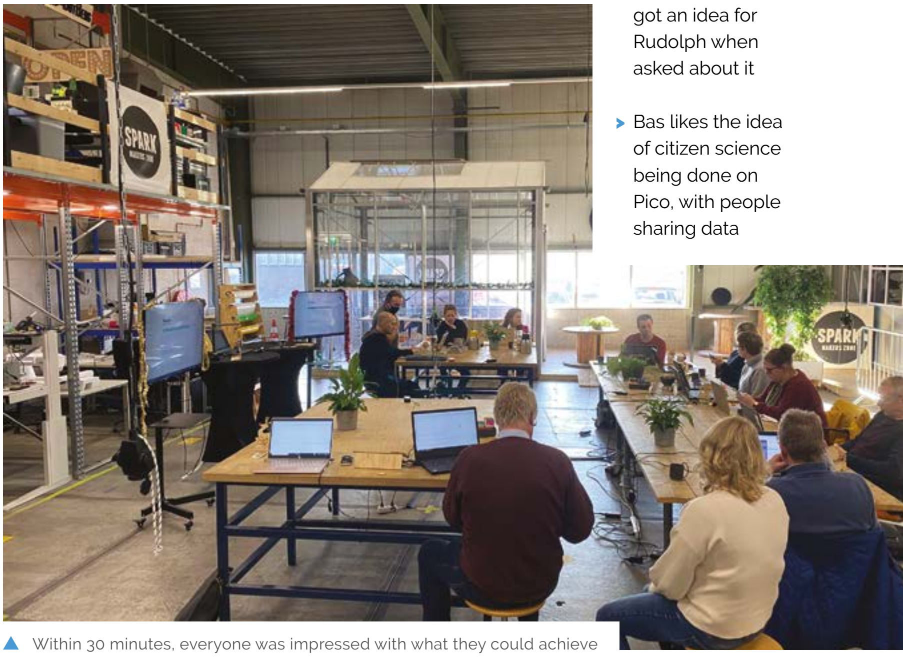
Quick FACTS

- Price was also a concern when putting together plans
- All Rudolphs were tested before the workshop
- Bas realised how much Pico could do when getting familiar with MicroPython…
- …and immediately got an idea for Rudolph when asked about it
- Bas likes the idea of citizen science being done on Pico, with people sharing data

> After a starting introduction of 30 minutes, they could get the first Christmas lights blinking and the front leg moving

blinking and the front leg moving. Once they were able to understand how to program a scenes themselves and get interaction with the rotating star, I saw various applications which I hadn't foreseen myself."

Bas is currently planning other holiday-themed workshops like this – such as an Easter bunny – and is going to bring back Rudolph next Christmas: "Having fun with programming with a low-entry approach is applicable throughout the whole year, and shouldn't be limited to the Christmas season only."



Within 30 minutes, everyone was impressed with what they could achieve

Raspberry Pi Backpack

Rather than rummage around his bag to set up Raspberry Pi, Nicholas Hacault has ensured his computer is always to hand, as **David Crookes** explains

MAKER

Nicholas Hacault

Nicholas Hacault is a Cyber Security student who loves tech and inventing.

nicholashacault.com

Portability is one of the great benefits of owning a small, yet powerful, computer such as Raspberry Pi. So long as you have a screen and keyboard to hand, you can pop one into your bag and take it on your travels. Or, as Nicholas Hacault has shown, you can actually make Raspberry Pi a feature of your bag and gain instant access to it wherever you go.

Nicholas came up with the idea when he was looking for a way of transporting a Linux device. At first, he considered running a virtual machine on his laptop. "But that sounded like too easy a solution," he says. "Since I was already bringing a bag to school, I figured that a backpack would be the perfect housing for my project." With the idea in the bag, he was ready to get going.

Pocket plan

One of his first decisions was the choice of backpack itself. "There were a lot of factors I had to consider," he says. "It needed to have space for my laptop, chargers, and books, as well as enough room to fit Raspberry Pi and all the components that come along with it. I decided that a bag with lots of different pockets would work best because it would make separating the project's components easier."

Eventually, he came across a backpack with two front pockets: a smaller one for Raspberry Pi and battery, and a larger one for the screen. The next step was working out how to incorporate the tech needed to get his backpack computer up and running. "I decided the minimum the project needed was a portable power bank, a seven-inch touchscreen display, a Bluetooth keyboard and mouse, and a Raspberry Pi Model B with 8GB of memory housed in a case," he says.

The display is powered through a USB cable so it can draw its juice directly from Raspberry Pi. The computer, in turn, is connected to a 30,000 mAh portable power bank. "It has a USB Type-C input/output rated at 20 watts, and it connects to the outside of the bag for easy charging," Nicholas says. "It also has two USB Type-A outputs rated for up to 22.5 watts, which is enough to power Raspberry Pi, and there are four 20 mm fans – two to pull air in and two to vent it back out."

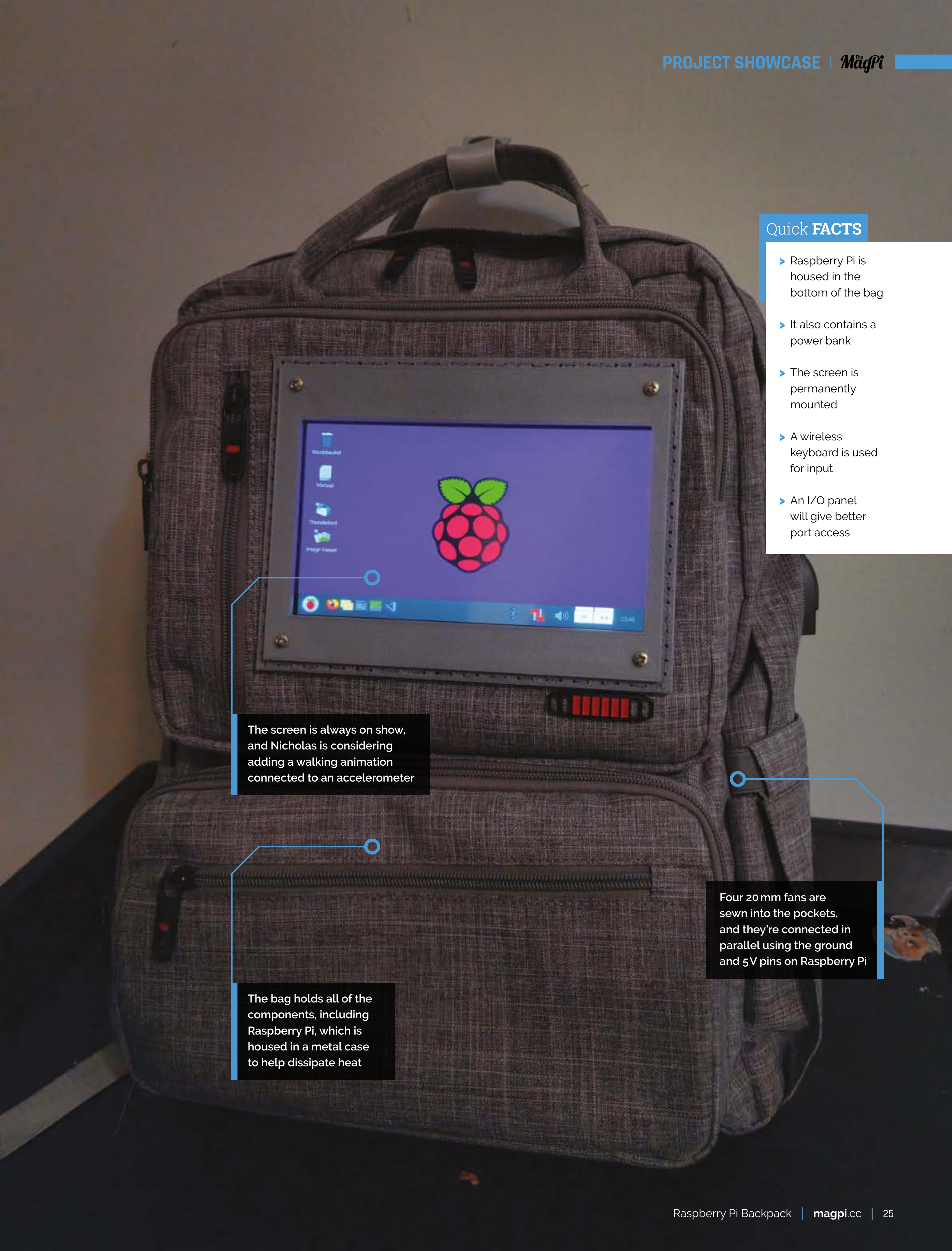
Packing away troubles

To house the components, Nicholas designed and 3D-printed a range of pieces. "I made a hole punch pattern around the edges of the prints in order to sew them on to the bag easily," he says. "The biggest challenge was working in a confined space and I went through many failed 3D-printed housings that always seemed to be slightly too big."

Thankfully, the programming side of the project was more straightforward. "There was super-minimal coding to get it running," he says. "All I had to do was follow the documentation to configure the touchscreen. The screen itself is actually installed upside down so the ports face the right direction. I needed to change the screen orientation in the Raspberry Pi config."

> " There was super-minimal coding to get it running "

Nicholas says there is still room for improvement. "The screen is always visible which is not great for waterproofing," he says. He also wants to be able to connect Raspberry Pi to a cell network or mobile hotspot so that it doesn't have to rely on Wi-Fi. Even so, it's proven very useful. "I can also use it to store items like a normal bag as well," he laughs.



The screen is always on show, and Nicholas is considering adding a walking animation connected to an accelerometer

Four 20 mm fans are sewn into the pockets, and they're connected in parallel using the ground and 5 V pins on Raspberry Pi

The bag holds all of the components, including Raspberry Pi, which is housed in a metal case to help dissipate heat

Quick FACTS

- Raspberry Pi is housed in the bottom of the bag
- It also contains a power bank
- The screen is permanently mounted
- A wireless keyboard is used for input
- An I/O panel will give better port access

Alternative Flight Simulator

One inventive aviation fan has cloned the ultimate cabin experience. **Nicola King** searches for her boarding pass

MAKER

Alex Shakespeare

Alex works in IT consultation, while using his free time to build unique gadgets for his home.

magpi.cc/alexair

Alex Shakespeare relishes the opportunity to fly anywhere and just has to have the window seat. Therefore, clearly, the only reasonable thing to do was to build his very own passenger jet plane experience, named 'Alex Air', but on terra firma. As a collector of aeronautical odds and ends, all he needed were some authentic plane seats (purchased from eBay), a plane window, and a few other flying-related gizmos. With the help of a Raspberry Pi Zero, Alex set about creating a truly unique and distinctive addition to his home.

First class upgrade

Apart from the seats, and various aesthetic aircraft appendages, there are two key sections to the Alternative Flight Simulator that help provide an authentic passenger experience. Firstly, Alex has constructed a 'flight selector panel' that the passenger uses to select the airport where they wish to land (and there is a wide choice).

Once a location is selected, a standard PC monitor behind the window plays the landing video, while two speakers emit some pertinent audio, so the passenger can fully immerse themselves in their approach into Innsbruck, or maybe New York. Alex managed to find all of the film/sounds for each specific flight approach on YouTube, then scaled and rotated them to fit the screen correctly. An overhead fluorescent light strip also kicks into action to give a realistic experience.

All of the above is controlled by a Raspberry Pi Zero: "I needed something that supports I2C/general IO, was capable of playing video and audio and, most importantly, was cheap," says Alex. Raspberry Pi Zero takes inputs from the selector panel and outputs to its small LCD display using I2C. "It's also responsible for video/audio playback, using the HDMI port and a USB sound card."

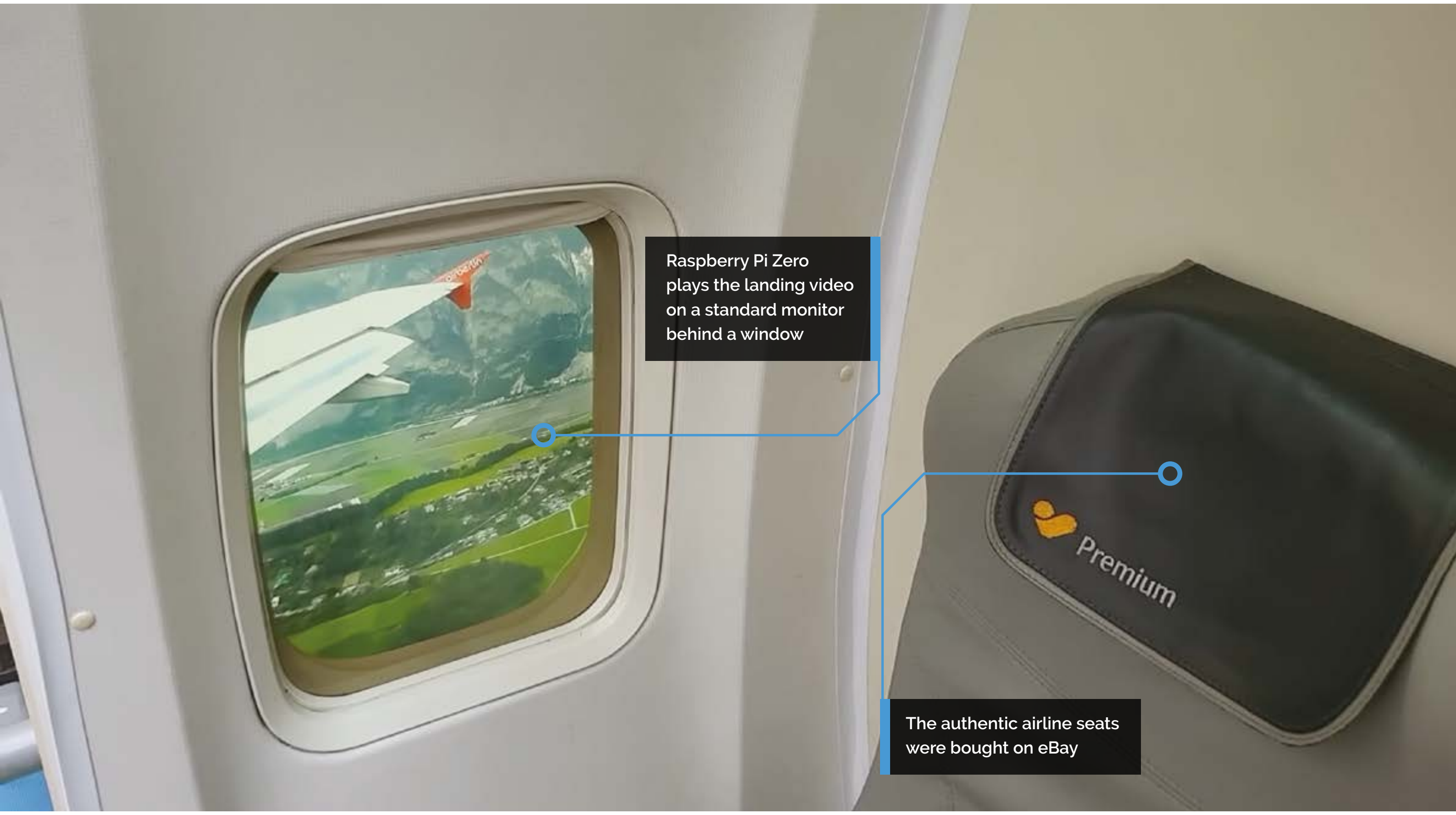
The second key section, independent of the selector panel, is the overhead 'customer service board' control panel, reclaimed from an old plane. This unit is powered by an ESP32 microcontroller, and the passenger can use it to switch on the light above, be cooled by the three-speed fan, or press the assistance button. Cleverly, Alex has arranged for his Google speakers to utter those immortal words "assistance required" whenever the passenger presses this button. Aside from an in-flight meal tray containing some unappetising foodstuffs, there really is very little missing from the Alex Air customer flight encounter, as he has even included a whimsical button which plays a Ryanair jingle (played in the firm's aircraft when they land on time).

> "I use it as a meeting room when taking video calls for work"

Brilliant backdrop

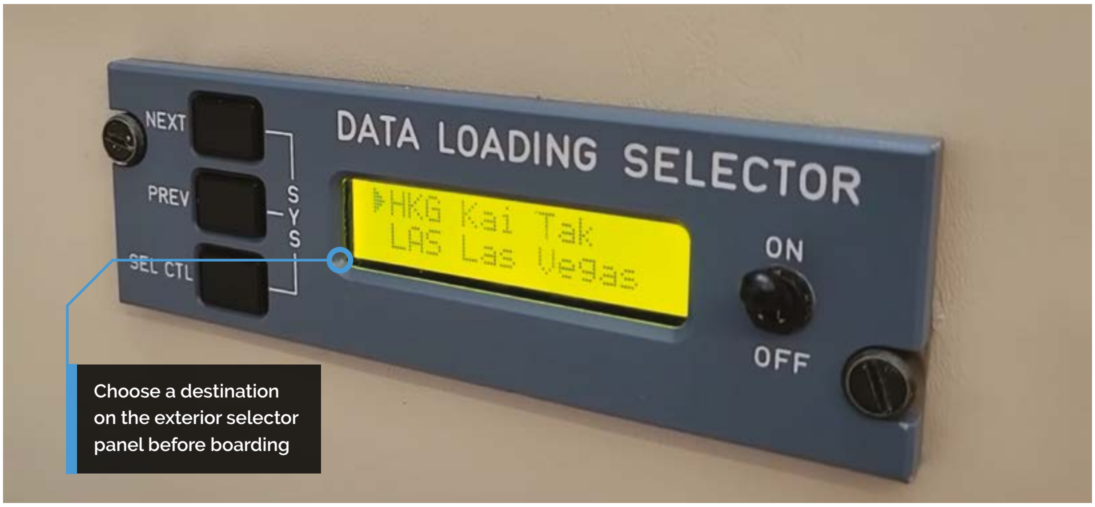
Housed in his conservatory, it looks like the Alternative Flight Simulator is there for the long haul, so to speak: "It's pretty much screwed into the wall so it's not going anywhere else. I use it as a meeting room when taking video calls for work… it beats just having a boring IKEA sofa in the conservatory." Furthermore, Alex Air has been very well-received, even featuring in a British national newspaper, a radio interview, and in blogs far and wide.

Alex admits that: "People think I'm a bit nuts but, having seen my other projects, generally expect this kind of thing from me." Indeed, Alex's other ingenious makes include an automated



Raspberry Pi Zero plays the landing video on a standard monitor behind a window

The authentic airline seats were bought on eBay



Choose a destination on the exterior selector panel before boarding

Quick FACTS

- The seats cost Alex around £150
- The window and other parts cost approximately £170
- He already had a spare monitor and Raspberry Pi Zero on hand
- Over three months, Alex spent around 40 hours on his project
- See Alex's informative simulator video: **magpi.cc/alexairvideo**

cocktail maker, an owl barometer, and an automatic pneumatic door.

While he doesn't plan on making any changes to his Flight Simulator, saying, "I think I'm done for now – it does the job," Alex does intend to post some source code soon, once he's had the opportunity to tidy it up a little. Keep an eye on his GitHub page (**github.com/shakso**) if you are interested in replicating this high-flying and deliciously eccentric venture.

Geiger Counter

Concerned about the potential of rising radiation levels in Ukraine, Dmytro Panin has set about monitoring them, as **David Crookes** explains

MAKER

Dmytro Panin

Dmytro Panin is a programmer based in Kyiv, Ukraine, and he wrote his first line of code aged eight. He works for a large provider of nearshore software engineering services.

magpi.cc/onlinegeigercounter

Our initial contact with Dmytro Panin came in August 2021 when we looked at his Air Quality Traffic Light – a project designed to monitor the pollution in the atmosphere of his home city of Kyiv in Ukraine. Never did we ever imagine, little more than a year later, he would be seeking to detect dangerous levels of radiation in the air, but that's where he finds himself today.

"The vast majority of the projects I build are created out of necessity," he tells us. "This one, unfortunately, is no exception. With almost daily threats of a nuclear catastrophe at one of our power plants due to Russian shelling or hints of a nuclear strike, some of us started to wonder how to detect an increasing level of radiation if something were to happen."

This prompted him to create a device using a Raspberry Pi Pico W. "Radiation is not something you can feel, see, taste, hear, or smell," he says. "So, I created a device designed to be a silent observer of the ionising radiation levels in my city. I wanted it to become the sixth missing sense, if you will."

Keeping count

As well as Raspberry Pi Pico W, the build includes an M4011 Geiger–Müller tube. It's old technology, for sure – the concept was developed back in 1908 at the University of Manchester and it came into being 20 years later. "But I've never had an excuse to work with it, so I took it as a way to improve my knowledge," Dmytro says. "It may be old, but it's very important tech."

At first, he tried to find all the components required to build a rudimentary Geiger counter from scratch before realising it would take him too long to locate everything he needed. "I then came across a pre-built module based on the M4011 which has a digital pin that can relay the tube's pulse caused by an ionising event," he continues.

"I configured an interrupt on one of the pins of my Raspberry Pi Pico W so it can listen and 'remember' the number of 'ticks' that the Geiger tube experiences over a period of time. Based on that, I can interpolate (or sometimes extrapolate) the current CPM (counts per minute)."

Stark warning

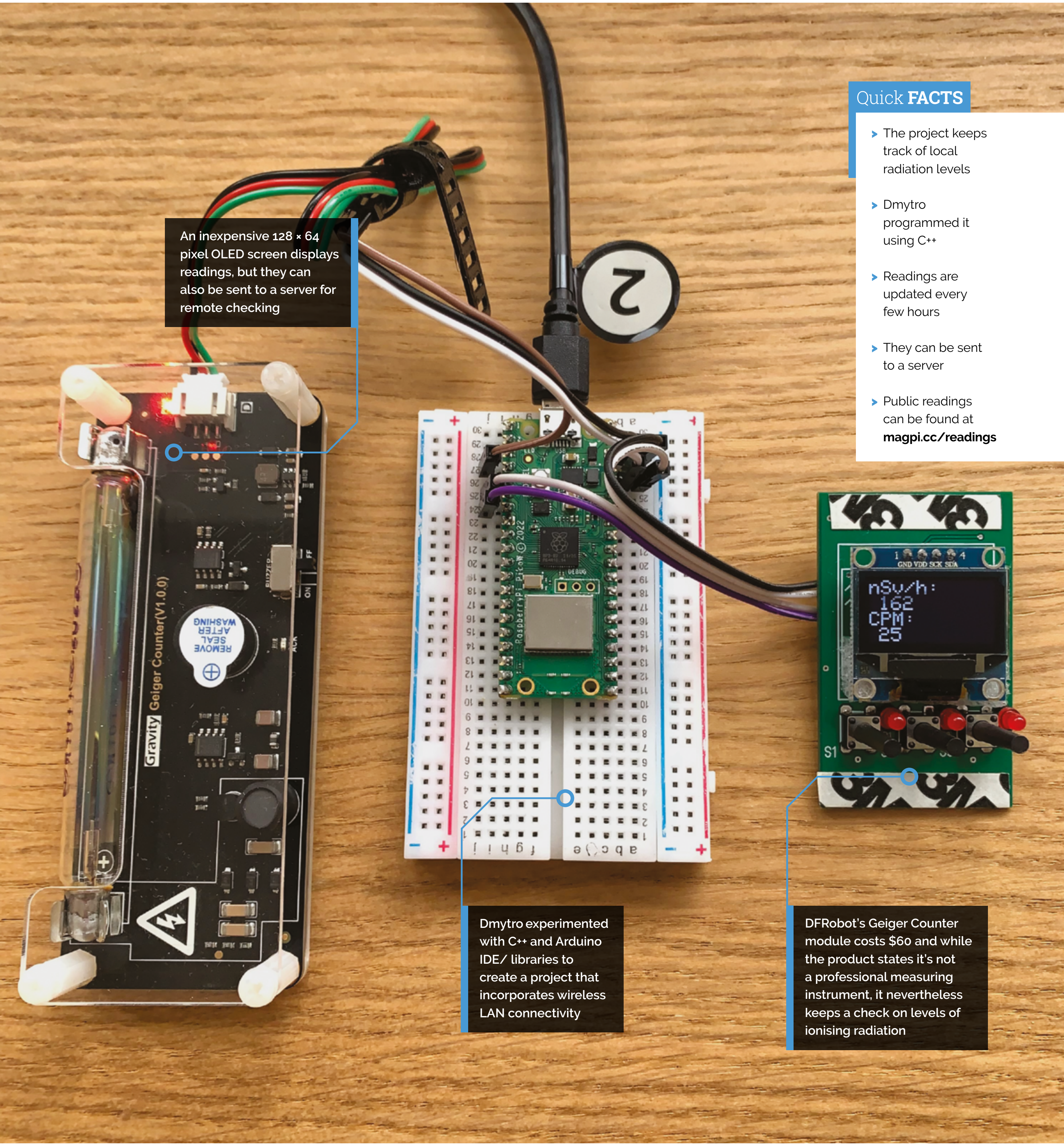
Understanding that CPM – while generally accepted and widely used in measuring radiation – doesn't universally equate to dose rate, he implemented a conversion to sieverts per hour, a unit that allows radiation levels to be expressed. "Fortunately for me, DFRobot, the creator of the module, had an easy conversion formula for M4011," Dmytro says. "So I decided to use that."

Dmytro has placed the device next to a window in his apartment and it's connected to an OLED screen so that he can keep an eye on the current radiation level. It's also possible to check the device from afar.

> "Due to the power blackouts we're having, the readings are not updated very often"

"Raspberry Pi Pico W can push the readings to a server every few hours so I can keep track of what's happening with the ionising radiation levels even if I'm not home," he explains. "But, unfortunately, due to the power blackouts we're having, the readings are not updated very often as I don't keep the device plugged in any more."

So far, the device has been showing around 25 CPM. "That's what you expect from a tube like this in a 'normal' environment," Dmytro says. "As for what reading would indicate a problem, well, I hope we won't need to learn this," he says. "But, overall if the readings are drastically higher than what they are at the moment, I'd interpret it as a problem that requires more thorough investigation." Fingers crossed it never comes to that.



An inexpensive 128 × 64 pixel OLED screen displays readings, but they can also be sent to a server for remote checking

Dmytro experimented with C++ and Arduino IDE/ libraries to create a project that incorporates wireless LAN connectivity

DFRobot's Geiger Counter module costs $60 and while the product states it's not a professional measuring instrument, it nevertheless keeps a check on levels of ionising radiation

Quick FACTS

- The project keeps track of local radiation levels
- Dmytro programmed it using C++
- Readings are updated every few hours
- They can be sent to a server
- Public readings can be found at **magpi.cc/readings**

SUCCESS STORY magpi.cc/success

BioBusiness

Stable supply and a "golden" relationship with technical staff supported the medical equipment company to scale

Egyptian medical equipment manufacturer BioBusiness needed to integrate IoT capability into its successful line of patient monitoring devices. Raspberry Pi technology allowed them to do it. BioBusiness was founded in 2009 by Mostafa Elwakeel and two medical and software engineer colleagues, with a focus on equipment, manufacturing, and design, as well as IoT related to medical devices. The three entrepreneurs, all recent graduates, set up a medical equipment business with a primary R&D centre in Cairo, Egypt and a production factory in Serdang, Malaysia.

The trio recognised the potential for Internet of Things-based monitoring products in medical settings, and set about creating a patient-monitoring ECG machine, ventilators, and other critical equipment. Demand for such devices was immediate: as well as manufacturing for sale under its own brand in Egypt, BioBusiness soon found itself making ventilators and other equipment, such as CPAP monitors, on behalf of some of the biggest names in pharmaceutical and medical equipment, including Philips, Siemens, and GE.

THE CHALLENGE

BioBusiness soon began manufacturing ventilators, as well as IoT devices that captured and shared medical imaging data, including ultrasounds, radiation, X-rays, and MRIs. For its next product line, BioBusiness wanted to develop IoT monitoring devices on an OEM (Original Equipment Manufacturer) basis, for sale to the likes of Siemens, GE, and Philips. The devices needed to provide detailed environmental information including temperature, humidity, and air quality readings, as well as measuring and analysing three-phase power.

However, the need to connect everything to the cloud complicated things. The field engineer support team needed to be able to remotely monitor MRI and CT machines which, in a hospital environment, may need to be mobile. More pressingly, BioBusiness needed to secure components for its growing Medicare portfolio in sufficient quantities that it could scale up manufacturing as its client base increased and, with it, demand for its products.

THE SOLUTION

BioBusiness chose Raspberry Pi Compute Module 4 and RP2040 microcontrollers when it became challenging to meet growing demand with its usual microcontroller supply and semiconductor RAM. Its CPAP (continuous positive airway pressure) device uses RP2040 to measure oxygen levels over time. "It's the main portable device monitor that controls the power inputs and pressure sensors, and runs a display with LCD data needed by a patient," says Mostafa Elwakeel. It also sells a Raspberry Pi-based oxygen concentrator that makes efficient use of the oxygen in ambient air and lets the user adjust the flow, plus a power quality monitor.

> "The company is almost unique in Egypt in terms of its medical equipment offerings"

WHY RASPBERRY PI?

BioBusiness chose Raspberry Pi for these life-critical monitoring machines after a thorough review of the technical specification of both Compute Module 4 and RP2040. An initial order of 14,000 RP2040s duly arrived, providing a sizable inventory to draw on as demand stepped up. Although the size of the order represented a significant outlay for the 49-employee company, their existing client list of approximately 500 customers in Egypt, as well as their multinational Medicare clients, and their confidence in Raspberry Pi's reliability and support, meant it was a calculated risk. "We appreciate the good availability of the RP2040," confirms Elwakeel. Having an established relationship with Raspberry Pi helped ensure continuity of supply of CM4 when silicon shortages hit global supply chains.

Ease of integration with existing products, and the flexibility to use a choice of components in a range of existing and planned Medicare products, mean that Raspberry Pi and Linux have proven a

good choice for BioBusiness, while the relationship built up with technical staff at Raspberry Pi has been in marked contrast to Elwakeel's experience with other computing component suppliers. He describes the relationship with Raspberry Pi as "golden". Steady reliability, with any issues encountered due to Linux rather than hardware, means BioBusiness is keen to continue using CM4 and RP2040 in its products, as it seeks to expand its portfolio as well as its geographical customer base.

THE RESULTS

BioBusiness has gone from strength to strength, and now uses Raspberry Pi's RP2040 microcontroller and Compute Module 4 in its oxygen concentrator, CPAP machine, and power quality monitor products. The company is almost unique in Egypt in terms of its medical equipment offerings, something Elwakeel would like to see change. "Unfortunately, ours is the only company in Egypt that produces this kind of R&D. It's good for us, but not good for the country."

Most of BioBusiness's client list has been built up through a combination of meetings and representation at industry events such as MEDICA, and awareness-raising in the medical trade press. Government support during the Covid-19 pandemic was important too: the Egyptian government encouraged BioBusiness to expand to help meet a critical need for medical equipment to diagnose cases and care for patients.

BioBusiness already has a strong business presence in Southeast Asian countries such as Indonesia, Malaysia, Singapore, Thailand, and Micronesia. For the future, Elwakeel sees further expansion in Asia and the Middle East. He envisages new products based around Raspberry Pi, with a focus on integrating RTLS (real-time location systems).

SUBSCRIBE TODAY FOR JUST £10

Get 3 issues + FREE Pico W

Subscriber Benefits

- FREE Delivery
  Get it fast and for FREE
- Exclusive Offers
  Great gifts, offers, and discounts
- Great Savings
  Save up to 35% compared to stores

Subscribe for £10

- Free Pico W
- 3 issues of The MagPi
- Free delivery to your door
- £10 (UK only)

Subscribe for 6 Months

- Free Pico W
- 6 issues of The MagPi
- Free delivery to your door

£30 (UK)	£35 (USA)
£35 (EU)	£45 (Rest of World)

☎ Subscribe by phone: **01293 312193**

Subscribe online: **magpi.cc/subscribe**

✉ Email: **magpi@subscriptionhelpline.co.uk**

Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A free Pico W is included with a 6-month subscription in USA, Europe and Rest of World.

SUBSCRIBE TODAY AND GET A

FREE Raspberry Pi Pico W

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

WORTH **$6**

- A brand new RP2040-based Raspberry Pi Pico W 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

This is a limited offer. Not included with renewals. Offer subject to change or withdrawal at any time.

Buy now: magpi.cc/subscribe

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
50 HACKS & HINTS
#MONTHOFMAKING
Know your Raspberry Pi better!
BUILD A GEIGER COUNTER
RECREATE A BBC MICRO
GET STARTED WITH ELECTRONICS & CIRCUIT BUILDING

SUBSCRIBE on app stores

From **£2.29**

Available on the App Store

GET IT ON Google Play

50 HACKS & HINTS

Unleash the full power of Raspberry Pi and Pico with these handy hints.

By **Phil King**

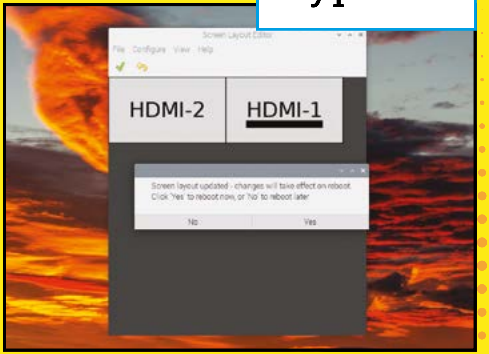
Dual monitor setup

Type: **OS**

With a Raspberry Pi 4 or 400, you can connect two monitors.

01 With the power off, connect two micro-HDMI to HDMI cables to the micro-HDMI ports on Raspberry Pi. Note that on Raspberry Pi 4, the primary display port is marked 'HDMI0', next to the power input; on Raspberry Pi 400, it's the one next to the microSD card slot. Connect each of the full-size HDMI ends of the cables to an HDMI input on one of your monitors/TVs. Turn on the displays and then the Raspberry Pi. When it boots up, a desktop should appear on each monitor.

02 The primary display, with a taskbar at the top, should be on the left-hand monitor. If not, go to Preferences > Screen Configuration to open a dialog showing the layout of the monitors. To switch them around, drag the left-hand one to the bottom, then the right-hand one to the left, and the bottom one back up to the right. Click the green tick icon to confirm and you'll be asked to reboot; click Yes.



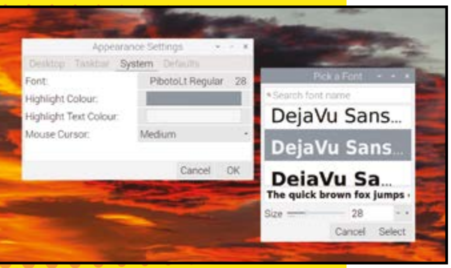
Screen layout updated - changes will take effect on reboot.
Click 'Yes' to reboot now, or 'No' to reboot later

03 With the monitors in the correct orientation once rebooted, you should now be able to move the mouse pointer between them, as if they were one long, continuous desktop. You can also drag windows between them to make full use of that increased screen real estate.

System font

Type: **OS**

You can change the system font and its size for Raspberry Pi OS. Right-click the desktop and select Desktop Preferences. In the Appearance Settings dialog, select the System tab and choose the desired font and point size. You can also alter the mouse cursor size. Select OK when done (or Cancel). Note that this won't change the text size in applications such as Chromium and Terminal.



Appearance Settings

Font: PibotoLt Regular 28
Highlight Colour:
Highlight Text Colour:
Mouse Cursor: Medium

Pick a Font

Size 28

Take a screenshot

Type: **OS**

To take a grab of the whole screen, press the **PRTSCN** key, found to the right of the function keys on most keyboards. Alternatively, use the `scrot` command in a Terminal window; it can be used with the `-d` switch to add a delay in seconds, e.g:

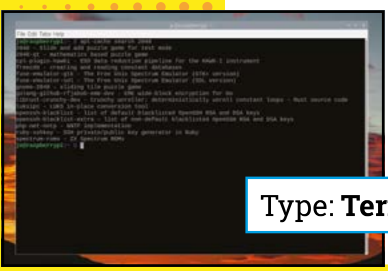
```
scrot -d 10
```

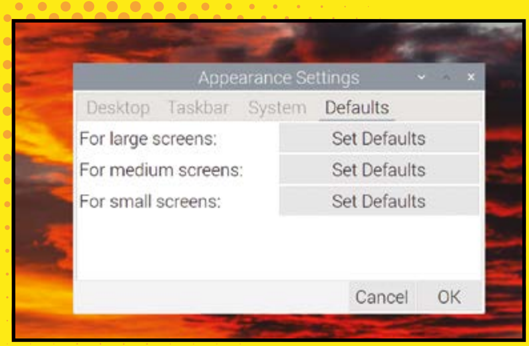
You can also add the `-s` switch and then select a window or drag an area of the screen to grab. By default, grabs are saved as PNG files with a timestamped name in the home directory, but you can add a path and file name to the command. See **magpi.cc/screenshot** for more details.

Software search

Type: **Terminal**

To search for software packages in the APT (Advanced Package Tool) repository from the command line, enter `apt-cache search` followed by the search term. Alternatively, download the **Sources.gz** file from **magpi.cc/aptsourcesbullseye** and use gunzip to extract it for the full list of software packages. Install one with `sudo apt install` followed by the name.





Appearance Settings

For large screens: Set Defaults
For medium screens: Set Defaults
For small screens: Set Defaults

Desktop icon size

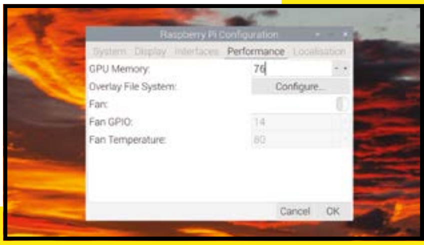
Type: **OS**

After connecting Raspberry Pi to a larger, higher-res TV or monitor (or a smaller one), you may find that the desktop icons in Raspberry Pi OS are too small (or large). To fix this, right-click the desktop and select Desktop Preferences. In the Appearance Settings dialog window, select the Defaults tab and then click 'Set Defaults' for a large, medium, or small screen. Then click OK to confirm, or Cancel the changes.

GPU memory

Type: **Hardware/OS**

By default, Raspberry Pi OS reserves a certain amount of RAM for the GPU (graphics processing unit): on models with 1GB or less of RAM, it's 64MB; on others, it's 76MB. You can alter this by going to Preferences > Raspberry Pi Configuration, selecting the Performance tab, and entering a new value in the GPU Memory field. You may need more for graphics-intensive applications, or less for other uses.



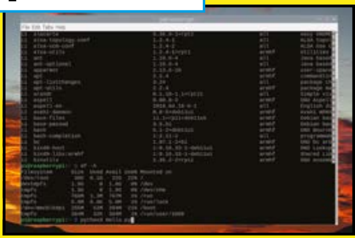
Raspberry Pi Configuration

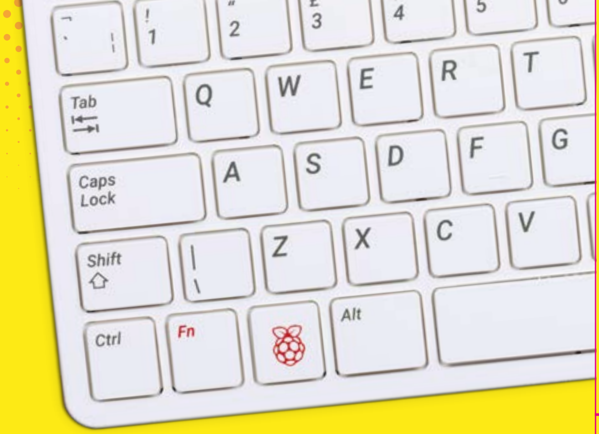
GPU Memory: 76
Overlay File System: Configure...
Fan:
Fan GPIO: 14
Fan Temperature: 80

It's history

Type: **Terminal**

A very handy feature at the command line is the ability to cycle through previous commands you've entered, simply by pressing the up arrow key repeatedly, so you can re-enter them quickly. You can press the down arrow to move back down through them.



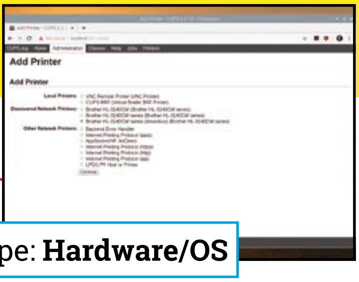


Menu shortcut

Type: **OS**

A quicker way to bring up the applications menu, without clicking the raspberry icon at the top left, is to press the Raspberry Pi logo key on an official keyboard, or **CTRL+ESC**. Then use the arrow keys to navigate its options, and **ENTER** to select.

Printing


Add Printer

Add Printer

Local Printers: VNC Remote Printer (VNC Printer)
CUPS-BRF (Virtual Braille BRF Printer)
Discovered Network Printers: Brother HL-3140CW (Brother HL-3140CW series)
Brother HL-3140CW series (Brother HL-3140CW series)
Brother HL-3140CW series (driverless) (Brother HL-3140CW series)
Other Network Printers: Backend Error Handler
Internet Printing Protocol (ipps)
AppSocket/HP JetDirect
Internet Printing Protocol (https)
Internet Printing Protocol (http)
Internet Printing Protocol (ipp)
LPD/LPR Host or Printer

Type: **Hardware/OS**

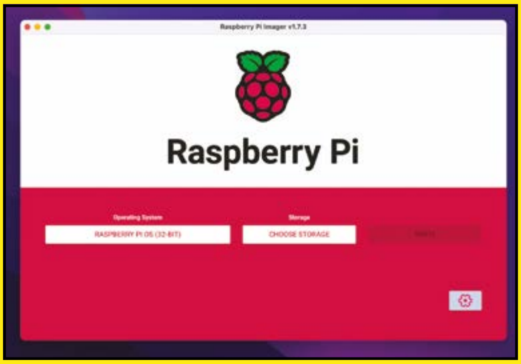
Want to print something out from your Raspberry Pi? You'll need CUPS, which should be pre-installed on the latest version of Raspberry Pi OS – if it's not, you can install it from a Terminal (see **magpi.cc/printing**). Point your Chromium browser at **localhost:631**, bypass the security warning, and you'll see the CUPS interface. From here, under Administration, after logging in with your username and password, you can add a network printer. You can then select it when printing with **CTRL+P**.

Custom OS setup

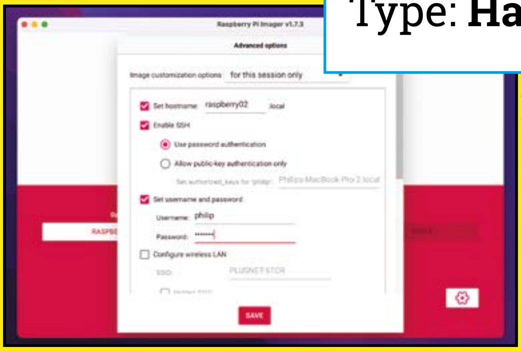
Type: **Hardware/OS**

Preconfigure a custom Raspberry Pi OS setup with the Imager tool.

01 Download Raspberry Pi Imager (from **magpi.cc/imager**) for your computer and launch it. Connect a microSD card to the computer using a USB card reader. Click on Choose OS and select a variant of Raspberry Pi OS. Then click the gear icon that's now appeared at the bottom right to open the advanced options.



02 At the top, you can set a custom hostname for Raspberry Pi – useful for finding it more easily on the Wi-Fi network and connecting via SSH, which you can enable here. Next, you can select a username and password.

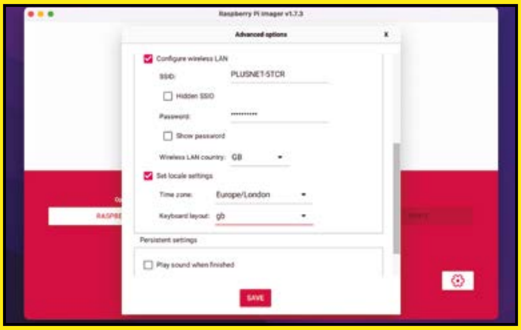

Raspberry Pi Imager v1.7.3

Advanced options

Image customization options: for this session only

☑ Set hostname: raspberry02 .local
☑ Enable SSH
● Use password authentication
○ Allow public-key authentication only
Set authorized_keys for 'philip': Philips-MacBook-Pro-2.local
☑ Set username and password
Username: philip
Password: •••••••
□ Configure wireless LAN
SSID: PLUSNET-5TCR

03 You can connect Raspberry Pi automatically to your Wi-Fi network when it boots up. Enter the SSID (name) of your router and its password. Select your Wi-Fi country, then your locale, timezone, and keyboard. Click Save, then write the OS to the microSD card as normal.


Raspberry Pi Imager v1.7.3

Advanced options

☑ Configure wireless LAN
SSID: PLUSNET-5TCR
□ Hidden SSID
Password: ••••••••••
□ Show password
Wireless LAN country: GB
☑ Set locale settings
Time zone: Europe/London
Keyboard layout: gb

Persistent settings
□ Play sound when finished

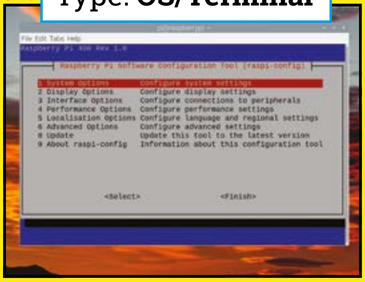
Alt window control

Type: **OS**

To move a window around, hold the **ALT** key and drag it from the middle – or hold the right mouse button and drag to resize it. To make the current window full-screen, press **CTRL+ALT** and the up arrow; to revert, press **CTRL+ALT** and the down arrow. To send it to the left or right half of the screen, press **CTRL+ALT** and the left or right arrow.

Raspi-config

Type: **OS/Terminal**



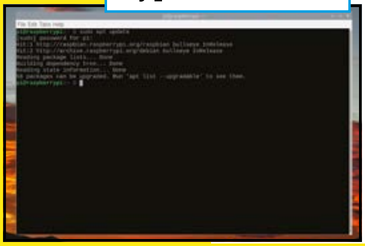
```
Raspberry Pi 400 Rev 1.0

┤ Raspberry Pi Software Configuration Tool (raspi-config) ├

1 System Options       Configure system settings
2 Display Options      Configure display settings
3 Interface Options    Configure connections to peripherals
4 Performance Options  Configure performance settings
5 Localisation Options Configure language and regional settings
6 Advanced Options     Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config   Information about this configuration tool

        <Select>                <Finish>
```

While you can alter common system settings in Preferences > Raspberry Pi Configuration on the desktop, there is an alternative method accessible from the command line. In a Terminal window or SSH connection, enter the command `sudo raspi-config` to open this configuration tool. Use the arrow keys and **ENTER** to navigate the menus. After changing settings, select Finish to save them – you may be asked to reboot. See **magpi.cc/raspiconfig** for all the details.

> “you can alter common system settings in Preferences”

Sudo it

Type: **Terminal**

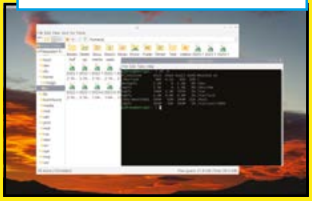


```
pi@raspberrypi:~ $ sudo apt update
[sudo] password for pi:
Hit:1 http://raspbian.raspberrypi.org/raspbian bullseye InRelease
Hit:2 http://archive.raspberrypi.org/debian bullseye InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
58 packages can be upgraded. Run 'apt list --upgradable' to see them.
pi@raspberrypi:~ $
```

While you can enter many commands as they are, some require ‘superuser’ (aka root) privileges to execute, such as when installing software. In this case, you need to prefix the command with `sudo` (short for ‘superuser do’) – you'll then be asked to enter your password to proceed.

Check storage left

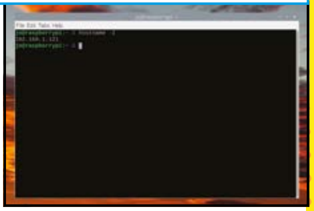
Type: **OS/Terminal**



When on the desktop, you can easily see the free system storage remaining at the bottom of the File Manager window. Alternatively, if working headless, you can enter the Terminal command `df -h`.

Check IP address

Type: **Terminal/Network**



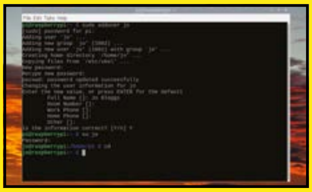
```
jo@raspberrypi:~ $ hostname -I
192.168.1.121
jo@raspberrypi:~ $
```

While you can hover the mouse pointer over the Wi-Fi symbol on the desktop to see your Raspberry Pi's IP address, you can also find it from the command line. Just enter the command `hostname -I`.

Add and switch users

Type: **Terminal**

You're not limited to one user on Raspberry Pi. You can add more users and switch between them. To add a user, enter the command sudo adduser followed by the desired username. You'll be asked to enter the current user's password, then a new one for the new user. You can then enter optional user details or leave them blank by pressing **ENTER** repeatedly. To switch to another user, enter `su` followed by the username and enter their password.



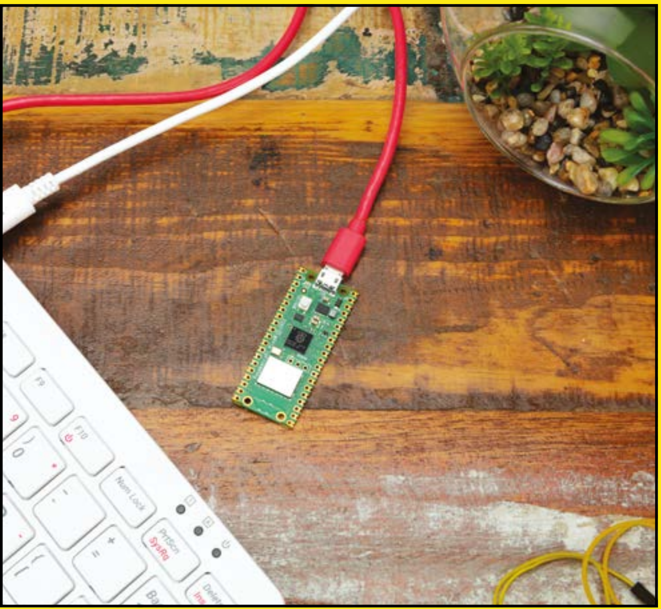
```
pi@raspberrypi:~ $ sudo adduser jo
[sudo] password for pi:
Adding user `jo' ...
Adding new group `jo' (1002) ...
Adding new user `jo' (1002) with group `jo' ...
Creating home directory `/home/jo' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for jo
Enter the new value, or press ENTER for the default
        Full Name []: Jo Bloggs
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
pi@raspberrypi:~ $ su jo
Password:
jo@raspberrypi:/home/pi $ cd
jo@raspberrypi:~ $
```

Connect Pico W to Wi-Fi

Type: **Pico**

Get Pico W to connect to your wireless network.

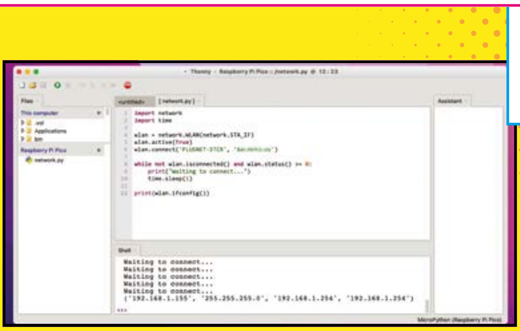
01 Unlike a standard Pico board, Pico W comes with Wi-Fi. To connect it to your wireless network, you'll need to write a short program. After flashing Pico W with the MicroPython firmware (from **magpi.cc/micropython**), connect it to a computer via USB, open the Thonny IDE, and select the 'MicroPython (Raspberry Pi Pico)' interpreter at the bottom right.



02 At the top of our program, we import the network and time modules:

```
import network
import time
```

We then set up the wireless connection. Replace 'SSID' and 'Password' here with your own wireless network's details:



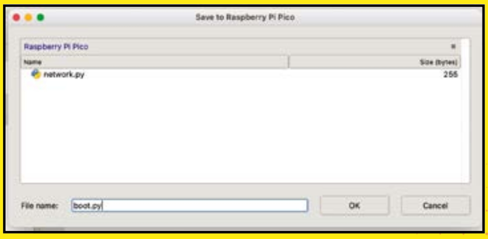
```
import network
import time

wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect('PLUSNET-5TCX', '[illegible]')

while not wlan.isconnected() and wlan.status() >= 0:
    print("Waiting to connect...")
    time.sleep(1)

print(wlan.ifconfig())
```

```
Waiting to connect...
Waiting to connect...
Waiting to connect...
Waiting to connect...
Waiting to connect...
('192.168.1.155', '255.255.255.0', '192.168.1.254', '192.168.1.254')
>>>
```

```
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect('SSID', 'Password')
```

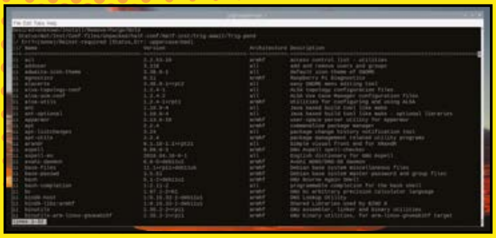
We then run a loop to display a message until the connection has been made. Once it has, Pico W's IP address will be shown (the first of the four displayed).

```
while not wlan.isconnected() and
wlan.status() >= 0:
     print("Waiting to connect...")
     time.sleep(1)

print(wlan.ifconfig())
```



File name: boot.py

03 Save and run the program to ensure a successful connection is made. You may well want Pico W to connect automatically to your wireless network whenever it boots up. If you save your Wi-Fi connection program as **boot.py**, it'll automatically run upon Pico W bootup. You could also embellish the program by handling connection errors and/or even making Pico W's on-board LED flash during connection. To disconnect from a network, run the command: `wlan.disconnect()`. Note that wireless connection when using the Pico C SDK involves a different process – see **magpi.cc/internetpicow** for details.



See installed packages

Type: **Terminal**

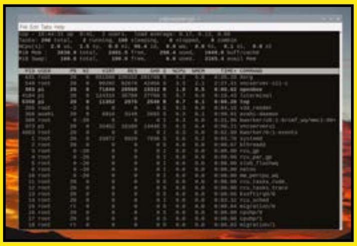
To find out which software packages are already installed on Raspberry Pi, enter the command `sudo dpkg -l`. You'll be presented with a list of packages; use the arrow keys to scroll up and down, and **Q** to exit.

> Find out which software packages are installed on your Raspberry Pi

See what's running

Type: **Terminal**

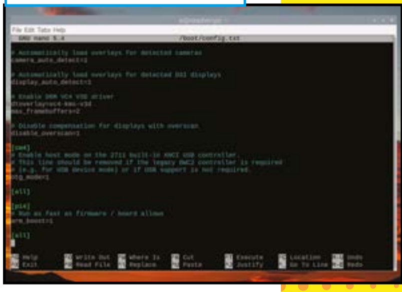
To list the processes running on your Raspberry Pi, enter the `htop` command to list them in descending order of CPU usage. This is a good way to discover a rogue process that's slowing down your machine. You can search and filter the processes by name or user and kill a rogue process that is using too much memory – or use the `sudo kill` command with its PID.



Edit config.txt

Type: **Terminal**

In place of the BIOS found on most PCs, a Raspberry Pi computer has a **config.txt** file. Found in the boot partition of the microSD card, it runs at bootup to configure certain settings. To edit it from the command line, enter `sudo nano /boot/config.txt`. Or access it directly from the microSD card inserted into another computer. By uncommenting certain lines in the file, you can trigger options such as forcing an HDMI output to a monitor. Or you can add extra lines such as to rotate an LCD display. For more details, see **magpi.cc/configtxt**.

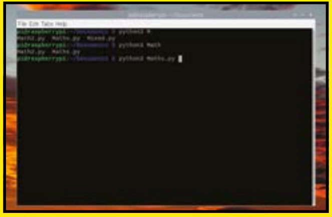


```
  GNU nano 5.4                    /boot/config.txt
# Automatically load overlays for detected cameras
camera_auto_detect=1

# Automatically load overlays for detected DSI displays
display_auto_detect=1

# Enable DRM VC4 V3D driver
dtoverlay=vc4-kms-v3d
max_framebuffers=2

# Disable compensation for displays with overscan
disable_overscan=1

[cm4]
# Enable host mode on the 2711 built-in XHCI USB controller.
# This line should be removed if the legacy DWC2 controller is required
# (e.g. for USB device mode) or if USB support is not required.
otg_mode=1

[all]

[pi4]
# Run as fast as firmware / board allows
arm_boost=1

[all]
```

Tab auto-complete

Type: **Terminal**

When entering commands in a Terminal window or via SSH, this feature can help speed things up. Start typing a command, file, or directory name and then press **TAB** to either auto-complete it fully or partially – in the latter case, press **TAB** again to see the possibilities, then type some more characters.

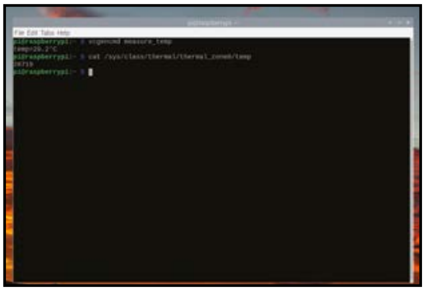


```
pi@raspberrypi:~/Documents $ python3 M
Math2.py  Maths.py  Mixed.py
pi@raspberrypi:~/Documents $ python3 Math
Math2.py  Maths.py
pi@raspberrypi:~/Documents $ python3 Maths.py
```

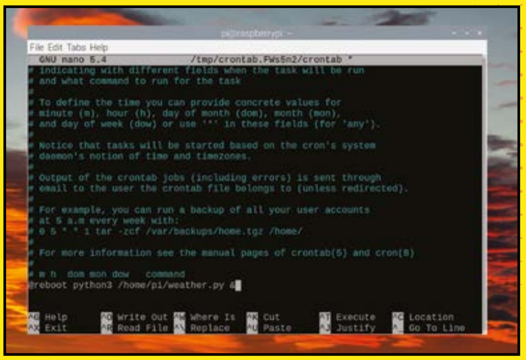
Check temperature

Type: **Terminal/Hardware**

Want to know how cool your Raspberry Pi is running? Check the temperature of its GPU (graphics processing unit) with the command `vcgencmd measure_temp`. For the CPU temperature, enter `cat /sys/class/thermal/thermal_zone0/temp` – note that you'll need to divide this figure by 1000 to get the temperature in degrees Celsius.



```
pi@raspberrypi:~ $ vcgencmd measure_temp
temp=29.2'C
pi@raspberrypi:~ $ cat /sys/class/thermal/thermal_zone0/temp
28719
pi@raspberrypi:~ $
```



```
  GNU nano 5.4                  /tmp/crontab.FWs5n2/crontab *
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
@reboot python3 /home/pi/weather.py &

^G Help       ^O Write Out  ^W Where Is  ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File  ^\ Replace   ^U Paste      ^J Justify    ^_ Go To Line
```

Auto-run programs

Type: **OS/Terminal**

If you've created a project that you want running all the time, you'll want to ensure that its program starts up automatically whenever Raspberry Pi is booted up. There are several ways of doing this, but one of the simplest is to create a cron job to run it. From a Terminal, enter `crontab -e`. If this is the first time you've used crontab, you'll need to select a text editor, such as Nano, from the options. At the bottom of the file, you can add a line to run one or more programs at set times or upon reboot. For the latter, enter `@reboot` followed by the command to run the code, including the full path to the file, e.g. `@reboot python3 /home/pi/weather.py &`. That `&` ensures it'll run in the background. Press **CTRL+X**, then **Y** to save it. For more details, see the crontab manual: **magpi.cc/crontab**.

Alternative OS

Type: **OS**

While Raspberry Pi OS is a great operating system, there are numerous alternatives you might want to try, some of them designed for specialist purposes such as 3D printing, digital signage, and CCTV. Check out our feature on Raspberry Pi operating systems in issue 111: **magpi.cc/111**.

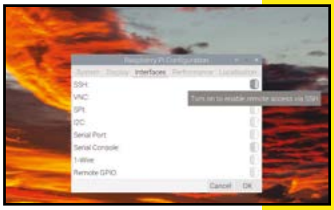
SSH remote access

Type: **Remote**

Secure Shell (SSH) is a useful way to access and control your Raspberry Pi from another computer or device. You'll first need to enable it in Raspberry Pi OS: go to Preferences > Raspberry Pi Configuration, select the Interfaces tab, and turn on the switch next to SSH. Alternatively, you can enable it in the raspi-config tool from the command line.

Once enabled, you can SSH into Raspberry Pi from another computer. On Linux, macOS, or Windows 10, open a Terminal and enter `ssh <user>@<IP>` – for example, `ssh pi@192.168.1.121`. Enter the password to connect. You can now enter commands on your Raspberry Pi, just as if you were in a Terminal on it.

Alternatively, you can use the PuTTY application (**putty.org**) for Windows to connect. There are also SSH apps available for Android and iOS devices.

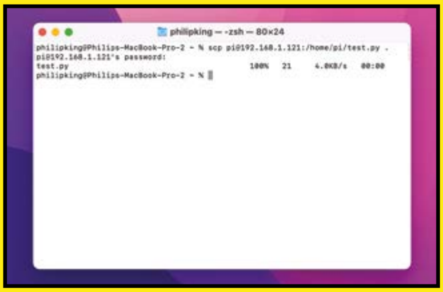


SCP secure copy

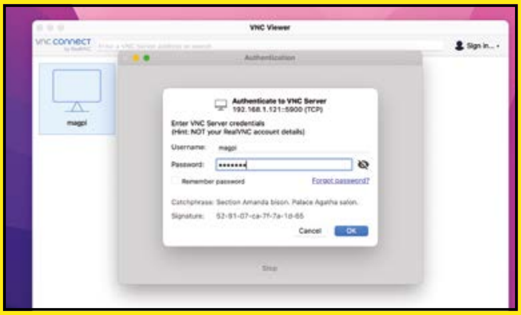
Type: **Remote**

With SSH enabled, you can transfer files securely from Raspberry Pi to another computer (and vice versa) using SCP (secure copy protocol). To do so, from a Terminal on the other computer, use the command: `scp <user@IP>:path/to/file path/to/` destination. For example, enter `scp pi@192.168.1.121:/home/pi/test.py /home/magpi/program.py`, to copy the **test.py** file on Raspberry Pi to **program.py** in the **magpi** directory on your local computer. You'll be prompted for the password. Note: you can omit the second file name to use the same name and/or replace the destination path with a dot (`.`) to use the current directory.

To copy a local file to the remote Raspberry Pi, use the format: `scp path/to/file <user@IP>:path/to/destination`. For example, `scp /home/magpi/program.py pi@192.168.1.121:/home/pi/test2.py`.



```
philipking@Philips-MacBook-Pro-2 ~ % scp pi@192.168.1.121:/home/pi/test.py .
pi@192.168.1.121's password:
test.py                                       100%   21     4.0KB/s   00:00
philipking@Philips-MacBook-Pro-2 ~ %
```



VNC Viewer

Authentication

Authenticate to VNC Server
192.168.1.121::5900 (TCP)

Enter VNC Server credentials
(Hint: NOT your RealVNC account details)

Username: magpi
Password: •••••••

□ Remember password

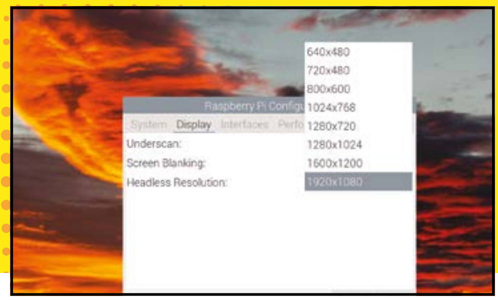
Catchphrase: Section Amanda bison. Palace Agatha salon.
Signature: 52-91-07-ca-7f-7a-1d-65

VNC remote access

Type: Remote

Another way to access Raspberry Pi remotely is to use VNC (Virtual Network Computing). This gives you full access to its GUI desktop. To enable VNC on Raspberry Pi, go to Preferences > Raspberry Pi Configuration, select the Interfaces tab, and turn on the switch next to VNC to enable it, then click OK. Alternatively, you can enable it in raspi-config from the command line. You'll see the VNC icon in the taskbar.

On another computer, download the VNC Viewer application from **magpi.cc/vncviewer** and launch it. Open a new connection and enter the IP address for your Raspberry Pi in the first field, give the connection an optional name, then click OK. Double-click the new connection, then enter the username and password to connect. You can now see and use Raspberry Pi's desktop as if you were running it locally.



Raspberry Pi Configu

Underscan:
Screen Blanking:
Headless Resolution:

640x480
720x480
800x600
1024x768
1280x720
1280x1024
1600x1200
1920x1080

Headless resolution

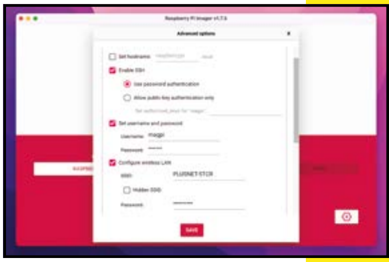
Type: Remote

You can set the desktop resolution for your remote VNC connection independently to that of a monitor to which Raspberry Pi is connected. Go to Preferences > Raspberry Pi Configuration, select the Display tab, and click on the value for Headless Resolution to select the desired resolution, e.g. 1920×1080.

Headless setup

Type: Remote

Rather than setting up Wi-Fi and enabling SSH from within Raspberry Pi OS, you may want to preconfigure it as a headless setup so you don't need to connect it to a monitor in the first instance. You can do so when writing the OS to a microSD card with Raspberry Pi Imager. Click on Choose OS and select a version of Raspberry Pi OS. Now click the gear icon to open the advanced options. Here, you can enable SSH and set the username and password, as well as adding connection details for your Wi-Fi router. You may also want to set a custom hostname, to use that to connect via SSH instead of the IP address. Once done, click Save, then flash your microSD card as usual. See 'Custom OS setup' (page 37) for details of the other advanced options.



Advanced options

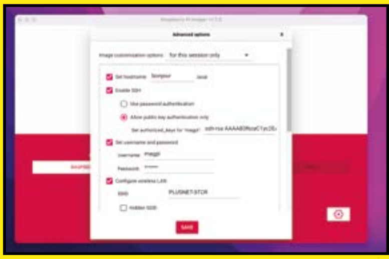
□ Set hostname: .local
☑ Enable SSH
● Use password authentication
○ Allow public-key authentication only
☑ Set username and password
Username: magpi
Password: ••••••
☑ Configure wireless LAN
SSID: PLUSNET-STCR
□ Hidden SSID
Password: ••••••••

SSH public key

Type: Remote

A more secure alternative to using a password for your SSH remote connection is to employ a public key. In Raspberry Pi Imager's advanced options, enable SSH and then select the 'allow public key authentication only' option. In a Terminal on your local computer, you'll need to generate the public and private keys: enter the command `ssh-keygen`. You have the option of renaming the files, or just pressing **ENTER** to skip, and adding an optional passphrase for extra security. Once generated, copy the contents of the public key (**id_rsa.pub** by default) into the 'set authorized_keys' field in Imager. Then click Save and flash the OS to the microSD card as usual.

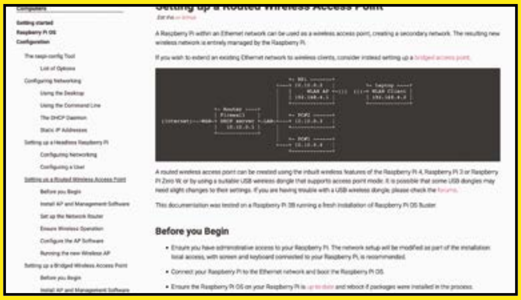
Now, when you SSH in from that computer, with the usual `ssh <user>@<IP or hostname>` command, it'll connect, showing the SHA fingerprint; type 'yes' to confirm. If you set a passphrase, you'll also have to enter that.



Advanced options

Image customization options: for this session only

☑ Set hostname: .local
☑ Enable SSH
○ Use password authentication
● Allow public-key authentication only
Set authorized_keys for 'magpi': ssh-rsa AAAAB3NzaC1yc2E
☑ Set username and password
Username: magpi
Password: ••••••
☑ Configure wireless LAN
SSID: PLUSNET-STCR
□ Hidden SSID



Computers

Getting started
Raspberry Pi OS
Configuration
The raspi-config Tool
List of Options
Configuring Networking
Using the Desktop
Using the Command Line
The DHCP Daemon
Static IP Addresses
Setting up a Headless Raspberry Pi
Configuring Networking
Configuring a User
Setting up a Routed Wireless Access Point
Before you Begin
Install AP and Management Software
Set up the Network Router
Ensure Wireless Operation
Configure the AP Software
Running the new Wireless AP
Setting up a Bridged Wireless Access Point
Before you Begin
Install AP and Management Software

Setting up a Routed Wireless Access Point

Edit this on GitHub

A Raspberry Pi within an Ethernet network can be used as a wireless access point, creating a secondary network. The resulting new wireless network is entirely managed by the Raspberry Pi.

If you wish to extend an existing Ethernet network to wireless clients, consider instead setting up a bridged access point.

```
                     +- RPi -------+
                 +---+ 10.10.0.2   |          +- Laptop ----+
                 |   |     WLAN AP +-)))  (((-+ WLAN Client |
                 |   | 192.168.4.1 |          | 192.168.4.2 |
                 |   +-------------+          +-------------+
 +- Router ----+ |
 | Firewall    | |   +- PC#2 ------+
(Internet)---WAN-+ DHCP server +-LAN-+---+ 10.10.0.3   |
 |   10.10.0.1 | |   +-------------+
 +-------------+ |
                 |   +- PC#1 ------+
                 +---+ 10.10.0.4   |
                     +-------------+
```

A routed wireless access point can be created using the inbuilt wireless features of the Raspberry Pi 4, Raspberry Pi 3 or Raspberry Pi Zero W, or by using a suitable USB wireless dongle that supports access point mode. It is possible that some USB dongles may need slight changes to their settings. If you are having trouble with a USB wireless dongle, please check the forums.

This documentation was tested on a Raspberry Pi 3B running a fresh installation of Raspberry Pi OS Buster.

Before you Begin

- Ensure you have administrative access to your Raspberry Pi. The network setup will be modified as part of the installation: local access, with screen and keyboard connected to your Raspberry Pi, is recommended.
- Connect your Raspberry Pi to the Ethernet network and boot the Raspberry Pi OS.
- Ensure the Raspberry Pi OS on your Raspberry Pi is up to date and reboot if packages were installed in the process.

Wireless hotspot

Type: **Remote/Camera**

When you're using your Raspberry Pi out and about – maybe with a Camera Module – without access to a Wi-Fi network, you may well want to access it from another device, such as a phone. For this, you can set Raspberry Pi up as a wireless access point (aka hotspot). For more details, see **magpi.cc/rpihotspot**, skipping the 'Enable Routing and IP Masquerading' section. Once connected to the hotspot, you should be able to SSH in to Raspberry Pi.

> A more secure alternative to using a password for your SSH remote connection is to employ a public key

USB boot

Type: **Hardware/OS**

The default option for booting a Raspberry Pi is from a microSD card, but it's possible to boot it from a USB mass storage device – such as a flash drive, hard disk, or SSD – instead. There are different methods for setting up USB boot mode depending on which Raspberry Pi model you're using. See **magpi.cc/bootusb** for details.

Portable power

Type: **Hardware**

If you want to use your Raspberry Pi away from a mains power point, you'll need some form of portable power. The simplest method is to use a standard mobile phone charger or power bank: just connect it via a USB cable to Raspberry Pi's power port. Alternatively, for a smaller footprint, it's possible to use a LiPo battery pack with the addition of a special HAT or SHIM. Some can be combined with mains power as a battery backup for an uninterruptible power supply (UPS) and they may also offer power management options.

Warning!
Battery Safety

Be careful when working with lithium ion batteries. They are highly flammable. Be sure to keep battery projects in a case.

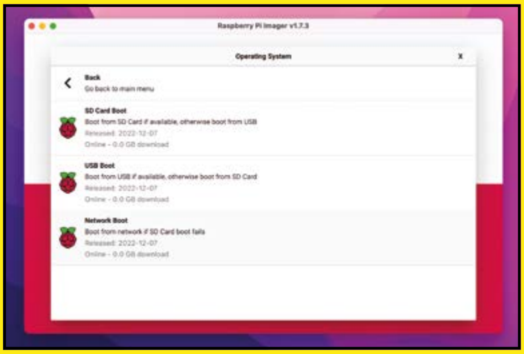
magpi.cc/lithion

Network boot

Type: **Hardware/OS**

It's also possible to boot a Raspberry Pi computer via a network connection, which is useful for overcoming any issues with microSD card corruption. Note that it's best to use an Ethernet connection rather than Wi-Fi, for higher speeds.

There are currently two ways of doing this. The first involves setting up a DHCP/TFTP server – see **magpi.cc/networkboot** for full instructions. The second, Network Install, enables you to download and run the Raspberry Pi Imager tool from a network connection, so you can install an OS directly on Raspberry Pi without needing a second computer – see **magpi.cc/installnet** for details.



Raspberry Pi Imager v1.7.3

Operating System

Back
Go back to main menu

SD Card Boot
Boot from SD Card if available, otherwise boot from USB
Released: 2022-12-07
Online - 0.0 GB download

USB Boot
Boot from USB if available, otherwise boot from SD Card
Released: 2022-12-07
Online - 0.0 GB download

Network Boot
Boot from network if SD Card boot fails
Released: 2022-12-07
Online - 0.0 GB download

Overclocking

Type: **Hardware/OS**

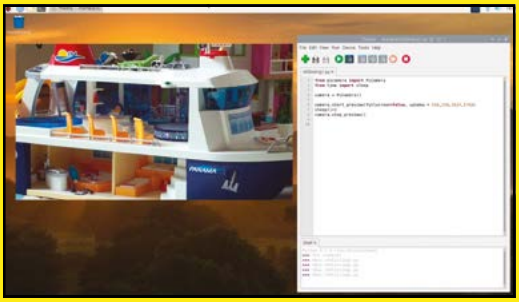
Each Raspberry Pi model is configured to run at a set maximum CPU clock speed by default. For instance, later Raspberry Pi 4 models will run at 1.8GHz. However, it is possible to boost this speed by 'overclocking' the chip. This involves editing the config.txt file in the boot directory – see the 'Edit config.txt' tip (page 52) – and adding lines to set the CPU frequency in MHz (e.g. `arm_freq=2000`), along with `over_voltage` and `gpu_freq`. See **magpi.cc/overclock** for details.

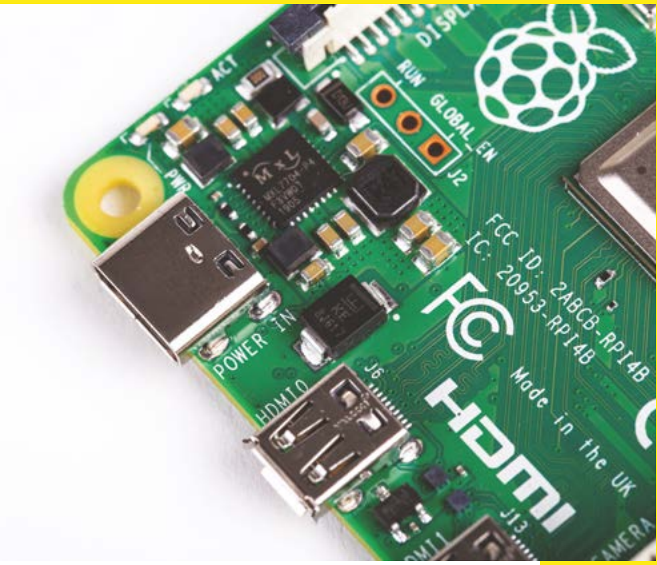
Note that overclocking may cause system instability and reduce the CPU's life span. In addition, to keep the temperature down, you may need to use a cooling solution such as a heatsink, stand, or fan.

Picamera 2

Type: **Camera**

Instead of issuing individual commands in a Terminal, you can write Python programs to control your Raspberry Pi Camera Module or HQ Camera. A new Picamera2 library is currently in beta and usable with the new libcamera-apps. It is now pre-installed in the latest version of Raspberry Pi OS. Go to **magpi.cc/picamera2doc** to read the manual.





Blinking LED

Type: **Hardware**

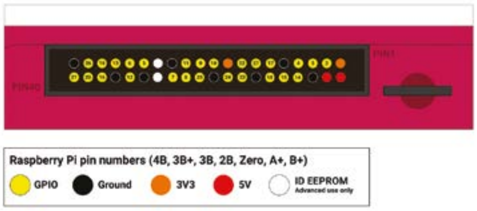
Full-size Raspberry Pi computers have a red power LED and a green 'ACT' LED near the power input. The latter blinks erratically during bootup to indicate everything's working. It also blinks special patterns to indicate errors – see **magpi.cc/actled** for the full list.

> "Each Raspberry Pi model is configured to run at a set maximum CPU clock speed by default"

Raspberry Pi 400 GPIO

Type: **Electronics**

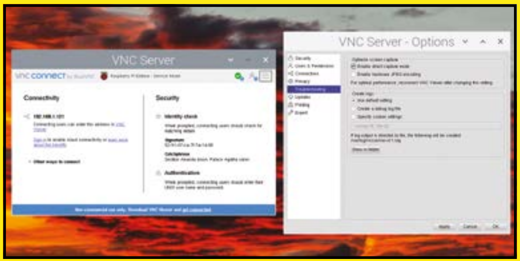
Raspberry Pi 400 has the same 40-pin GPIO header as other Raspberry Pi computers. Using a standard HAT with it isn't ideal, however, due to the header being at the rear of the keyboard unit, which means that a display or LED matrix HAT faces backwards. This can be overcome with a GPIO adapter (such as **magpi.cc/pi400adapter**) or a ribbon cable (**magpi.cc/pi400ribbon**). Make sure you connect it the right way up – physical pin 1 (3V3) is at the top right and pin 40 (GPIO21) at the bottom left. as indicated on the case.



VNC camera preview

Type: **Remote/Camera**

By default, you won't be able to view the camera preview window when accessing your Raspberry Pi remotely from another computer via VNC. However, there is a setting to make the window appear. On Raspberry Pi, click the VNC icon in the taskbar, open the menu (top right icon), go to Options > Troubleshooting, then tick 'Enable direct capture mode' and click OK.



Check camera connection

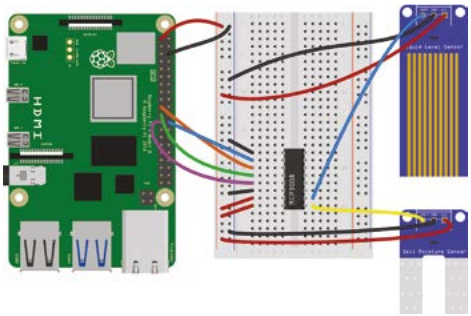
Type: **Camera**

There's an easy way to double-check that your Camera Module or HQ Camera is connected correctly and has been detected by Raspberry Pi. In a Terminal, enter the command `vcgencmd get_camera`. The result should include 'detected=1'. If not, check your camera cable is connected correctly at both ends (Camera Module and Raspberry Pi). If using an older version of Raspberry Pi OS, the camera may not have been enabled – check in Raspberry Pi Configuration or raspi-config.

Using an ADC

Type: **Electronics**

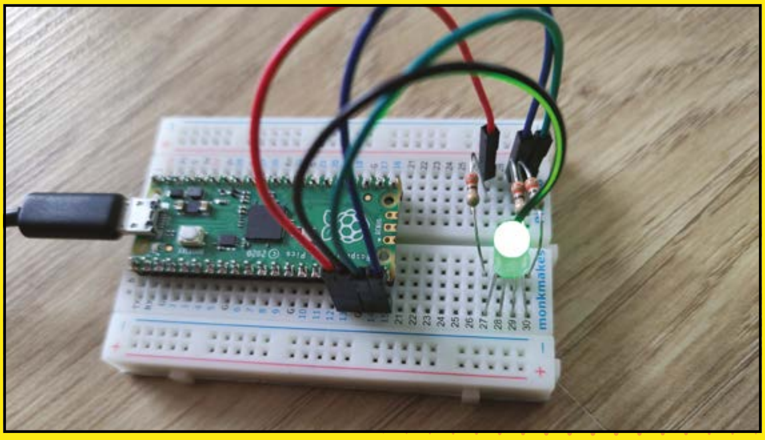
Unlike Pico (see 'Read Pico's analogue inputs' tip, overleaf), Raspberry Pi computer models only allow digital inputs via the GPIO header. If you need to connect a component with an analogue output signal, such as a potentiometer or certain sensors, you will need to do so via an ADC (analogue-to-digital converter) chip such as the MCP3008. Careful wiring is required, including connecting four of the ADC's pins to the SPI pins on Raspberry Pi. See our Plant Monitor tutorial in issue 114 (**magpi.cc/114**) for an example.



Using PWM

Type: **Electronics**

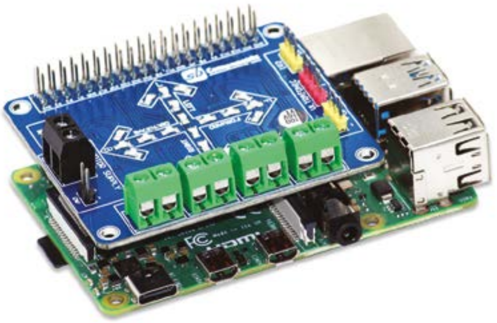
Pulse-width modulation (PWM) is a way of modulating a digital output signal to adjust the speed of a motor or the brightness of an LED. It works by pulsing a GPIO pin output on and off at high frequency, e.g. 50Hz. By altering the 'duty cycle' – the proportion of the time the pin is on versus off – you can alter the average level of voltage output to the component.



Drive motors

Type: **Electronics**

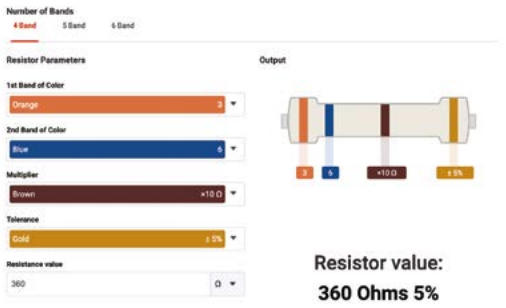
While servos can be controlled from a GPIO pin using PWM, for standard motors you'll need an H-bridge, usually in the form of a breakout board or HAT. The H-bridge enables you to drive a motor in forward or reverse, while PWM is used to determine the speed of the motor. A dual H-bridge enables you to connect two DC motors – or one stepper motor – and control them independently.



Read resistor codes

Type: **Electronics**

The coloured bands around a standard through-hole resistor tell you its rating in ohms (Ω). On a four-band resistor, the first two bands represent digits, the third is a multiplier, and the fourth shows the tolerance level. For instance, orange-orange-brown-gold is 33 × 10 (i.e. 330Ω) with a ±5% tolerance. Five-band resistors have a third digit band and the multiplier on the fourth band. For a handy calculator, go to **magpi.cc/resistorcodes**.



Number of Bands
4 Band | 5 Band | 6 Band

Resistor Parameters

1st Band of Color: Orange 3

2nd Band of Color: Blue 6

Multiplier: Brown ×10 Ω

Tolerance: Gold ± 5%

Resistance value: 360 Ω

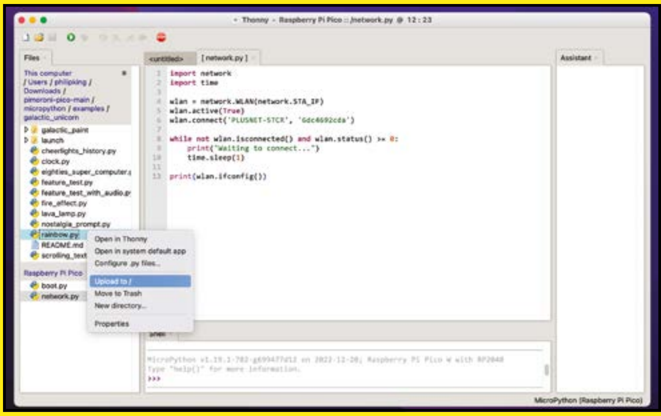
Output

Resistor value:
360 Ohms 5%

Copy files to Pico

Type: **Electronics**

When programming Raspberry Pi Pico with Thonny on a connected computer, you can save a MicroPython file directly onto Pico, but what if you want to copy any other files or folders to it? In the Files pane (select View > Files if not shown), navigate to the file or folder within the computer file structure shown at the top. Then right-click and select 'Upload to /' to copy the file or folder onto Pico.



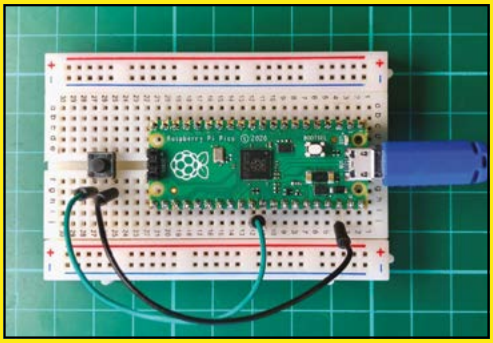
```
import network
import time

wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect('PLUSNET-STCR', '6dc4692cda')

while not wlan.isconnected() and wlan.status() >= 0:
    print("Waiting to connect...")
    time.sleep(1)

print(wlan.ifconfig())
```

```
MicroPython v1.19.1-782-g699477d12 on 2022-12-20; Raspberry Pi Pico W with RP2040
Type "help()" for more information.
>>>
```

Reset Pico

Type: **Pico**

While Pico doesn't have a reset button, it's possible to quickly reboot it by connecting the RUN pin (physical pin 30) to GND. This saves you having to unplug and reconnect the USB cable every time you want to flash a new UF2 file. For more convenience, you can even wire those pins to a push-button. See **magpi.cc/picoreset** for details.

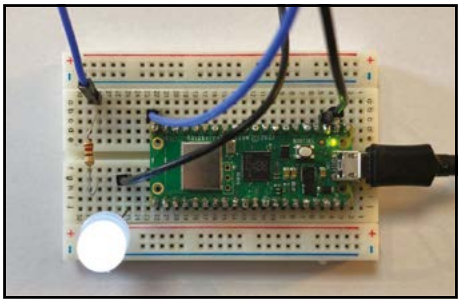


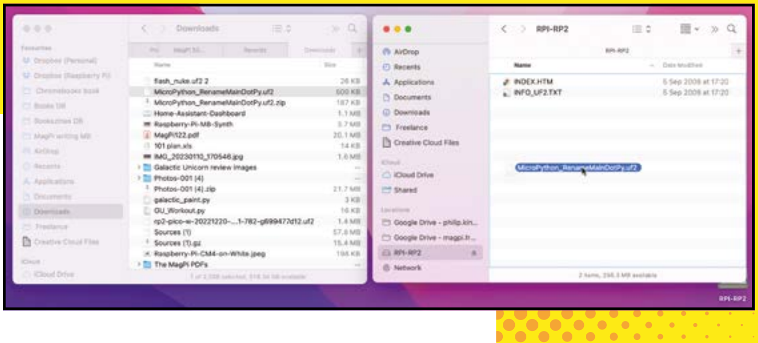
> Pico can accept analogue inputs on some of its GPIO pins

Pico W web server

Type: **Pico**

With Pico W's Wi-Fi connectivity, it's possible to use it to generate a simple web server to access from another device. This can then be used to remotely control components attached to Pico W, such as an LED. See **magpi,cc/picoserver** for more info on how to set it up.





Name	Size
flash_nuke.uf2 2	26 KB
MicroPython_RenameMainDotPy.uf2	600 KB
MicroPython_RenameMainDotPy.uf2.zip	187 KB
Home-Assistant-Dashboard	1.1 MB
Raspberry-Pi-MB-Synth	3.7 MB
MagPi122.pdf	20.1 MB
101 plan.xls	14 KB
IMG_20230110_170546.jpg	1.6 MB
Galactic Unicorn review images	--
Photos-001 (4)	--
Photos-001 (4).zip	21.7 MB
galactic_paint.py	3 KB
GU_Workout.py	16 KB
rp2-pico-w-20221220-...1-782-g699477d12.uf2	1.4 MB
Sources (1)	57.8 MB
Sources (1).gz	15.4 MB
Raspberry-Pi-CM4-on-White.jpeg	195 KB
The MagPi PDFs	--

Name	Date Modified
INDEX.HTM	5 Sep 2008 at 17:20
INFO_UF2.TXT	5 Sep 2008 at 17:20

MicroPython_RenameMainDotPy.uf2

Re-flash locked Pico

Type: **Pico**

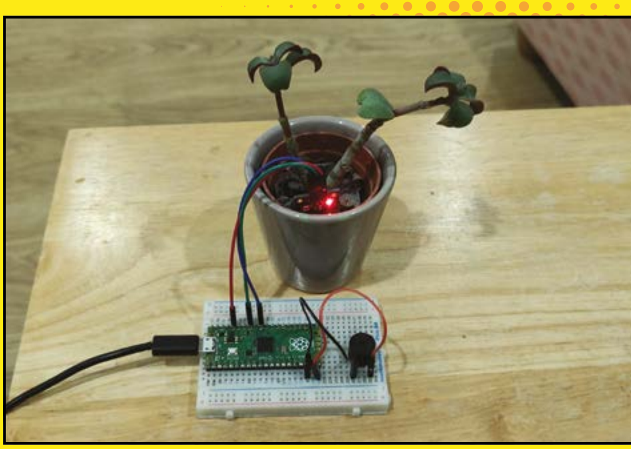
If you mistakenly create an auto-running **main.py** program that blocks REPL, Pico could refuse to communicate with Thonny or other IDEs. In this case, there are two options, both involving flashing a special version of the MicroPython UF2 firmware to Pico. The first, downloaded from **magpi.cc/picomainrename**, simply renames the **main.py** file so it doesn't auto-run. If that doesn't solve the issue, the alternative is to flash Pico with the 'flash_nuke' UF2 file from **magpi.cc/piconuke**, but this will erase all the programs on it!

Read Pico's analogue inputs

Type: **Pico**

Unlike Raspberry Pi computers, Pico can accept analogue inputs on some of its GPIO pins, as they're connected to an on-board ADC. As well as accepting digital inputs, GP26, GP27, and GP28 accept analogue inputs (up to 3.3 V) as ADC0, ADC1, and ADC2. The GND pin between them (on physical pin 33) doubles as an AGND. A fourth analogue channel is wired to Pico's on-board temperature sensor.

To assign an analogue input in MicroPython, you use `machine.ADC` followed by the GPIO pin number in brackets, e.g. `(26)`. To read it, you use the `.read_u16()` suffix instead of the normal `.read` for digital pins; this turns it into an unsigned 16-bit integer between 0 and 65535. For more details, see **magpi.cc/picoanalogue**.



Emulate a BBC Micro on Raspberry Pi 400

With an optimised version of the B-Em emulator, Raspberry Pi 400 can take us back to the heyday of the British micro that inspired it

MAKER

K.G. Orphanides

KG is a writer, developer and software preservationist.

magpi.cc/hauntedowlbear

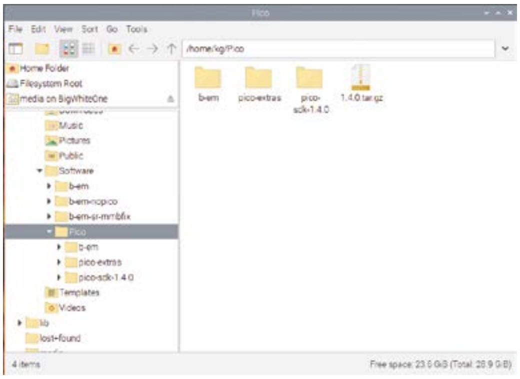
You'll Need

- Pico SDK **magpi.cc/picosdk**
- Pico Extras **magpi.cc/picoextras**
- Kilograham's B-Em fork **magpi.cc/bem**

From schools to bedrooms, Acorn Computers' BBC Micro helped to kick-start a revolution in software and hardware development. Its influence is imprinted on products ranging from the ubiquitous Arm processors to modern gaming blockbusters like Elite Dangerous.

But emulating the BBC on modest hardware has traditionally been a chore. There are several excellent BBC emulators, and you can run most of them on Raspberry Pi, but only at the cost of major performance issues.

That's no longer the case, thanks to Raspberry Pi Pico SDK architect Graham 'kilograham' Sanderson who's forked an optimised version of the B-Em emulator that'll run smoothly on both the Raspberry Pi Pico microcontroller and Raspberry Pi SBCs.



Make sure that the Pico SDK, Pico Extras, and B-Em source code folders are all in the same directory before you compile

01 Check your graphics settings

When building this B-Em fork for Raspberry Pi, you need to make sure that the compositor is disabled and that you're not using the legacy video driver. If you're running a fresh Raspberry Pi OS Bullseye install on Raspberry Pi 400, neither of these should be the case, but it's worth checking.

To do so, open a Terminal window and type:

```
sudo raspi-config
```

Select '6 Advanced', then 'A3 Compositor'. Make sure this isn't enabled. If your hardware and OS versions support the legacy video driver, you'll also find this in the advanced menu. Similarly, make sure it's disabled. Reboot.

02 Gather your dependencies

At the Terminal type:

```
 sudo apt install build-essential cmake
libdrm-dev libx11-xcb-dev libxcb-dri3-dev
libepoxy-dev ruby libasound2-dev
```

Although we're building B-Em for Raspberry Pi, we'll still need the Pico SDK. The Pico SDK, Pico Extras, and B-Em folders all need to be in the same directory. We'll make a directory called **Pico** in our home root to put everything in, then put all the software we need in it.

You should substitute the wget URL in the next step with the current release version of the Pico SDK (**magpi.cc/picosdk**).

03 Build B-Em

Enter the following Terminal commands to download and make the software:

```
 cd
 mkdir Pico
 wget https://github.com/raspberrypi/pico-
sdk/archive/refs/tags/1.4.0.tar.gz
 tar -xzf 1.4.0.tar.gz
 git clone https://github.com/raspberrypi/
pico-extras.git
 git clone https://github.com/kilograham/b-
em.git
 cd b-em
 mkdir pi_build
 cd pi_build
 cmake -DPICO_SDK_PATH=path/to/pico-sdk -DPI_
BUILD=1 -DPICO_PLATFORM=host -DDRM_PRIME=1
-DX_GUI=1 ..
 make -j4
```

Following these instructions, the path to Pico SDK will be something like: **~/Pico/pico-sdk-1.4.0**.

04 Is this thing on?

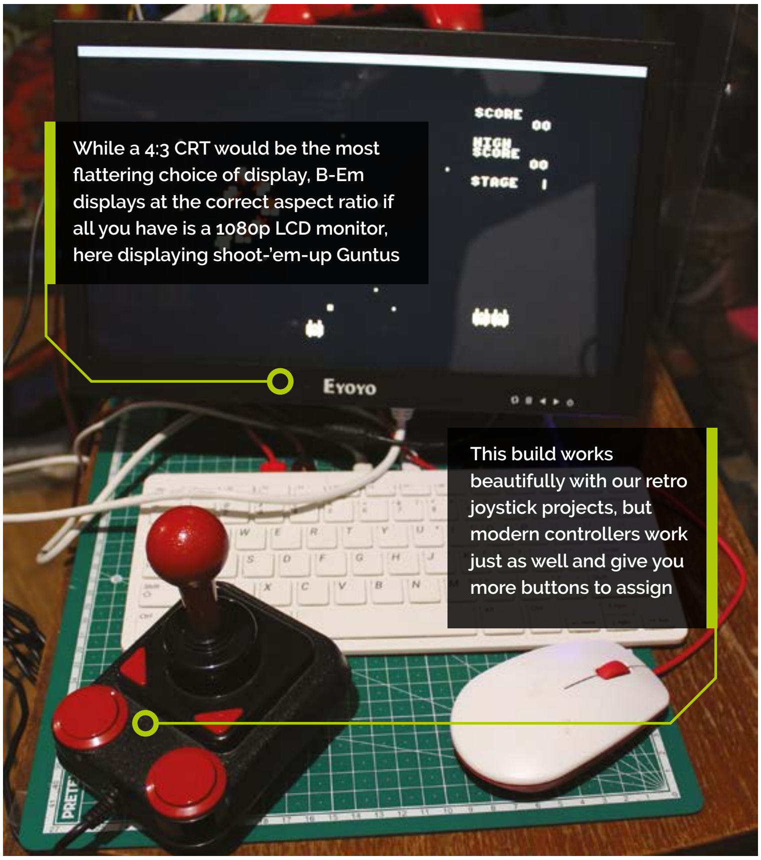
B-Em emulators for the BBC Micro (xbeeb) and Master (xmaster) will be built two directories below the **pi-build** directory we made earlier, in **src/pico**. Assuming you still have the Terminal open from the previous step:

```
 cd src/pico
 ./xbeeb
```

You should find yourself looking at an emulation of a 32K BBC Micro running Acorn's 1770 DFS (disk filing system).

05 Would you like to play a game?

Unlike other BBC emulators, including the fully loaded versions of B-Em, there's no graphical interface to browse and load disk image files. Fortunately, B-Em for Pico loads a copy of Eben Upton's port of puzzle game 2048 by default. Not all the keys and characters on Raspberry Pi 400 map tidily to those of a BBC Micro – see **Figure 1** overleaf. In B-Em, type:



While a 4:3 CRT would be the most flattering choice of display, B-Em displays at the correct aspect ratio if all you have is a 1080p LCD monitor, here displaying shoot-'em-up Guntus

This build works beautifully with our retro joystick projects, but modern controllers work just as well and give you more buttons to assign

```
 *.
```

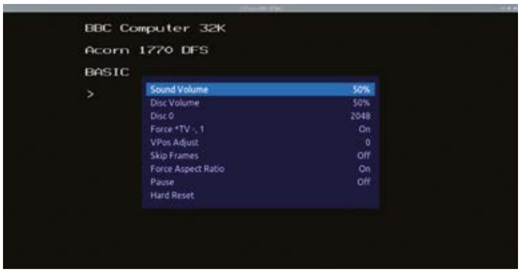
...to list the files on the virtual disk, then:

```
 *EXEC !BOOT
```

...to run the program.

06 Key combinations and commands

In this version of B-Em, **F11** opens the menu, navigable using the mouse or arrow keys. Press **ENTER** to select an option and, if it flashes



```
BBC Computer 32K

Acorn 1770 DFS

BASIC

>
```

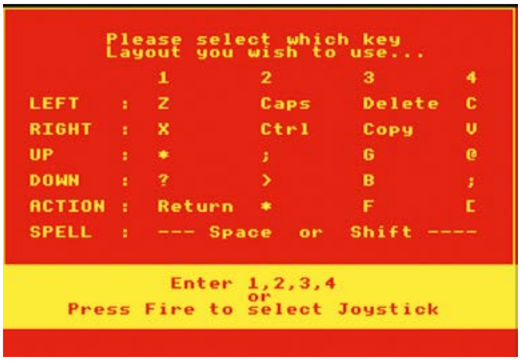
Sound Volume	50%
Disc Volume	50%
Disc 0	2048
Force *TV -, 1	On
VPos Adjust	0
Skip Frames	Off
Force Aspect Ratio	On
Pause	Off
Hard Reset	

This version of B-Em has a stripped-down **F11** menu but it includes critical sound and video options, plus embedded disc image handling

Top Tip

Build for Pico!

After this tutorial, you'll also be set to build B-Em for Raspberry Pi Pico. You'll need a VGA board to handle graphics output.



```
    Please select which key
    Layout you wish to use...
              1          2         3      4
LEFT    :   Z          Caps      Delete   C
RIGHT   :   X          Ctrl      Copy     V
UP      :   *          ;         G        @
DOWN    :   ?          >         B        ;
ACTION  :   Return     *         F        [
SPELL   :   --- Space   or   Shift ----

          Enter 1,2,3,4
               or
    Press Fire to select Joystick
```

There weren't many consistent standards for game or cursor control layouts in the early 1980s, leading to some creative approaches, demonstrated here by Citadel

after selection, to confirm it. Press **ESC** to exit the menu or opt out of a flashing selection. The pared-down menu gives you sound and graphics settings, access to embedded disks, and a hard reset. To exit B-EM, close the window or press **ALT**+**F4** to terminate the program.

We've already seen that `*.` lists the files on a disk and `EXEC` executes a program. You can find a helpful lookup sheet for other commands and for BBC BASIC in general at **magpi.cc/bbcbasic**. An HTML version of the BBC Micro User Guide exists at **magpi.cc/bbcuserguide**.

07 Loading disk images

Although there's no disk image browser, you can launch B-Em to run a specific disc image. The .SSD and .DSD formats are supported, these are the most commonly used to distribute modern BBC games.

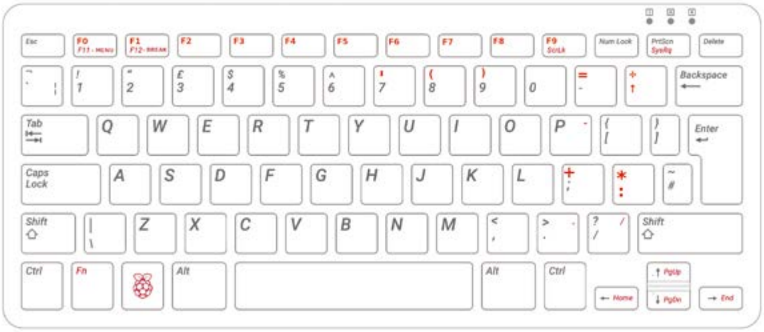
Download Infuto's Guntus shoot-'em-up from **magpi.cc/guntus** (direct download) and save it to your **Downloads** folder.

To load it, the command with full paths should be:

```
~/Pico/b-em/pi_build/src/pico/xbeeb
~/Downloads/Disc172-Guntus.ssd
```

We'll make some shortcuts to tidy that up later. In B-Em list the files by typing `*.`, then load Guntus with:

Figure 1 BBC Micro keyboard map for the UK Raspberry Pi keyboard. BBC keys are marked in red


Esc F0 F11 - MENU F1 F12 - BREAK F2 F3 F4 F5 F6 F7 F8 F9 ScrLk Num Lock PrtScn SysRq Delete
¬ ` ¦ ! 1 " 2 £ 3 $ 4 % 5 ^ 6 ' 7 (8) 9 0 = - + ↑ Backspace
Tab Q W E R T Y U I O P - { [}] Enter
Caps Lock A S D F G H J K L + ; * : ~ #
Shift | \ Z X C V B N M < , > . ? / / Shift
Ctrl Fn Alt Alt Ctrl ← Home ↑ PgUp ↓ PgDn → End

```
CHAIN"GUNT-IN"
```

We could also have used `*EXEC !BOOT`, as a boot file is present.

08 Embedding disks

You can also embed disk images at build time. By default, B-Em boots with two embedded disks: 2048 and the iconic BBC Welcome disk. You can switch between them in the **F11** menu, by going to Disc 0, using the left and right arrow keys to navigate to the option you want, then pressing **ENTER**.

In our **~/Pico/b-em/src/pico/discs/** directory, you'll find a file called **beeb_discs.txt**, alongside several disk images. Add new embedded disks by inserting a line at the bottom formatted thus:

```
Your Label For The Disk = /path/to/disk/
image.dsd
```

Alternatively, you can create a file in the same directory, called **beeb_discs_user.txt**, with entries formatted in the same way. This will automatically supersede **beeb_discs.txt**. Labels can have spaces; paths and file names cannot.

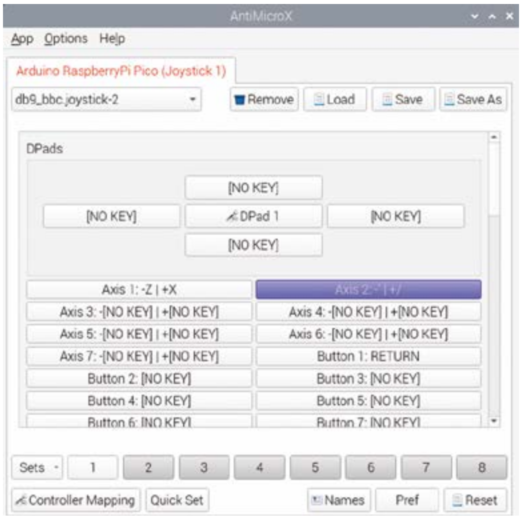
Once you've added your list of files to embed, repeat the final `make -j4` command in the **pi_build** directory from Step 3.

09 Sticks of joy

Joystick support isn't built into this version of B-Em, but you can use mapping software to work around this. AntiMicroX can map any recognised joystick, from a Sony DualShock 4 to the Pi Pico-powered DB9 joystick adapter we made in issue 125, or our older GPIO joystick adapter (**magpi.cc/db9joystick**). At the Terminal:

```
sudo apt install antimicro
antimicrox
```

A popular control schema for games used **Z** to move left, **X** to move right, **:** for up, and **/** for down, with **SPACE** and **RETURN** configured as jump and/or action keys. If you only have one joystick button, assign it to whichever you need most often. AntiMicroX supports multiple control profiles, so it's easy to switch between the most common.



Axis 1: -Z | +X
Axis 2: -' | +/
Axis 3: -[NO KEY] | +[NO KEY]
Axis 4: -[NO KEY] | +[NO KEY]
Axis 5: -[NO KEY] | +[NO KEY]
Axis 6: -[NO KEY] | +[NO KEY]
Axis 7: -[NO KEY] | +[NO KEY]
Button 1: RETURN
Button 2: [NO KEY]
Button 3: [NO KEY]
Button 4: [NO KEY]
Button 5: [NO KEY]
Button 6: [NO KEY]
Button 7: [NO KEY]

Mapping a controller with AntiMicroX is a convenient way of both adding retro joystick support to this version of B-Em, and replacing old-school keyboard layouts with something more familiar

10 A smoother startup

Let's add some symlinks and shortcuts

Open a Terminal and type:

```
  sudo ln -s ~/Pico/b-em/pi_build/src/pico/
xbeeb /usr/bin/local
```

You can now launch xbeeb by name at the Terminal, no path required. You'll still need to provide paths to disc files you wish to launch, though.

We'll also make a desktop launcher. Create a text file called **b-em.desktop** on the desktop, or add it to Raspberry Pi OS's main menu by placing it in **~/.local/share/applications**. See the code listing for the full launcher file. If you want to launch from the desktop, embed the disks you use regularly, as there's no file browser. To ensure that .desktop files are executed by default, in the file manager go to Edit > Preferences and tick 'don't ask options on launch executable file'.

11 Getting new software

There's a vibrant homebrew scene for the BBC Micro. The non-commercial section (**magpi.cc/bbcnoncomm**) of the Complete BBC Micro Games Archive is home to 29 games released in 2022 alone, plus hundreds of older freeware and public domain titles ranging from the 1980s to the present. New releases are usually announced on the forums of mainline B-Em and BeebEm maintainer group Stardot: **magpi.cc/bbcstardot**.

b-em.desktop

Language: **DESKTOP**

```
[Desktop Entry]
Encoding=UTF-8
Version=1.0
Type=Application
Terminal=false
Name=B-Em
Comment=BBC Micro emulator
Exec=~/Pico/b-em/pi_build/src/pico/xbeeb
Icon=~/Pico/b-em/icon/b-em.png
```

> "There's a vibrant homebrew scene for the BBC Micro"

12 Learn to program like it's 1985

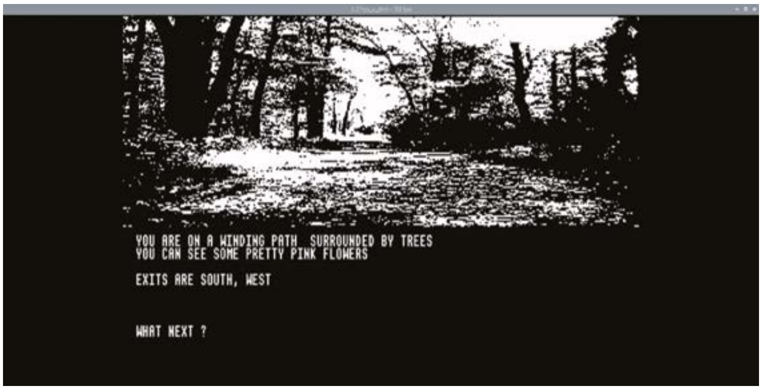
Many of Usborne Books' classic 1980s guides to computer programming are now available for free at **magpi.cc/usborne1980s**. BBC Micro projects feature heavily.

There are even modern publications that review and analyse software for 8-bit platforms including the BBC Micro, such as *The Classic Adventurer* (**classicadventurer.co.uk**), while historians and enthusiasts have archived contemporary magazines and fanzines at sites including the Educational Software Archive for the BBC Micro (**magpi.cc/bbceduarchive**), 8BS (**magpi.cc/8bsmag**), and The Internet Archive (**magpi.cc/microuserarchive**).

Top Tip

Now I am the Master

B-Em includes a BBC Master emulator, which you'll find in the same directory, with an executable file named **xmaster**.



```
YOU ARE ON A WINDING PATH  SURROUNDED BY TREES
YOU CAN SEE SOME PRETTY PINK FLOWERS

EXITS ARE SOUTH, WEST


WHAT NEXT ?
```

Modern BBC Micro games like Brian Tierney's Blue Wizard are able to use impressive graphical techniques not readily available to BBC devs in the early 1980s

Part 01

Beginning electronics with a breadboard and an LED

MAKER

Stewart Watkiss

Also known as Penguin Tutor, Stewart is a maker and YouTuber who loves all things Raspberry Pi and Pico. Author of *Learn Electronics with Raspberry Pi*.

twitter.com/stewartwatkiss

Get started with electronic projects using simple components. Find out what makes the electrons flow using a LED light and switch

In this project, you will learn how to create circuits on a breadboard, along with basic information you need to know when starting out with electronics. Find out how to work out which way the LED goes and follow the easy guide to find the right-value resistor. Add the switch and you have the basics of a first circuit.

01 Using a breadboard

A prototyping breadboard is a way of creating electronic circuits by pushing components into the board. Using a breadboard is a great way to learn about electronics. You can use it to create a circuit, swap out some components, and experiment with different values. When you've finished, you pull the components out and they can be reused for your next project. They are designed so that integrated circuits can be plugged into the centre of the board, and you can connect components to the various pins. Most include power rails that run horizontally across the top and bottom, and vertical rails with five holes each – see **Figure 1**.

02 Wire up an LED circuit

You can start by creating a simple circuit using a battery, LED, and resistor. You'll also need some jumper wires, or solid core wire to create the complete circuit. Avoid stranded wire as that is difficult to insert and the individual strands can break off. The wires for battery packs normally have stranded wires, but hopefully they are tinned, which makes them easy to use. If not, then you may need to push them in using one of the jumper wires.

The first circuit is shown in **Figure 2**.

03 Testing the first circuit

Has the LED lit up? If so great. If not, don't worry: it may be that the LED is the wrong way around. We didn't mention it before, but the LED does need to be connected in a certain direction for it to work. Using a breadboard means that it's easy to experiment: just unplug the LED and insert it with the leads the opposite way around. If the LED still doesn't light, then look for a switch on the battery connector or recheck the wiring.

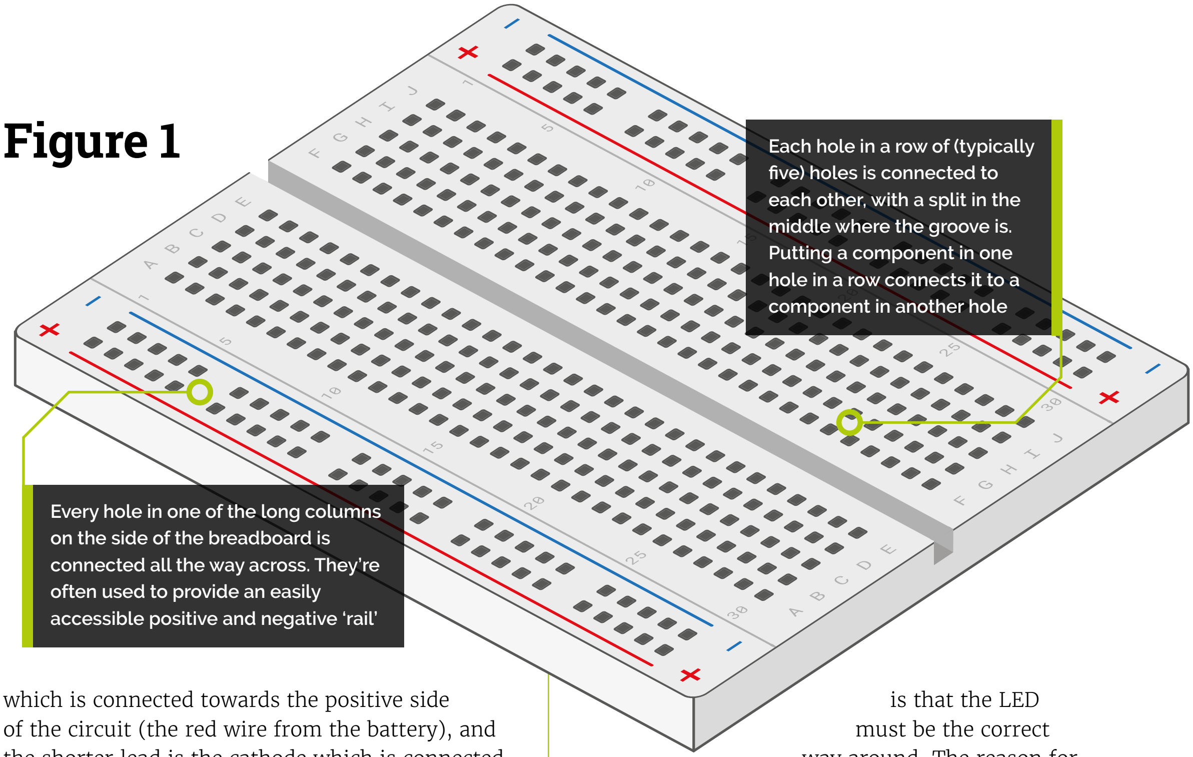
04 LED direction

When using a breadboard, it's easy to swap components around, but it's not so easy when you create a permanent circuit. There are a few ways to tell which way around an LED needs to go, the easiest (but less reliable) is to look at the length of the leads. The longer lead is normally the anode

You'll Need

- Breadboard **magpi.cc/breadboardhalf**
- Battery holder **magpi.cc/batteryholder**
- LED **magpi.cc/diffusedleds**
- 220Ω resistors **magpi.cc/220resistors**
- Jumper wires **magpi.cc/mmjumpers**
- PCB switch **magpi.cc/tactileswitch**

Figure 1



Each hole in a row of (typically five) holes is connected to each other, with a split in the middle where the groove is. Putting a component in one hole in a row connects it to a component in another hole

Every hole in one of the long columns on the side of the breadboard is connected all the way across. They're often used to provide an easily accessible positive and negative 'rail'

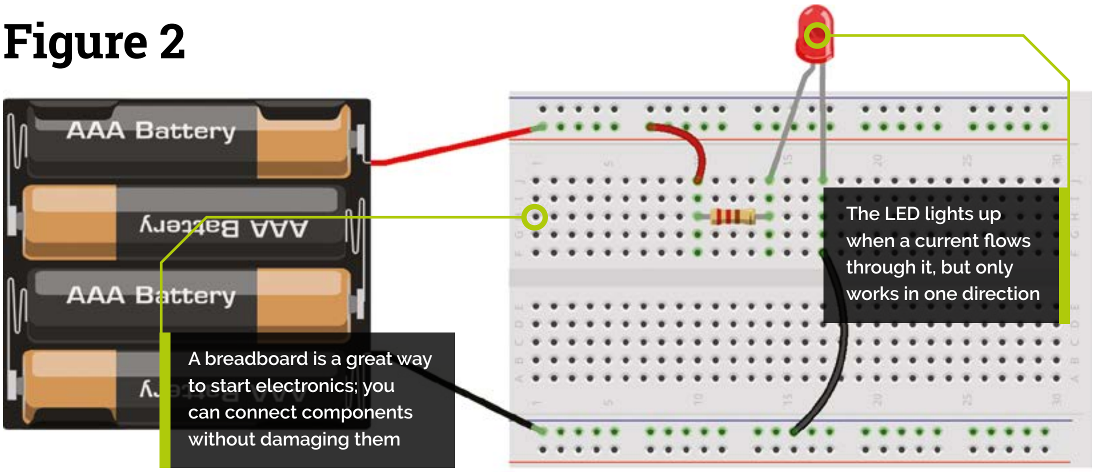
which is connected towards the positive side of the circuit (the red wire from the battery), and the shorter lead is the cathode which is connected towards the negative side of the circuit (the black wire from the battery). A more reliable way of determining the anode and cathode is to look for a flat area on the body of the LED; the flat area normally denotes the cathode (negative) side of the LED.

05 How the circuit works

For an electronic circuit to work, there need to be connections going from the positive terminal of the battery, through the components, and then back to the negative terminal of the battery. This completes a full circuit. The additional factor here is that the LED must be the correct way around. The reason for this is that it is a semiconductor device. Semiconductors have special properties that allow them to work as a conductor or an insulator when certain conditions are met. In this case, the LED is a type of diode which acts like a one-way valve. It acts as a conductor when

> “Without the resistor, the LED will allow too much current to flow and burn itself out”

Figure 2



The LED lights up when a current flows through it, but only works in one direction

A breadboard is a great way to start electronics; you can connect components without damaging them

Top Tip

Datasheets are your friend

Datasheets are available for most electronic components. They will provide the information you need to use them in your circuits.

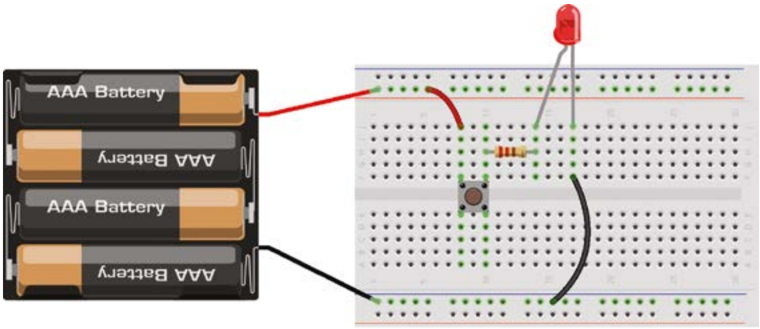


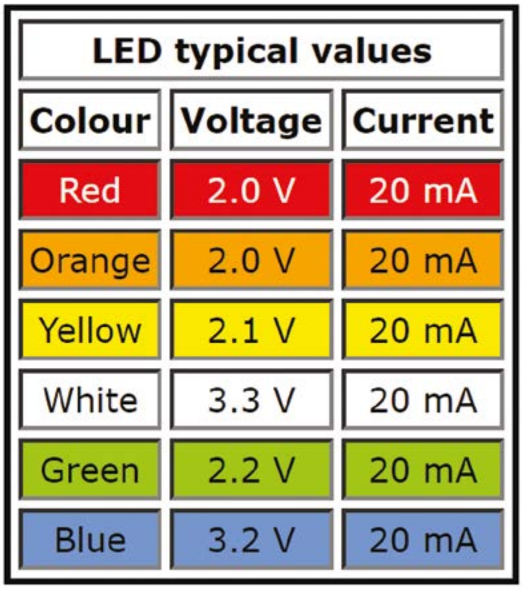
Figure 7 The updated circuit includes a switch to turn the circuit on and off

in the forward direction and an insulator when in the reverse direction.

06 The resistor

The other component included in the circuit is a resistor. As its name implies, this resists the current around the circuit. This is needed to protect the LED; without the resistor, the LED will allow too much current to flow and burn itself out.

The resistor used in this circuit is 220 Ω (ohms), but why do we use this value? To understand how to choose resistors, you need to understand some basic electrical circuit theory. This may sound a bit daunting, but you only need to understand the very basics.



LED typical values		
Colour	Voltage	Current
Red	2.0 V	20 mA
Orange	2.0 V	20 mA
Yellow	2.1 V	20 mA
White	3.3 V	20 mA
Green	2.2 V	20 mA
Blue	3.2 V	20 mA

Figure 4 Typical LED values for voltage and current. Check datasheets for non-standard LEDs

07 Voltage, current, and resistance

The three fundamental measurements of an electrical circuit are voltage, current, and resistance.

Voltage is the amount of electric pressure. Voltage is measured in volts, which uses the symbol V.

Current is the flow of electric charge going through a conductor. It is measured in amperes (normally referred to as amps), which has the symbol A. When used in electronics, the current is normally very small, so is often shown in milliamps (mA) which are thousandths of an amp.

Resistance is how much a material opposes the flow of electric current. It is measured in ohms, denoted by the Greek omega character, Ω. You may also see values in thousands of ohms, known as kiloohms (kΩ), or even millions of ohms, known as megaohms (MΩ).

08 Ohm's law

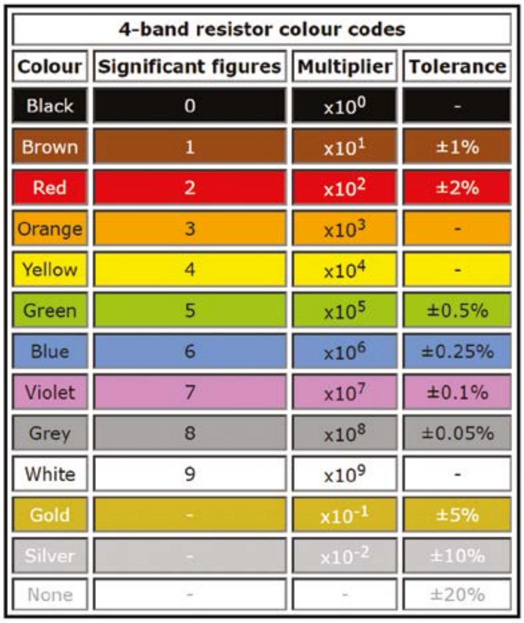
The voltage, current, and resistance are all related to each other. This is all described in Ohm's law. The easiest way to understand this is by using Ohm's law triangle (see **Figure 3**). This has V for the voltage, I for the current, and R for resistance. Cover the letter for the value you want to find, and you have the formula. If you know the current (I) and the resistance (R) but need to know the voltage (V), then cover up the V and it leaves you with I × R. If you want to know the resistor value needed for a specific voltage and current, then cover the letter R and you have V ÷ I.

09 Calculating the resistor voltage and current

To calculate the resistance required, we first need to know the voltage across the resistor. This will be the voltage of the battery, minus the voltage dropped across the LED. The voltage across the LED is specific to the type of LED used, but you can use approximate values based on the typical values.

We are using four AA batteries. These are normally 1.5 V per battery, providing a total of 6 V. We're using a red LED based on the typical values, which is 2 V. This gives 6 – 2, which is 4 V across the resistor.

The current that the LED requires depends upon several factors, including type, size, and brightness.



4-band resistor colour codes			
Colour	Significant figures	Multiplier	Tolerance
Black	0	x10^{0}	-
Brown	1	x10^{1}	±1%
Red	2	x10^{2}	±2%
Orange	3	x10^{3}	-
Yellow	4	x10^{4}	-
Green	5	x10^{5}	±0.5%
Blue	6	x10^{6}	±0.25%
Violet	7	x10^{7}	±0.1%
Grey	8	x10^{8}	±0.05%
White	9	x10^{9}	-
Gold	-	x10^{-1}	±5%
Silver	-	x10^{-2}	±10%
None	-	-	±20%

Figure 5 The resistor colour table. For four-band resistors, there are two significant figures, a multiplier, and a tolerance band

We have used 20 mA across the table (**Figure 4**), which is a realistic value for a standard 5 mm LED.

10 Calculating the resistor value

We now have the voltage across the resistor (4 V) and current (20 mA), so we can use that to calculate the required resistance. Using the formula R = V ÷ I, we calculate 4 / 0.02 = 200 Ω.

You may notice that earlier we specified a 220 Ω resistor. This was not a mistake. You can indeed use a 200 Ω resistor, but they are less commonly available.

> The current that the LED requires depends upon several factors, including type, size, and brightness

Resistors are available in different values, but it would be impossible to have resistors for every calculated value. There are different series of values known as E-series. Your author normally uses E12 series, which provides a reasonable selection of values, with the nearest values being 180 Ω or 220 Ω. We've chosen the higher value, as that is safer than risking too high a current. The value we used is 220 Ω.

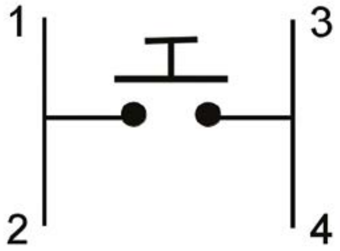


Figure 6 Many switches have four pins. The pins on each side are connected together, with the switch going between them

11 How to recognise the correct resistor

If you look at the resistors, they don't include the resistance value written on them. Instead, they normally use bands of different colours. A resistor will normally have four or five bands of colour. The four-band variant is the most common, so we will explain that. There is usually a bigger gap between the third and fourth bands so you know in which order to look at the resistor.

The first and second bands denote the first and second significant figure of the value, followed by the third band which is a multiplier (effectively the number of zeroes after the first and second values). For our 220 Ω resistor, it will be red (2), red (2), followed by brown (×10). The fourth band is the tolerance, which for E12 resistors is often gold (5%). See **Figure 5**.

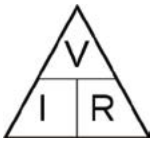
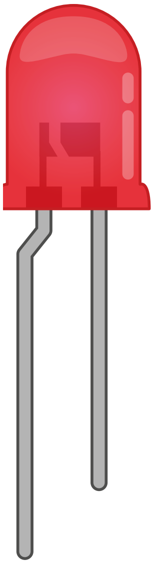


Figure 3 The Ohm's law triangle, makes it easy to find the formula for calculating resistance, voltage, or current

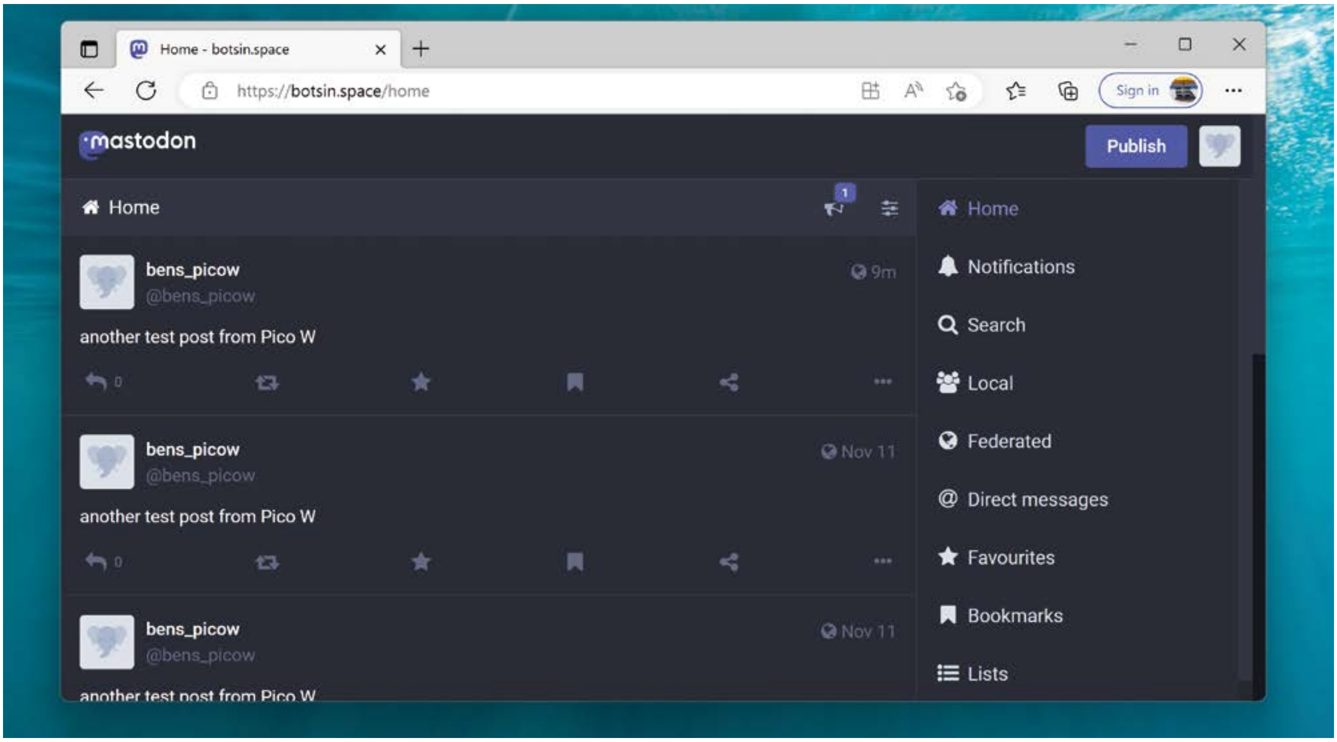
12 Add a switch

At the moment, the circuit stays on all the time. You can add a switch to the circuit so that it only comes on when this is pressed. This requires a push-to-make switch, which is one which completes the circuit when the switch is pressed.

These switches normally have four pins, but these are wired in two pairs, as shown in **Figure 6**. When you insert these into the breadboard, the pins on the left are connected, as are those on the right. Just remove the jumper wire going to the resistor with a wire from the supply to one side and connect the other side of the switch to the resistor (**Figure 7**).



The LED needs to be connected the correct way around. Look for the longer lead (anode) or the flat area (cathode)



bens_picow
@bens_picow
9m
another test post from Pico W

bens_picow
@bens_picow
Nov 11
another test post from Pico W

bens_picow
@bens_picow
Nov 11
another test post from Pico W

Pico W with CircuitPython

Getting online with a microcontroller

Ben Everard

@ben_everard

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

Above The Mastodon social network has an API that's easy to connect to and use from embedded devices

Raspberry Pi Pico W microcontroller has just got support for CircuitPython. Let's take a really quick look around.

For the most part, using Pico W is exactly the same as using Pico. The pinout is the same, and you'll have exactly the same features available. However, you also have added network connectivity.

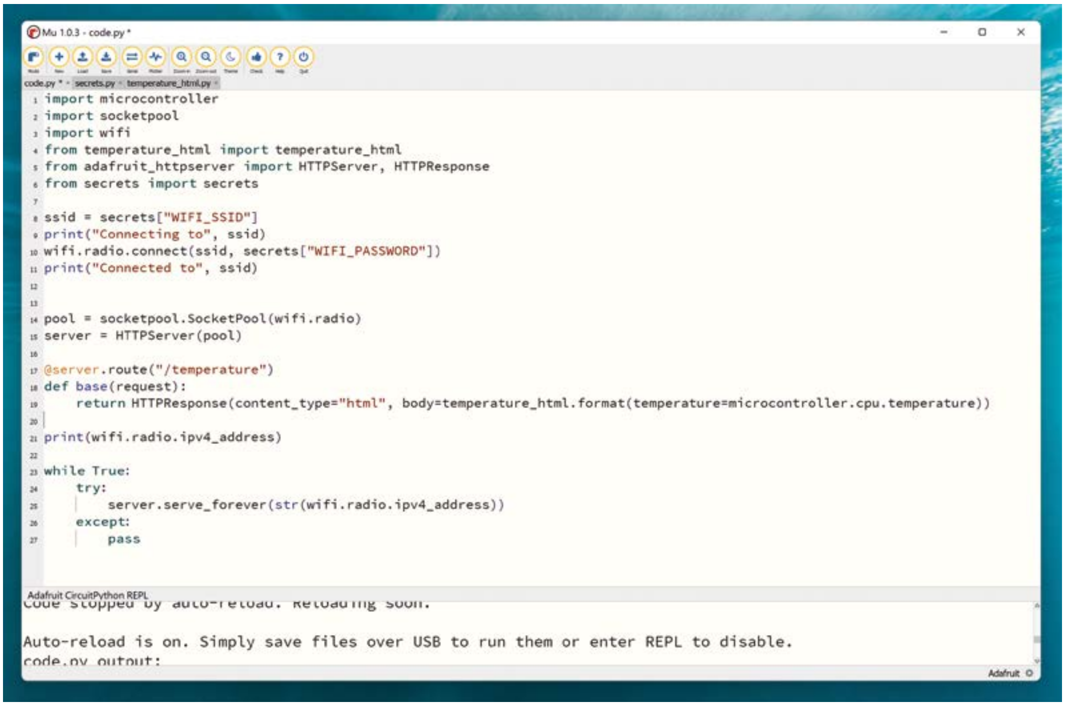
First, you need to get the software. You can download a UF2 file of CircuitPython for Pico W from **hsmag.cc/PicoWCircuitP**. At the time of writing, the latest version was 8.0.0-beta.4. There wasn't yet a stable version for Pico W. This should be fine for most cases, but obviously, a beta version might have bugs in it. The stable version should be available shortly.

In order to get that onto Pico W, you need to hold down the BOOTSEL button, then plug Pico W into your computer (unplug it first, then plug it in). You should see a new removable drive appear, and you can drag and drop the UF2 file onto this. It should disappear and a new CircuitPython drive should pop up.

If you have not used CircuitPython before, then there is a good 'getting started' guide at **hsmag.cc/AdafruitCircuitP**.

For our first experiment, let's try interacting with the social media network du jour, Mastodon. You'll need a Mastodon account for this. It can be a personal account, or you can set one up just for your bot. In fact, there's a Mastodon instance set up specifically for bots – **botsin.space**. Be sure to read the rules so you can properly write your request if you want an account on that instance. If you're looking to set up a personal account, you'll need to pick an instance to join. Head to **joinmastodon.org** to look at the options. Don't worry too much, as you will be able to move your account to a different instance later if you want, and you can interact with people on any instance.

Once you've got an account, you can get an API key. In your edit profile page, go to Development, then click on New Application and give it a name, then


Mu 1.0.3 - code.py *
code.py * secrets.py temperature_html.py
Adafruit CircuitPython REPL
Adafruit

```
import microcontroller
import socketpool
import wifi
from temperature_html import temperature_html
from adafruit_httpserver import HTTPServer, HTTPResponse
from secrets import secrets

ssid = secrets["WIFI_SSID"]
print("Connecting to", ssid)
wifi.radio.connect(ssid, secrets["WIFI_PASSWORD"])
print("Connected to", ssid)


pool = socketpool.SocketPool(wifi.radio)
server = HTTPServer(pool)

@server.route("/temperature")
def base(request):
    return HTTPResponse(content_type="html", body=temperature_html.format(temperature=microcontroller.cpu.temperature))

print(wifi.radio.ipv4_address)

while True:
    try:
        server.serve_forever(str(wifi.radio.ipv4_address))
    except:
        pass
```

```
Code stopped by auto-reload. Reloading soon.

Auto-reload is on. Simply save files over USB to run them or enter REPL to disable.
code.py output:
```

Left You can build an HTTP server in under 30 lines of CircuitPython, so it's an easy way of adding an interface to your project

press Submit. You'll get redirected to the application list page. Click on the name of your new app, and this should take you to the details page where you'll see the access token for this app. You'll need this shortly.

Posts on Mastodon are short bits of text. Exactly how short depends on your instance, but 500 characters is pretty common. They can also have images or video attached to them, but we won't worry about that at the moment.

Let's look at the code to create a post on Mastodon.

GETTING DATA

In the first example here, we used a POST request to send data to an external server. In the second example, we set up a server and got data from it. You might be wondering if it's possible to combine the two and use POST requests to send data to Pico W. The answer is: yes, you can.

We don't have space to go through it all here, but Liz Clark has written a guide to this for Adafruit – you can see it at **hsmag.cc/PicoWHTTPServer**.

```
import microcontroller
import socketpool
import wifi
import adafruit_requests as requests
import ssl
from secrets import secrets

ssid = secrets["WIFI_SSID"]
print("Connecting to", ssid)
wifi.radio.connect(ssid, secrets["WIFI_PASSWORD"])
print("Connected to", ssid)


pool = socketpool.SocketPool(wifi.radio)
requests = requests.Session(pool, ssl.create_
default_context())

response = requests.post("https://botsin.space/
api/v1/statuses",
                    headers={"Authorization" :
secrets["mastodon_access_token"]},
                    data={"status":"another
test post from Pico W"})
```

As you can see, this uses the Adafruit requests module that can be used much like Python 3's requests module. You just need to make sure that you copy it over the **adafruit_requests** folder from the CircuitPython Library Bundle (which you can download from **circuitpython.org**).

You will also need to create a **secrets.py** file. This will contain all the things that you don't want to be public, including your Wi-Fi password and Mastodon key. →

ESPHOME

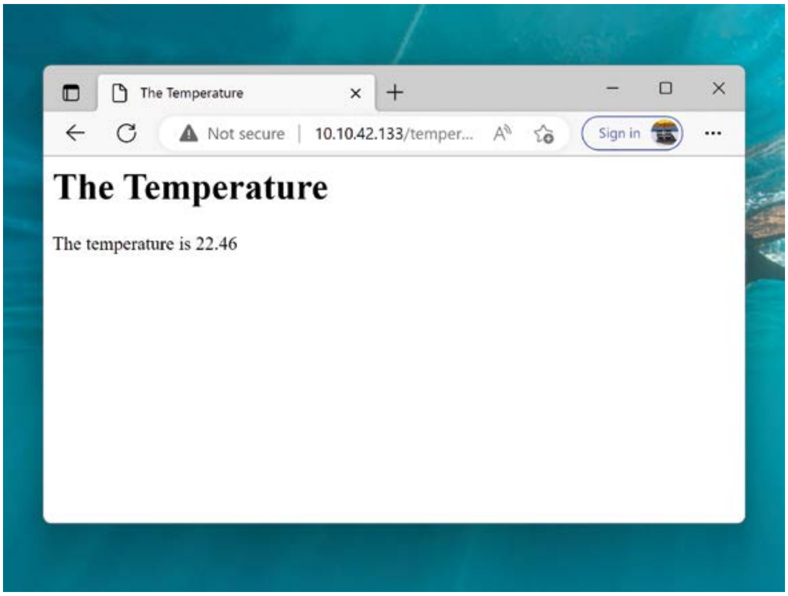
ESPHome is a platform for building and interconnecting DIY Internet of Things devices. As the name suggests, it was originally built for ESP8266- and ESP32-based devices. However, there's now support for Pico W. This gives you a really easy low-code way of linking up sensors, actuators, and other 'thing' devices. You can create configurations in the YAML markup-language and deploy these to devices. Using this, you can very quickly make a network of interconnected things using commonly available maker components and control them from the Home Assistant smart home software.

The format should be:

```
secrets = {"WIFI_SSID" : "<your ssid>",
        "WIFI_PASSWORD" : "<your password>",
     "mastodon_api_key" : "Bearer <your token>"
}
```

In most of these cases, you just need to slot in the appropriate bit of secret information. The Mastodon token is a bit different just because you need to include the word 'Bearer' before the token.

Once you've set this up, you can save the above code as **code.py** on your Pico W, and it will connect to your Wi-Fi network, post the message to Mastodon, and then finish. This is probably a little dull by itself, but you can use it to log information, send messages, or just generally to scream into the void.



The Temperature

The temperature is 22.46

Below OK, this isn't the best-looking website, but it gets the information across

AT YOUR SERVICE

The second thing that we're going to look at is setting up a simple HTTP server. This will let you connect to Pico W and read information about what's going on.

```
import microcontroller
import socketpool
import wifi
from temperature_html import temperature_html
from adafruit_httpserver import HTTPServer,
HTTPResponse
from secrets import secrets

ssid = secrets["WIFI_SSID"]
print("Connecting to", ssid)
wifi.radio.connect(ssid, secrets["WIFI_
PASSWORD"])
print("Connected to", ssid)
print(wifi.radio.ipv4_address)

pool = socketpool.SocketPool(wifi.radio)
server = HTTPServer(pool)

@server.route("/temperature")
def base(request):
    return HTTPResponse(content_type="html",
body=temperature_html.
format(temperature=microcontroller.cpu.
temperature))

while True:
    try:
        server.serve_forever(str(wifi.radio.
ipv4_address))
    except:
        pass
```

There's a slightly unusual bit of syntax in the line `@server.route("/temperature")`. This is called a decorator, and you put them before methods. They're not used that much in CircuitPython, but are a bit more common in regular Python. Essentially, they are a way of wrapping a method up in another bit of code. We won't go into everything you can do with decorators, but in this case we're using the `server.route` method to register the method **base** with the server at the location **/temperature**. This way, anyone who visits **http://<server_IP>/temperature** will see the output of this function.

We use the `HTTPResponse` method (that we've imported from `HTTPServer`) to create this response. This method takes care of most things; we just have to tell it that we can't return HTML and that the

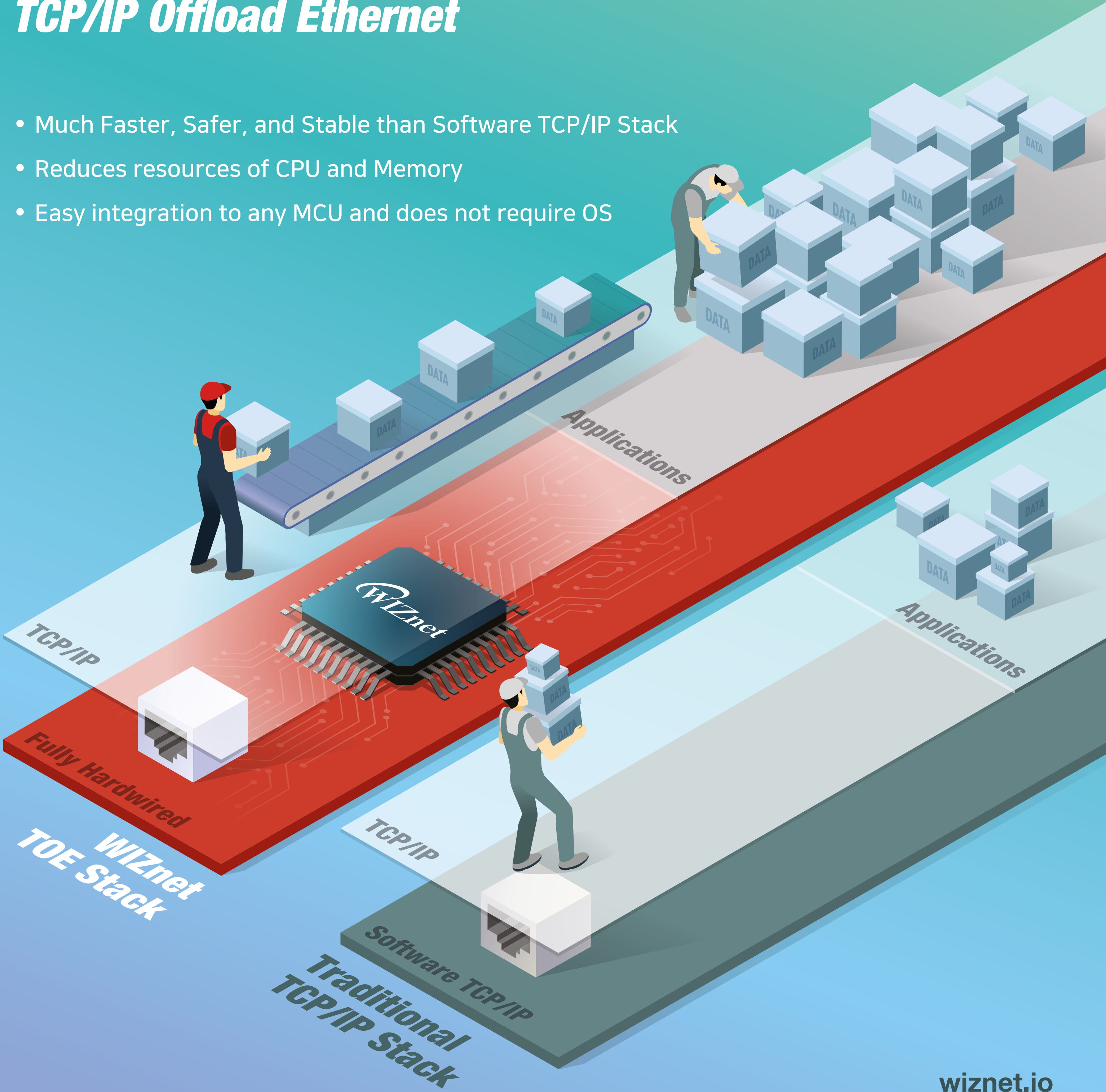
WIZnet

Internet Offload co-Processors for IoT

WIZnet TOE

TCP/IP Offload Ethernet

- Much Faster, Safer, and Stable than Software TCP/IP Stack
- Reduces resources of CPU and Memory
- Easy integration to any MCU and does not require OS



wiznet.io

sinclair

Commodo

the COMPUTERS THAT MADE BRITAIN

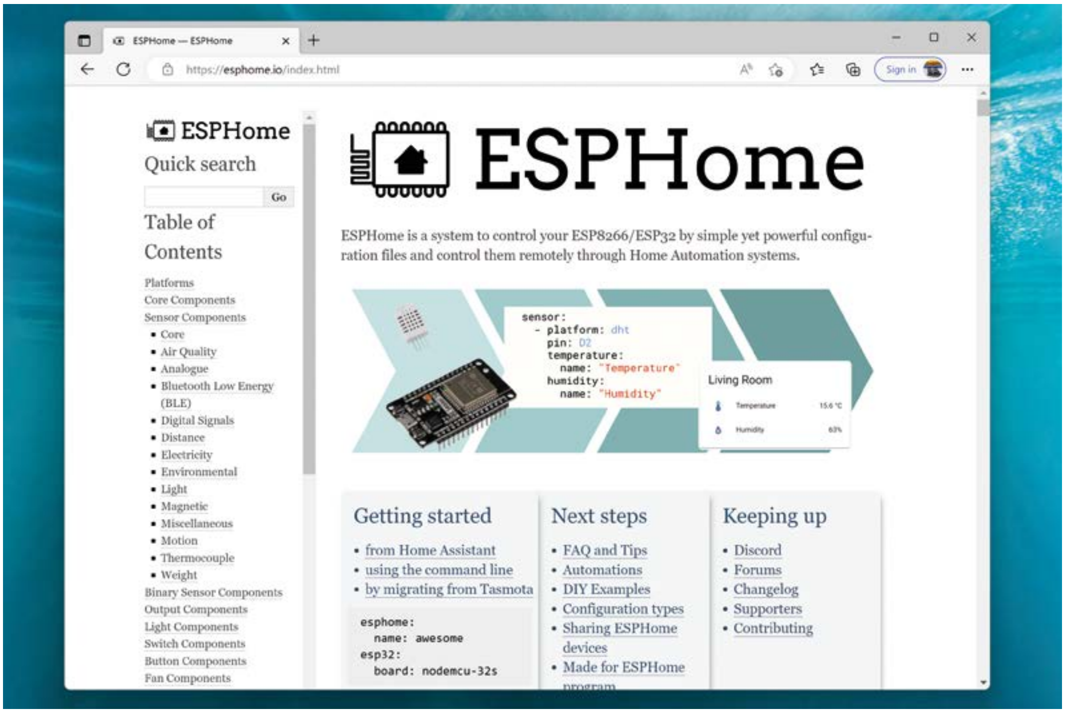
OUT NOW

"The Computers That Made Britain is one of the best things I've read this year. It's an incredible story of eccentrics and oddballs, geniuses and madmen, and one that will have you pining for a future that could have been. It's utterly astonishing!"

- **Stuart Turton**, bestselling author and journalist

Buy online: **wfmag.cc/ctmb**

Available on amazon



ESPHome

Quick search

Go

Table of Contents

Platforms
Core Components
Sensor Components

- Core
- Air Quality
- Analogue
- Bluetooth Low Energy (BLE)
- Digital Signals
- Distance
- Electricity
- Environmental
- Light
- Magnetic
- Miscellaneous
- Motion
- Thermocouple
- Weight

Binary Sensor Components
Output Components
Light Components
Switch Components
Button Components
Fan Components

ESPHome

ESPHome is a system to control your ESP8266/ESP32 by simple yet powerful configuration files and control them remotely through Home Automation systems.

```
sensor:
  - platform: dht
    pin: D2
    temperature:
      name: "Temperature"
    humidity:
      name: "Humidity"
```

Getting started

- from Home Assistant
- using the command line
- by migrating from Tasmota

```
esphome:
  name: awesome
esp32:
  board: nodemcu-32s
```

Next steps

- FAQ and Tips
- Automations
- DIY Examples
- Configuration types
- Sharing ESPHome devices
- Made for ESPHome program

Keeping up

- Discord
- Forums
- Changelog
- Supporters
- Contributing

Left The new support for Pico W in ESPHome gives access to a huge range of IoT hardware with minimal coding

body of the response should be a formatted version of **`temperature_html`**.

When working with web pages, we're going to need to manipulate quite a bit of text. This isn't a huge problem from a processing point of view, but we want to be a bit careful of how we manage our files to ensure that they're not too messy. There are a few ways of doing this, but we've stored the HTML in a string in another file and so are importing that string from the file. We're using Python's built-in string formatting tools to insert data into this HTML string. In its simplest form, this lets us put **`{0}`** in the string somewhere, and then the **`format`** method will replace this with the first parameter to format. If you want to insert more bits of text, you can add **`{1}`**, **`{2}`**, etc. to the string and use extra parameters to format to inset these into the text. The **temperature_html.py** file is:

```
temperature_html = """
<html><head><title>Temperature</title></head>
<body><h1>Temperature</h1>
<p>The temperature is: {0} degrees C</p>
</body>
</html>
"""
```

HTML is a markup language for creating web pages. We're not going to go into it too deeply here, but very simply, text is encased in tags with angled brackets. A **`/`** tag closes the brackets. There's a **`<head>`** section with things like the title, and the body which is the main text of the page (and not to be confused with the body of the HTTP request which includes the whole of the HTML).

> You can use it to build up complex interfaces and **easily get data out of your build without having to physically plug in a cable**

`<h1>` is heading level 1 (i.e. the largest heading); **`<p>`** is a paragraph.

With all that in place, we just need to kick off our server with **`server.serve_forever(str(wifi.radio.ipv4_address))`**. We've wrapped this up in a try-except block in an infinite loop. This is in case there are any problems while it's running. HTTP servers don't have to be particularly complex, but there's a lot of things that can behave strangely in there. The network could lose packets. The client requesting the data might send a poorly formed request, or there could be a bug in our code. This will just attempt to restart the server should anything go awry. It might not last forever, but it will make it a bit more resilient.

There we have it. A simple HTTP server for interacting with a Pico W. You can use it to build up complex interfaces and easily get data out of your build without having to physically plug in a cable – which can make the physical setup much easier. We've looked at the most basic use case here, but it can get as complex as you like. □

HackSpace

This tutorial is from HackSpace magazine. Each issue includes a huge variety of maker projects inside and outside of the sphere of Raspberry Pi, and also has amazing tutorials. Find out more at **hsmag.cc**.

Make a giant Pomodoro timer using Galactic Unicorn

Go large or go home with this jumbo-sized timer for getting more done

MAKER

Nik Rawlinson

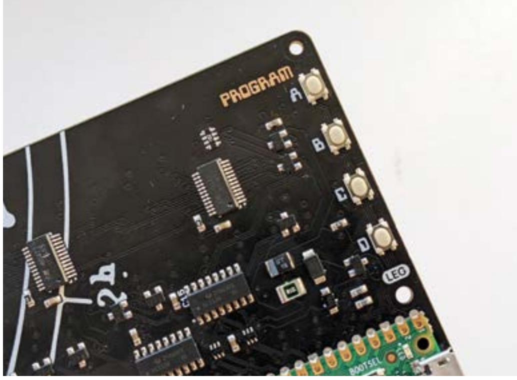
Coffee-drinking typewriter lover with a penchant for pencils and paint. Frequently found staring at the sea from the back of a camper-van.

nikrawlinson.com

The Pomodoro technique is a simple system of effort and reward: work for 25 minutes and you get five minutes off. After every fourth block of work, you can rest for 20 minutes. That's it. But, if you're keeping track of those blocks with the timer on your phone, it's easy to miss a break – or to get so distracted actually playing with your phone that you get nothing done. In this tutorial, we'll solve both of those problems by using the stunning Galactic Unicorn to build a dedicated Pomodoro timer that automatically alternates between work and rest periods. Press button A on the back to start it, C to manually switch between work and rest, and D to end the cycle altogether.

01 Set up the environment

The first four lines of our **pomo.py** code (listed overleaf) import the libraries needed for our routine. We then create a frame buffer, called `buffer`, in which we can temporarily stage any graphics we want to send to the display before committing them to the LEDs. We then declare the variable `gu`, which represents the Galactic Unicorn in our routine, and store its width in the variable `displaywidth`. We'll use that width to calculate the panel's midpoint when we want to centralise text later.



We'll use buttons A, C, and D to control the Pomodoro timer

You'll Need

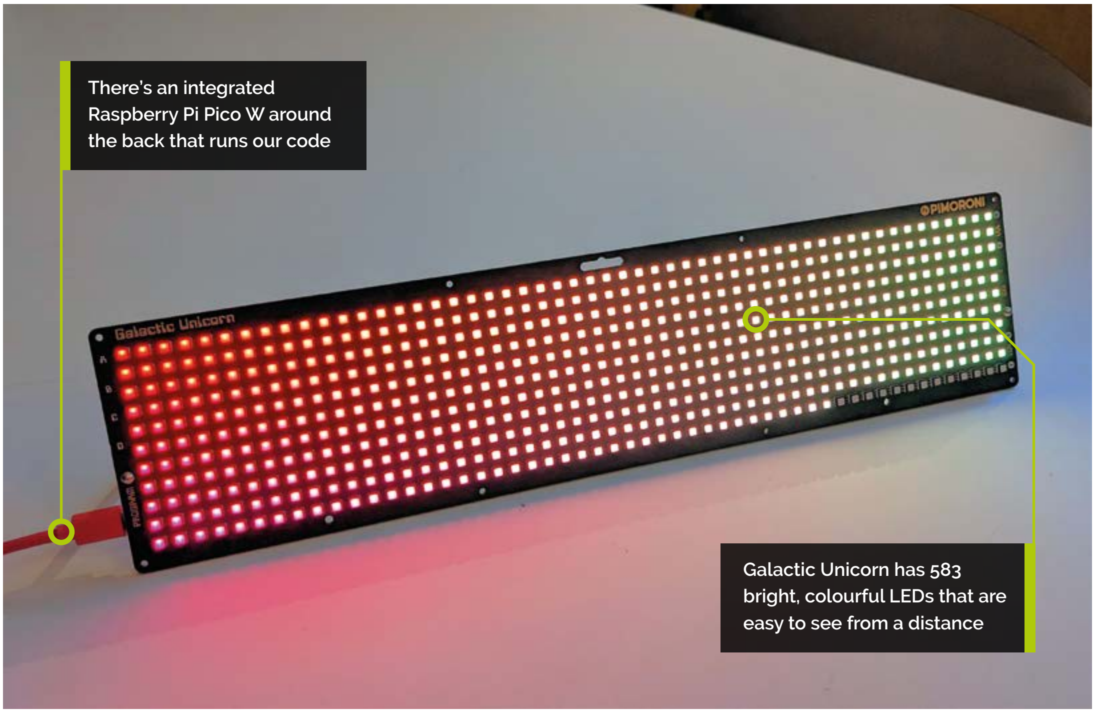
- Galactic Unicorn **magpi.cc/galacticunicorn**
- Thonny. Comes with Raspberry Pi OS; download for other platforms from **magpi.cc/thonny**

02 Define three routines

We've split the bulk of our code into three functions: `cycle()`, `fill()`, and `empty()`. The `cycle()` function manages the transition between `fill()` and `empty()` and keeps track of whether we're in a Pomodoro work cycle or rest cycle. The `fill()` function handles wiping and then illuminating every pixel on the display in a subtle red-biased gradient for a work cycle, and a green-biased gradient for a rest cycle. The `empty()` function extinguishes each LED in turn, at a rate determined by the period: work, rest, or extended rest. Let's take a look at the `cycle()` function in more detail.

03 The cycle() function

Our first job, on line 19, is to define a flag – called `killswitch` – and set its condition to `True`. As you'll see from line 23, we've set the `cycle()` function to carry on repeating so long as that condition doesn't change. We've also established a counter, called `tally`, which will increment by one at the end of every work cycle. Why? Because most Pomodoro routines allow for five minutes of rest after a regular work cycle, and 20 after every fourth work cycle. By keeping track of how



There's an integrated Raspberry Pi Pico W around the back that runs our code

Galactic Unicorn has 583 bright, colourful LEDs that are easy to see from a distance

many work cycles we've completed, we can check whether the number stored in `tally` is a multiple of four when we get to the `empty()` function.

04 Explain what's happening

Lines 24 to 30 tell the user what's going on. We set the pen colour to hot pink, having previously assigned that value to the variable `fg` (for foreground) on line 12, then measure the width of our message, which happens to be "Go get 'em". We subtract the message length from the width of the Galaxy Unicorn display and divide the result by two. The result of this calculation gives us the position of the left-hand margin necessary for centralising the text on the display. We then use:

```
utime.sleep(1)
```

...to pause for a second, which should be long enough to read the display before calling the `fill()` function.

05 Clear the display

When calling `fill()`, we pass it the variable '`type`'. This variable is used throughout the code, and will always be set to either 'work' or 'rest'. It's used to define both the colours used to illuminate the LEDs (in the `fill()` function), and the interval between each LED blinking out (in the `empty()` routine). The `fill()` function's first job is to wipe away the instructions we just gave the user, so it first switches to the black pen, which we assigned to the value `bg` (for background) on line 13, then works through each of the 53 columns and 11 rows, turning off each pixel in turn. We could have blanked the whole screen at once using the command:

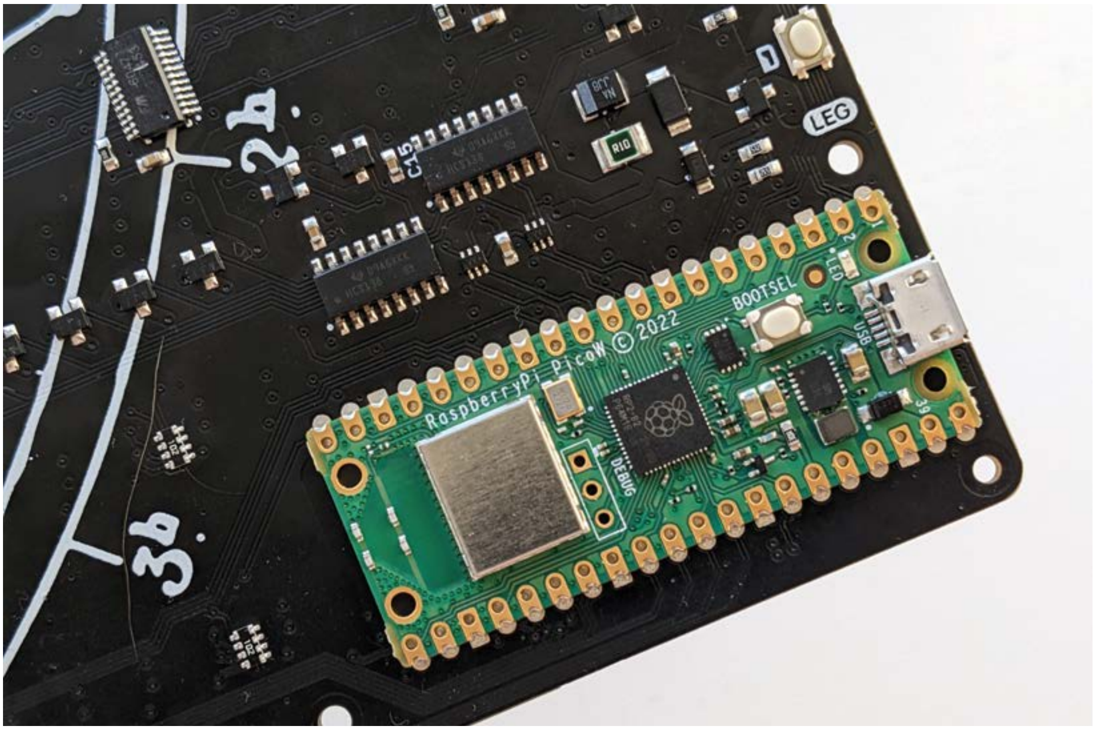
```
buffer.clear()
```

...but doing it one pixel at a time, with a very slight pause between each LED, is a more visually interesting option.

> "We subtract the message length from the width of the Galaxy Unicorn display and divide the result by two"

06 Rainbows everywhere

Once every light has been extinguished, we light them up again. Notice how, on lines 45 to 47, we set the values for red, green, and blue to maximum (255 each). If we used these values to illuminate the LEDs, every light on the board would be white. That would be boring. Instead, we want to introduce a rainbow or gradient effect, so on lines 60 and 62 we reduce the strength of the green tone if it's a work cycle, and the red tone if it's a



As Raspberry Pi Pico W is on board, there's no need to solder any headers

“With every LED illuminated, we're returned to the cycle() function”

rest cycle, by dividing the maximum strength of either by 53 (the number of columns of LEDs) and multiplying the result by the column number to deliver a gradient. Again, we pause briefly between each LED being illuminated, purely for effect.

07 Check the killswitch

With every LED illuminated, we're returned to the `cycle()` function, which immediately calls the `empty()` function in a way that will pipe any value returned by that function to our `killswitch` variable:

```
killswitch = empty(type, tally)
```

This way, we can check whether the `killswitch` was activated while the `empty()` function was running. Notice that, at the same time, we're passing two variables to `empty()`: `type` and `tally`. That's so `empty()` knows how long to leave between each LED going out, and whether we've earned ourselves a longer rest period.

08 Mind the gaps

Lines 78 to 81 effectively set the delay between each LED extinguishing to 2572ms (2.572 seconds) and 515ms (0.515 seconds) for work and rest cycles respectively. We then test whether the number stored in the `tally` variable is a multiple of four on line 83, and, if it is, we extend the gap between each LED going out in the upcoming rest period to 2058ms (2.058 seconds). Multiply each of these timings by 583, which is the total number of LEDs on the board, and you'll see how long each cycle lasts. A work cycle is 1,499,476ms (1499.476 seconds, or 24.99 minutes), a regular rest cycle is 300,245ms (300.245 seconds, or five minutes), and an extended rest cycle is 1,199,814ms (1199.814 seconds, or 19.99 minutes).

09 Test for button presses

Between lines 57 and 109, we extinguish the pixels one at a time, starting at the bottom right of the board and ending up at the top left. Notice line 92:

```
for zzz in range(delay):
```

This is where our calculations come in. Rather than pausing for 2058ms at a time between each work cycle LED, the function actually repeats 2058 times, pausing for 1ms (0.001 seconds) each time, and testing to see if button C or D on the back of

Top Tip

Add audio

The Galactic Unicorn has a built-in speaker, which you can address using the PicoSynth library (**magpi.cc/picosynth**).

pomo.py

▶ Language: **Python**

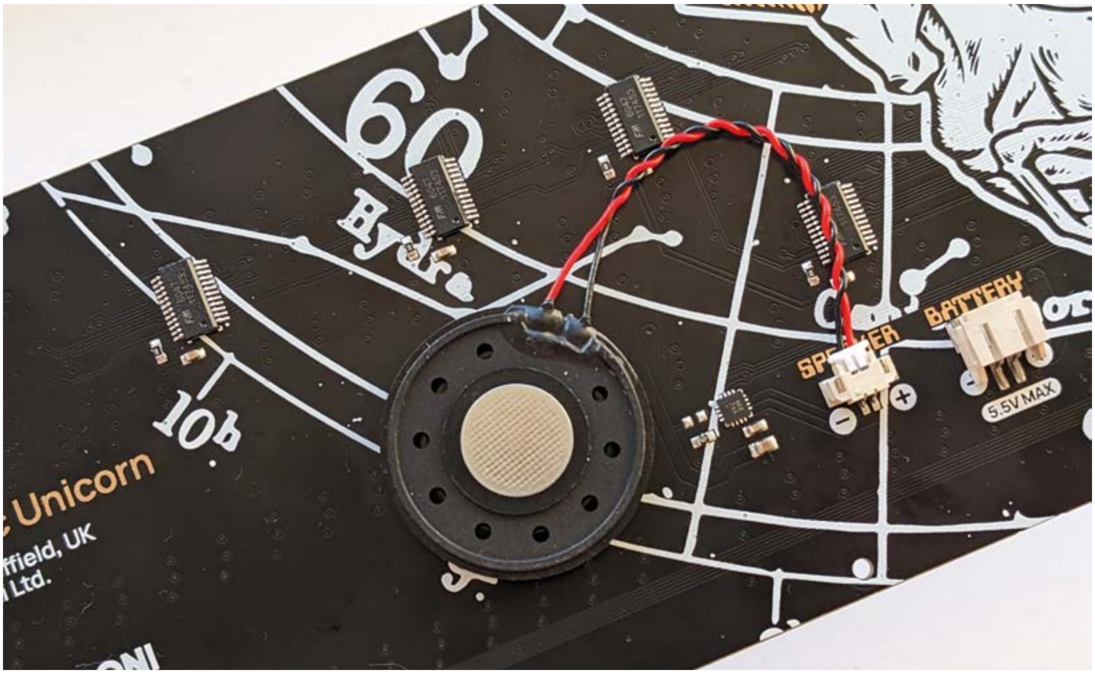
DOWNLOAD THE FULL CODE:

magpi.cc/pomopy

```python
import math
import utime
from galactic import GalacticUnicorn
from picographics import PicoGraphics, DISPLAY_GALACTIC_UNICORN

# define buffer handling elements
buffer = PicoGraphics(display=DISPLAY_GALACTIC_UNICORN)
gu = GalacticUnicorn()
displaywidth = GalacticUnicorn.WIDTH

# define colours and font
fg = buffer.create_pen(255,20,147)
bg = buffer.create_pen(0,0,0)
buffer.set_font("bitmap8")

def cycle(type):

    #set initial variables
    killswitch = True
    message = "Go get 'em"
    tally = 0

    while killswitch:
        buffer.set_pen(fg)
        width = buffer.measure_text(message, 1)
        display_size = GalacticUnicorn.WIDTH
        start_position = int((display_size - width) / 2) + 1
        buffer.text(message, start_position, 2, -1, 1)
        gu.update(buffer)
        utime.sleep(1)

        fill(type)

        killswitch = empty(type, tally)

        if type == "work":
            tally = tally + 1
            type = "rest"
            message = "Break time!"
        else:
            type = "work"
            message = "Go get 'em"

def fill(type):
    red = 255
    green = 255
    blue = 255

    #erase status message
    for a in range(53):
        for b in range(11):
            buffer.set_pen(bg)
            buffer.pixel(a, b)
            gu.update(buffer)
            utime.sleep(0.0001)

    #fill with red to yellow or blue to green rainbow
    for x in range(53):
        if type == "work":
            green = int(4.81 * x)
        else:
            red = int(4.81 * x)
        for y in range(11):
            blue = int(4.81 * y)
            dotpen = buffer.create_pen(red, green, blue)
            buffer.set_pen(dotpen)
            buffer.pixel(x, y)
            gu.update(buffer)
            utime.sleep(0.0001)

def empty(type, tally):
    #set variables
    row = 10
    column = 52
    buffer.set_pen(bg)

    #calculate rest between each LED extinguishing
    if type == "work":
        delay = 2572
    else:
        delay = 515

    if ((tally & 3) == 0) and type == "rest":
        delay = 2058

    #blink out one pixel at a time
    while row > -1:
        while column > -1:
            buffer.pixel(column, row)
            gu.update(buffer)
            column = column - 1
            for zzz in range(delay):
                if gu.is_pressed(GalacticUnicorn.SWITCH_C):
                    buffer.set_pen(bg)
                    buffer.clear()
                    gu.update(buffer)
                    killswitch = True
                    return killswitch
                elif gu.is_pressed(GalacticUnicorn.SWITCH_D):
                    buffer.set_pen(bg)
                    buffer.clear()
                    gu.update(buffer)
                    #set termination flag
                    killswitch = False
                    return killswitch
                else:
                    utime.sleep(0.001)
        column = 52
        row = row - 1
    killswitch = True
    return killswitch

while True:
    while gu.is_pressed(GalacticUnicorn.SWITCH_A):
        buffer.set_pen(bg)
        buffer.clear()
        gu.update(buffer)
        cycle("work")
```

Elektor
DAS MAGAZIN FÜR DIE RASPBERRY PI COMMUNITY
MagPi
Das offizielle Raspberry Pi Magazin
März/April 2022 Nr. 2
magpi.de
BETRIEBS-SYSTEME
TÜFTELN, LÖTEN & PROGRAMMIEREN MIT DEM ZERO 2
RASPBERRY PI SMART GADGETS
+ ÜBERTAKTUNG BEIM RASPBERRY PI ZERO 2 W
57 SEITEN PROJEKTE UND PRAXIS
#29 Juli/August 202
Raspberry Pi Pico Grundlagen
LoRa m
Raspbe Pico
48 SEITEN PROJEKTE UND PRAXIS

Raspberry Pi Pico with pinheaders
BONUS
Elektor
MagPi
Achtung – Meiner ist +8 GB
€5
Herzlich willkommen!



Although our project doesn't use it, the Galactic Unicorn has a built-in speaker that could be used to sound an alert between cycles

the Galactic Unicorn has been pressed. If either has been touched, the function ends and we return to `cycle()`. For regular and extended rest cycles, the function repeats 515 and 2058 times respectively.

10 How it all began

If it was button C that sent us back to `cycle()`, we switch between the work and rest routines (or vice versa) and, as the `killswitch` flag hasn't yet been set to `False`, the loop continues. However, if it was button D, the flag will be set to `False`, and the loop will terminate. We'll be returned to the code that called `cycle()` in the first place, which is the six lines from 113 to 118, at the very end of our script. This puts the integrated Raspberry Pi Pico W into a watchful state, where it's waiting for someone to press button A. When someone does this, it makes sure the screen is blank, then launches `cycle()`, and we're back where we started.

11 Test your code

To copy the code to your Galactic Unicorn, make sure the board is connected to your computer, paste the code into Thonny, then click File, followed by Save As. Click Raspberry Pi Pico on the dialog, then save it on the device using any name you choose, such as 'pomo.py'. To run it from Thonny but on the Galactic Unicorn, click the status line at the bottom right of the Thonny window and select 'MicroPython (Raspberry Pi Pico)', then click the play button on the Thonny toolbar.

12 Run at boot

If you want to be able to run the timer when your Galactic Unicorn isn't connected to a computer, make sure you save the code as **main.py**, on the device itself. As there is already a routine called **main.py** on board, Thonny will ask if you're sure you want to overwrite it. Click Yes (you may want to make a copy first in case you ever want to reinstate it). Once the script has been saved, you can unplug your Galactic Unicorn from the computer and plug it into a regular USB adapter to run the Pomodoro timer wherever you choose.

Top Tip

Something smaller

If you don't have space for Galactic Unicorn, you could code a similar timer using a smaller board like the Pico Unicorn (**magpi.cc/picounicorn**).

Once we know the size of the display, we can work out the midpoint for centralised text

#MONTHOFMAKING ASK THE EXPERTS

Top tips from expert makers to help you with your next project

By expert writer Rob Zwetsloot

It's that time of the year again – time for *The MagPi* community far and wide to make something and share their progress online. This collaborative build event can be joined by anyone and with any kind of project you wish! Features editor Rob will be sewing himself.

This year, we thought we'd ask some expert makers about their specialty subject. Flip through to get inspired and maybe build your dream project this March.

See y'all online.

THE RULES OF #MONTHOFMAKING

- Work on a project, new or old
- Take photos of your progress and completed projects
- Share it to Twitter, Mastodon, and/or Instagram with a helpful description
- Make sure to include the #MonthOfMaking hashtag
- That's it!

If you don't have social media, you can always email your projects to **magpi@raspberrypi.com**

PAST #MONTH(S)OFMAKING

The MagPi #79

We introduced #MonthOfMaking in *The MagPi* #79 with eleven projects you could try out.
magpi.cc/79

The MagPi #91

In *The MagPi* #91, we showed how to document and share your projects with the community.
magpi.cc/91

The MagPi #103

We explored other forms of making in *The MagPi* #103, including wearables, embroidery, and upcycling.
magpi.cc/103

The MagPi #115

Microcontroller Madness was the theme last year, showing you how to make great things with a Raspberry Pi Pico.
magpi.cc/115

3D PRINTING EXPERT

Turn ideas into physical objects with these top 3D printing tips

Allie Katz

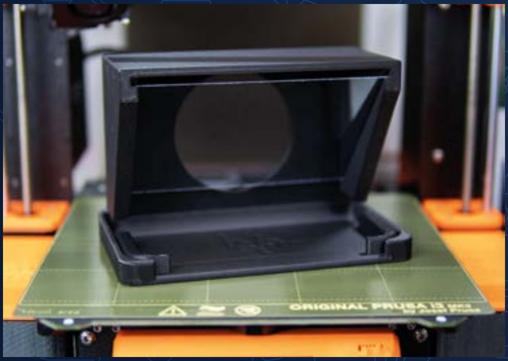
Allie is a YouTuber and creative technologist who does a lot of 3D print designing and… printing. They're a pro at turning an idea into a real thing, and they also work with ceramics and wearables! **katzcreates.com**

Q: How did you start with 3D printing?

A: A friend and a generous gift for the most part! A friend of mine showed me his 3D printer, lighting a bit of a flame in me to want to try it myself and resulting in it becoming the top thing I wanted to get for myself. Amazingly, I received my first 3D printer as a birthday gift, and as they say, the rest is history!

Q: What are some of your favourite prints you've done?

A: I've got a pretty deep love for Big Buff Pikachu designed by Chan Yen Yee; that was one of my early prints. It's just so silly and beautiful and fun. Beyond that, I get kind of boring rather quickly as I think most of my favourite prints are practical things I've designed for myself to make my life easier, like my DIY Teleprompter: **magpi.cc/prompter**.



Q: What do you think is an important thing for makers to consider?

A: I think an often overlooked consideration when it comes to 3D printing is to know *your* printer. Not everyone's printer can do the same thing, and I don't just mean in terms of size limitations! Knowing what your printer is capable of, where its strengths and weaknesses are, will save you a lot of time and wasted plastic later when it comes to printing things. Building out custom printing profiles that work with these strengths and weaknesses as well will save you a lot of time later!

Q: Are there any ways to practise designing and/or printing?

A: Start small and work your way up to bigger and more complex things. Test and calibration prints are a great way to get to know your printer's capabilities and limitations, but if you find that uninteresting and want to be designing and printing your own things, then try tackling one difficult thing at a time, instead of trying to throw it into a single design. I also recommend reading up on the basics/theory of 3D printing before diving in too far, particularly since there are some amazing, very easy-to-digest primers out there. I highly recommend Billie Ruben's guides: **magpi.cc/3druben**.

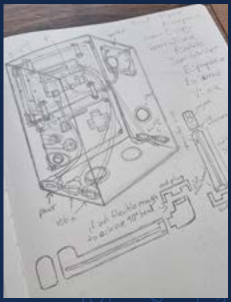
CAD DESIGN TIPS FOR 3D PRINTING

BY BILLIE RUBEN

Like all production processes, 3D printing has constraints which are to be considered during the design phase in order to make high-quality, functional, and beautiful objects. I love seeing people make their own designs to fulfil their specific needs and desires, so I've collected many of these considerations to help you hit the ground running! ♥

Make walls a multiple of your extrusion line width for a
Filament must be laid upon existing material, so avoid
Vertical holes are fine, but horizontal ones should be
Pointed arches are better than round ones as they
Vertical edge fillets increase quality by reducing inertia

STEPS TO MAKE

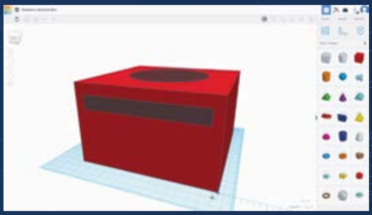


01 Planning out a 3D print

"Think about what you want/need and ask yourself the question: is 3D printing best for this? If you need something that is small, very detailed, or requires certain properties that will be well served being plastic, then 3D printing is likely the best solution! It's important to recognise though that not everything is best 3D-printed, and having a good understanding where 3D printing is ideal is incredibly useful!"

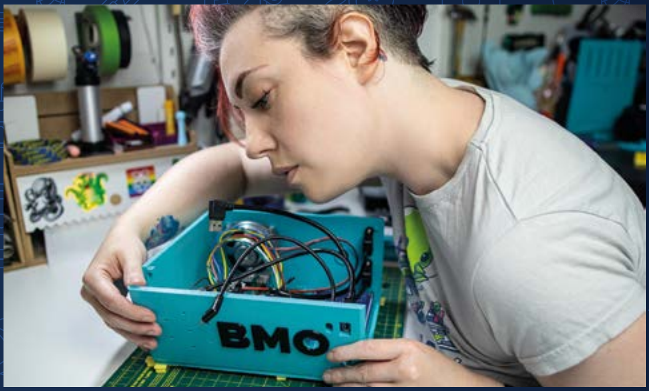
02 Designing a 3D print

"Always keep your printer's tolerances in mind while designing pieces that need to fit together (or into something else)! There will always be a small amount of space 'lost' when 3D printing because of nozzle inaccuracies and thermal expansion of the plastic, and if you don't create gaps between your pieces in the design software, there's a very good chance that the physical pieces won't fit together when printed. As mentioned before, all printers are different, and it's key to know what your printer is capable of!"



03 Printing

"If designing anything with parts that fit together or has specific size requirements, print prototypes that test one particular aspect before printing the whole piece at full quality! Printing only a certain part of a piece if you're testing the fit, or a piece with infill only (no top or bottom layers) to check on sizing, are great examples of how you can print faster and less material-dense prototypes before jumping into the final piece."



ALLIE'S PROJECTS

Clean air necklace

3D printing, electronics, sensors, wearables

Make sure the air you're breathing above your neck is good, and also make it fashion.

magpi.cc/3dnecklace

3D-printed LED name badges

3D printing, electronics, sensors, wearables

After a lengthy hospital stay, Allie made these badges to thank their nurses.

magpi.cc/3dbadge

BMOctoprint

3D printing, web services, programming, PCB design

This cute robot from Adventure Time helps keep Allie's 3D printer running.

magpi.cc/bmoctoprint

RESOURCES

Thingiverse

MakerBot-owned repository for 3D print files of all kinds. It's been around for nearly 15 years, so there's a lot on there.

thingiverse.com

Printables

Prusa-owned repository for all things 3D print files. It also has great articles on how to actually 3D-print.

printables.com

Etsy

A marketplace for small creators, many people sell their own 3D designs on here, so give it a look if you're struggling to find what you want.

etsy.com

ROBOTICS EXPERT

Automatons, rovers, and arms; there's a lot you can make under the robotics umbrella

Kevin McAleer

Friend of *The MagPi* Kevin McAleer is a master of robotics, wowing us nearly weekly with a new robot (or just a cool coat for his dog). He shows his process on his YouTube channel, a great resource for learning about robotics in itself.
kevsrobots.com

Q: How did you start with robotics?

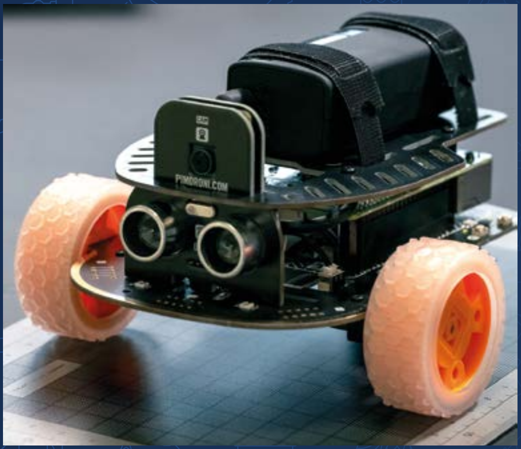
A: I started building robots with a single blinking LED on an Arduino, this quickly moved onto buzzing buzzers, moving servos, and detecting light with photosensors. Then I bought a 3D printer, and this changed everything: I was able to download 3D-printable robots and started to learn more about how to make them (it's how **smarsfan.com** came into being).

Q: What are some of your favourite robots?

A: My favourite robots are small, 3D-printable robots that you can build and modify yourself. These include OpenCat, OttoDIY, SMARS (obviously), and Danielle Boyer's Scobots. I love Raspberry Pi Pico, Pico W, and full Raspberry Pi robots such as the Pimoroni Trilobot. All the robots I build use MicroPython, too, because its so easy to read and write.

Q: What are some great kits for starters?

A: The Trilobot (**magpi.cc/trilobot**) is a great starter kit, and can utilise the Raspberry Pi Camera Module to detect objects.



I designed the BurgerBot for beginners; its easy to construct and runs on either a Pico or Pico W, with parts from Pimoroni (see **magpi.cc/burgerbot** for the bill of materials).

CuteBot (**magpi.cc/cutebot**) from Elecfreaks (micro:bit-based robot).

Kitronik Autonomous Robotics Platform for Pico (**magpi.cc/picorobotics**).

Q: What are some things people tend to forget?

A: Start small, work on one problem at a time, and don't be afraid to ask for help.

Robotics is a deep topic, and can be a bit overwhelming, but don't give up! Look to your strengths in the three skill areas: mechanics, electronics, and programming. This will also help you understand where you need to develop your skills. (I'm still working on my electronic skills.)

STEPS TO MAKE

01 Planning

Whether you're building your first robot from scratch or gearing up for a Pi Wars contest, Kevin has this to say about robot planning: "What do you want to achieve? What should the robot do (and what are the bonus stretch goals)?"

02 Designing

"How will you make the robot do these things, what kit do you need, and what do you need to learn? Will you build this or buy it? I like to draw a sketch of the robot, and make a list of what it should do, in my robot lab notebook."

03 Building

"Have a project box to keep all the bits together and understand that building is an iterative process – none of my robots are finished! Decide on what is good enough when building. Think about the parts you'll need, the tools and materials you need, and time in your week when you can concentrate on building it."

KEVIN'S PROJECTS

Spooky Scary Skeleton

Robotics, 3D printing, sensors, spookiness

This Halloween decoration truly sends shivers down our spines.

magpi.cc/spookyskele

PicoCat

Robotics, 3D printing, servos, non-traditional locomotion (it walks)

This Pico-powered feline walks on its four legs using a series of servos, and also has an ultrasonic distance sensor so it can, feasibly, not bump into you.

magpi.cc/picocat

Cubie-1

Robotics, 3D printing, ROS, Lidar

A little experimental robot that uses Robot Operating System (ROS) on a Raspberry Pi 4.

magpi.cc/cubie1

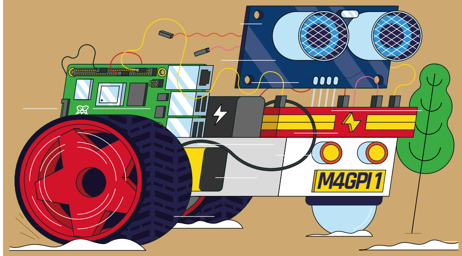
OTHER PROJECTS

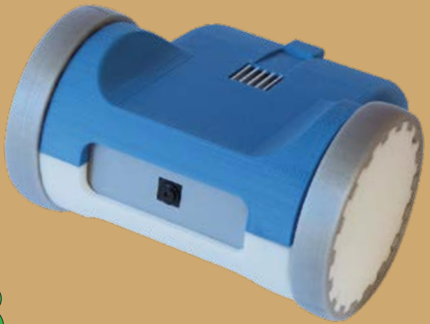
Try out these other robot projects!

Super-Simple Robotics

Our full guide to building a beginner robot can be found in issue 120, and is a great companion to Kevin's videos.

magpi.cc/120





ZeroBot

This cool little robot uses Raspberry Pi Zero in a 3D-printed case to create a tiny, programmable automaton.

magpi.cc/zerobot

CamJam EduKit #3

This very basic kit can be built using the box as a chassis, and you can always use the components for other robots.

magpi.cc/edukit3

RETRO GAMING EXPERT

Time to build your dream arcade cabinet or portable gaming device

K.G. Orphanides

K.G. Orphanides is *The MagPi*'s resident retro game writer, and you may have seen their regular tutorials on how to emulate an old console, or replicate a classic control scheme that works with modern hardware.
magpi.cc/hauntedowlbear

Q: How did you get into retro gaming?

A: When I started, it was just 'gaming', thanks! My family won an MSX in a raffle when I was seven, and my mum, an IT worker, taught me how to program in BASIC. Then she borrowed an XT from work until we saved up enough for our own 286. I kept using PCs and picked up other computers and consoles cheaply as they went out of fashion. I vividly remember going on eBay as a teen in the late 1990s and sending envelopes of dollars off to the US in exchange for giant boxes of remaindered classic DOS games that reached me on a very slow boat, with very cheap postage.

Q: What kind of projects do retro gaming enthusiasts work on?

A: I love emulation for the way it's kept old platforms and games relevant, for how accessible it is, and for the way it provides a window into the past.

Or you could buy or salvage old computers and consoles, or build a Pentium II or Athlon K7 based PC running FreeDOS. But platforms like Raspberry Pi (software emulation) and MiSTer FPGA (hardware emulation) mean you can build dedicated systems for running old games without the aging equipment.

FROM THE MAKERS OF MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
NEW 2022 UPDATE
RETRO GAMING WITH RASPBERRY PI
2nd EDITION
164 PAGES OF VIDEO GAME PROJECTS

Q: What are some of your favourite projects?

A: I love 1980s and 1990s audio hardware, so making an emulated USB version of a Roland Sound Canvas (my favourite sound card range) with mt32-pi (**magpi.cc/mt32pi**) was a blast.

I had no idea how much I wanted and needed an arcade cabinet in my life until I built one. You'll find that, along with a few of my other makes (like the DIY CD-ROM console I made to play my PS1 game discs!), in *Retro Gaming with Raspberry Pi* (**magpi.cc/retrogaming**).

Q: How do you find games to run on emulators?

A: **Itch.io** is your friend! Just search for the platform you're interested in, like NES, C64, ZX Spectrum, or MSX!

If you're after official digital releases of old commercial games for DOS and Windows, GOG (**gog.com**) and Zoom Platform (**magpi.cc/zoom**) should be your first ports of call.

Indie Retro News (**indieretronews.com**) announces new releases across most major 8- and 16-bit retro computers, consoles, and fantasy consoles.

STEPS TO MAKE

01 Planning

“Make things that bring you joy! It should be fun to build, fun to have or, ideally, both. Planning is playing! While working out what I want to make, I’ll research and experiment in software often just on my desktop workstation, often while procrastinating other work.

When I’m writing a project guide, I usually test all possible permutations of the software on the hardware I have in order to pick the one that produces the best result.”

02 Parts

“Make sure all the hardware and software you want to use is actually compatible! Your available hardware should generally determine what software you use. I’m a firm believer in recycling and reusing as much as possible. Actually paying money for new hardware is sometimes required, but some of my favourite builds reused old stuff I have around (a CRT TV, an elderly DVD drive) or draw on my stash of reusable components (SBCs, microcontrollers, adapters with screw-down terminators).”

03 Construction

“You should be pretty confident that everything works before you assemble parts into a inconvenient location. Where possible, build the innards of the project on the bench/your sitting room floor before permanently soldering, screwing, or otherwise mounting them into place. It’s been three years and I’ve still not rewired some arcade machine buttons I connected to the wrong GPIO pins, even though it preys on me sometimes. (I put in a software workaround instead.)

Finally: don’t be afraid of stickers, decals, stencils, and paint. They’re easier to position and reposition than you think, and any wonkiness is almost always completely invisible to everyone who isn’t you.” M

K.G.’S PROJECTS

Build a MicroPython handheld

Retro gaming, electronics, game programming, 3D printing

It turns out Pico is great for game programming! K.G. shows you how to make the most of it.

magpi.cc/123

Pico DB9-to-USB joystick

Retro gaming, programming, electronics

Instead of daisy-chaining twelve adapters, how about creating a custom DB9 to USB interface?

magpi.cc/125

Commodore 64 emulation

Retro gaming, software, emulation

Raspberry Pi 400 is somewhat inspired by 1980s computers. Here’s how to make it run like one.

magpi.cc/102

RESOURCES

RetroPie

A retro gaming system built upon Raspberry Pi OS that makes it very easy to set up and play games. It’s used in many retro projects!

retropie.org.uk

Make your own video games

K.G.’s guide on how to make games like in the old days. Many lessons can be brought forward to modern game development as well.

magpi.cc/73

Legal emulation information

Our handy guide on legal emulation, along with links to repositories of completely legal and legitimate ROMs you can get for your machines.

magpi.cc/legalroms

PiBoy XRS

SPECS

INPUT: 2 × analogue sticks, 1 × D-pad, 7 × face buttons, 4 × shoulder buttons

PORTS: 3.5 mm audio jack, mini HDMI out, 2 × USB 2.0, 2 × USB 3.0, 1 × Gigabit Ethernet

DIMENSIONS: 162 × 93 × 36 mm

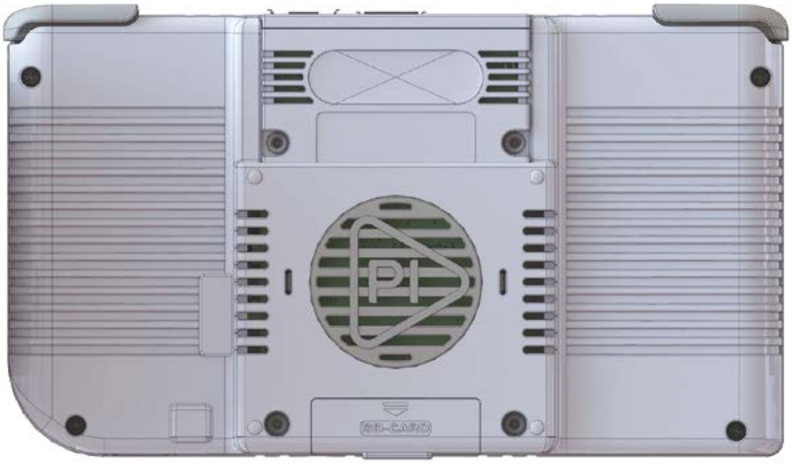
DISPLAY: 3.5" IPS LCD

▶ Experimental Pi ▶ magpi.cc/piboyxrs ▶ £125 / $150

A more ergonomic handheld experience, the PiBoy XRS takes the best parts of a lot of classic consoles. **Rob Zwetsloot** has a play

We were big fans of the PiBoy DMG, a Raspberry Pi 4-compatible handheld in the shape of the classic Game Boy, albeit with a few more buttons and an analogue stick. It's a great form factor for nostalgia purposes. However, even Nintendo switched to a horizontal format eventually. This brings us to the PiBoy XRS which, while quite like the DMG version, is definitely a superior product.

Taking design notes from the original Game Boy, Neo Geo Pocket, and just about every portable game system since the release of the Game Boy Advance, this updated kit also whacks on a second analogue stick for much improved playability. Instead of two shoulder buttons, there are now four, and while that does mean two fewer face buttons, for the vast majority of games it's a layout we prefer. It turns out there's a reason why handhelds went horizontal after the Game Boy.



▲ There's easy access to the microSD port, and plenty of ventilation for Raspberry Pi 4 too

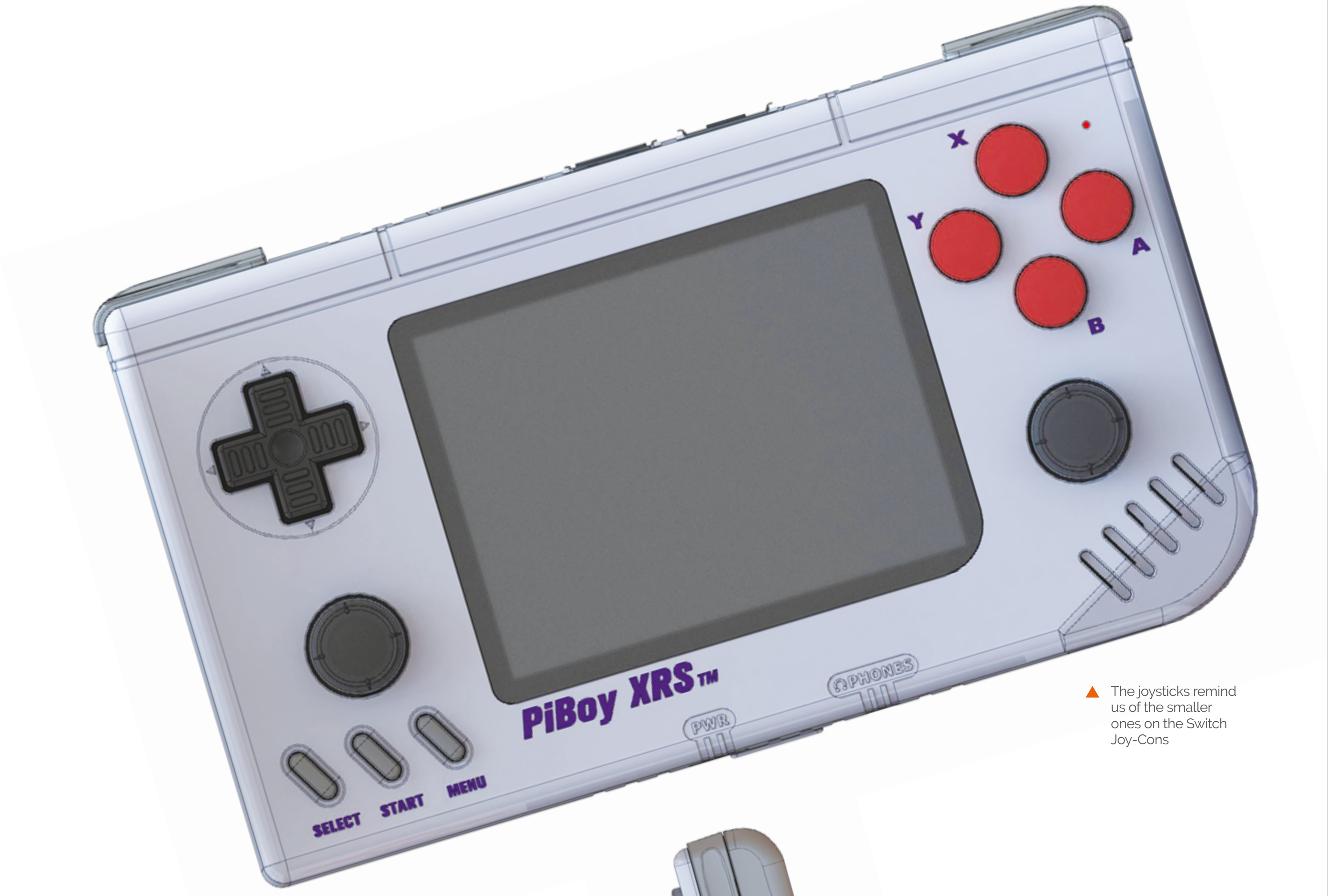
Hook it up

Installing a Raspberry Pi 4 is very simple – with the supplied screwdriver, you just need to take the eight screws off the back and then slot Raspberry Pi into the GPIO pins. There's some thermal paste for the processor, and an optional adapter for the mini HDMI out. Four of the screws on the back-plate go through the mount holes on Raspberry Pi, making for a very elegant reassembly process.

Power and headphone jacks are routed to the underside of the console, with rechargeable batteries, adding up to 5600 mAh, connected to that USB-C power port for hours of play time. The standard USB and Ethernet ports are then left open at the top of the XRS for easy, if not perhaps slightly awkward, access for peripherals and memory sticks. Those pop out of a fake cartridge slot, which is a nice little design touch, especially paired with the hand-detachable cover that allows you to change out microSD cards without having to get the screwdriver back out.

Playing with power

Like DMG, the whole package is solid and feels good to hold – we find some handheld Raspberry Pi consoles can feel a little flimsy and, while this may be a little heavier than those, we appreciate the added heft.



▲ The joysticks remind us of the smaller ones on the Switch Joy-Cons

Playing games is incredibly simple. Once the PiBoy image is installed, you can game basically straight away and as it's the familiar RetroPie interface, you can easily customise it to your liking. Compatible games run smoothly thanks to the extra oomph of Raspberry Pi 4, and everything on the screen looked lovely as we played. Our kit came with the mini HDMI adapter, and we had no problems playing games on a bigger TV with it plugged in either – although, due to the graphical prowess of some games, it did look better on the smaller screen with a higher pixel density.

> " Once the PiBoy image is installed, you can game basically straight away "

Also, as it has RetroPie, it means you can install Steam Link, so it's a pretty cool alternative to a Steam Deck.

◄ Access to the USB ports is handy however you play it. There are much fewer use cases for the Ethernet port, though

Verdict

The perfect retro handheld system is finally here, and you can even use it to comfortably play modern games as well.

10/10

Plasma Stick 2040 W (Pico W Aboard)

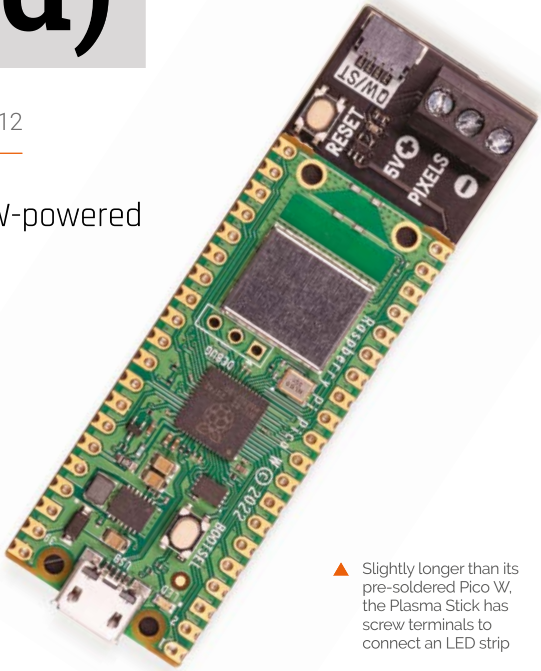
▶ Pimoroni ▶ magpi.cc/plasmastick ▶ £12 / $12

Light up your projects with this Pico W-powered NeoPixel controller. By **Phil King**

SPECS

FEATURES: 3 × screw terminals (5 V, ground, data), reset button, Qwiic/STEMMA QT port

DIMENSIONS: 67 × 22 × 12 mm



▲ Slightly longer than its pre-soldered Pico W, the Plasma Stick has screw terminals to connect an LED strip

With a pre-soldered Pico W on board, the Plasma Stick 2040 W is designed as an easy-to-use controller for addressable NeoPixels (aka WS2812B / SK6812 LEDs). Three screw terminals offer a simple and secure connection method for your LED strip's wires. Note that if the strip has female connectors, as ours did, you may need female-to-male jumper wires.

To get you up and running quickly, the Plasma Stick 2040 W is preloaded with Pimoroni's own brand of MicroPython firmware, complete with a fair few example programs, including fire, rainbow, snow, and sparkle effects. These can easily be adapted to suit the number of LEDs in your strip. Alternatively you can code in C/C++, or flash Pico W with CircuitPython to make use of Adafruit's LED Animation library.

Plasma programming

The MicroPython code examples are based around Pimoroni's Plasma library, as used for the firm's Plasma 2040 controller – although unlike that board, the Plasma Stick can't be used with DotStar/APA102 LEDs. The colour of individual LEDs can be set using RGB or HSV, along with the overall brightness level.

▲ The Plasma Stick uses the power of the RP2040 chip's PIO capability to drive LEDs at a high, flicker-free frame rate

> Pico W's Wi-Fi connectivity adds extra possibilities for your lighting projects

Just connect to a computer via USB to power the Plasma Stick's Pico W and run/edit the code. As usual, to get a program to run automatically on boot-up, save it as **main.py** and then connect a standard USB power source, such as a power bank.

Pico W's Wi-Fi connectivity adds extra possibilities for your lighting projects. Code examples include responding to the current Cheerlights colour (see **@cheerlights** on Twitter) or weather conditions sourced from the web. You can also take readings from any sensor attached to the Plasma Stick's Qwiic/STEMMA QT port.

Verdict

While you could always use a plain Pico W to control LED strips, the Plasma Stick makes it simpler and has the bonus of a Qwiic/STEMMA QT port and reset button.

8/10

AN INTRODUCTION TO C & GUI PROGRAMMING

All you need to know to write simple programs in C and start creating GUIs

Inside:

- Create simple command-line C programs
- Control flow with conditions and loops
- Handle variables, strings, and files
- Design graphical user interface applications in C
- Handle user input with buttons and menus
- Use advanced UI features such as data stores and dialogs

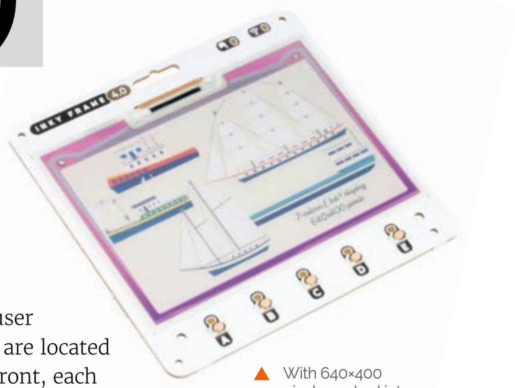
£10 with FREE worldwide delivery

Buy online: magpi.cc/cgui

Inky Frame 4.0” (Pico W Aboard)

▸ Pimoroni ▸ **magpi.cc/inkyframe4** ▸ From £70 / $71

This slick colour e-ink display benefits from Pico W's wireless connectivity. By **Phil King**



With 640×400 pixels packed into a 4-inch screen, the Inky Frame has excellent definition for displaying images

SPECS

DISPLAY:
4.01-inch e-ink, 640×400, seven colours

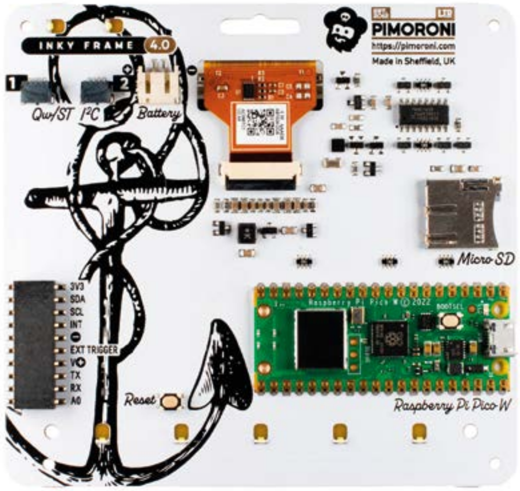
FEATURES:
Pico W, 5 × user buttons, reset button, LEDs, microSD card slot, breakout header, 2 × Qwiic/STEMMA QT ports, 2 × metal legs

DIMENSIONS:
102.8 × 96.7 mm

The latest in Pimoroni’s Inky range of e-ink screens, the Inky Frame 4.0” features a 4-inch, 640×400, seven-colour display that offers superb detail. The board is based around a Raspberry Pico W pre-soldered to the rear. As well as giving it Wi-Fi connectivity, this increases portability when used with a battery pack.

Also on the rear, there’s a handy reset button, microSD card slot (for extra storage), two Qwiic/STEMMA QT ports, a JST-PH battery connector, and an extension header (including six GPIOs).

> “It can even be put into a super deep sleep mode with Pico W shut down”



On the rear you’ll find Pico W, along with a microSD slot, extension header, battery connector, and two Qwiic/STEMMA QT ports for breakout sensors

Five user buttons are located on the front, each with an LED, with two more LEDs at the top to indicate activity and wireless connection. Two metal legs (supplied) can be used to stand up the board.

Refreshing stuff

The Inky Frame comes preloaded with Pimoroni’s brand of MicroPython firmware, including the PicoGraphics display library and several code examples. As soon as you connect it to a computer via USB, it’ll start a screen refresh which takes 25–30 seconds – typical for this kind of multicolour e-ink display.

By default, it auto-runs a launcher menu with five example programs. However, to run them you’ll first need to stop the launcher in Thonny and create a **secrets.py** file with your Wi-Fi credentials (see Pimoroni’s Getting Started guide: **magpi.cc/inkyframeguide**).

An optional accessories pack adds a 3×AA battery pack (with batteries), Velcro square to attach it to back of board, 16GB microSD card, and micro USB to USB cable. You could use a LiPo battery instead, but the board can’t charge it. Still, it uses a small amount of power while refreshing, and can even be put into a super deep sleep mode with Pico W shut down – and reawakened by the real-time clock, buttons, or a trigger pin on the extension header.

Verdict

A classy colour e-ink display whose Wi-Fi connectivity greatly extends its possible uses, including as a digital photo/art frame, life organiser, or low-power smart home dashboard.

9/10

10 Amazing:

Portable Projects

Carry Raspberry Pi around with you wherever you go with these cool builds and kits

Raspberry Pi is basically pocket-sized. You don't even really need big pockets either, and Raspberry Pi Zero and Pico will even fit in your wallet! Now that's small. Being so easy to take with you, that begs the question, what are the best ways to carry it around? We have ten answers for you.

▲ Mini laptop

Palm-sized Raspberry Pi

This DIY build is very cheap and very good as well – it runs on a Raspberry Pi after all! Upgrade it with a Raspberry Pi Zero 2 W and it's even more powerful.

magpi.cc/74

▲ Becca Cam

Raspberry Pi SLR

Built into an old SLR camera, this Raspberry Pi digital camera gets to use some of the benefits of an old SLR. Imagine this with a Camera Module 3…

magpi.cc/beccacam

▸ PiBoy DMG kit

Retro handheld

We reviewed the successor to this kit in this issue, the XRS. However, the original is still an excellent handheld that we take around with us to this day.

magpi.cc/piboydmg | £113 / $140

▾ CutiePi tablet

Tablet with handle

This Compute Module 4-powered tablet is very practical and useful, and the fancy handle can also be used to prop it up in multiple ways.

cutiepi.io | £185 / $229

Lunchbox Arcade

Breaktime combos

You could take your lunch in a lunchbox, or you could take Super Street Fighter II Turbo and find out who is the strongest world warrior.

magpi.cc/lunchbox

Cyberdeck Portable Entertainment System

Retro-future laptop

This suitcase is fit for the 1980s cyberpunk fantasy of your dreams, complete with Raspberry Pi desktop and the ability to play retro games.

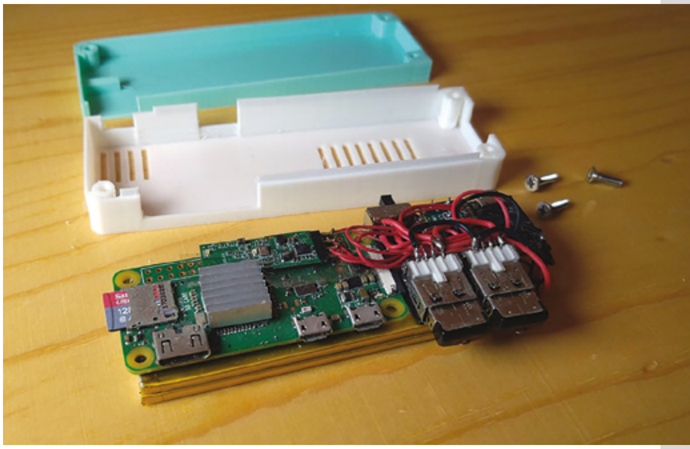
magpi.cc/cyberneon

Picade

Arcade cool

This is technically portable, especially if you get the smaller one, or even just get the plug-and-play version. It's a great way to game on the go.

magpi.cc/picade | From £195 / $225



Pocket Cloud Server

Tiny NAS

This wireless cloud server build on a Raspberry Pi Zero W is a great way to take the cloud with you for work or vacation.

magpi.cc/pocketcloud

CrowPi L

Lite laptop

This small, yet powerful, Raspberry Pi-powered laptop is also great for learning electronics with, as it has full GPIO access.

magpi.cc/crowpil | £193/$239

Mini Dongle Computer

Plug-in Zero

We have built this in the mag before, but the original build is an excellent mini-development Raspberry Pi you can take anywhere.

magpi.cc/minidongle

Learn Building and Making

Combine your love of tech with some amazing making, suggests **Rosie Hattersley**

Kev's Robots

AUTHOR

Kevin McAleer

Price: free

kevsrobots.com

We've been having quite the love-in with Kevin recently, featuring his sterling robot-making and MicroPython work in *The MagPi* (see his Cyberdog harness: magpi.cc/cyberdog) and profiling him in a recent HackSpace (magpi.cc/kevinmcaleerhs). Essentially, if you want to learn about how to design and create quirky robots from someone who has learned and blogged as he's gone along, and are also keen to see the possibilities of coding with MicroPython with a view to coming up with your own makes, head to Kevin's website or tune in to his weekly YouTube broadcasts.

Creative project sites

Get your making imagination working overtime

RASPBERRY PI PROJECTS SITES

We're absolutely blowing our own trumpet here: we're very proud of the Projects section of Raspberry Pi's website, where you can search by type of creative project and the experience level that might be needed.

▶ magpi.cc/projects

TOM'S HARDWARE

Our friends at Tom's Hardware regularly feature great Raspberry Pi builds, and their latest rundown of projects worth tackling – and why – is well worth browsing while you decide what you'd like to build and what's involved. We particularly like the Pico-based radio transmitter:

▶ magpi.cc/tomsprojects

INSTRUCTABLES

A super site for creative makes of all stripes, Instructables offers details of how to tackle dozens of Raspberry Pi projects, and even has instructions for teachers, such as this set of TJBot robot resources: **magpi.cc/runtjbot**.

▶ instructables.com

Raspberry Pi Projects For Kids

AUTHOR

Dan Aldred

Price:
£18/$20

magpi.cc/raspiforkids

RASPBERRY PI® PROJECTS FOR KIDS
CREATE AN MP3 PLAYER, MOD MINECRAFT®, HACK RADIO WAVES, AND MORE!
DAN ALDRED

Although this book, published by No Starch, is ideal for getting youngsters to be creative with Raspberry Pi, there's every chance you'll also find something you want to build yourself – whether a radio receiver, a music box, or a wildlife tracker. There's a helpful introduction to Python programming too, which will be invaluable if coding is new to you or your offspring, or you're making the switch from Scratch or another visual coding method. The nicely laid out projects and accessible, but not too simple approach, give it cross-generational appeal, which sounds like the perfect excuse to buy it for the kids, although it's also an informative treat for you.

HackSpace Wearable Tech Projects and The Book of Making 2

AUTHOR

HackSpace

Price:
£10/$10

magpi.cc/wearableprojects

magpi.cc/bookofmaking

HackSpace magazine is devoted to all kinds of making and creativity with two pertinent books: *Wearable Tech Projects* and *The Book of Making Volume 2*. Although these don't focus exclusively on Raspberry Pi, each publication provides plenty of details and inspiration, along with GitHub code downloads so you can replicate each project, or adapt an idea to make it your own. *Wearable Tech Projects* invites you to make, sew, and wear items you've stitched and coded, including replica sci-fi costumes and bling-laden outerwear and jewellery.

50 PROJECTS FOR MAKERS & HACKERS
BOOK OF MAKING
VOLUME 2
MAKE ELECTRONICS CIRCUITS WITH PLAY-DOH
TAKE TO THE SKIES WITH YOUR OWN ROCKET
PLAY GAMES WITH A CUSTOMISED CONTROLLER
CREATE MUSIC ON A HOMEMADE SYNTHESIZER
FROM THE MAKERS OF HackSpace MAGAZINE

Teaching resources

Tasked with helping younger tinkerers achieve? Read on

CODE CLUB

Code Club offers lots of fun project ideas, with full instructions for both makers and anyone helping them achieve their creative goals. Look out for items labelled as part of the Physical Computing Curriculum.

▶ magpi.cc/codeclubprojects

PROJECT ARTHUR

Project Arthur combines paper-building with Raspberry Pi Zero W tech to track the International Space Station and light up when it's overhead. A bonus is that it's named after the satellite dish at Goonhilly in Cornwall, site of the recent space satellite launch attempt.

▶ apollo50.co.uk

MARIA AWARD-WINNING HUMAN

Something to catch the eye of manga fans and other cool customers, Maria is a pseudo-holographic personal assistant to give Alexa a run for her money. As well as being a quirky but engaging Raspberry Pi project, Maria's features can be extended in multiple ways, customising music, weather and gossip feeds, depending on creators' preferences.

▶ maria.awardwinninghuman.com

VEEB

Meet two creatives who admire a minimialist aesthetic and enjoy making useful things from 'older' technology

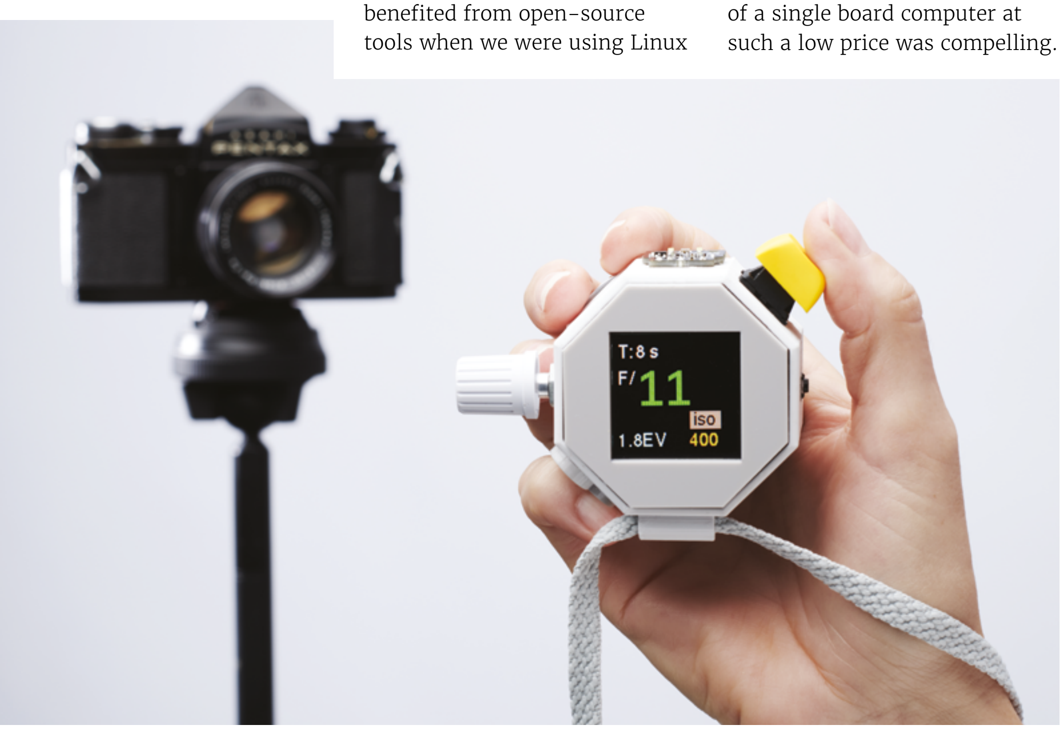
- Name **Vanessa Bradley and Martin Spendiff** | Occupation **Photography**
- Community role **Makers** | URL **veeb.ch**

Have you ever just wanted to get away? Vanessa Bradley and Martin Spendiff did, and a decade ago they moved to Switzerland. After a few years of working in IT for big corporations, they then made another change to photography, with a side of making to go with it.

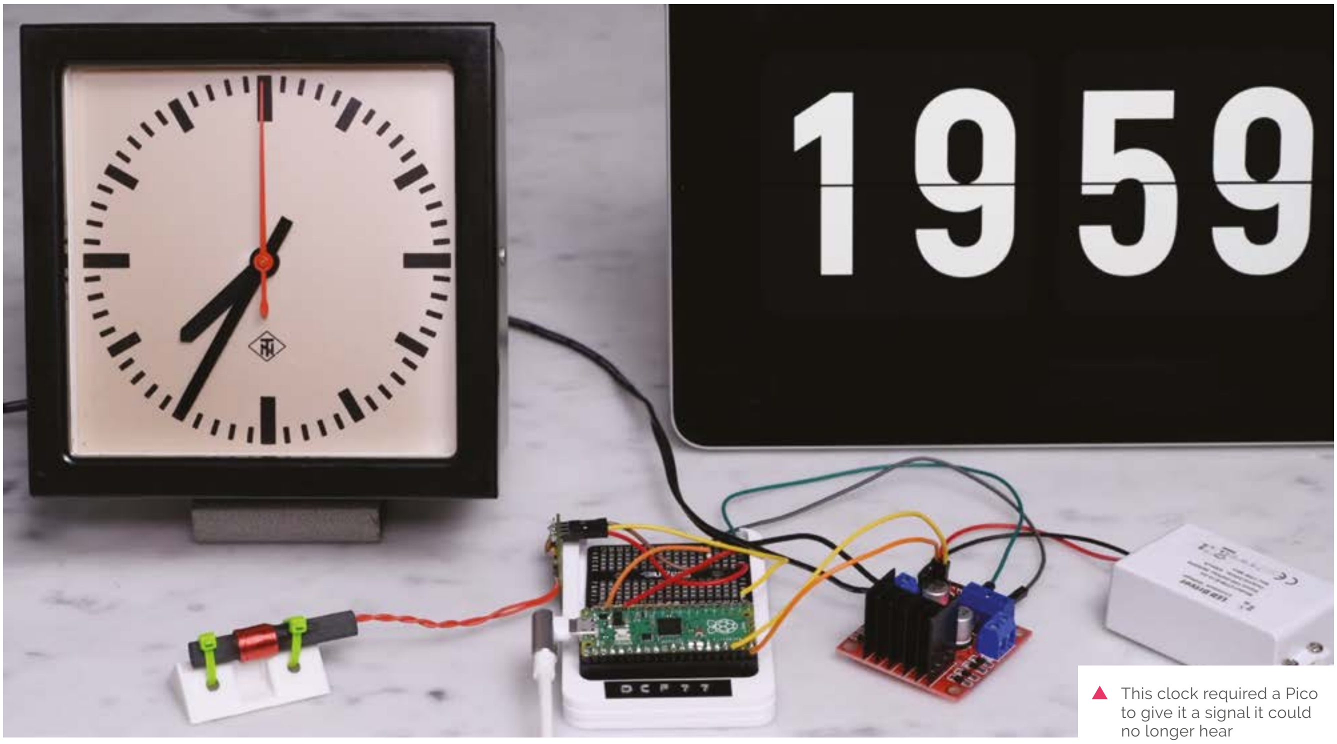
"We've always been early-adopters and interested in getting stuff to do new things that are a touch outside of the originally intended use-case," the two tell us. "We also benefited from open-source tools when we were using Linux wa(aaaaa)y back in the 1990s, so we are keen to share what we do in case anyone finds it useful."

When did you first learn about Raspberry Pi?

In 2012, just from being aware of new things in tech. The idea of a single board computer at such a low price was compelling.



From last issue, Photon got a big response online from photographers



This clock required a Pico to give it a signal it could no longer hear

Since then, they have always been dotted about running various things like Pi-hole, Nextcloud, and RetroPie etc.

Raspberry Pi Pico was a real revelation, as it suddenly made it easy to imagine bits of code living outside of the confines of an operating system that did all sorts of other things.

> The idea of a single board computer at such a low price was compelling

Where do you get your ideas from?

Laziness, a refusal to stop using things that do something well, or a sentence that starts with 'it would be nice if…'. There's also a thread of nostalgia that runs through a lot of things we end up doing.

What is a favourite thing you've made with Raspberry Pi?

The incident light meter, Photon. People occasionally make one and send us photos of them using them to do exotic kinds of photography. An honourable mention has to go to the LED physical progress bar that we made (**magpi.cc/LEDprogress**). It's a super-simple build that uses Google Calendar and a light strip to make something that is really handy to glance at. It's now an indispensable feature in our studio.

Where can people follow your stuff?

Most things we do end up going into a YouTube video (**magpi.cc/ytveeb**), as well as onto GitHub (**github.com/veebch**).

In a very Internet 1.0 way, we also add things to our own 'blog' (**veeb.ch/notes**) (do people even say 'blog' any more?)

A simple, but useful, solution for making sure that your plant is watered

The first project that we showcased from the two in the magazine, an automation throwback

MagPi Monday

Amazing projects direct from our Twitter!

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 have made.

Here's a selection of some of the awesome things we got sent this month – and remember to follow along at the hashtag #MagPiMonday!

01. Better check your scarf to see if you need to wear it. Let's hope it doesn't lie so you don't leave it at home

02. This tiny computer is very cute. We kinda want one

03. We love a good Raspberry Pi Camera Module project – this is definitely up there

04. This has to be seen to be appreciated – unfortunately headphones won't make it any less noisy

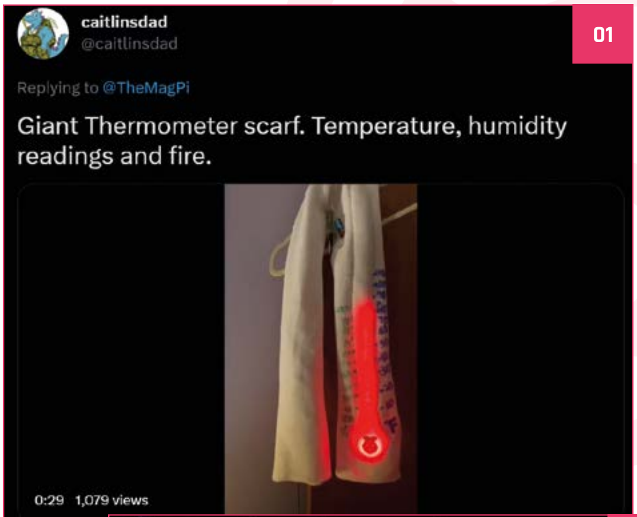
05. We always want to see pictures of critters and creatures caught with Raspberry Pi

06. Archie the owl has some competition it seems

07. While there were some Super Mario Bros. Arcade machines, they weren't quite as cool as this one

08. We already have project plans ourselves for next Christmas...

09. We like how compact this robot is



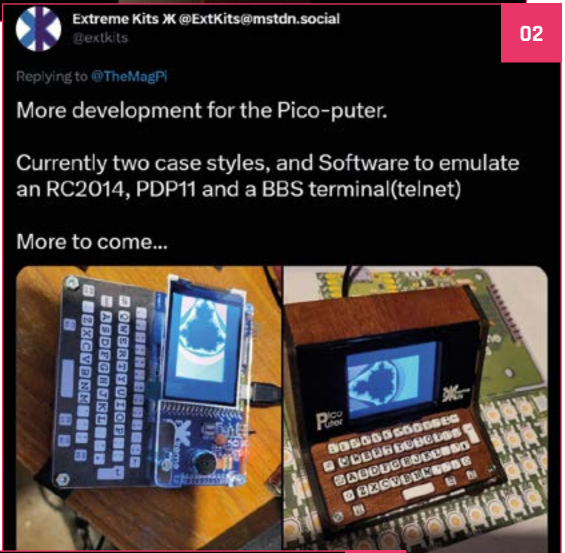
01

caitlinsdad
@caitlinsdad

Replying to @TheMagPi

Giant Thermometer scarf. Temperature, humidity readings and fire.

0:29 1,079 views



02

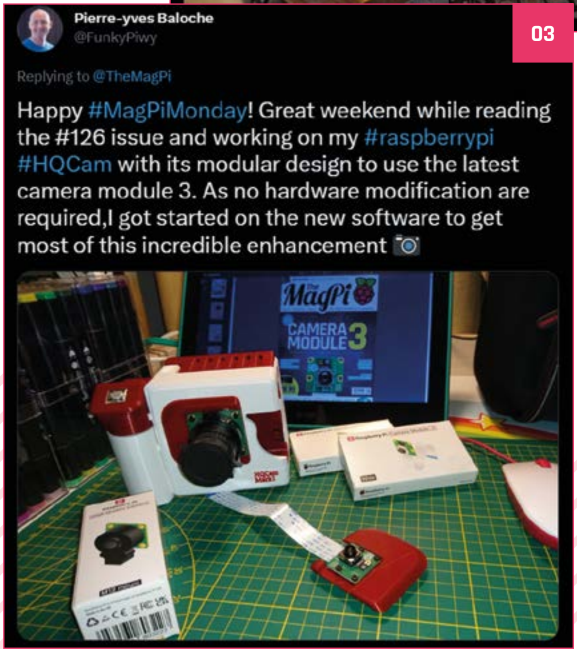
Extreme Kits ЖK @ExtKits@mstdn.social
@extkits

Replying to @TheMagPi

More development for the Pico-puter.

Currently two case styles, and Software to emulate an RC2014, PDP11 and a BBS terminal(telnet)

More to come...

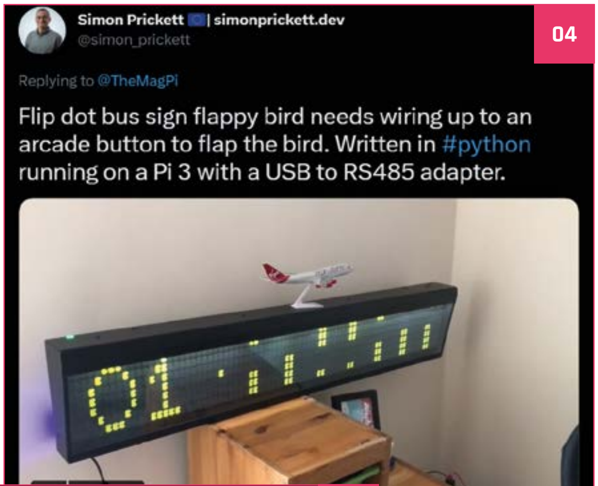


03

Pierre-yves Baloche
@FunkyPiwy

Replying to @TheMagPi

Happy #MagPiMonday! Great weekend while reading the #126 issue and working on my #raspberrypi #HQCam with its modular design to use the latest camera module 3. As no hardware modification are required,I got started on the new software to get most of this incredible enhancement 📷

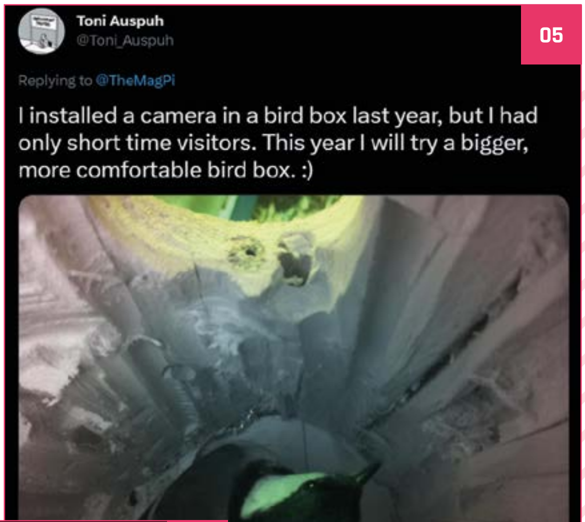


04

Simon Prickett 🇪🇺 | simonprickett.dev
@simon_prickett

Replying to @TheMagPi

Flip dot bus sign flappy bird needs wiring up to an arcade button to flap the bird. Written in #python running on a Pi 3 with a USB to RS485 adapter.

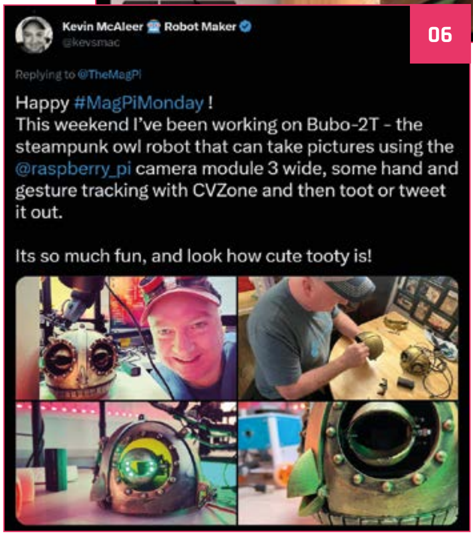


05

Toni Auspuh
@Toni_Auspuh

Replying to @TheMagPi

I installed a camera in a bird box last year, but I had only short time visitors. This year I will try a bigger, more comfortable bird box. :)



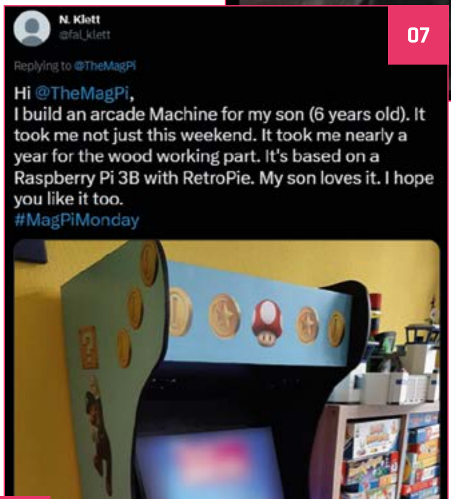
06

Kevin McAleer 🤖 Robot Maker
@kevsmac

Replying to @TheMagPi

Happy #MagPiMonday !
This weekend I've been working on Bubo-2T - the steampunk owl robot that can take pictures using the @raspberry_pi camera module 3 wide, some hand and gesture tracking with CVZone and then toot or tweet it out.

Its so much fun, and look how cute tooty is!

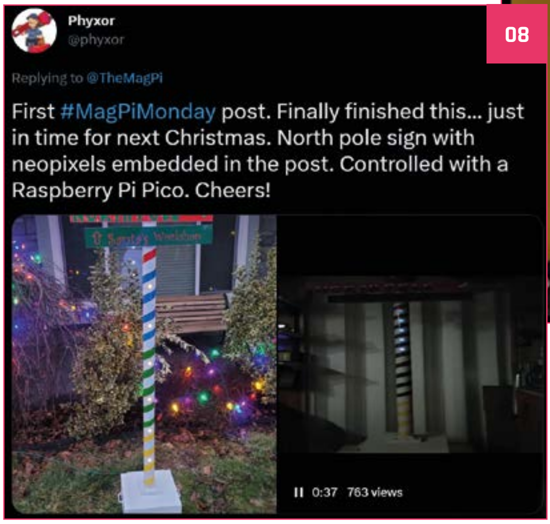


07

N. Klett
@fal_klett

Replying to @TheMagPi

Hi @TheMagPi,
I build an arcade Machine for my son (6 years old). It took me not just this weekend. It took me nearly a year for the wood working part. It's based on a Raspberry Pi 3B with RetroPie. My son loves it. I hope you like it too.
#MagPiMonday

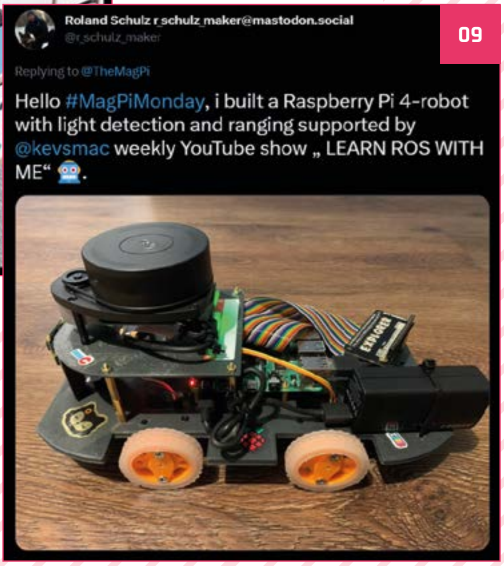


08

Phyxor
@phyxor

Replying to @TheMagPi

First #MagPiMonday post. Finally finished this... just in time for next Christmas. North pole sign with neopixels embedded in the post. Controlled with a Raspberry Pi Pico. Cheers!



09

Roland Schulz r.schulz_maker@mastodon.social
@r_schulz_maker

Replying to @TheMagPi

Hello #MagPiMonday, i built a Raspberry Pi 4-robot with light detection and ranging supported by @kevsmac weekly YouTube show „ LEARN ROS WITH ME“ 🤖.

CamJam returns!

Cambridge's premier Raspberry Jam is back this April

Great news folks! As the world starts to get a little more open, Raspberry Jams and events are making their way back to physical spaces, and this includes the venerable CamJam (camjam.me) on Saturday 22 April.

The MagPi regular Brian Corteil has taken the reigns for its return and promises that "There will be a series of talks, organised workshops for both the Raspberry Pi and Pico, plus a 'show and tell' area for people to show off their projects. We are fully inclusive of technology: if your project is based around an Arduino, micro:bit, or something else, we would still love to see it!"

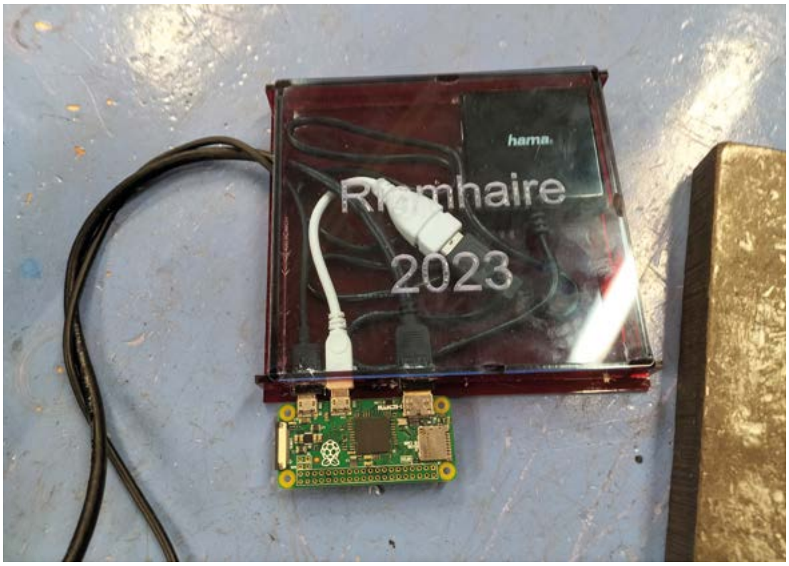
It's being held at Makespace, a fantastic community workshop in Cambridge, and kids enter free. Find out more info at the Eventbrite page here: **magpi.cc/camjam23**.

▶ CamJam is one of the oldest Raspberry Jams, running in the home of Raspberry Pi

Affordable bilingual computing

Student Conall Kavanagh presents a solution for Irish and English speakers

▼ Raspberry Pi and Raspberry Pi OS can make for cheap computing for bilingual learning



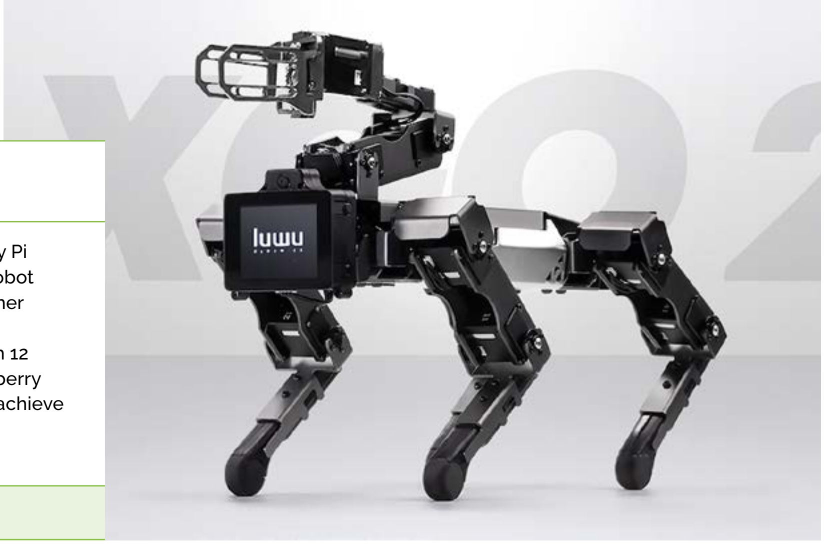
"My hypothesis is that there are serious failings in the provision of technology in the Irish second-level education sector," reads Conall's report 'Low-cost bilingual computing for the education sector'. He presented his project at the BT Young Scientist and Technology Exhibition (**btyoungscientist.com**).

"Much of the technology that is recommended for use is extremely expensive," Conall continues. "This makes it unaffordable for many Irish families, and puts some students at a severe disadvantage compared to more affluent peers."

Conall presented a specialised version of Raspberry Pi OS on a Raspberry Pi Zero 2 W as part of his solution to the problem, complete with a compact dock.

Crowdfund **this!**

Raspberry Pi projects you can crowdfund this month

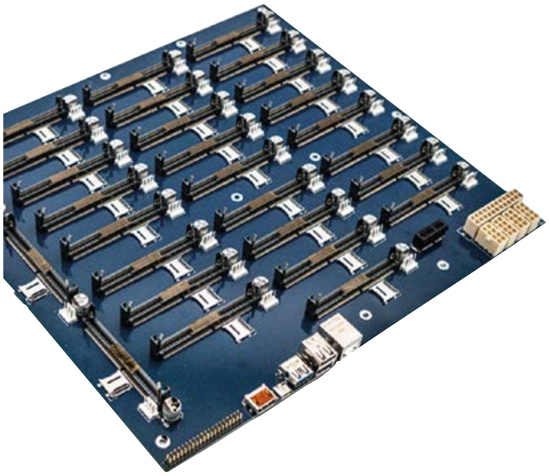


XGO 2

According to the website, XGO 2 is the first Raspberry Pi robotic dog with an arm – and by that they mean a robot arm attached behind the head, along with its four other robotic limbs.

"XGO 2 is a desktop-sized AI quadruped robot with 12 degrees of freedom and a gripper powered by Raspberry CM4 for AI edge computing applications, which can achieve omnidirectional movement, six-dimensional posture control, posture stability, and multiple motion gaits."

▶ **kck.st/40fpLzG**



ARMedOne

The ARMedOne is a compact and modular carrier board for up to 28 compute modules, using a standardized E-ATX form factor. The Kickstarter page claims it can handle every type of Compute Module which uses SODIMM-Pins, which includes Raspberry Pi Compute Modules, among others.

▶ **kck.st/3WT29hS**

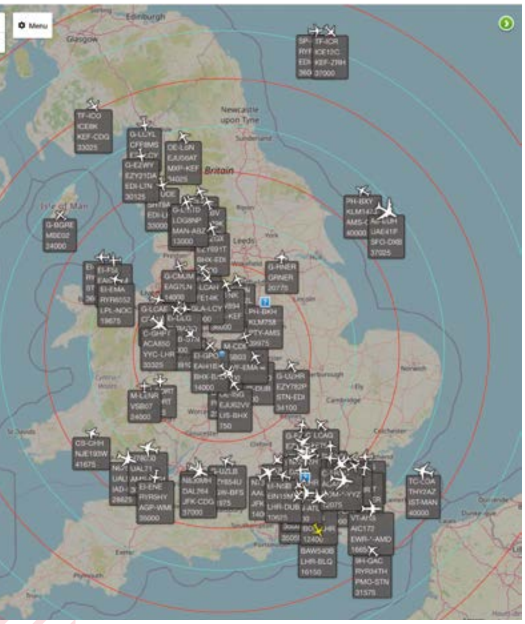
Your Letters

Top antenna tips

I was interested to see your article on flight tracking.

This is something I've been experimenting with for about four years.

You mention, as a top tip, that antenna position is key. My experiments prove this. Having a clear line of sight, and a good antenna is very important to pick up ADS-B signals… with a general purpose discone antenna on my garden shed, with 5 m of cable between the antenna and the SDR. It manages to pick up less than 20 aircraft, with a maximum range of 120km to the north and 50km to the south.



▲ Antenna placement is very important, as Ian points out

The screenshot [pictured] is using an ADS-B specific antenna, at rooftop height and with 1 m of cable between the antenna and SDR. The SDR is mounted in a waterproof box on the antenna pole, and connects to an indoor Raspberry Pi via a 10 m active USB cable. This picks up over a 100 aircraft, with a maximum range of 400km.

Living near Birmingham, I can 'see' aircraft over a large part of the country, as long as they are above the horizon. A plane at 40,000 feet has a radio horizon of about 450km. So, I can see planes flying over London, but not landing at Heathrow because they're below the horizon.

Another part of my experimentation has been to use a second Raspberry Pi to take screenshots every few seconds, then stitch them together to make a time-lapse movie. It was very interesting to see the effect of the pandemic on flight movements. Nearly all passenger planes stopped, but there were still a large number of freight flights taking place.

Rather than use Flightradar24, or other packages that send data to a cloud-based server, I keep the data in-house. I use Dump1090 (I believe Flightradar24, and most of the other packages, use Dump1090 internally) to control my SDR USB stick, and use Virtual Radar Server to generate the map, with aircraft plotted on it using Dump1090's data stream. Both packages are on the same Raspberry Pi. I can then web-browse to my Pi from any device on my home network to see what's in the air.

I find it quite interesting to see what's up there, and I've learnt a few things. For example, East Midlands Airport is very busy at night with freight aircraft coming in from the USA. I used to think it was mainly summer holiday flights.

Ian via email

Thank you for your extra information about this! Keeping the data in-house is a great idea, and the time-lapse is a cool concept too. Hopefully, any readers who read the original article, and were inspired to make their own, can use this knowledge to do some extra bits to theirs.

▲ We interview the makers of the excellent Photon on page 86

PDF error

I think there's an error in issue 126 that I'm reading now.

In the Contents, it says project PHOTON on page 26 (I'm very interested because photography is one of my greatest interests), but on this page the project is 'Raspberry Pi Backpack', and Photon does not appear in the magazine.

Roberto via email

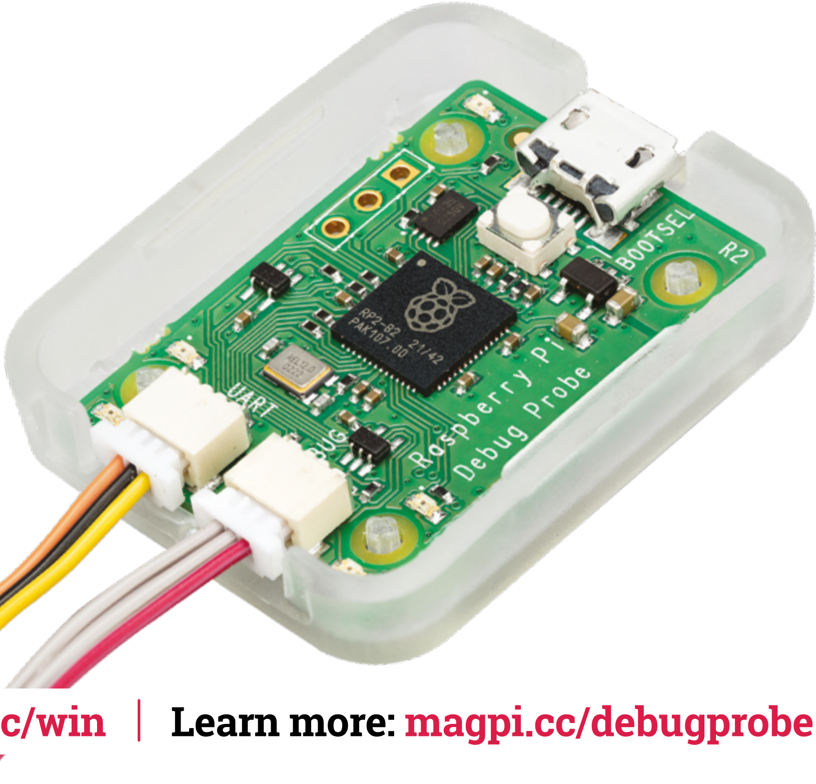
There was a slight error with the PDF for issue 126 when we first published the magazine, which has since been updated. Print issues have the project on the correct page, but if you downloaded the PDF in the first couple of days, you may need to re-download it.

Contact us!

- Twitter **@TheMagPi**
- Facebook **magpi.cc/facebook**
- Email **magpi@raspberrypi.com**
- Online **forums.raspberrypi.com**

WIN 1 OF 10 RASPBERRY PI DEBUG PROBE

The new Raspberry Pi Debug Probe is built on RP2040 and can help you to debug your Raspberry Pi Pico projects in a way that a computer lacking GPIO pins cannot. We have ten of the probes to give away.

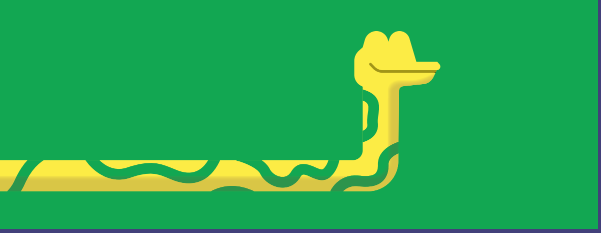


Head here to enter: magpi.cc/win | **Learn more: magpi.cc/debugprobe**

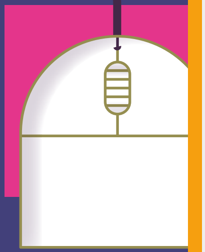
Terms & Conditions

Competition opens on **22 February** and closes on **30 March**. 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 The MagPi 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 or any other companies used to promote the service.

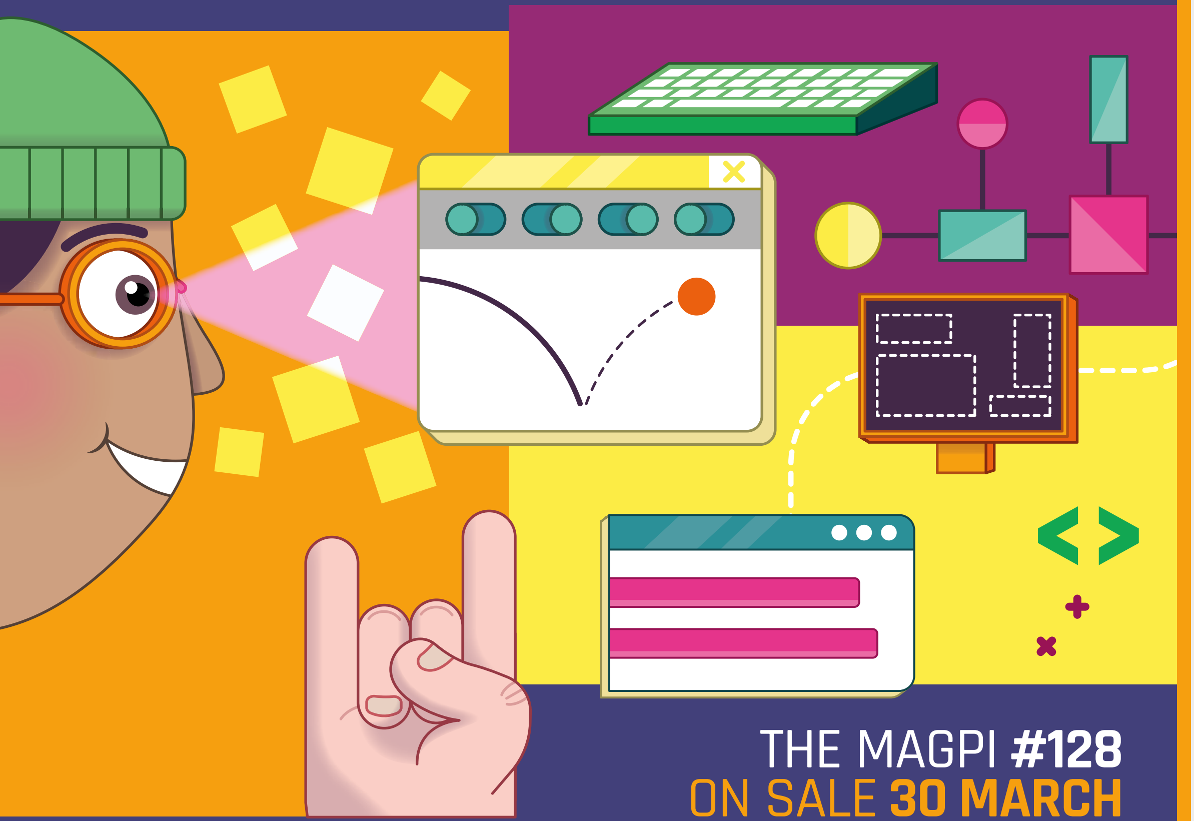
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
NEW 2022 UPDATE
RETRO GAMING WITH RASPBERRY PI
2ND EDITION
164 PAGES OF VIDEO GAME PROJECTS
BUILD AN ARCADE MACHINE



LEARN TO CODE WITH PYTHON



GET STARTED WITH COMPUTER PROGRAMMING



THE MAGPI **#128**
ON SALE **30 MARCH**

Plus!

Astrophotography with Raspberry Pi

Build a machine-learning digital transcriber

Your introduction to microcontrollers

DON'T MISS OUT!
magpi.cc/subscribe

TWITTER	@TheMagPi
FACEBOOK	fb.com/MagPiMagazine
EMAIL	magpi@raspberrypi.com

EDITORIAL

Editor
Lucy Hattersley
lucy@raspberrypi.com

Features Editor
Rob Zwetsloot
rob@raspberrypi.com

Sub Editor
Nicola King

ADVERTISING

Charlotte Milligan
charlotte.milligan@raspberrypi.com
+44 (0)7725 368887

DESIGN

criticalmedia.co.uk

Head of Design
Lee Allen

Designers
Olivia Mitchell, Ty Logan, Sam Ribbits

Illustrator
Sam Alder

CONTRIBUTORS

David Crookes, Ben Everard, Rosemary Hattersley, Nicola King, Phil King, KG Orphanides, Nik Rawlinson, Stewart Watkiss

PUBLISHING

Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com

Director of Communications
Liz Upton

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
magpi.cc/subscribe
magpi@subscriptionhelpline.co.uk

recycle When you have finished with this magazine, please recycle it.

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.

The MagPi magazine is published by Raspberry Pi Ltd, Maurice Wilkes Building, St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS. 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.

Raspberry Pi PRESS

ISSN: 2051-9982.

Artificial Artists

Can machine-learning technology can be put to better use? By **Lucy Hattersley**

Artificial Intelligence is still in the news, and for good reason. There are incredible strides appearing in the world of computer-generated images, video and text. Generative art and video is based upon an open-source technology called Stable Diffusion, which can create any artwork you desire based on text prompts. You can test it out online at **stablediffusionweb.com** if you want to paint the cheese moon landing, or the distracted boyfriend meme by Picasso. The open-source Stable Diffusion 2 GitHub code is found at **magpi.cc/stablediffusiongit**.

If you think artists have it bad, then spare a thought for your lonely writers. We're staring down the barrel of ChatGPT and Google's response, Bard.

The creative mill

As Rob pointed out last month (**magpi.cc/126**), ChatGPT's copy is generously described as 'hit-and-miss'. A more accurate assessment would be: "dumber than a tall person hopping at the very peak of Dunning-Kruger's mount stupid" (**magpi.cc/mountstupid**). It is convincing though and a lot of online copy is equally misguided. Which may be the point. Getty Images is suing Stable Diffusion for $1.8-trillion because: "It is Getty Images' position that Stability AI unlawfully copied and processed millions of images protected by copyright."

Beyond copyright and clarity, there are legitimate ethical concerns. Since last month tech wizards with flexible morals have already got around safeguards by getting ChatCPT to role-play as a different AI called 'DAN' which stands for 'Do-anything-Now'.

AI can create anything as dark as the mind can conjure. A quick traipse through the internet will reveal some very dark thoughts indeed.

> “ Machines should work; People should think ”

Positive thoughts

It's easy to knee-jerk against this advancement. I love what I do, but technology should replace people. I can quite happily sit on a beach and read poetry to my friends while computers do all the work.

The IBM/Jim Henson's Machines Should Work; People should think speech from 1967 comes to mind (**magpi.cc/machinesshouldwork**). Computers are supposed to take out the trash, clean the floor, do the dishes and the washing up and we (the humans) are supposed to sit on the beach and share our stories, make up music, and paint pictures.

The future of machine-learning technology is, I suspect, in more mundane, but ultimately useful, applications. The robot vacuum cleaner that picks up the socks from your 'floordrobe' and puts them back in a drawer (or in the wash) is ultimately far more valuable to us than one that paints selfies.

Of course, that machine would probably need wheels and arms and a mechanical nose alongside the AI smarts to know what scruffy socks look like. Oh and look, scientists from Tel Aviv are working on a robot with a biological nose (**magpi.cc/bionose**). Of course, it has Raspberry Pi inside.

This is the real stuff of the future. Where hardware and software and robotics and AI-smarts come together to do useful stuff while we all chill.

AUTHOR

Lucy Hattersley

Lucy is editor of *The MagPi* and currently getting AI to paint a picture of her cat fetching her a cup of tea in the morning.

magpi.cc/lucyhattersley

PiKVM
VIDEO OUT
ACT
CON
PWR