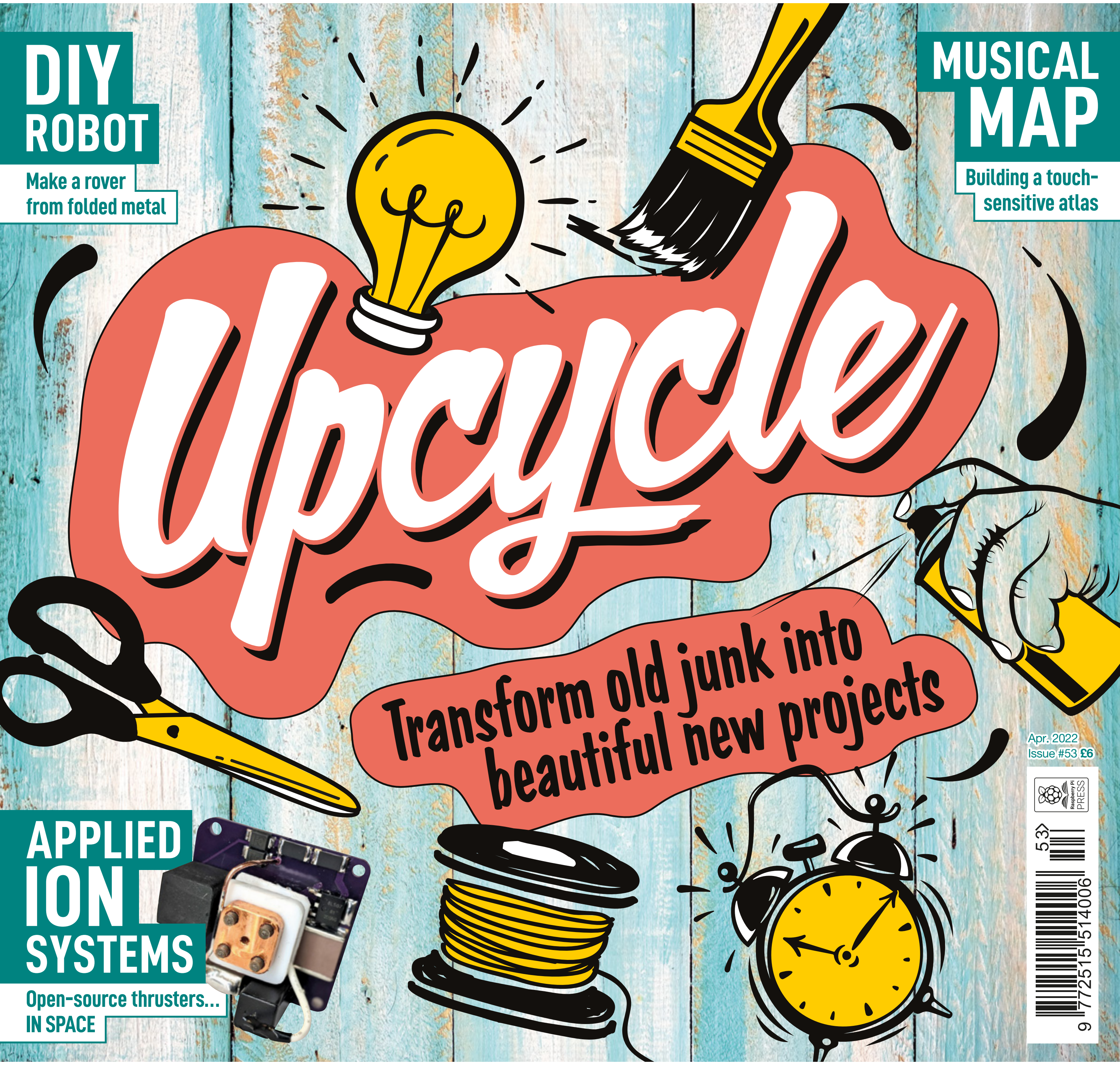
MAKE | **BUILD** | HACK | **CREATE**

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc | April 2022 | Issue #53



DIY ROBOT
Make a rover from folded metal

MUSICAL MAP
Building a touch-sensitive atlas

Upcycle

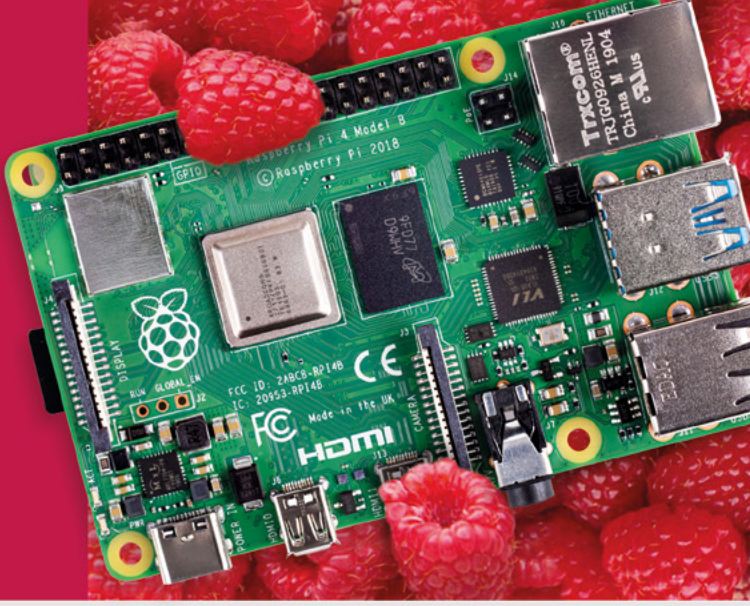
Transform old junk into beautiful new projects

APPLIED ION SYSTEMS
Open-source thrusters... IN SPACE

Apr. 2022
Issue #53 £6

LEDS ANALOGUE **R2040** KEYBOARDS

American
Raspberry Pi
Shop



- Displays
- HATs
- Sensors
- Cases
- Arcade
- Swag
- Project Kits
- Cameras
- Power Options
- Add-on Boards
- Cables and Connectors
- GPIO and Prototyping

Partner and official reseller for top Pi brands:

adafruit sparkfun ELECTRONICS PIMORONI HiFiBerry and many others!

Price, service, design, and logistics support for **VOLUME PROJECTS**

USA PiShop.us

Canada PiShop.ca

Raspberry Pi APPROVED RESELLER

Welcome to HackSpace magazine

We, as humanity, are using too many raw materials. It is simply unsustainable for us to keep producing, using, and disposing of things at the rate we currently do. We, as makers, have to be aware of our place in the wider world, and our responsibilities to the generations that follow. While it's true that the most significant part of this consumption comes from factory-made goods, we makers have to choose whether we want to be part of the problem or part of the solution. Here at HackSpace mag, we want to be part of the solution. This doesn't mean not making cool stuff, but it does mean being mindful of the amount of virgin material we use. It means making stuff that will stand the test of time, and it means reusing or repairing things where possible.

It is simply unsustainable for us to keep producing, using, and disposing of things **at the rate we currently do**

This issue, we're looking at the craft of upcycling. That means upgrading or reusing existing objects in our builds which minimises the use of new material and gives us some cool-looking parts as an added bonus. Let's make cool stuff, responsibly.

BEN EVERARD
Editor ben.everard@raspberrypi.com

Got a comment, question, or thought about HackSpace magazine?

get in touch at **hsmag.cc/hello**

GET IN TOUCH

hackspace@raspberrypi.com
hackspacemag
hackspacemag

ONLINE

hsmag.cc

Available on the App Store

GET IT ON Google Play

PAGE 42
FREE PICO WHEN YOU SUBSCRIBE

EDITORIAL

Editor
Ben Everard
ben.everard@raspberrypi.com

Features Editor
Andrew Gregory
andrew.gregory@raspberrypi.com

Sub-Editors
David Higgs, Nicola King

DESIGN

Critical Media
criticalmedia.co.uk

Head of Design
Lee Allen

Designers
Sam Ribbits, Olivia Mitchell, Ty Logan

Photography
Brian O'Halloran

CONTRIBUTORS

Jo Hinchliffe, Marc de Vinck, Rob Miles, Andrew Lewis, Rosie Hattersley, Phil King, Mike Bedford, Ian Dunn

PUBLISHING

Publishing Director
Russell Barnes
russell@raspberrypi.com

Advertising
Charlie Milligan
charlotte.milligan@raspberrypi.com

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

To subscribe
01293 312189
hsmag.cc/subscribe

Subscription queries
hackspace@subscriptionhelpline.co.uk

This magazine is printed on paper sourced from sustainable forests. The printer operates an environmental management system which has been assessed as conforming to ISO 14001.

Contents

68

06 **SPARK**

06 **Top Projects**
Some really nice things to grab your attention

18 **Objet 3d'art**
A 3D-printed robot that's smarter than we are

20 **Meet the Maker:** Timon Skerutsch
Turning a successful crowdfunder into a product

26 **Letters**
Your take on these pages

31 **LENS**

32 **Upcycle**
Turn junk into something you can admire/use/sell

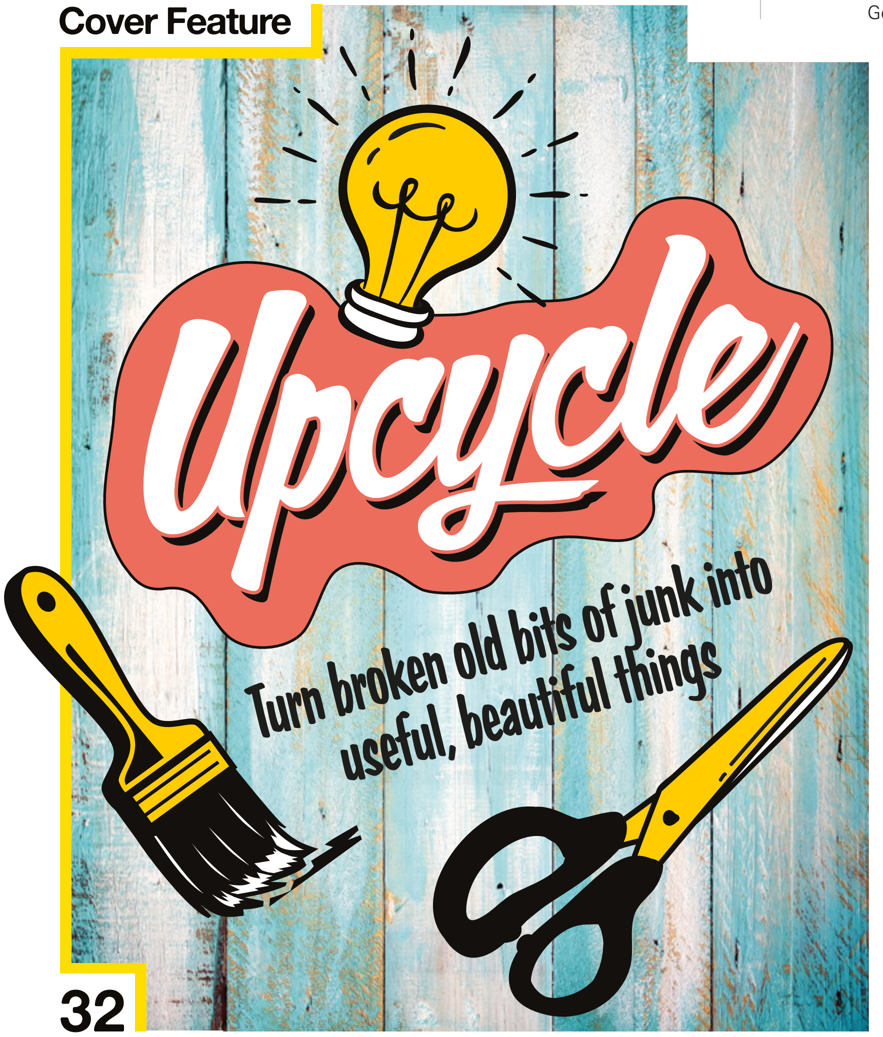
44 **How I Made:** Mechanical Pico keyboard
Satisfying key presses with a Pico controller

50 **Interview:** Michael Bretti
Open-source space thrusters. We can't believe it either

58 **Improviser's Toolbox** Tyres
Turns out, rubber and steel are good for making things

62 **In the workshop** Musical map
Geography, meet music. Music, meet capacitive touch…

Cover Feature



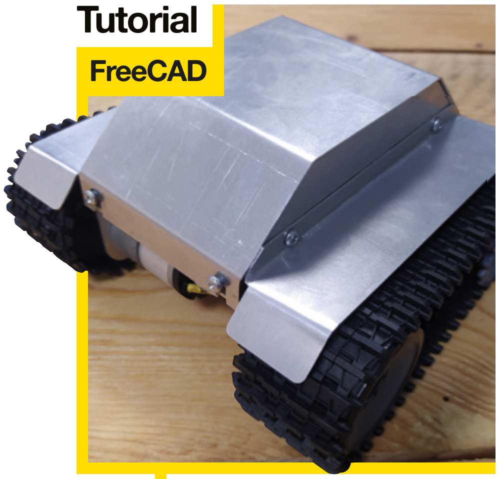
32

Tutorial
Laser cutter

88 Optics: lenses and lasers to carve through hot plywood

44

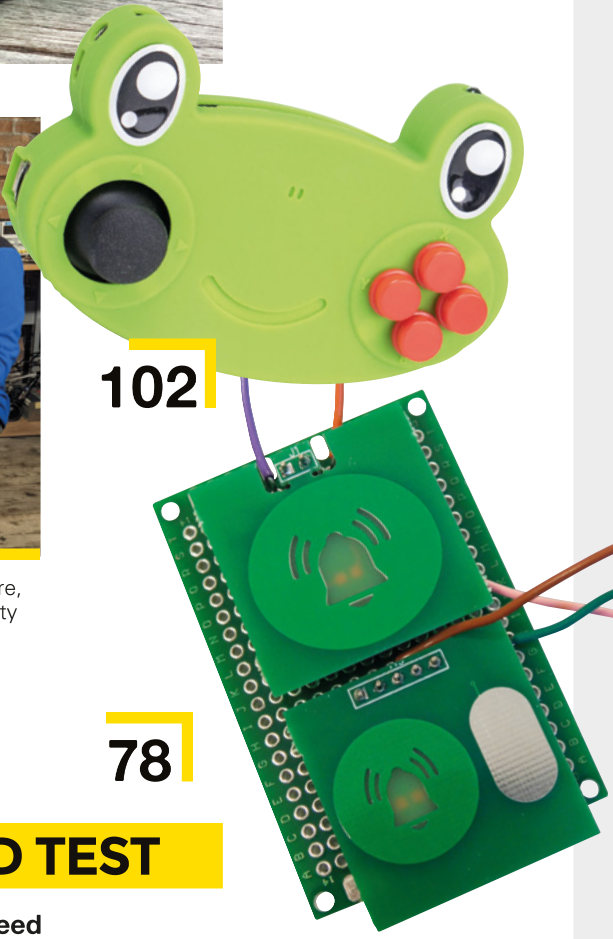
Tutorial
FreeCAD



94 Design and build a robot out of immutable metal

Interview
Michael Bretti

50 Open-source space hardware, developed for the community



67 **FORGE**

68 **SoM** Pico Chord keyboard
Turn key presses and code into letters and words

74 **SoM** Sensory world
Make a system to monitor your plants

78 **Tutorial** PCBs
Integrate buttons with PCBs

82 **Tutorial** 74HC
Back to basics with logic chips

88 **Tutorial** Laser cutter
Get the optics and laser right

94 **Tutorial** FreeCAD
Build a metal chassis for a robot rover

101 **FIELD TEST**

102 **Best of Breed**
Rediscover the power of the micro:bit

108 **Review** Prusa MINI+
An excellent mid-level printer that won't break the bank

112 **Review** Solder Party RP2040 Stamp
Everything you need for an RP2040 project in 25 mm × 25 mm

Teak Bluetooth Speaker

By SimranWasu

hsmag.cc/TeakSpeakers

This beautiful object by SimranWasu would be nice enough to have in your home even if it didn't do anything. It's a Bluetooth speaker/amplifier, made of CNC teak hardwood and CNCed aluminium. Upgrading an obsolete piece of hardware, such as a radio, by adding Bluetooth capability to it is nothing new – and it's produced some lovely builds that we've featured in these pages – but the thing that sets this build apart is that it's been done from scratch so, so beautifully.

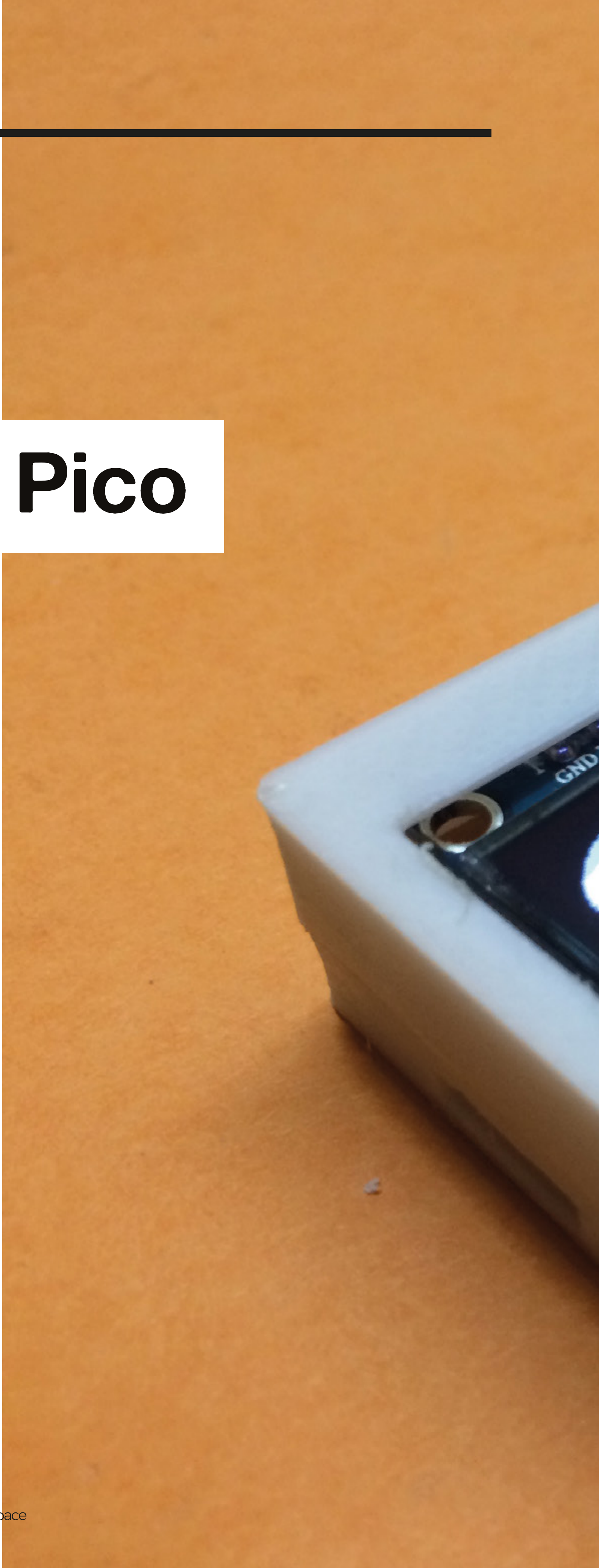
Left
You can't see it from the finished image, but the sandwiched aluminium layer is just as lovely as the wooden exterior

Raspberry Pi Pico calculator

By Makertronics

hsmag.cc/PicoCalc

It's a cliché that the team that landed the first spacecraft on the moon did so using no more computing power than you can get in a modern calculator. One of the first things students do when they're learning to program is to build calculator applications, so why not take that one step further and build an actual, physical calculator, like Anil has done, here? The bill of materials is small: just a Raspberry Pi Pico, some surface-mount buttons, an OLED display, a LiPo battery and charge circuitry, and a custom, single-layer PCB. Wrap it all up in a 3D-printed case and you've got a great physical computing build. □



Right On startup, the PiCalc displays a logo on the OLED screen

±
π
C CE
+
−
=
3
6
X
2
9
/
1
5
.
4
8
0
7
S
VDD SCK SDA
πCalc

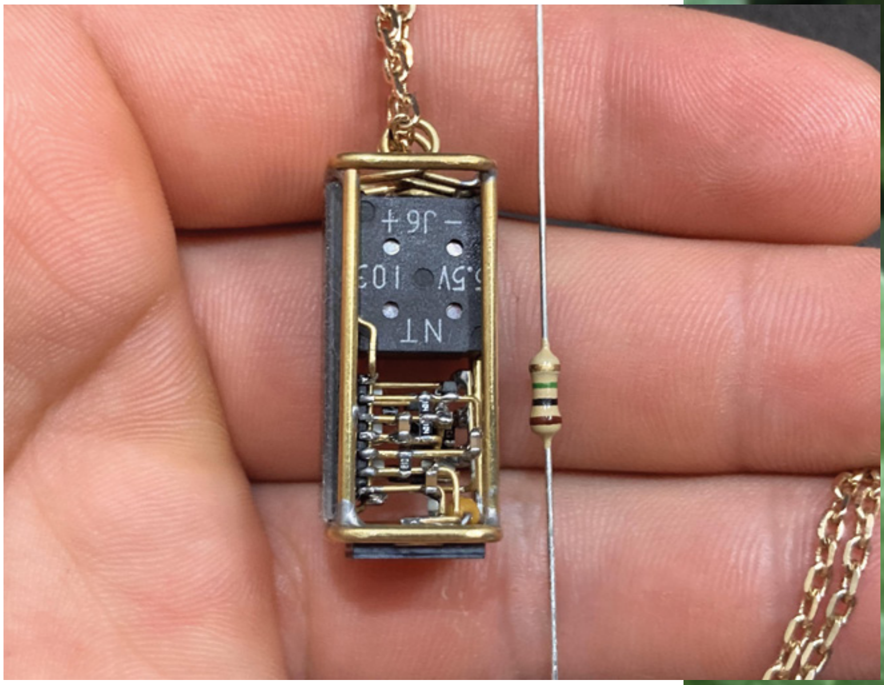
REGULAR

Chirping bird pendant

By **Jack Spiggle** hsmag.cc/ChirpingBirdPendant

This tiny free-form soldering circuit by Jack Spiggle (aka NanoRobotGeek) is a masterpiece of miniaturisation. The key to this build is the KXOB25-02X8F-TB solar panel, which generates 5.5 volts in the bright Australian sunshine. As with Jack's other builds, the solar panel charges a capacitor, which, when it's full, activates the circuit – in this case, that's four oscillators making a 'chirp' sound via a piezo speaker.

We can only imagine how hard it must be to perform free-form soldering on a circuit this small and get it so neat (Jack says that working with brass is a headache, as it conducts heat away from solder so quickly). □



Right The through-hole resistor for scale shows you just how precise the soldering work is here

J6
5.5V
NT

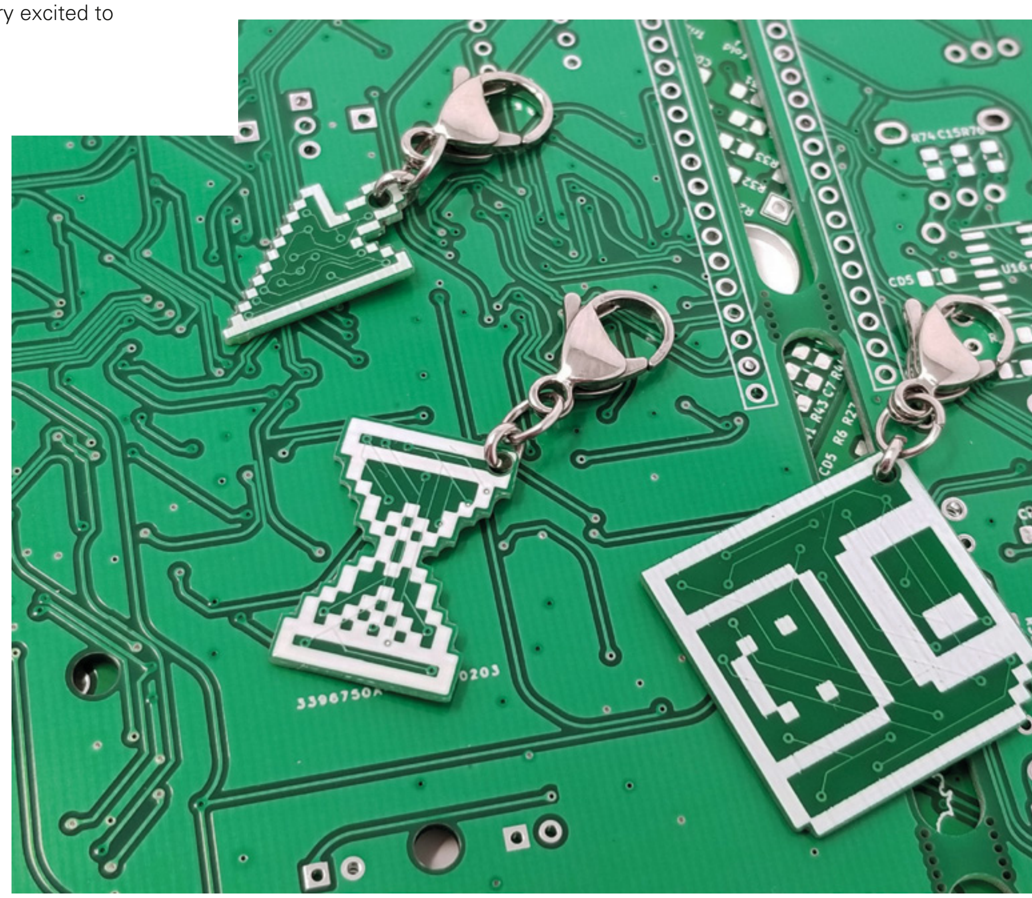
Hardware charms

By **Happy Hardwear** hsmag.cc/HappyHardwear

We found these PCB charms on Tindie, and we liked the look of them, so we asked the creator to tell us about them. Here's what they said:

"Happy Hardwear is a new company making original, tech-inspired wearables out of circuit boards. We were looking for another creative outlet and started making pixel art, inspired by our nostalgia for early internet culture.

"We tried placing our art on circuit boards in KiCad and quickly realised it was a perfect fit for jewellery and accessories. We've been thrilled with the results so far, and are very excited to keep growing!"



Right Sometimes, when we've been staring at a screen for too long, reality only seems real when it's pixelated

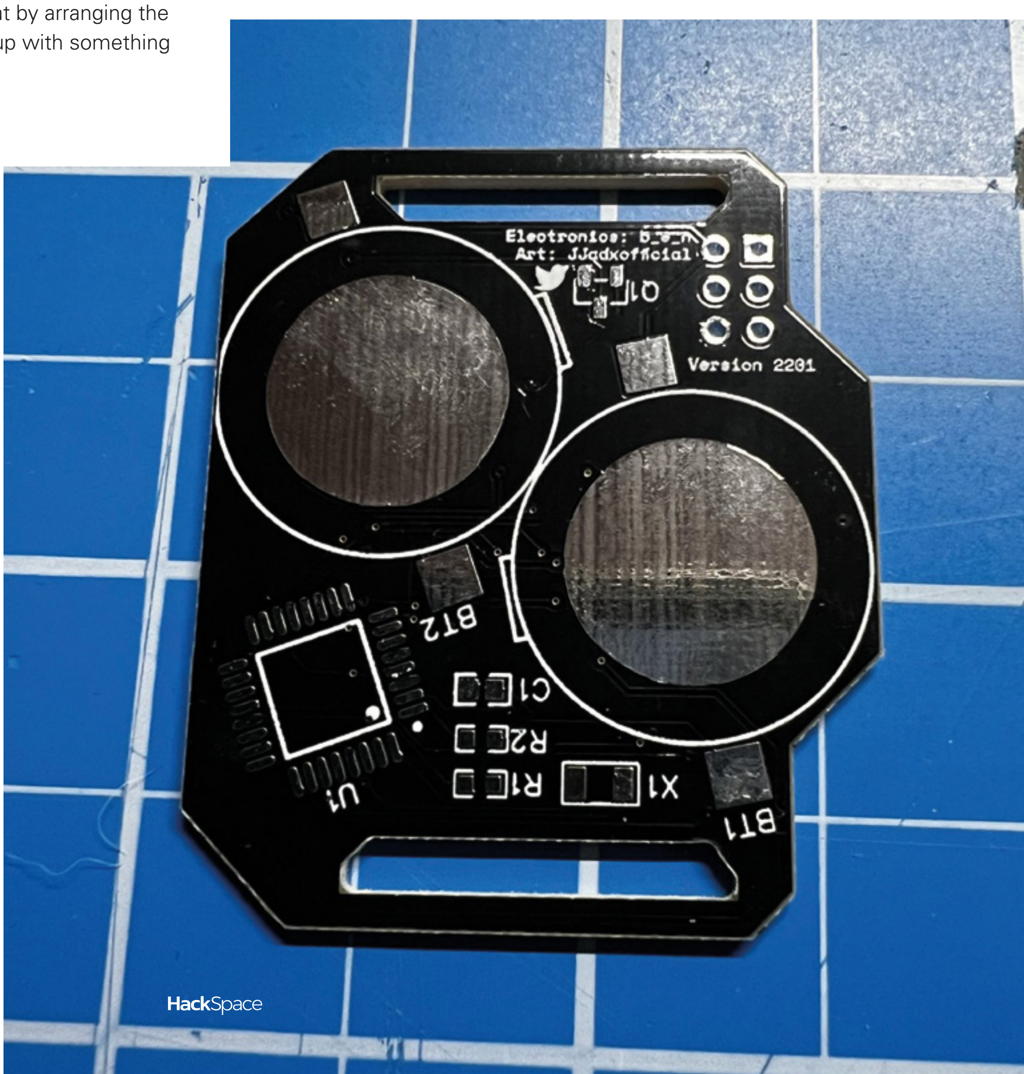
Cyberpunk watch

By Bennett Warner

hsmag.cc/CyberpunkWatch

Cyberpunk 2077 had a mixed reception when it was released to an eager gaming public in late 2020. One thing that united opinion though was the game's incredible aesthetics. In the best traditions of *Blade Runner*, *Alien*, and *Red Dwarf*, the world it portrayed was simultaneously futuristic and run-down, grubby and seedy, and neon-lit. And that's why fans loved it.

One such fan is Bennett Warner, who made this watch crafted in the style of the game's Arasaka Corporation. We love it because it looks so original: who would have thought that by arranging the digits in a square, rather than a line, you'd end up with something that looks so unfamiliar?



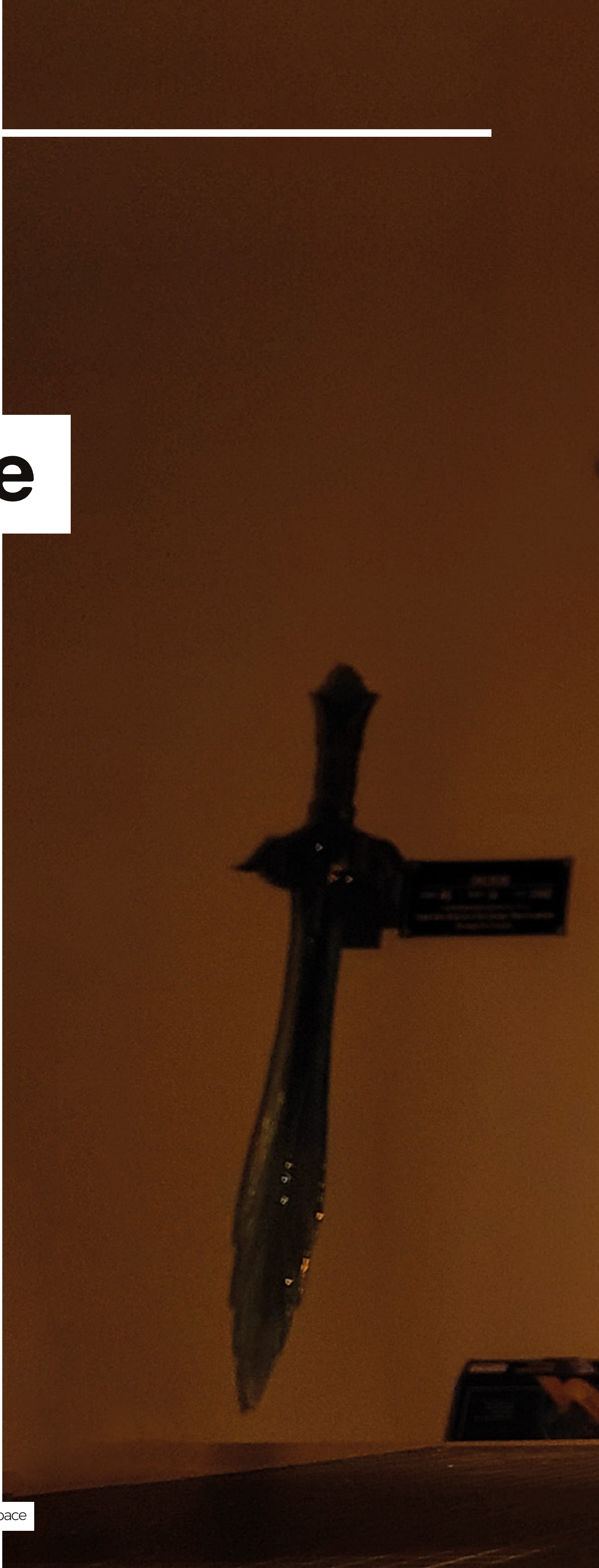
Right If you don't have Bennett's skills with PCB design, you can buy one of these premade from the Tindie store

10
35
UNIT CHARGE
arasaka
20771

Discovery Space Shuttle

By When Geeks Craft   hsmag.cc/GlassShuttle

Before it became a venue for billionaire show-offs, space used to be very, very cool. The US Space Shuttle was, along with the ISS, the crowning achievement of that golden age of space hardware. Fatter than an aeroplane, it always looked to this writer as though it shouldn't be able to fly at all. It took a series of rockets to get it into space, delivered its payload, and would then fall back to Earth, gliding down to the runway through the atmosphere, covered in heat-resistant tiles that I told myself were exactly the same as the ones in our kitchen. It's an amazing thing, and worth recreating any way we can. This glass version of the Space Shuttle is no less awesome; it started life as a papercraft pattern that was transferred to pieces of coloured glass. The edges were covered in copper tape, and then soldered together. If you've fallen in love with slow, deliberate, lovingly made build videos over lockdown, check out the accompanying video from When Geeks Craft – it's superb. □



Right Space used to be cool – and it can be cool again, as you'll see in our interview on page 50

Objet 3d'art

3D-printed artwork to bring more beauty into your life

Behold the nanosaur by Raffaello Bonghi, a 3D-printed, autonomous, tracked robot. Rather than the more familiar Raspberry Pi, the nanosaur uses an NVIDIA Jetson Nano, NVIDIA's platform for embedded AI. This means that if you build your own nanosaur (there are clear, detailed instructions on Raffaello's site: **nanosaur.ai**), you'll be able to program it to pootle around your desktop avoiding the edges, follow lines, and whatever else you can come up with using a camera and some insanely powerful consumer-grade AI hardware.

The nanosaur measures 10 × 12 × 6 cm, weighs only 500g, and uses a bunch of kit, including Raffaello's custom PCB, a WiFi dongle, 64GB SD card, a power bank, a Pololu Micro Gearbox, and an Adafruit motor controller. You can choose your own camera module from a limited range, which makes 3D printing perfect for this project – Raffaello has included STL files to fit a number of camera modules, so make your choice and get printing. □

Meet The Maker: Timon Skerutsch

Bringing the power of Linux to hardware development

The Arduino and its ecosystem are ubiquitous in hardware development, thanks to its open architecture, cheapness of price, and abundance of ports; the Raspberry Pi is ubiquitous in computing, thanks again to its low price, the exposed GPIO pins, and its free Linux operating system. Wouldn't it be great if you could combine them, taking a Raspberry Pi Compute Module and adding some extras to make it slot more easily into hardware development, just like the Arduino?

That's the idea behind the Piunora. We featured this little board when it was in its crowdfunding stage and, now it's being manufactured and distributed, we thought we'd check in with its maker, Timon Skerutsch, to find out how the journey from crowdfunding to product has been going. Spoiler alert – it's been a lot of hard work…

" **There were several weeks of back and forth with my manufacturer,** building up that partnership "

"Back in December 2020, I posted on Twitter a prototype of Piunora. I had no idea at that point that I'd turn it into a product; it was just that the Raspberry Pi Compute Module was always something that interested me. I made some hacky carriers for the older modules and [was] just playing around.

"The initial idea came from Scott Shawcroft at Adafruit, who works on CircuitPython. He was like, somebody should just make Compute Module 4, but in an Arduino form factor. And that stuck with me. I thought, I can do that, it's not too hard.

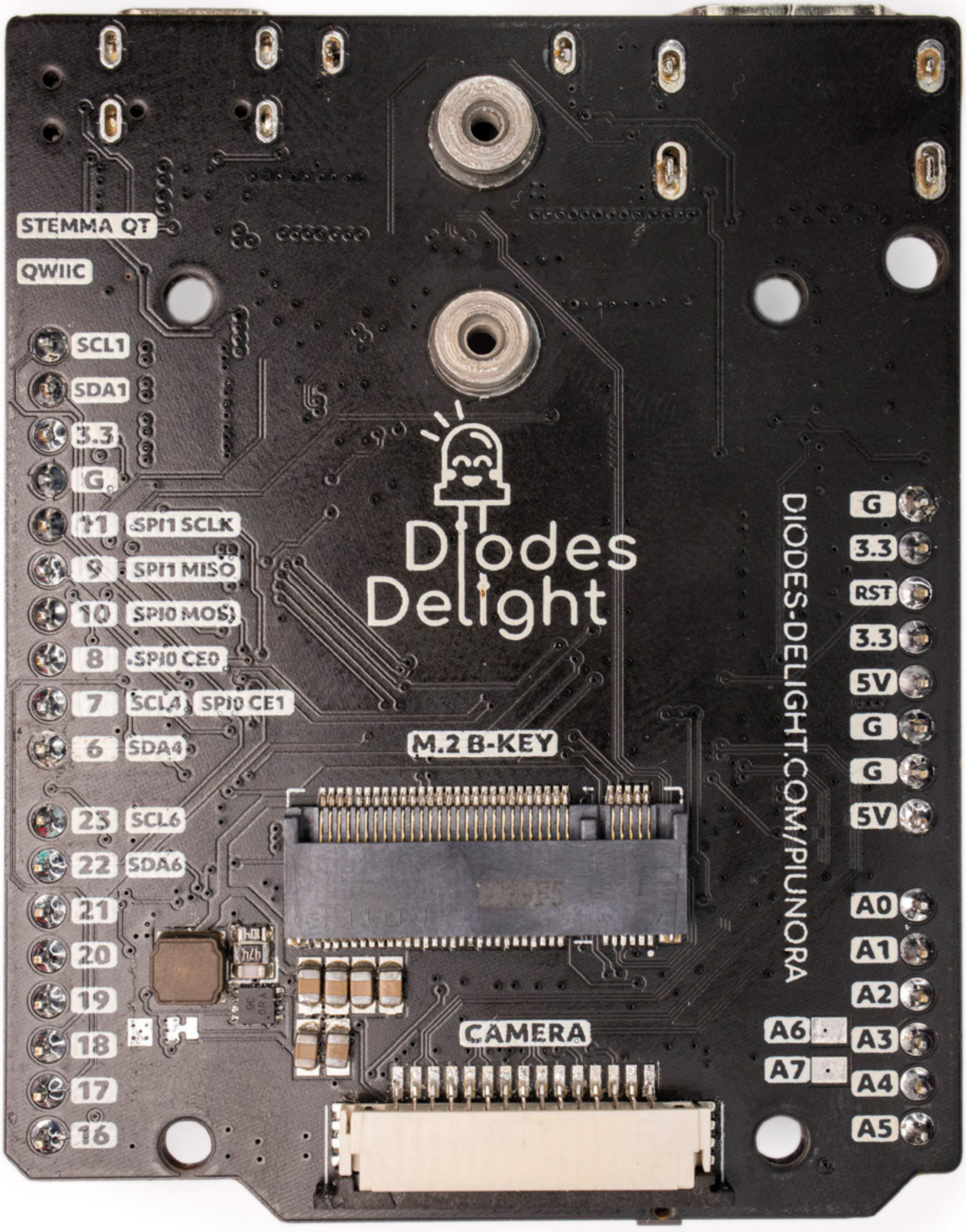
"Over a weekend I threw out that first design, and people were, like, really into it. I'd always been thinking about starting a side business – before that I was working on this Linux tablet thing – and manufacturing process-wise, it's fairly easy to make these kind of boards: it's just a PCB. It's a good startup product, you know.

"I started to set up a company, which was a fair bit of work, at least in Germany, all of the paperwork. And then I got in contact with Crowd Supply. Helen Leigh there, she's a friend of mine, and she kind of talked me into that. I was thinking, you know, it's not so expensive, I could get the funds together from my own money. But it's less risk if I have someone else's money for that, and anyway, a [crowdfunding campaign] helps you to gauge demand for a product.

"And then it was three months of work, of preparation for the campaign. The trickiest part was gauging pricing – you have to give people a price without actually knowing what the production will cost. There are so many factors – in four months' time pricing will be different, components might not be available. The shortage was already beginning, so there was pressure on to lock in a price for components before they went up. There were several weeks of back and forth with my manufacturer, building up that partnership more. And then making all of the media content for the campaign and PR stuff – that was a ton of work. I actually took a vacation for that – I took three weeks off my day job.

"So, that definitely was a full-time job, for at least a month, to do all of the writing and comparison stuff →

Right The back of an early version Piunora. The pins are labelled on the silkscreen, which is a usability feature Timon was keen to include


STEMMA QT
QWIIC
SCL1
SDA1
3.3
G
11 SPI1 SCLK
9 SPI1 MISO
10 SPI0 MOSI
8 SPI0 CE0
7 SCL4 SPI0 CE1
6 SDA4
23 SCL6
22 SDA6
21
20
19
18
17
16
Diodes Delight
DIODES-DELIGHT.COM/PIUNORA
M.2 B-KEY
CAMERA
G
3.3
RST
3.3
5V
G
G
5V
A0
A1
A2
A3
A4
A5
A6
A7

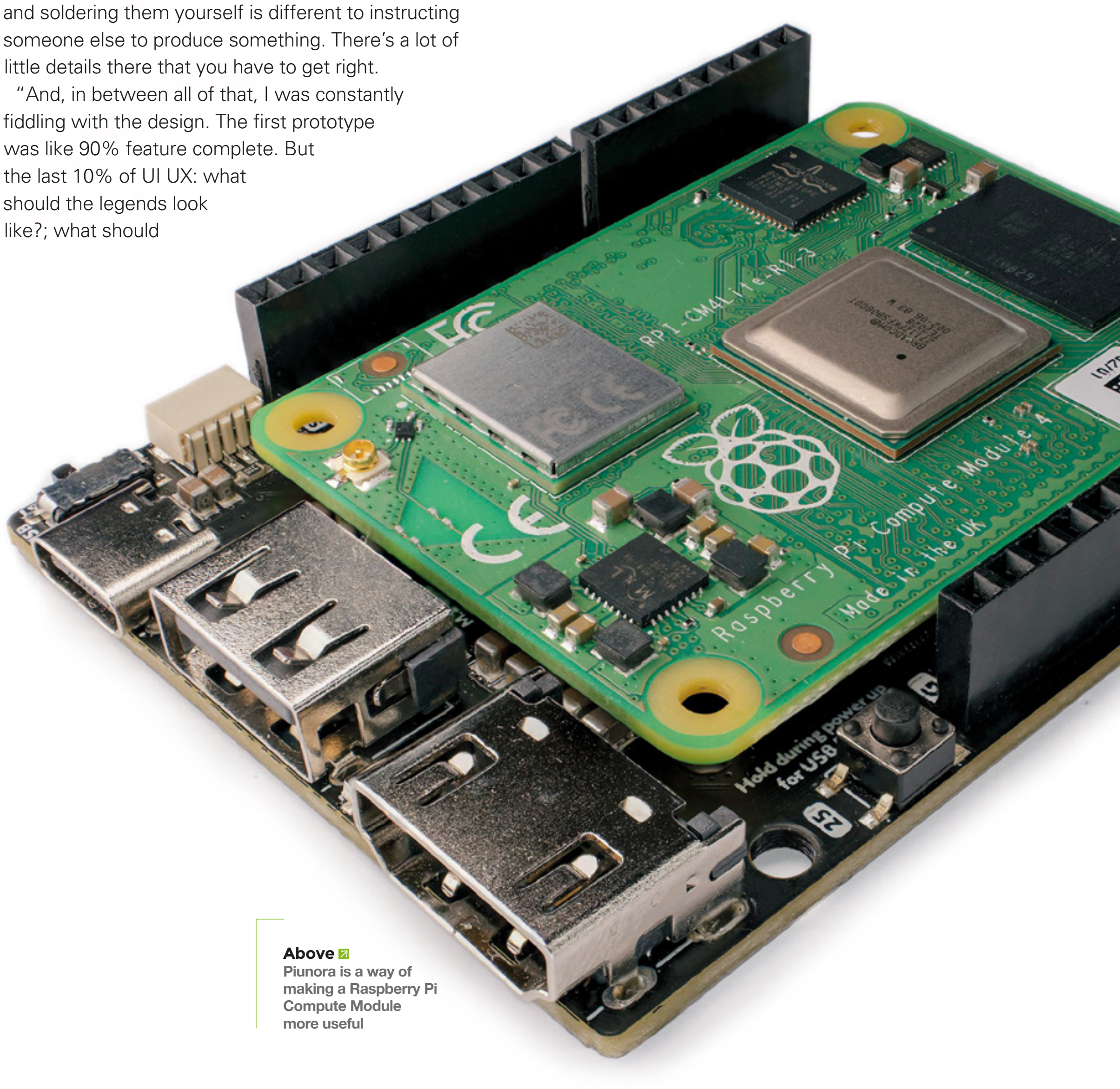
that you have to make: how am I different to others? And what do I offer you?

"The video alone took two and half weeks, and that's just a 1-minute, 20-second teaser video. For anyone who's thinking of crowdfunding a product launch, the best thing you can do is have people who you can work with, who can support you – don't do everything by yourself.

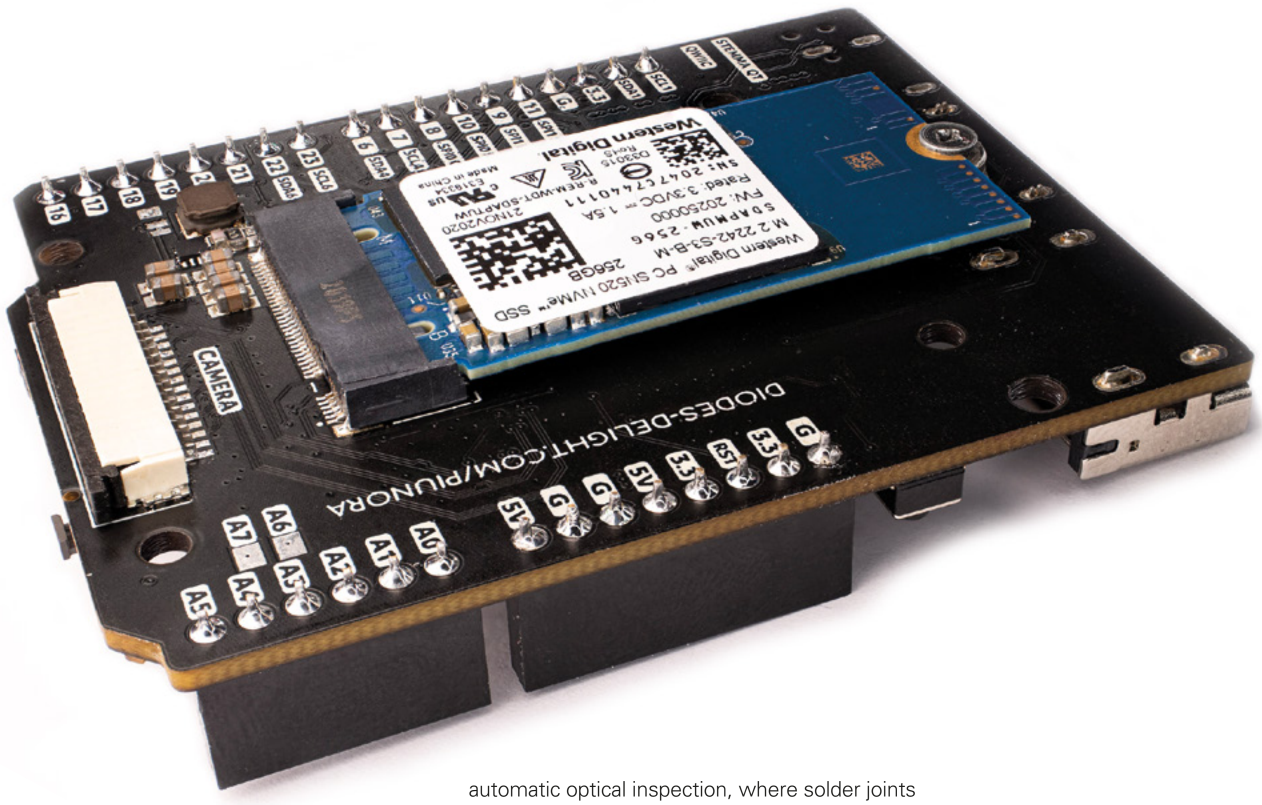
"I think at the end of the crowdfunding, I'd reached 120%,130% of the initial target. I'd hoped for a little bit more, but I got funded, which is what counts at the end of the day. After that, I started beta testing with people; I made the first initial run with a manufacturer to test. First of all, you have to test the manufacturing: is everything the way I want it to be? Ordering PCBs and soldering them yourself is different to instructing someone else to produce something. There's a lot of little details there that you have to get right.

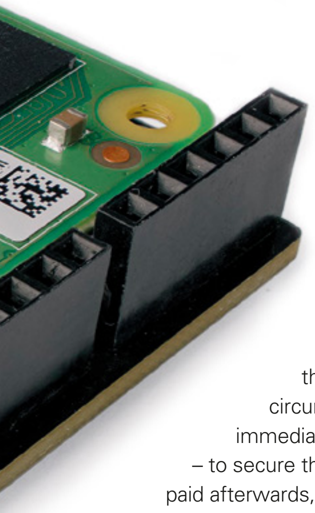
"And, in between all of that, I was constantly fiddling with the design. The first prototype was like 90% feature complete. But the last 10% of UI UX: what should the legends look like?; what should I put there? Like, what information is important? Where should I place connectors; where are they most accessible? That's a huge usability thing. If you want to make it a good product – one that people actually use, rather than buy and stick in a drawer somewhere – that takes quite a lot of time. So, I would say that the optimisation part took me another six weeks' worth of work, drawn out over about five months.

"Shortly after the campaign, I bought all of the components immediately, which was kind of a risk because the design wasn't 100% finalised. But, you know, the shortage was hitting, and everything I needed was depleting in stock



Above
Piunora is a way of making a Raspberry Pi Compute Module more useful





rapidly. So I had to buy it, even if I didn't know for sure I'd be using it. And the other thing was that I didn't even have the funds at that point. I had to buy components with my own money – luckily I had enough savings to be able to do that.

"In other circumstances I would have probably not needed to do that but, because of the circumstances, I just needed to do it immediately. That's why I put out the money – to secure the components. Everything else, I paid afterwards, like PCBs and manufacturing costs.

"There was a lot of logistical back and forth, a lot of corresponding with people; that was the biggest strain, because I had a day job and, on top of that, there was this constant list of five or six background tasks. Even if it was just an email that I had to send, it was constantly on my mind. Because if you don't do anything, it's not getting done, because you're the only person doing it – that's another reason to have a team member, someone to help you.

"There was a delay in production with a high-density connector for the Compute Module; that was another three or four weeks, because the manufacturer was not confident to just use an optical inspection [an automatic optical inspection, where solder joints are inspected visually]. It was so fine-pitched that it was hitting the limits of their machines so, at the last minute, I had to make this electrical tester that dropped in in place of the Compute Module and checked the connections electrically. For more complex products, the testing process can sometimes exceed the engineering time of the actual product. It's quite a task to make these testers, and I was hoping I wouldn't have to do that, because it's a simplistic product without [many] active components. I didn't anticipate that.

"**I also didn't realise that I needed CE certification to sell in Europe,** because they removed some exceptions in recent years"

"I also didn't realise that I needed CE certification to sell in Europe, because they removed some exceptions in recent years that they had in place before. So, I also had to rush to get CE certified. Yeah, there's so much documentation you have to write for all of these things. If you do it all yourself, build things yourself, test things yourself, you don't have to communicate anything, you can just do it because you know what to do. But, if you have to educate a whole other team on what's in your head and what to do →

and it's just you know… even down to making designs for the little labels that go on the box, designing the box, specifying how it should be packaged. You have to tell people to add a length of adhesive tape for sealing the box. It's so many little things that add up to a ton of work, really. Buying stickers – it's not something you think of when you're making electronics, but it's part of the thing.

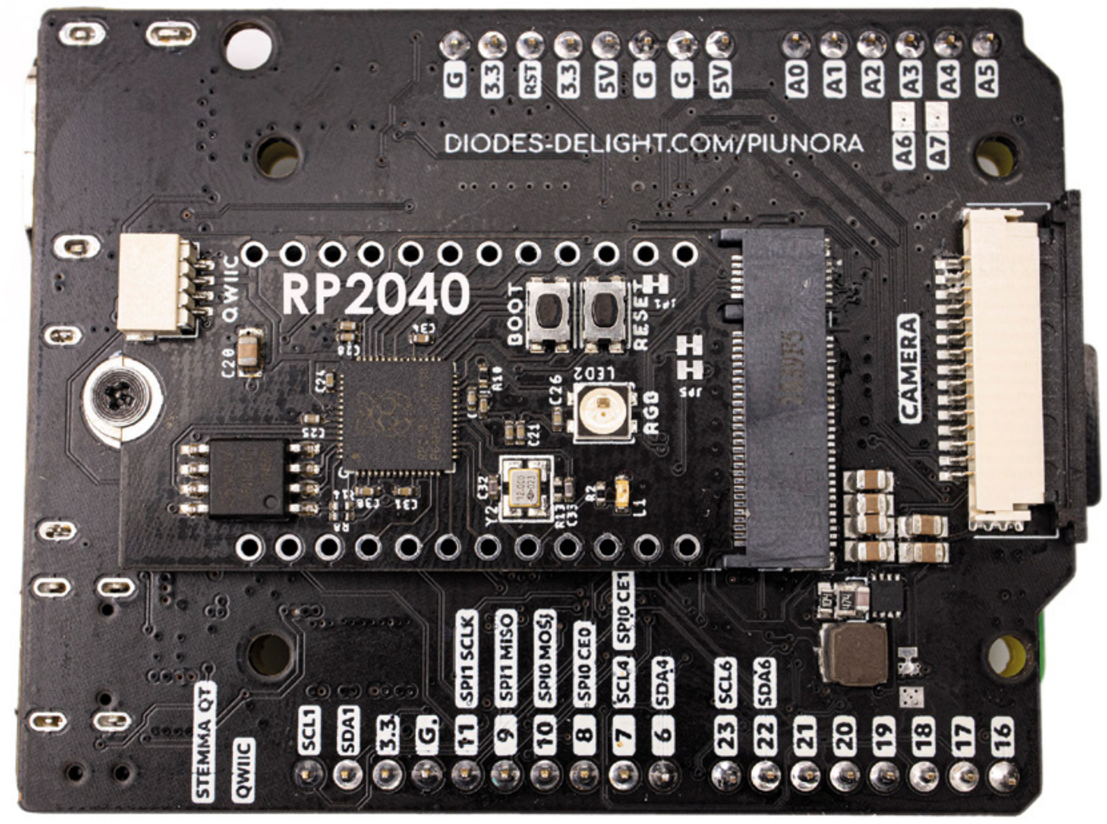
THE PIUNORA

"I made the Piunora because there wasn't anything out there that was Linux, catering to electronics prototyping specifically. The Raspberry Pi, for example, is a computer. You can do electronics with it, but it's not made for that purpose. It's got this [unlabelled] 40-pin header interface, you know, you have to count pins to find the right spot, and there are a lot of little details that add up to a wall that you have to go over for starting a prototype.

"With a Piunora, you plug it into a computer, it's easy to hook up the wires; there's silkscreen [labelling] so you know what everything is. You have analogue interfaces, which the Raspberry Pi doesn't have, you have some buttons and some inputs and outputs that you don't have to add when you're prototyping; you can just get on with it.

"I've added a NeoPixel LED, an analogue input, an ADC. One thing that was very important for me because it's so big in the electronics space, is the Qwiic connector for I2C, which enables you to hook up these little boards that SparkFun and Adafruit make… It's just a lot of these small little things that are nice to have when you're doing electronics prototyping. I'm working on something at the moment, and I have my accelerometer connected. I don't have to find out where the I2C ports are: it's just plug and play – basic. Load up the Python driver and that's it. It's also important to me that all of the pins are labelled, at least important ones. And one other thing the Raspberry Pi doesn't have is that it doesn't break out the PCI Express ports; being able to add an SSD, or even a SIM card connector, is useful in some sensor deployment applications.

"I've not completely finished my vision for USB mass storage on the Piunora, but it's kind of there. And you do have that on the Raspberry Pi 4. But the issue is the



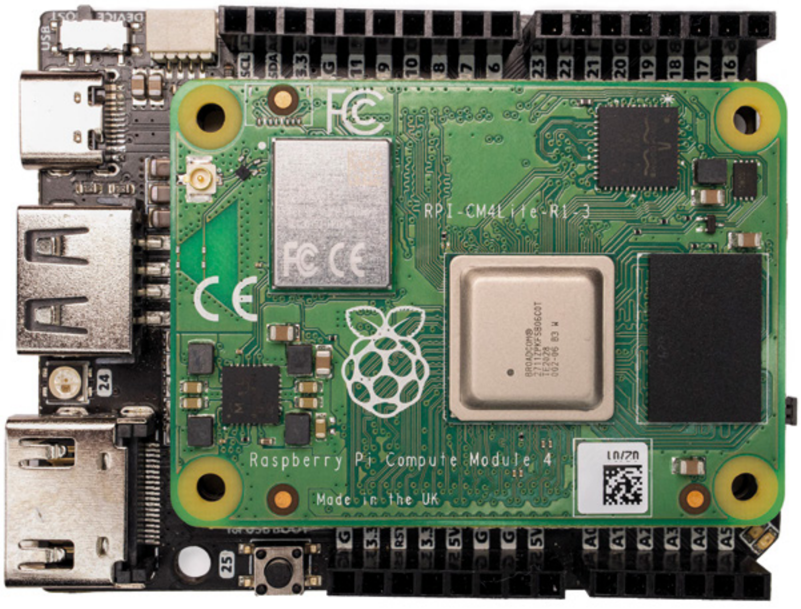
Right Hmm, the RP2040. We've heard of that somewhere…

Raspberry Pi 4 has much higher base power consumption because more peripherals are present. With Piunora, it's about 560 milliamps at peak, if you don't have anything connected. That means you can connect it to a computer via a USB 3.0 cable (a USB 3.0 port can do 900 milliamps). And, if you connect it to the USB-C port, it can then be a USB device, which is where I always wanted it to be. It's always felt like, for prototyping, it's kind of tricky to have to connect to WiFi, or have a screen plus keyboard, and find out the IP address, SSH into it, enable SSH… it's like a process. I wanted [it] more like Arduino: plug it in, it boots, and it becomes accessible via a serial port. And that's something that the Raspberry Pi can't do.

"At the moment, I can kind of do Arduino-style – I can SSH in, or I can sit at a porch terminal kind of broke with it. But, if you've worked with CircuitPython before, when you connect, it becomes a mass storage device where you just edit the code directly on the microcontroller.

"That's very specific to CircuitPython, and I'm a huge fan of that. You just plug it in. It's like a USB stick that pops up and there's a code.py file. OK, and you just write your code and press Save. And then you know, the microcontroller reloads and has the new code running, which is super-cool for rapid prototyping. I wanted to also have mass storage, so you can drop files on it and have them transferred to the active file system, and then be able to work with them without having to do SFTP or something: just like drag and drop files around, that kind of workflow. I had it working as like a proof of concept, but I want to get it more robust.

"It's still early days, but I've had a bit of feedback already from people using the Piunora. It's always kind of hard, because a lot of people buy it and do things but they never get back to you about it. Somebody connected it to a kind of Motorola laptop – Motorola made this laptop without any brains with it, where you can plug in your phone and then it becomes like a laptop – just HDMI and USB input. And people hacked that together like a Raspberry Pi and it had this huge clunky thing on the back. They have apparently [got] a similar device, and they made a laptop out of it, which is kind of cool because everything is on one side. So, you can just make this whole pluggy thing with it, because all the connectors are on the front.



Above Piunora hit the shelves of its first distributor on 28 February, Raspberry Pi's 10th birthday

"I can't tell you the name, but there is a large research institute using them for medical research, which I never expected to happen.

"I saw a lot of IC companies order it, so I assume a lot of people use it for testing jigs, or something like that. FTDI bought a lot of them. I've no idea what they exactly do – I guess some lab applications.

"I think it's communication work that I still have to do to – to show people what exactly you can do with it"

"One reason why I want to do some more tutorials, or publish projects with the Piunora, is to give people something to work off, because I feel a lot of people get started by following Adafruit guides or something similar. I packed a ton of features into the Piunora, and I know what you can do with it. But I think it's communication work that I still have to do to – to show people what exactly you can do with it, and get the imagination flowing.

"One thing that I've done this year is make a Piunora operating system image, which is a modified Raspberry Pi OS image that sets everything up for you, so you don't have to fiddle around with scripts and stuff – everything's ready to go.

"I think it's easy to get started right away at the moment, but I want to show what things you can combine. You can, I don't know, control NeoPixels from a browser, for example. I want to take the powerful Linux computer that the Raspberry Pi 4 is and show people what you can do with it when you combine it with electronics.

"That's why I made Piunora – to make that hurdle a bit easier."

Letters

ATTENTION ALL MAKERS!

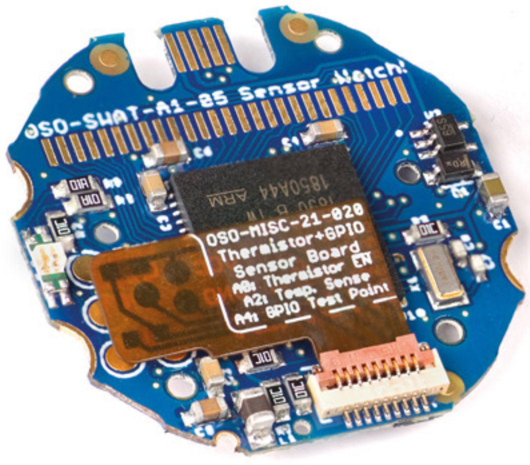
If you have something you'd like to get off your chest (or even throw a word of praise in our direction) let us know at **hsmag.cc/hello**

CROWDFUNDING

What a mad, nostalgic rush I got seeing the Sensor Watch [Crowdfunding, issue 52]. As a proud geriatric millennial (I prefer 'millennial elder' or 'ur-millennial' myself), the Casio F-91W is part of my internal aesthetic. It looks like a watch should look, like the way Keanu Reeves in *The Matrix* thinks of himself as having a full head of hair, even though he's been bald all his life. I want one! I don't have any idea what I'm going to do with it, other than invent a time machine so I can go back to 1990 and show off to my ten-year-old self. Shut up and take my money!

Robert Wilson

Ben says: Whoa there! Yes, it's a brilliant project, but the nature of all crowdfunded projects is that it will take a while for this idea to be made flesh (well, silicone) and distributed to backers. You can think of that as a form of time travel if you like: present you is buying a gift for future you.



FLEXIBLE PCBS

I'd already seen some of the work of Carl Bugeja before you spoke to him [in issue 52]. Instead of winding coils of wire around a spindle, you etch the copper spirals onto a PCB, and that enables him to produce an electric motor that's small and light. It's a brilliant, sideways approach to solving a problem, and I remember being impressed when I first saw it. The other stuff he's doing is completely bonkers: a flexible PCB with LEDs stuck to it, that wobbles about and creates a persistence of vision effect? Brilliant!

Pete D
Manchester

Ben says: There's also the flexible PCB-as-actuator that Carl's been working on which, so far, has seen application in a robotic fish, a jumping robot, and more. And you're absolutely right. As PCBs have become cheap enough for hobbyists to afford, we (or at least, I) have spent a bit of time and energy on making things look nice, but, as Carl shows, there's a lot more that you can do with a little bit of imagination.

THINK OF THE CHILDREN

I was amused to read your interview with Odd Jayy, about robotics and machine learning. It's cool and a bit frightening to think of the machines gaining sentience and taking over the world one day; it's also cool to realise that you can fool them by simply putting a mask on.

James Fitzwilliam

Ben says: Jayy says that training AI is like teaching a child: you need to keep repeating yourself, and you need to be patient. Eventually the AI will decide that your taste in music is awful, and will moan about having to eat its vegetables. Seriously, there's no magic to artificial intelligence, and that's really what Jayy's trying to show by doing it himself: by detailing his successes and failures, he's showing that machine learning isn't a mysterious black box, it's just a tool for achieving an end. Sentient super-robots aren't coming any time soon.

CROWDFUNDING NOW

Jolly

Updating the classic Uno

From €12 | jolly-dev.com | **Delivery:** September 2022

The Arduino Uno (and its clones) was, for a long time, the dominant microcontroller for makers, and it's still one of the most iconic circuit boards. However, the board has some drawbacks. The microcontroller isn't particularly powerful by modern standards, and there's no network connectivity.

Jolly is an in-place upgrade for your Uno. You pop the microcontroller out of its socket and replace it with the Jolly circuit board, which combines a microcontroller and WiFi controller chip into the same shape as the original microcontroller.

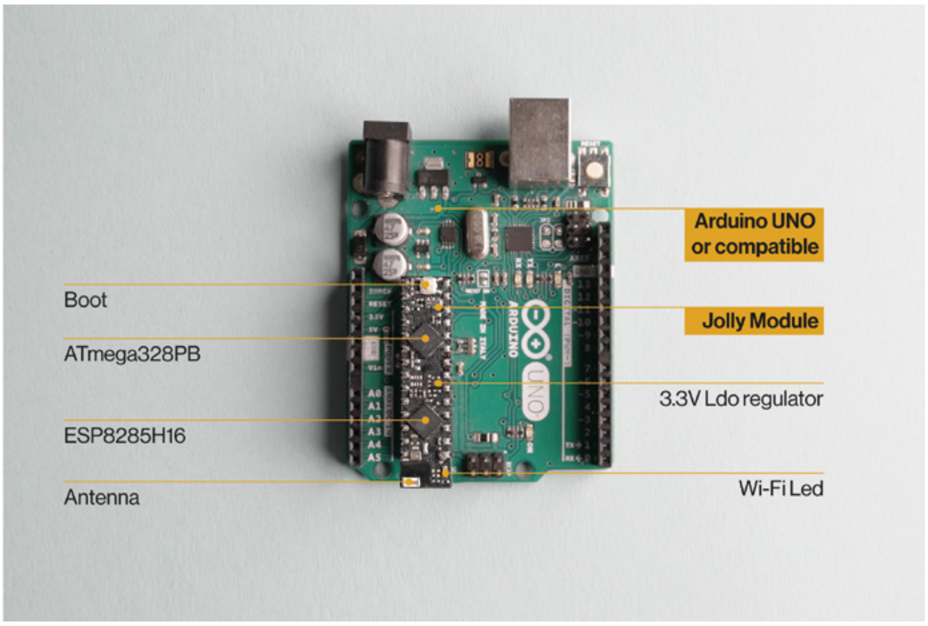
There are a few advantages to this approach over, say, just using a WiFi-capable dev board. Firstly, the vast majority of WiFi-capable dev boards run at 3.3V rather than 5V of the original Arduino. In many cases 3.3V might be better, but if you've got a project set up and running on a 5V Arduino and want to WiFi-enable it, it might be easier to switch out the microcontroller for a Jolly than to meddle about with the circuit to make it work with 3.3V. By putting everything on the main microcontroller module rather than on a shield, they aren't interfering with the I/O pins that are broken out.

"You pop the microcontroller out of its socket and replace it with the Jolly circuit board"

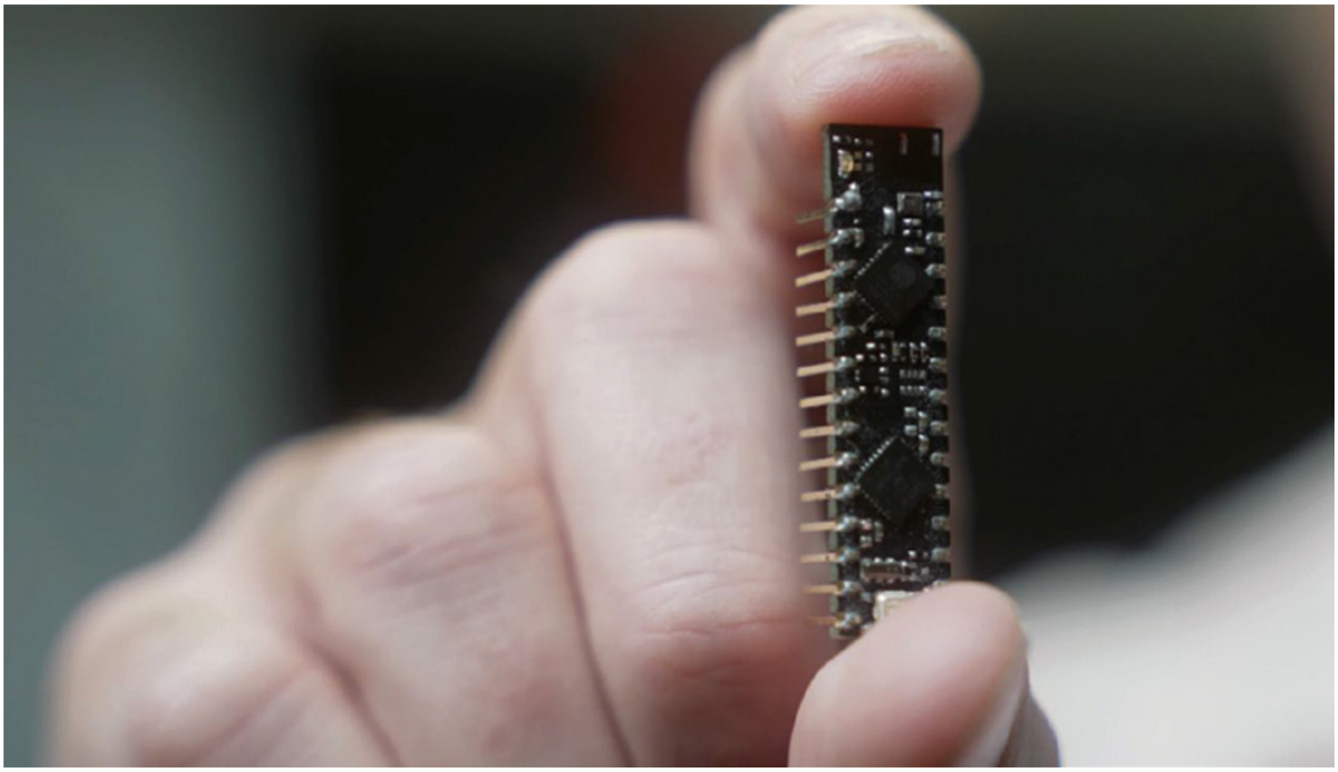
There are, of course, trade-offs on the other side. If you're starting a new project, this is an expensive way of getting a dev board since you'll need both the original donor board and the Jolly module. Also, you'll end up with a system with quite an outdated microcontroller. Whether or not this is worth the compatibility with Arduino Uno projects is up to you. □

BUYER BEWARE

When backing a crowdfunding campaign, you are not purchasing a finished product, but supporting a project working on something new. There is a very real chance that the product will never ship and you'll lose your money. It's a great way to support projects you like and get some cheap hardware in the process, but if you use it purely as a chance to snag cheap stuff, you may find that you get burned.



Below The entire circuit board is the same size as the microcontroller chip in the original Uno



FREE SHIPPING ON ORDERS OVER £33 OR $50 USD*

Design starts here

More products
More suppliers
More NPI

Digi-Key® ELECTRONICS

0800 587 0991
DIGIKEY.CO.UK

Start with Digi-Key

*A shipping charge of £12.00 will be billed on all orders of less than £33.00. A shipping charge of $18.00 USD will be billed on all orders of less than $50.00 USD. All orders are shipped via UPS, Federal Express, or DHL for delivery within 1-3 days (dependent on final destination). No handling fees. All prices are in British pound sterling or United States dollar. Digi-Key is a franchised distributor for all supplier partners. New products added daily. Digi-Key and Digi-Key Electronics are registered trademarks of Digi-Key Electronics in the U.S. and other countries. © 2022 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA

ECIA MEMBER
Supporting The Authorized Channel

HackSpace
TECHNOLOGY IN YOUR HANDS

LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG 44

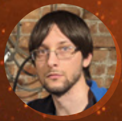
HOW I MADE: **MECHANICAL KEYBOARD**

A pleasing clickety-clackety input device with a Raspberry Pi Pico at its heart

PG 50

INTERVIEW: **MICHAEL BRETTI**

The open-source space hardware keeping satellites moving around our Earth



PG 58

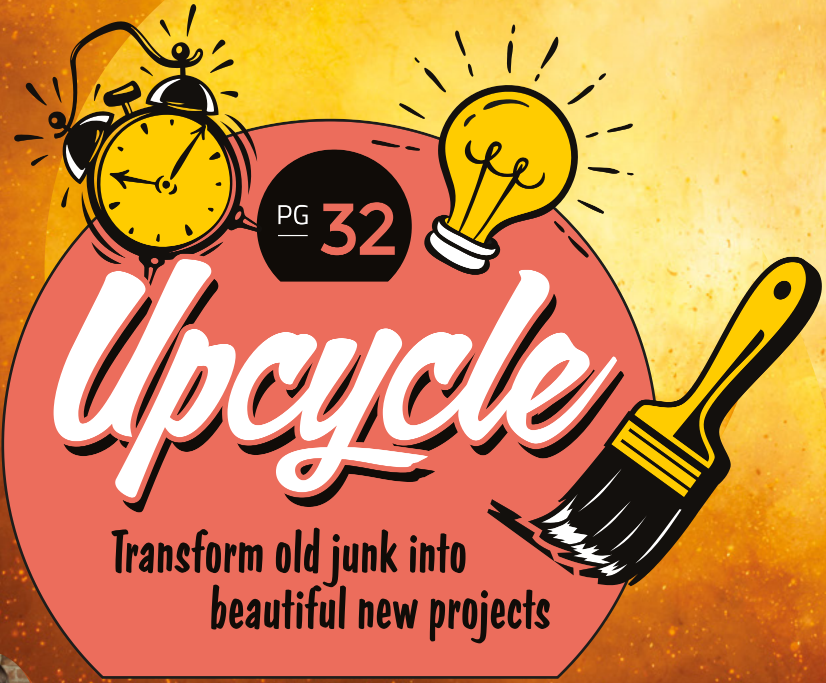
IMPROVISER'S TOOLBOX: **TYRES**

Don't burn old tyres! Make something useful with them instead!

PG 62

IN THE WORKSHOP

Bring the musical map of Scotland to life with capacitive touch



Upcycle

Transform old junk into beautiful new projects

Upcycling, or remaking, is the art of taking something that's no longer needed and turning it into something new and useful. It might be broken, or obsolete, or just unwanted. There's a huge variety of things you can do to upcycle old or unneeded things – it might be something fairly straightforward, such as adding a bit of flair to bring old clothes back into fashion again, or it could be a complex rebuild. The important thing is that you're keeping something out of landfill and, at the same time, reducing your dependence on virgin materials. This combination of environmental benefits is perhaps the biggest advantage to upcycling, but the other big plus is that you get design features from the previous life. How you incorporate this into your new item is up to you. Sometimes it makes sense to flaunt it; other times you may choose to hide it away.

Before we get started, though, let's decide what upcycling actually is.

Is giving something a new coat of paint upcycling? Is that just maintenance or mending? What about adding more decorative elements to change the fundamental design style of the original piece?

What about stripping something down to its component parts – is that upcycling or recycling?

It's a grey area, and does it really matter? While there are some clear-cut items, there are also many that could fit into more than one category.

For the purposes of this article, recycling is going through an industrial process. So, plastic that gets melted down and re-extruded, we're calling recycled. Materials that are removed by hand and put to a second use – to us, that's upcycling.

Fundamentally, though, our philosophy on this is that it doesn't really matter. If you're taking something old and unused and converting it into something new and useful, then that's a good thing, regardless of what you call it. →

Wood

Save the trees by reusing their products

Wood is a versatile material and, when kept dry, can last almost indefinitely. It's expensive to buy new, and reused wood picks up character from each use.

There are a couple of ways of repurposing wood. Firstly, you can use wooden objects and keep their features. Once upon a time, it was relatively common for things to be built with joinery. Dovetails and box joints were the strongest ways of making corners and so were often seen on products. When made well, they can look stunning, and reusing old joints is the quickest and easiest way of getting good joinery in your builds. If you're lucky enough to come across such products today, it can be worth keeping them intact and showing off their joinery – such things are seldom seen in today's world of mass-produced items made of engineered wood.

The second option is to break it down to individual pieces of wood and use these for your projects. Perhaps the most famous option for this are wooden pallets that large objects are delivered on, but it's also common to come across second-hand scaffolding board. These are both relatively rough

Left Convert pallet offcuts into comfy seating

Above
The same techniques could be used to build a wide variety of garden furniture

Safety

Pallets aren't designed to be turned into furniture, so there are a few safety considerations if you choose to do this. There may be stray nails that aren't fully sunk. It's not unusual to find nails sticking half-out of a pallet.

The wood is often very prone to splintering, and can be sharp if it's cracked. It typically needs plenty of sanding before it's a final product, and you may need to remove some broken planks to make it safe.

"A particular attraction of pallets is that they are already made into frames"

boards, so you'll need to at least sand them down heavily for most projects. While, these days, there are no cheap sources of wood anymore, reclaimed scaffolding boards and pallets can be more affordable, but the quality can be questionable depending on… well, a lot of things. Both often have cracks and other imperfections to deal with. If you have the time and inclination, though, they can produce beautiful makes.

Paint your pallet blue and grey

A particular attraction of pallets is that they are already made into frames. If what you're making can use these frames, it can be a really quick way to get things up and running.

YouTuber Well Done Tips has a great guide for making a garden chair from pallet wood – **hsmag.cc/GardenChair**. This uses leftover wood from other pallet projects, so it's not only upcycling pallets, but upcycling leftover pallets. It could be a great project if you end up with some partly broken pallets.

Sticking with the garden theme, Instructables user DanPro made a potting bench from old pallets. In this build, DanPro shows us how to work with pallets in smaller forms – **hsmag.cc/PottingBench**.

Pallet projects don't have to be this complex, though. Tables and chairs can be made by simply stacking pallets on top of each other, and perhaps adding a back.

Pallet furniture is most popular in the garden – it can be a little rustic for many tastes as indoor furniture. But that doesn't mean that there aren't indoor uses for pallets. Using the wood as cost-effective cladding is now almost obligatory for a certain type of bar or coffee shop, and the wood can be used for a huge range of projects. →

No two pallets

It's common to refer to a pallet as though it's a standard thing, but it's not. They come in different sizes and shapes for different purposes. They're also often reused until they're pretty worn out, so it's common to only be able to get hold of older, broken pallets. If you're going to work with pallets, you have to be prepared to work with the limitations of the particular pallets you've got. If you can do this, it's often possible to pick a few up relatively easily, especially if you have a vehicle that you can fit them in.

It's possible to buy new and second-hand pallets online, but if you post a message on your local neighbourhood social media group, you may well end up with a few free of charge.

Below
With minimal work, a pallet makes a great planter

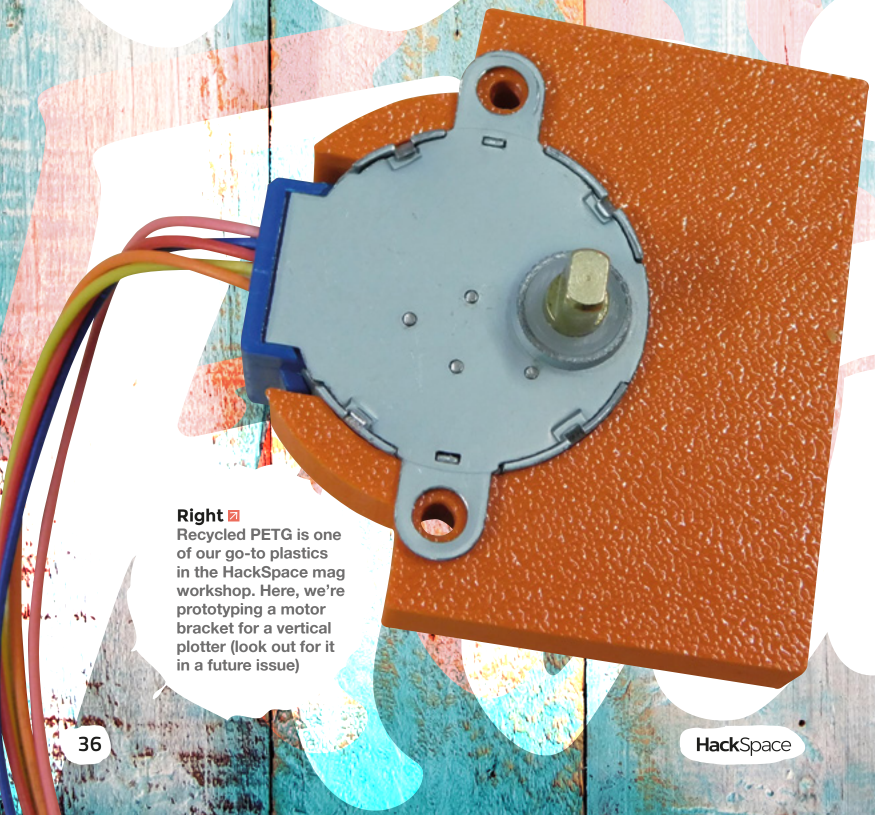
Recycled Filament

Keeping precious plastic in circulation

There's a bit of a blurry line between upcycling and recycling. They both involve turning something old into something new. Generally, recycling involves breaking down things into their raw materials and reusing these, just as you would any virgin material, whereas upcycling involves keeping some assemblies or parts from the original.

The term upcycling comes from the fact that recycled parts have traditionally been lower grade than virgin raw materials, and so recycling is sometimes moving them down the quality spectrum, whereas upcycling is seen as moving materials up the quality spectrum.

While recycling does pose some materials quality challenges, it's not a given that recycled materials will be lower quality than virgin raw materials. Perhaps the one recycled material makers can use that has the most impact is recycled plastic. While much plastic we consume is technically recyclable, unless there are robust streams of people using recycled plastic, the recyclable plastic will continue to be



Right
Recycled PETG is one of our go-to plastics in the HackSpace mag workshop. Here, we're prototyping a motor bracket for a vertical plotter (look out for it in a future issue)

incinerated or buried. Using recycled plastic means that you're not using up new raw materials, and you're helping find a market for the vast amounts of recyclable plastic that are collected.

PLA-y time

The environmental credentials of PLA are somewhat convoluted. It's a bioplastic that's usually made from corn starch, and is biodegradable. However, it doesn't biodegrade very easily and needs relatively high temperatures (over 60 °C) for the plastic to break down. In the UK, some food waste collection does reach this temperature, but many do not, and those that do may not accept PLA.

Perhaps the kindest thing for the environment, then, is to recycle it. There are a few PLA collection projects around accepting PLA, but an important part of recycling isn't just sending material off to be recycled: it's buying the stuff that's made with it.

We tested out some of Reflow Filament's rPLA to see if it's up to the task. We loaded it up into our test printer and it worked just as well as regular PLA. The colours are earthy tones that reflect the material's eco credentials.

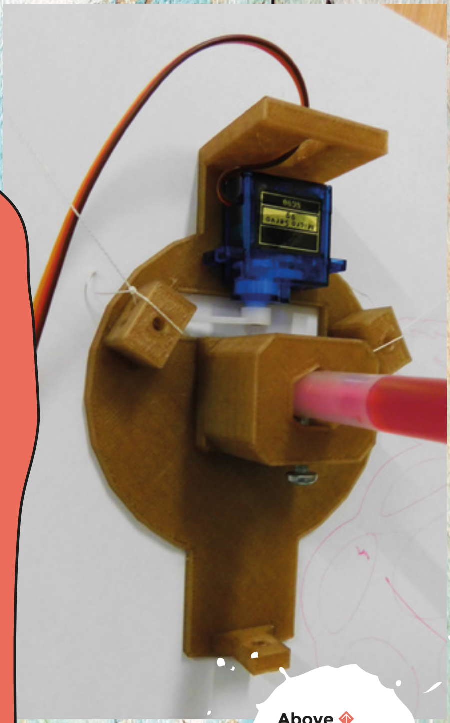
At the moment, recycled PLA is more expensive than virgin material. However, depending on what you're printing, the cost-per-make may not actually be that significantly higher. For example, a Benchy uses around 13g of filament. Using a particularly cheap spool of PLA, this comes in at about 20p. Using recycled filament, it will come in around 40p.

Post Industrial PETG

While the options around recycling vs composting PLA may not be clear-cut, if you need to use a different filament, such as PETG, then the case for using recycled filament is much more robust. Plastic recycling is in its infancy. Despite the fact that many of us have been dutifully sorting our rubbish for years, the full chain from waste plastic through to new plastic thing is still not fully formed. One of the most useful things that we can do as makers is help create a market for recycled plastic. PETG is, after PLA, the most popular 3D-printed plastic – and of its constituent polymers, PET, is common in waste.

We tested out FilaPrint's Post Industrial PETG (PIPG) and it printed excellently with the standard PETG settings on a Prusa MK3. Recycled PETG is a similar cost to regular PETG, and we certainly plan to switch over our PETG use to recycled filament in the future.

Filament choice is a very personal decision, with a lot of factors playing into it. However, here at HackSpace magazine, we're now switching all our in-house 3D printing over to recycled filament. The filaments we've tested in this issue have printed as well as we'd hoped and, frankly, without a good reason not to use recycled plastic, we can't justify the environmental cost of using virgin plastic. →


Above Francofil's Beer PLA works well for this pen holder

Left This Benchy in recycled PLA came out a little stringy, but it's nothing that a heat blast from a heat gun won't fix

Beer PLA

If your end goal is to be as environmentally friendly as possible, there is another route to go down: bioplastics. Yes, we know that PLA is a bioplastic, but PLA filament isn't entirely PLA. It has other bits added to make it flow better, to add colour, and to generally improve it as a filament. These other bits can come from a wide range of sources, and it can be hard to find exactly what is in your PLA.

The Francofil brand has a range of filaments made entirely of biologically derived ingredients. Alongside the base PLA, there's leftovers from beer brewing, waste from the shellfish industry, and more. If you have the capabilities to compost PLA, then this means that you'll only be composting biologically sourced bits – great news if you don't want the other additives in PLA getting in your compost.

Adding New Technology

Raid thrift stores and car boot sales for your next project

Above Items with large bodies often need internal structures adding to them – Martin Mander uses LEGO

Over the years since the start of the Industrial Revolution, fashions and available materials and processes have changed many times. These days, it's rare to find new items made of cast iron or bakelite. Even 'real' wood is now rare (as manufactured wood is easier to work with on an industrial level). While these materials are rare in new items, there remains a huge collection of old objects sat around in garages and lofts. It's not our place to say whether these look better or worse than modern mass-produced items, but we certainly like the aesthetic of these retro objects.

However, while they look cool, their usefulness is often limited by the age they were designed for. The cassette player might look cool, but do you really want your music collection on cassettes rather than streaming media?

There are loads of ways of repurposing and upcycling using existing objects that could be enhancing their original purpose with additional parts or newer technology, or using them for an entirely new purpose.

New tech

Just because something is obsolete, it doesn't mean it's ready for the bin. Modern electronics are often many times smaller than old-fashioned parts, so many electrical objects can be upgraded with new technology.

Sound systems are an area ripe for getting technological enhancements. The technology has changed several times over the past few decades. Vinyl, cassettes, and CDs have all come and gone. While there is still some joy to be had in physical media, the convenience of digital streams has now won out for many people. While the medium of storing music has changed, older speakers and amplifiers can still create a great sound.

There are a few things to consider when upcycling an old audio system. Firstly, what parts you want to use. You might want to use the speakers and housing, or the amplifier as well. If you want to use the amplifier, you'll need a way of getting sound into the amplifier – this is easiest if you have an auxiliary input port.

What audio source do you want to use? Bluetooth audio receiver boards are cheap and plentiful. We have had quality issues using boards with built-in amplifiers. The downside of this is you need to keep your mobile (or other Bluetooth device) near the sound system. A slightly more complex build is to create a media streamer that can play audio directly from internet radio, Spotify, or other similar system. There are boards that will do all this (take a look at the selection from **arylic.com**), or you can build one using a Raspberry Pi – there's a build guide for one in The MagPi issue 96.

This is, of course, just one idea. There are literally hundreds of ways older hardware can be brought back to life. →

Above We took an in-depth look at upgrading this old radio back in issue 30. It's still going strong and streaming audio. Take a look at how we did it in issue 30: hsmag.cc/issue30

"Many electrical objects can be upgraded with new technology"

Below The Pi VCR brings a Raspberry Pi and a flat screen into the body of an old video recorder

Old tech new spec

Martin Mander is one of our favourite makers working with old technology. We've featured his work a few times in Top Projects, and had an in-depth chat with him in issue 11. You can read about it at hsmag.cc/issue11, or follow along on his blog at kyliemander.com.

Repurposing Older Items

Shining lights and ticking clocks

Some of the most popular repurposing upcycling uses we've seen are converting things into lamps and transforming them into clocks. In both of these cases, the conversion can be as simple or complex as you like. The main reason to upcycle in this way is that it lets you take something that has a beautiful design but is no longer needed (either because it's broken or obsolete) and keep it on display as a useful object. The range of objects that can be converted into one of these two things is almost limitless.

From a circuit point of view, making a lamp is simple. There's power, a switch, and a bulb holder. However, when you're working with mains voltages, you have to

Below Dubchinsky converts old phones to lamps. See its range at: hsmag.cc/Dubchinsky be careful – not just about how it's wired up now, but how it might fail if it's damaged. If you're going to work with mains power, you have to be confident that you understand all the risks that it can pose, and how to mitigate them.

If you'd rather not work with mains power, then fear not! Modern LEDs are efficient enough that they can put out plenty of light from low-voltage sources. This could be a battery or a sealed power supply.

Tick tock

The simplest way of converting an object into a clock is to drill an appropriately sized hole in it, and then poke a premade clock mechanism through. These mechanisms are cheap and easily available, but make sure you get the right one for your project. The two biggest differences we've found in our work between different clock movements are:

Shaft length – you want one that just pokes through your object without holding the hands too far off the surface.

Sound – some tick, some are smooth. It's a personal preference, but this author can't stand the sound of a ticking clock.

Poke the movement's shaft through the hole and secure with a nut, and the hands push into place. Beyond that, it's just a case of making sure you get the look you want.

Of course, an analogue clock isn't your only option. Modern microcontrollers can drive a huge range of output devices, including seven-segment displays, LEDs, and Nixie tubes (Soviet-era tubes that have a filament that glows in the shape of a number).

Right Old objects and Nixie tubes are one of the classic combinations for maker clocks

Depending on your microcontroller, you might need a real-time clock (RTC) module that goes alongside the main microcontroller and counts the time. There are libraries for working with these in most languages. See how to work with them in CircuitPython at **hsmag.cc/RTCCircuitPython**, and take a look here for details in Arduino: **arduino.cc/en/Reference/RTC**.

"Modern microcontrollers can drive a huge range of output devices"

Over to you

These are just a few of our favourite ways of upcycling things. Because the range of starting material is so huge, the range of projects you can do is also huge. The starting point is an object or thing that you find interesting that's not living up to its full potential. That thing might be something that's just cluttering up your house, something that you were about to throw away, or something that you saw for sale second-hand somewhere. Remember to keep a lookout and remember that someone else's junk might be your treasure. ◻

Left Find out how to make this hard drive clock, by The Unknown Chef, at: hsmag.cc/HDClock

Left Metal with holes produces a very striking shadow pattern. Colanders and washing-machine drums work, as well as cheese graters

MAKE | BUILD | HACK | CREATE
HackSpace
TECHNOLOGY IN YOUR HANDS
hsmag.cc | April 2022 | Issue #53
MUSICAL MAP
Building a touch-sensitive atlas
Upcycle
Transform old junk into beautiful new projects
APPLIED ION SYSTEMS
Open-source thrusters... IN SPACE
LEDS ANALOGUE R2040 KEYBOARDS

HOW I MADE

By **Ian Dunn**

A PCB MECHANICAL KEYBOARD

Building a customised input device

I made an 87-key mechanical keyboard with a conventional key layout, constructed from three layers of FR-4 PCB. While making a keyboard from PCB material may sound a little unconventional, I think it's ideal; it's strong, easily customised, and very easy to clean. What makes this keyboard really unique is its open-source firmware written in the Arduino IDE that can run on the Raspberry Pi Pico.

AN OVERVIEW OF MY PROCESS

Briefly put, I started by designing a layout, turned the layout into a schematic, and then placed 87 keys in the correct place on a PCB. Once the layout was complete, I placed a diode reasonably close to each switch. The circuit isn't complicated, but it is a really large circuit and a big PCB. If you want to take on this challenge, this article will save you from making some of the beginner mistakes that I made. I've open-sourced my schematic and key layout for you to use as a reference, or to customise to your own liking without you having to start from scratch, at this link: **boltind.com**.

DESIGNING A LAYOUT

I recommend designing your layout with: **keyboard-layout-editor.com**. This is a handy tool that allows you to drag and drop keys where you want them, adjust the →

Above This is my keyboard, but yours can be any design you like

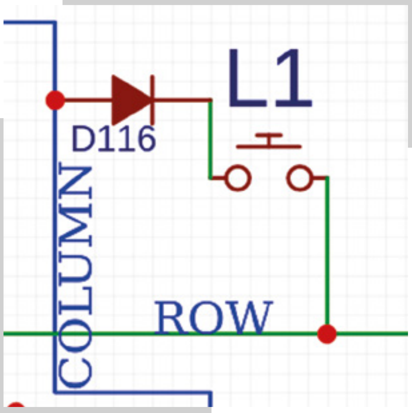


Figure 1 Make sure the diodes are the right way around

Below The main circuit board with switches and other components soldered in place

size to your liking, change the legends, and otherwise customise your keys.

The other tool I used is the Keyboard Firmware Builder, at **kbfirmware.com**. You can copy and paste the 'raw data' from Keyboard Layout editor into this nifty tool and it will give you a schematic diagram of how your rows and columns should be laid out. It also generates custom firmware, but I don't recommend using it for this. It's no longer maintained and there are better ways of building firmware, which I'll get to later on.

Once you've got your layout planned, you can draw a schematic in your preferred EDA software. I recommend EasyEDA if you're a first-time EDA user. EasyEDA is really simple, yet it's packed with every feature you could ask for. You'll want to start by locating a switch symbol with a CHERRY MX-style PCB footprint. This form factor is pretty much the gold standard for mechanical keyboard switches, and makes it easy to place a switch on the schematic for every button on the keyboard based on your layout.

Start with the first row of keys. Label each one carefully before moving on to the next row. The prefix naming conventions can get a little confusing, because EDA software usually names components 'R-1, R-2, R-3' and so on. If you label the **F1** key as 'F1', then you will find that you've already got an F1 when you place the letter F key. Just find a naming convention that works for you, and be careful not to place the 'F1' key in the place of the letter F key when you layout your PCB. I've been there, done that!

KEYBOARD MATRIX

Once you've got the switches placed on the schematic, draw the matrix. Unless you're designing a macro pad, you'll find there's not enough pins on any microcontroller to have each key on its own GPIO. This means that you have to use a matrix to scan each row and column, one at a time. For the matrix to work properly, a diode is required for each key. Place a diode for each switch. The 1N4148 switching diode is the only way to go. It's economical and it comes in through-hole, as well as surface-mount, packages. It's important to note that the direction of the diode depends on the firmware. Most firmware will allow you to select the direction of the diode. If you're going to use BMK, my Arduino IDE firmware, the diodes must be placed from column to row. See **Figure 1**.

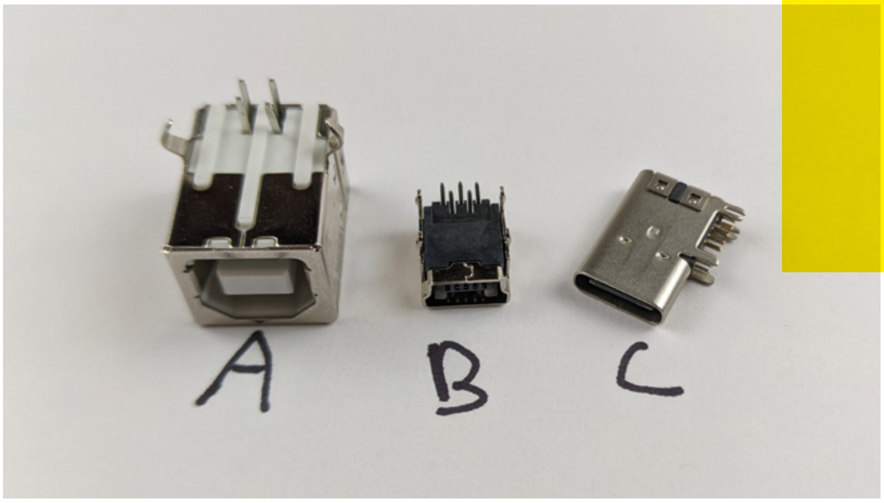
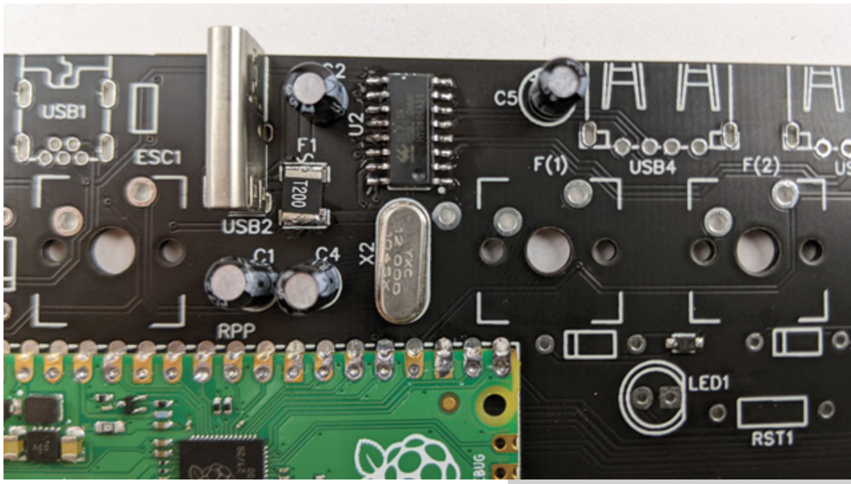


Figure 2
I found these USB connectors were the easiest to solder



Above
There's a bit of additional circuitry for the on-board USB hub

KEYBOARD LAYOUT IN AN EDA PROGRAM

A small gap between keys is required so that the keys don't rub against each other. Typically, the 'unit' on a keyboard is ¾". This means the keys are spaced ¾" on centre, and every key size is some multiple (or fraction) of ¾"which is equal to 750 mils (for anyone more familiar with metric than imperial measurements, a mil in this context is a thousandth of an inch, NOT a millimetre). For example, the square letter keys are spaced 750 mils on centre and the rows are offset by some fraction of 750 mils. When laying out keys, we can just assume each key is 750 mils and we'll end up with the correct spacing between keys because the key caps are a little less than 750 mils.

The exception is the larger keys, like the **SPACE** bar, **TAB, CAPS, SHIFT, ALT**, etc. These keys can be 1.25, 1.75, 2, 2.25, or 2.5 units wide. Because the key sizes are in increments of .25 'units', and the rows are offset from one another, you will need to set your snap size to 93.75 mils while laying out the keys because 93.75 mils is ⅛ of a ¾" key. Also, set the grid to 750 mils.

STABILIZERS

Once you've got your keys laid out, you will need to add stabilizers. Stabilizers come in two sizes: 6.25 units and 2 units. The former stabilizers are for the **SPACE** bar. The two-unit stabilizers are for **SHIFT, ENTER**, and **BACKSPACE**. There's also seven-unit stabilizers out there, but every **SPACE** bar I've ever come across is 6.25 units. →

"YOU WILL NEED TO ADD STABILIZERS"

Above The extra USB ports let me connect additional devices

TOP AND BOTTOM TRIM

The next step is to route your PCB traces. You could stop here and have a working keyboard, but I highly recommend adding a top and a bottom made from FR-4 PCB. This will cover up all the components in a way that looks attractive and will add a lot of rigidity. To screw the three layers together, you'll want to use M3 × 8 mm screws, M3 nuts, and M3 × 10 mm standoffs that are tapped all the way through. Screw everything together tightly so that the finished keyboard will be nice and stiff.

To create a top or bottom, copy the main layer and then make changes accordingly. For the top, trace the 750 mil outline of all the switches with a board outline trace. Don't worry about adding a gap; key caps aren't quite 750 mils wide. Once you've got every 'group' of keys traced, delete all of the switches and you should have a finished top trim. The top trim is a great place to add legends for your custom Macro or Unicode keys.

The bottom can be identical to the main layer to save cost, or you can make another board with the same outline and no components. If you added screw holes to the main layer, just leave them in the same place when you copy the main layer for the top and bottom and they'll line up perfectly.

CONNECTING TO USB

You can certainly use the USB port on your microcontroller, but I found that the Pico fits best between rows, horizontally. This means that the micro USB would awkwardly stick out the side of the keyboard. To get around this, you can connect a separate USB port to the USB pins on the bottom of the Pico. They are labelled TP2 and TP3. TP2 is D- and TP3 is D+. Making a USB device can get really complicated, but the main thing to know is that the D+ and D- lines need to be right next to each other, and they must have identical impedance. If you follow these two rules, and don't try to do anything too fancy, you shouldn't have any problems.

Most USB connectors require very fine surface-mount soldering. I've come across three through-hole USB connectors that I like. All three of them can be purchased from LCSC. See **Figure 2**, overleaf.

The first (A) is a USB-B plug. This one is big and ugly, but it's really easy to solder. Its part number is USB-BF90 from Valuepro. You could also salvage one from just about any old printer.

The next (B) is a USB-mini B connector. Its part number is 920-462A2021D10102 and is also from Valuepro. This one is perhaps the best option because it's reasonably compact but still easy to solder.

The last (C) is a USB-C connector. Its part number is U264-141N-4BAC10 and is from XKB Connectivity. This one is nice because it's USB-C, but it's also the most difficult to solder. Its footprint is as small as through-hole soldering gets. It is a through-hole part, but it's not any bigger than most surface-mount USB connectors. You'll need a needlepoint soldering iron and some very small solder. USB-C is normally a USB 3.0

connector, but this one is USB 2.0 only. The power, D+ and D-, need to be connected in two places each. You may also want to add a 5.1KΩ pull-down resistor to each of the two CC pins. This will tell the upstream USB 3.0 device that the keyboard is a peripheral device. Once enabled, you can plug your keyboard directly into your USB-C phone or tablet and it will work like magic.

I also added a two-port USB 2.0 hub to my keyboard. This is really handy, because you can plug your USB peripherals directly into your keyboard. You can even plug a mouse into the keyboard while it's connected to your phone, so you can have the use of a mouse and a keyboard on your phone.

FIRMWARE

The final step is to flash a firmware to your Pico. A good friend of mine, Brian Di Donna, wrote what I call 'BMK' – an abbreviation for Bolt (Industries) Mechanical Keyboard. BMK is cool because it's all done in the comfort of the Arduino IDE. BMK can do just about anything a regular keyboard can do, but is completely customisable so it offers endless possibilities. It's really fun to write keyboard macros in the Arduino IDE.

Another firmware candidate is KMK. KMK runs on CircuitPython, so it works nicely with the Raspberry Pi Pico. Once you've got KMK installed, making changes to your keyboard is just as simple as opening a text document on the Pico 'flash drive'. Simply make your changes and press Save.

I hope you'll experience the satisfaction I have found as you build something just right for you. □

"ANOTHER FIRMWARE CANDIDATE IS KMK"

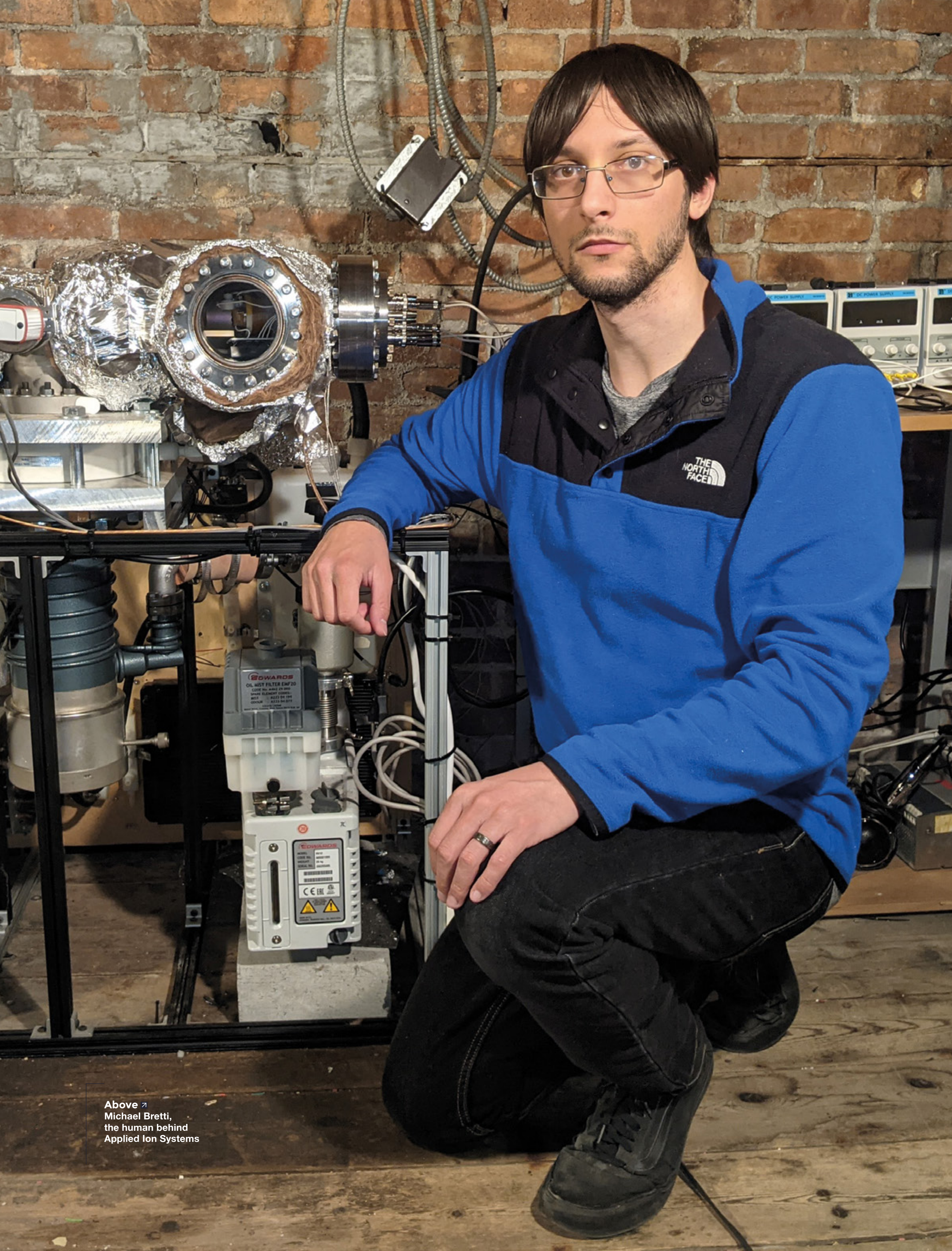
Above There is a huge range of key caps available, but I like these black and red ones

HackSpace magazine meets...

Michael Bretti

Open Source space hardware

Space: the final frontier. These are the voyages of Michael Bretti, the man behind Applied Ion Systems. His continuing mission: to develop low-cost, low-power thrusters to control the movements of satellites. They have to be small, to fit on a 5 cm cube. They have to be affordable, because Michael works out of his home in Upstate New York, not for NASA or SpaceX or any of the big boys. And more pertinent to us at HackSpace magazine, they're open-source hardware, being developed completely in the open. Every test, every failure, every success is documented and shared, so anyone can learn a little something about space thrusters. We spoke to Michael and learned the difference between a plasma thruster and a rocket, plus other mind-bending space physics. →


Above ↗
Michael Bretti, the human behind Applied Ion Systems

HackSpace Let's get something out of the way early on. When I see 'space', I think 'rockets'. But the things you make aren't rockets, are they?

Michael Bretti No, not at all. They follow similar basic principles: you throw something out the back end, and you move something in space, but they're very different mechanics.

I've always been fascinated with ion beam, particle beam stuff, plasma physics, things like that. I've worked on a bunch of projects in atmospheric conditions, because you don't need any special vacuum hardware to run the stuff for that.

And I always wanted to get into vacuum-based plasma physics, and those types of things. So a couple of years ago, I finally bit the bullet. I sat down and did proper research and design for a vacuum chamber, all set up and configured, and I took a year to become kind of an expert in the vacuum engineering side of things, so I'd put things together and know that it would just work, and I'd know what I was doing.

That took a lot of time scrounging and locating materials. One of the nice things about being in the US is our surplus: the surplus of components and stuff like this on eBay is very, very prolific. You can go on eBay, and you can buy any type of scientific equipment you want. So there's a lot of surplus vacuums, hardware, and stuff like that floating around, and other surplus vacuum shops in the US.

So, I spent a year doing that stuff, getting a vacuum system put together. Originally I had no intention of actually doing electric propulsion: it was for another project. I had been really fascinated with particle physics. So I had been working towards designing a very high-power pulsed electron gun.

So huge system for injecting kiloamp level beams at, you know, tens of nanoseconds wide, really, really intense pulse things. There's an issue with that, because when you have very high voltage, high-power electron beams, you have a lot of X-rays. And that's a huge safety issue. So while I was trying to figure out the shielding and everything for that, I had some spare leftover stuff.

I've always been fascinated with propulsion things. Like, back in high school, I remember looking up home builds of turbojet engines and pulse jets and stuff like that, that people have done. And electric propulsion allowed me to combine all my interests in one thing, so plasma physics, particle beam physics, vacuum physics, high voltage, pulsed power propulsion, all into one category.

So I figured, you know, ion and plasma thrusters are pretty cool. Let's see if, on the side, I can play around with this stuff. So I put together that chamber, and I started designing and playing around with thrusters and posting about it on Twitter and stuff. Eventually, from there, people like Jo Hinchliffe took notice and reached out and connected me to people especially in the PocketQube [satellite] community where there's almost no propulsion work at that scale.

> "I've always been fascinated with ion beam, particle beam stuff, plasma physics, things like that"

So it just took off from there. I started focusing and tailoring my effort towards really, really tiny, really low-power systems and developing specifically for the community. And, eventually, I developed my first full thruster system. I actually have it still here. Let's grab that. So this is the first full thruster I ever built.

HS That's tiny. I've seen the image of it before [on page 57], but I had no idea it was that small until seeing you holding it.

MB It's tiny because, when you're talking about PocketQube stuff, you're talking about a cross-section of five by five centimetres, and then a single cube is five centimetres deep. So super, super tiny systems.

I built them specifically for PocketQubes, and I delivered them to an AMSAT Spain group. Unfortunately, that rocket flew recently, just last year, but it failed to reach orbit. So we lost the two units, unfortunately.

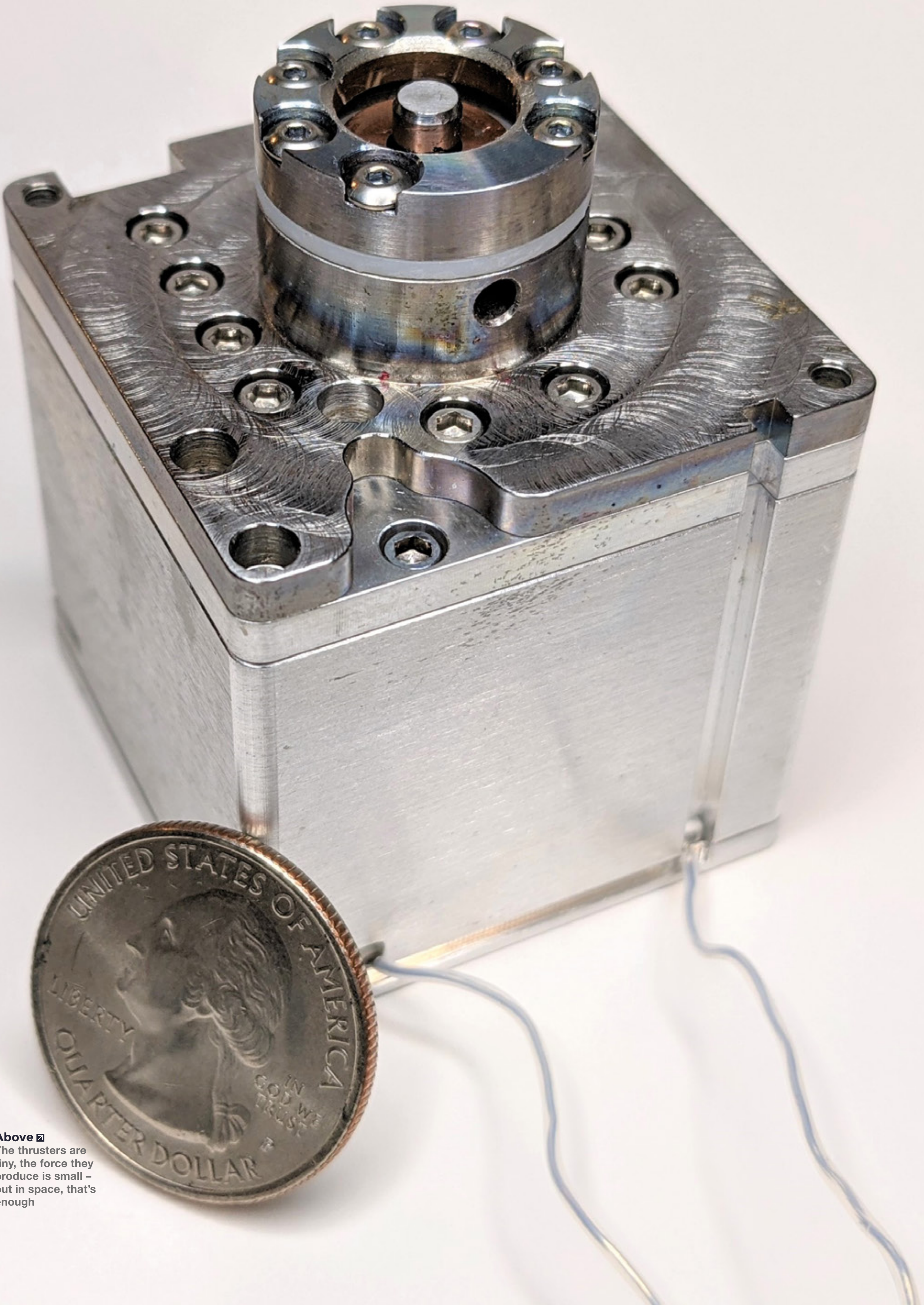
But this one actually did make it to orbit last March aboard the Care Weather Technologies CubeSat that was launched on a Rocket Labs rocket.

And you can download all the design files for this thruster. So the PCB, the mechanical [data], everything for it.

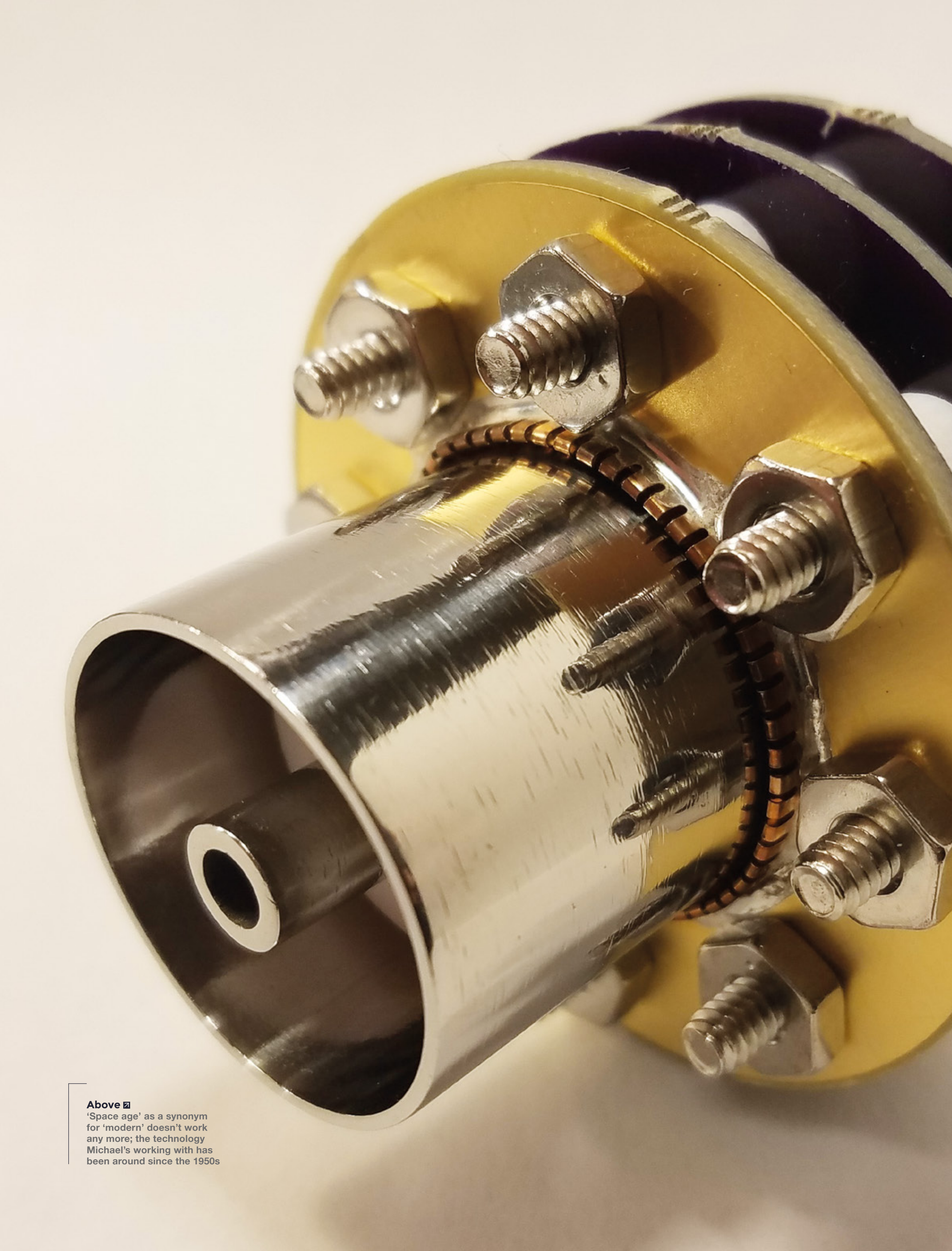
HS Isn't that quite unusual, for space hardware to be open-source?

MB Very much so. I mean, there are definitely people doing open-source PocketQube stuff. And there's the Open Source CubeSat Workshop that Libre Space Foundation hosts every year. But in terms of propulsion, it's completely unheard of. And really, at this scale, outside of well-funded research facilities like MIT or NASA, or multimillion-dollar startups, you don't see this stuff at home.

So, I've been embracing the open approach, and it's really allowed me to create a unique niche for myself. I'm doing something different than anyone else is doing, and ultimately making it more understandable for people. For accessibility to mean anything, it's not just about making things cheaper, or even necessarily easier to manufacture or open source. It's also important to provide resources and show people that, you know, this stuff isn't black magic. When NASA publishes articles and stuff about 'next-generation ion thrusters' and things like that, it's always spun in a way that makes it seem so sci-fi. 'This is state-of-the-art; this is super-futuristic.' But it's really not: this stuff has been around →



Above ↗
The thrusters are tiny, the force they produce is small – but in space, that's enough



Above ↗
'Space age' as a synonym for 'modern' doesn't work any more; the technology Michael's working with has been around since the 1950s

for, like, 60, 70 years. It's all really, really old technology.

I mean, this thruster here, this uses Teflon fuel. The circuit is literally just like a camera flash circuit. It's really, really, simple. And all the systems I'm doing – anything from pulse plasma thrusters to Hall thrusters, and stuff like that. The fundamentals of the stuff are really, really simple, and I want to show people the behind the scenes of how it's done. It's a lot of failures and a lot of struggles and some pretty cool successes.

HS Forgive me if it takes a while for me to grasp what ion propulsion systems are. So, there's an amount of a fuel source, such as Teflon, like you mentioned earlier, or something called adamantine, which sounds cool; and then there's some sort of electric field around it, and then that fires ions from the fuel source out into the void of space.

MB Yes – for one of the throttle thrusters that I'm working on, adamantine is the fuel source for that particular thruster.

Essentially, for all electric propulsion, regardless of the fuel type, you have some sort of fuel that you apply electricity to. So for the pulsed plasma thruster, for example, the way that's done is you have an ignition spark inside, and that little ignition spark ablates a little bit of Teflon in there, and it ionises in there and expands into a little gas.

And then that connects between an anode and a cathode where there's a capacitor that charges up to really high voltage. And that allows you to dump electricity through that in a bigger arc and accelerates it out. For the adamantane fuel, you have the solid fuel that you apply heat to.

So it sublimates into gas. And then you will have again, an anode and a cathode and a little magnetic field with that, and you apply high voltage to that again, and that discharge creates the plasma, and the interaction with the magnetic field allows the ions to get accelerated out. So in all cases, for electric propulsion, you're essentially taking some fuel – doesn't matter if it's solid, liquid, or gas – applying electricity to that fuel in a vacuum, to create a plasma, and then either plasma or ions are accelerated out, depending on the type of thruster you're doing.

HS A rocket uses a combustion effect, which is heat and expansion, which creates force. This is plasma, which is something that I've heard the word, but don't really understand what it is…

MB Essentially, going from the gas, you're adding more and more energy into it, and you're ionising it, so you're ripping away electrons. And you're creating the next state of matter, essentially, where you have this electrically charged gas. And different systems can do different things with it. So you can have just a bulk plasma exhaust, so you're just firing out the plasma, which is a mix of ions and electrons, and everything all just kind of spits out.

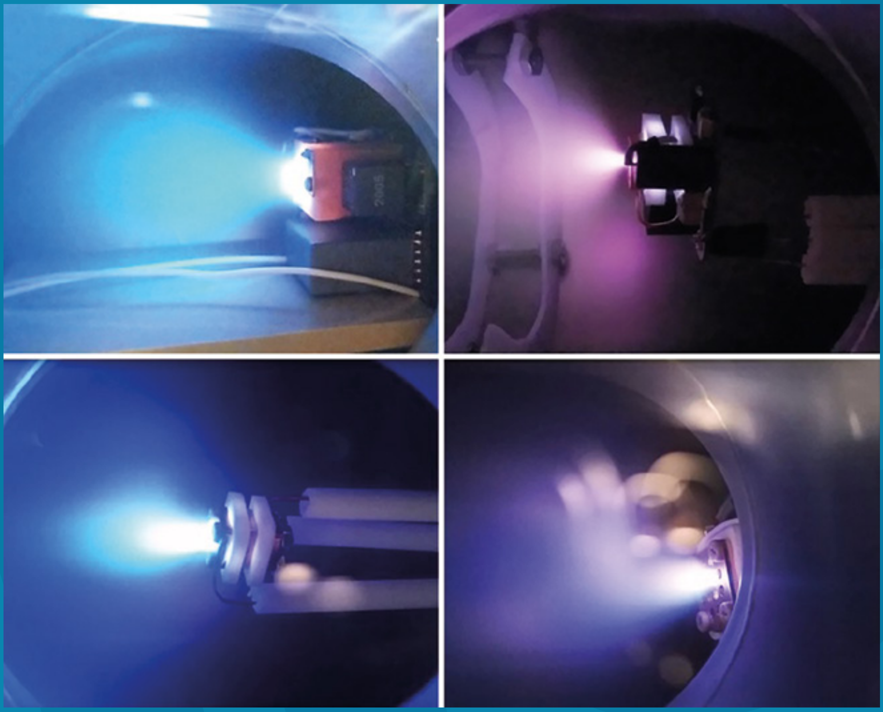
And then for other stuff, like the adamantane thruster, or even some of the other things that I've worked on, [those use] room temperature, molten salts for the fuel, that is sprayed out as an ion. So you have a plasma, and then from that plasma, you pull out the ions and accelerate them out as a beam.

So the mechanisms are different, but a lot of it is very related across the field.

HS I guess if the stuff that you're firing out is just the small particles of electrons or ions, there's not very much mass coming out from the fuel?

MB Very, very tiny mass. And actually, in some cases, it's so little that I can't even measure it, even after firing the system. The pulsed plasma thruster uses so little fuel, that if I weighed it before and after, I wouldn't even be able to tell [that it's used any fuel].

That's partly because this thing has a really limited lifetime; the capacitor fails catastrophically after about 500 shots. Some of the other stuff, like the adamantane, you can definitely →



Right ➔
Michael shares tests (whether successful or unsuccessful) on Youtube. That's open

measure how that sublimates down, you can check the level very easily. And some of the new stuff where I'm using bismuth fuel, it has such a high ion erosion rate, the rate at which it produces ions is so high that it's very, very measurable. Whereas if I use something like titanium for the same type of thruster, it's almost negligible. So different fuels in different thrusters will have different kinds of rates.

But essentially, that's the trade-off with electric propulsion: you have really high exhaust velocities, so tens of 1000s of metres per second exhaust velocity, but the mass is so tiny, that your thrust is really, really, really small. So we're talking about micronewtons of thrust at this scale.

HS How about battery life for the electric pulse that you send through the fuel sources? Is that a limiting factor?

MB The biggest single thing in electric propulsion that really governs the performance and essentially what you can do – and the scaling and what technology would be suitable for particular missions – a lot comes down to the power that's available. So all that power comes from the onboard battery supply on the satellites themselves, which are charged through solar panels that are mounted to the satellites. So really, as long as you have enough power to run the thruster, then the thruster will run, you know, assuming you have fuel and there are no other failure mechanisms in the electronics.

For this thruster, for example, it just takes the five volts from the satellite bus, turns it into 2000 volts with this tiny, high voltage supply, and then you have the rest of the circuitry on there that applies it. All the power comes from the battery on the satellite, and then that's converted into various voltages on board on the thruster.

HS Are you pleased with how the work's going so far?

MB Yeah. I think, when I first started, I had no idea what I was doing. So this is the first thruster I ever built. This is another Teflon pulsed plasma thruster; I had no idea what I was doing. I got the inspiration from old [vacuum] tube-based sockets. So you have a thruster with different electrodes that plugs into the PCB. I applied high voltage over the outside of the chamber. And if it didn't fire at all, I had horrible arcing in the chamber. I had a lot of electronics failures; it was disastrous. And then I reworked that and played around with a different design. And with that one, the thruster fired only once – I captured it on camera; it was just a single pulse.

" With that one, the thruster fired only once – I captured it on camera; it was just a single pulse "

I destroyed all of my electronics during the process, pushing the thing way, way, way too hard. So that was not fun. But it led to the thought that, you know, maybe I can actually do this. I got one shot; maybe I can get two next time, or ten. And it's really grown from there.

What's been so amazing has been the support from the community. When I started out, I was, you know, completely unknown, just posting random stuff on Twitter, trying to connect with people. And I had no intention of starting growing this into a startup on the side, or even trying to expand further. I never even considered the possibility of this stuff even going into space; I was just playing with it in the basement. So the fact that I've gotten something into space, and I have a lot of people I'm working with, developing a number of systems for a number of people and expanding to tons and tons of different technologies across the board at various power levels...and with all the support of the community, it's really blown me away. I would never have imagined that just playing with this stuff on the side would open up so many opportunities.

HS When you say community involvement, community support, does that mean that you get a lot of input and help from other scientists?

MB I do get input. But it's challenging because on one hand, I'm also learning as I go, so I'm experimenting with things like that and doing a lot of experimentation on the side, and it's still an area where you have people in industry who are doing this, and there are tons of companies and researchers all over the world doing this stuff, but outside of that, the maker community, with the hobbyist community, people know the high-level fundamentals.

But actually, the nitty-gritty details and the mechanisms and stuff, that's what I'm trying to bring to the community itself, through all these experiments and this open development and research. I want to show people from the ground level how it's done. So I think it's tough for people to give input, just because it's a field that you have to really spend a lot of time looking into and looking at the details. I'm also trying some really weird things because my budget is so limited, so I have to play around with some unusual configurations.

But in terms of support in general, like, all the encouragement, and obviously with Patreon, for example, it's helped me do a lot of research. All the encouragement from makers, from hobbyists, from the satellite people. Because there are tons of failures. And there are periods where it's just failure after failure after failure. And it really wears on you. The community don't let me give up when it gets really tough.

For example, when I first started with my vacuum system, I had a cheap refrigeration pump for one of the primary pumps. And that failed, and I didn't have the budget to keep going. So some members of the community [it was actually organised by Bruce Perens] started a GoFundMe to raise funds for a scientific-grade pump. And that's been running ever since and supporting a lot of development. I have a lot of trouble asking for help. I never asked for help, just because I really struggle with that. I think people have been really inspired by this stuff, and what I've been doing on the side, so they've been willing to pitch in and see how far this development can be pushed.

HS What are you working on right now?

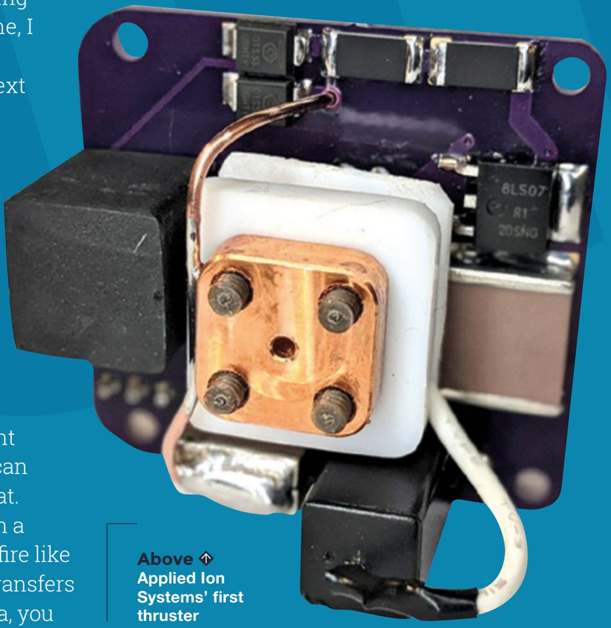
MB I'm working on another pulse thruster – it's called a vacuum arc thruster. So that uses the solid metal fuel that I was discussing earlier, working specifically with bismuth. It's about the same size as the image below, but it uses a very different mechanism. It's actually a little bit weird because, unlike that one, which uses about 2000 volts, this one generates the plasma at very low voltages, so 40 volts or something on the input. And it's very weird, you have this graphite layer on a ceramic washer that you apply a pulse to.

It causes these little things called cathode spots to form. So it explodes out electrons and gas and graphite from that layer and causes a plasma flash-over to arc between the electrodes. It's a really cool little device. Again, it's very good for scaling, so it can be made really small, really low power for these tiny PocketQubes. But it can also be scaled-up for big CubeSats and other stuff pretty readily. I've been working on a tiny system. So for a single one, I actually have boards in that I'll hopefully be running within the next week or two, firing three of them simultaneously in a vacuum just to start doing some lifetime qualification of them.

I've been doing a lot of pendulum tests. So I made a tiny ballistic pendulum with a super-thin piece of Kapton [a heat-resistant, electrical insulating polymer]. You put the thruster right in front, and you fire it. And by knowing the displacement and the mass and everything, you can calculate the force generated on that. There are some issues because with a classical ballistic pendulum, if you fire like a bullet into a block, it sticks, so it transfers all the momentum. With the plasma, you have particles that hit the target, and there's a ton that bounce off. There's some that stick; there's some that scatter. So the readings are really inaccurate with a flat pendulum, which is what I've been using.

I'll be changing to a conical shape – there was a research paper done in the 1980s where they used a cone shape facing the thruster, so when the particles hit, they bounce off radially, and that cancels out any additional momentum. I'm transferring the pendulum so you can get more accurate readings. So I'll be kind of playing around with that pendulum.

And then, also, I recently got some requests to see if I can do a much higher power system. So the small system runs at about 2.5 watts – really, really low power, it's about 10 micronewtons of thrust, in that ballpark range. And I'm working on a much bigger system, that would be a 50-watt class system. So much, much bigger. Really big plasma plumes coming out of that, and probably looking at 500 to 1000 micronewtons of thrust. So it's a much bigger scale. I'm focusing on this particular class of the vacuum arc thruster category, just because there's a lot of potential, and it's been one of my best firing systems yet. □


Above Applied Ion Systems' first thruster

IMPROVISER'S TOOLBOX

TYRES

Burning rubber is fun; burning tyres not so much.
Rosie Hattersley suggests alternatives to creating a noxious nuisance

Rosie Hattersley

@RosieHattersley

Rosie Hattersley writes tech, craft, and life hacks and tweets **@RosieHattersley**.

Every impoverished driver has probably contemplated retread tyres. They're a good budget option if you want to save on the cost of replacing worn-out ones, but there's a limit to how long you can spin things out in a bid to keep motoring costs manageable. Eventually, the depth of tread will no longer suffice, posing an issue of how best to dispose of your expired tyres. Some might say this is where the fun really starts!

A small proportion of tyres end up being recycled as artificial turf, playground surfaces and, of course, bumper zones around go-kart and other racetracks. Should you wish to make your own 'tyre mulch', you'll need an industrial shredder and a fair bit of patience (**hsmag.cc/TyreMulch**), though a handful of UK companies have turned it into a business: **hsmag.cc/MonsterMulch**.

"Curiously, rubber tyres were invented many years before the golden age of motoring"

You probably don't have the means to break down your old car tyres in order to reuse them as cushioning at the bottom of a kids' slide but, if you have the outdoor space, a rope swing dangling from a tree or tyre swings attached to an A-frame are a reliable DIY build. If you have toddlers, a half-tyre rocking horse is also a popular build: **hsmag.cc/Rocking-Horse**.

Curiously, rubber tyres were invented many years before the golden age of motoring – the patent for the first vulcanised rubber pneumatic tyre was awarded to Robert William Thomson in 1846, while the better-known Goodyear tyre first appeared in 1844. We've been enjoying tyre swings and coming up with innovative ways to make use of discarded ones ever since, not least go-karts with tracks marked out by dozens of old tyres!

Right Rocking horses are still one of the most popular children's toys

GARDEN PLANTERS AND STORAGE

You don't need to look hard online to find examples of old tyres being reused singly, or in stacks, as perfectly serviceable plant containers. In fact, you'll find many an independent petrol forecourt ringed by reused rubber tyres used this way. Tyres cut in half can also make useful edges for garden beds, or built up to form a makeshift wall – simply pack with a mix of sand and soil so that they are heavy enough not to shift. One of our favourites is the Minion-themed planters assembled by Mancunian Pete Leicester to impress his wife, Gill. In common with most upcycled planters, the brightly painted tyres make a virtue of reusing an otherwise redundant item.

"Other obvious outdoor uses for old tyres are as water butts and compost heaps"

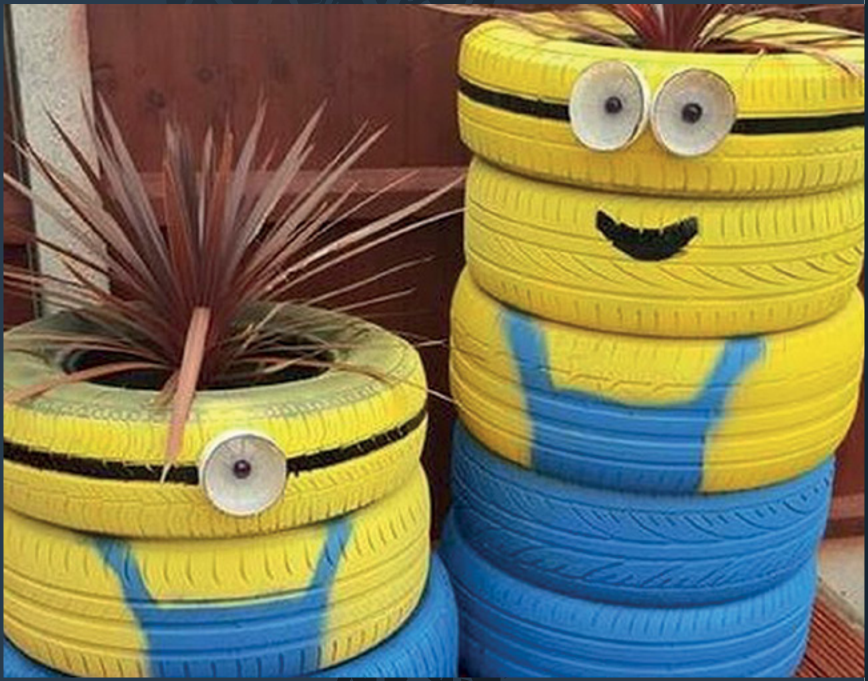
The tyres were free from a local junkyard, and Pete used the bottoms of plastic drinks bottles for the Minions' eyes, bringing the whole project in for a princely £3.

Other obvious outdoor uses for old tyres are as water butts and compost heaps, or as an instant store for plants pots and garden tools – just fashion a lid from discarded decking or pallet wood. →

Project Maker
Pete Leicester

Project Link
hsmag.cc/Minion-Planters

Below Other beloved fictional characters are possible



PET BED AND COMFY FOOTSTOOLS

Project Maker
AMARILDO SILVA

Project Link
instagram.com/amarildosilvaoficial

The curviness of a tyre can be quite enticing for little ones drawn to hide inside or to play in a makeshift sandpit. The cocoon also appeals to comfort-seeking pets keen to bed down somewhere secure. Lay down a squishy mat, place the tyre on top, and stuff a couple of pillowcases with wadding, or cut up old clothes and bedding, to form a soft interior for your lucky pooch. Brightly coloured paints, and perhaps animal-themed fabric, are almost obligatory. Brazilian Amarildo Silva took the concept of providing comfort for pooches further, by turning discarded tyres into makeshift beds for strays he encountered in his hometown of Campina Grande, even labelling each one with the dog's name and adding a homemade cushion. Art and architecture blog Dornob (among others) reported (**hsmag.cc/PetBeds**) that he then used the Instagram-friendly photos as a means of promoting kindness over street violence.

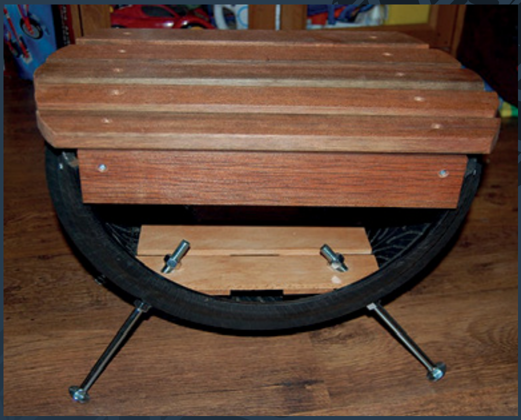
"The cocoon also appeals to comfort-seeking pets keen to bed down somewhere secure"

Right
Purr-fect for tyred pets

Below
A bit of paint can really brighten up an upcycled tyre

OUTDOOR TABLE

Painted, stacked tyres can work well as DIY outdoor furniture, but talented woodworker petachock's table is a more considered build. Half a car tyre became the base of a folding table. Once cut in half, the tyre is turned inside out – a clamp, foot, and brute force were needed to achieve this. Metal bolts were drilled in place as legs, and positioned at an angle. Wooden braces were added to provide stability, while curved supports ensure the tyre maintains its shape. A slatted wooden bench forms the table top element, with the added bonus that its wood was already treated for outdoor use and didn't need an additional protective coating.



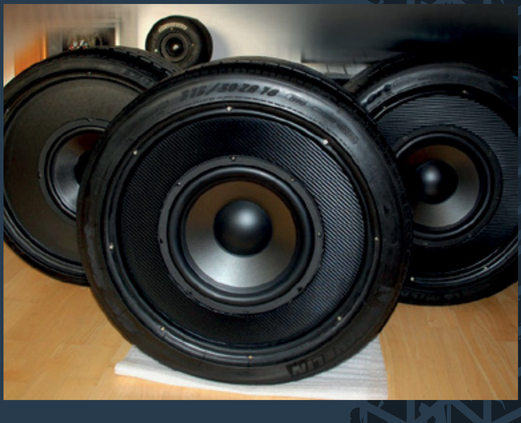
Project Maker
PETACHOCK

Project Link
hsmag.cc/Table-From-a-Tyre

Left There's no fiddly joinery with rubber, just bend it to the right shape

PERFORMANCE-TYRE SUBWOOFER

Car makeover shows may be reminiscent of the early 2000s, but the need for extremely loud speaker systems goes back to 1970s and 1980s block parties. In-vehicle sound systems have never really gone out of fashion, so it was little surprise to find bombastic speaker company and stretch Hummer aficionados, JBL, turn up at the 2019 Consumer Electronics Show in Las Vegas with the BassPro Hub tyre-based subwoofer: **hsmag.cc/Basspro-Hub-11,** while **99carstereo.com** even compares the best available spare tyre subwoofers. There's not much stopping you using a worn-out tyre to create your own, as aimzzz's Instructable shows. □



Project Maker
AIMZZZ

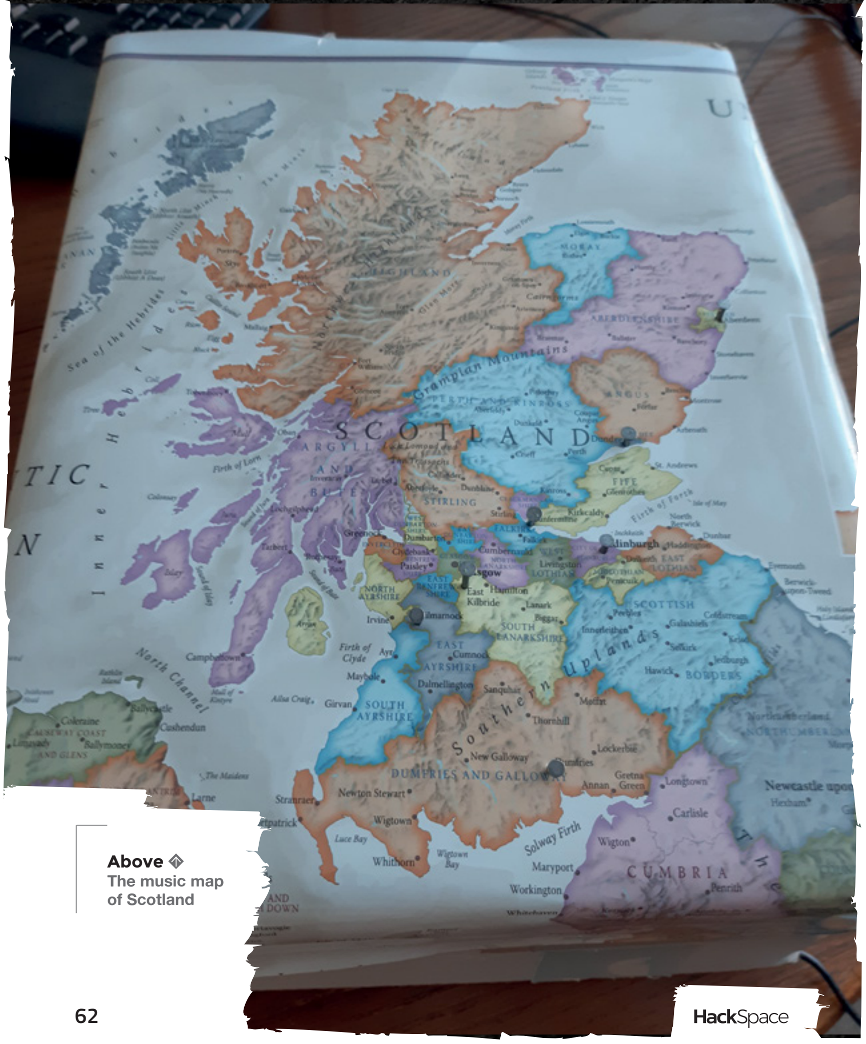
Project Link
hsmag.cc/Subwoofer

Left Less screeching tyres and more whoomp-whooming tyres

IN THE WORKSHOP: Capacitive touch map

By **Andrew Gregory**

Because maps need music



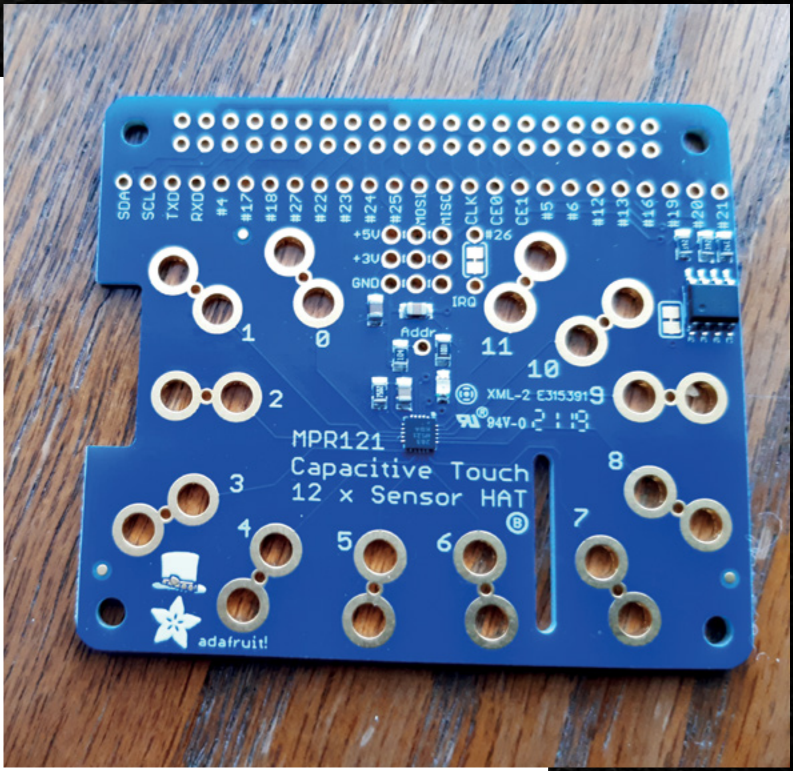
Above The music map of Scotland

We see loads of great-looking projects every week at HackSpace Towers. One of the most intuitive, literally hands-on builds we've seen, and something that has stayed with us since we first saw it, was the capacitive touch globe by Caroline Buttet. You touch a spot on the globe to pick an image gallery, then turn it to activate a rotary switch, and flick through the images in that album. If you haven't seen it, have a look: **hsmag.cc/GlobeTrotter**.

We've been working on a map of the United Kingdom, with a capacitive touch interface that you can touch to trigger an output that's relevant to the point on the map that you're touching. It's still in its prototype phase, as we're waiting for the price of hardwood to come down so that we can build a chunky frame to house the electricals, but for now, we're hard at work creating a proof of concept.

We need a map; we need something conductive to stick in the map (drawing pins would be ideal, but we only had the pins with the plastic bit on them, so we had to use small nails). We also need wires (crocodile clips in this case will do the job perfectly), a speaker, and a board that will enable us to program the logic.

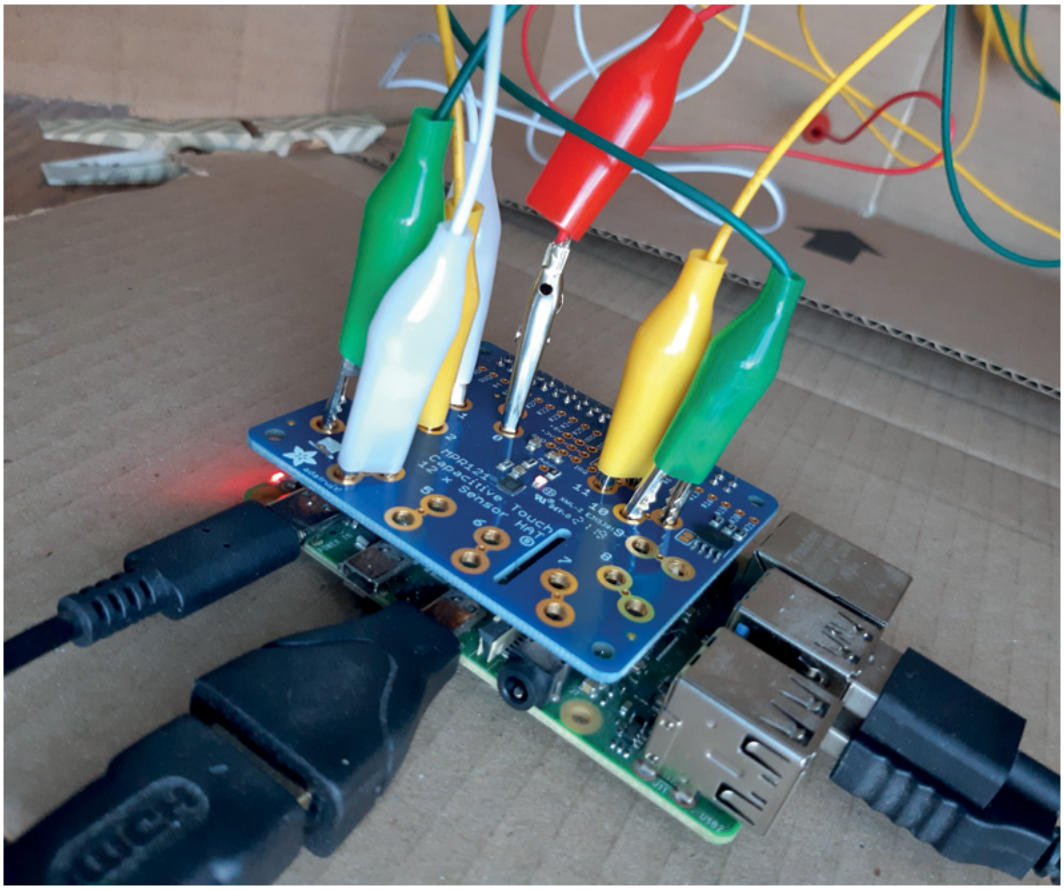
The musical map will play music related to the area on the map that the user touches: if you touch the nail stuck in Sheffield, you should hear some Arctic Monkeys or Human League; if you touch Oxford on the map, you'll get the soothing tones of Radiohead.



We could add shuffle, skip, and volume controls, but for now we just want to get a working prototype.

There are several boards on the market that will handle the (admittedly simple) duties of taking in a capacitive touch signal and turning it into something useful. The most common, and coolest use of capacitive touch in education, is to use it to make a keyboard, with a different piece of fruit standing for each note in an octave. If you wanted to take this seriously, with calculations done in real time, you'd need a Bela board. If you're more familiar with Arduino, the Touch Board from Bare Conductive gives you a load of connectivity options. We're going with Raspberry Pi though, because that's what we've got to hand.

Raspberry Pi doesn't offer capacitive touch capabilities out of the box; for that we need to add a HAT from Adafruit that will convert the conductivity →

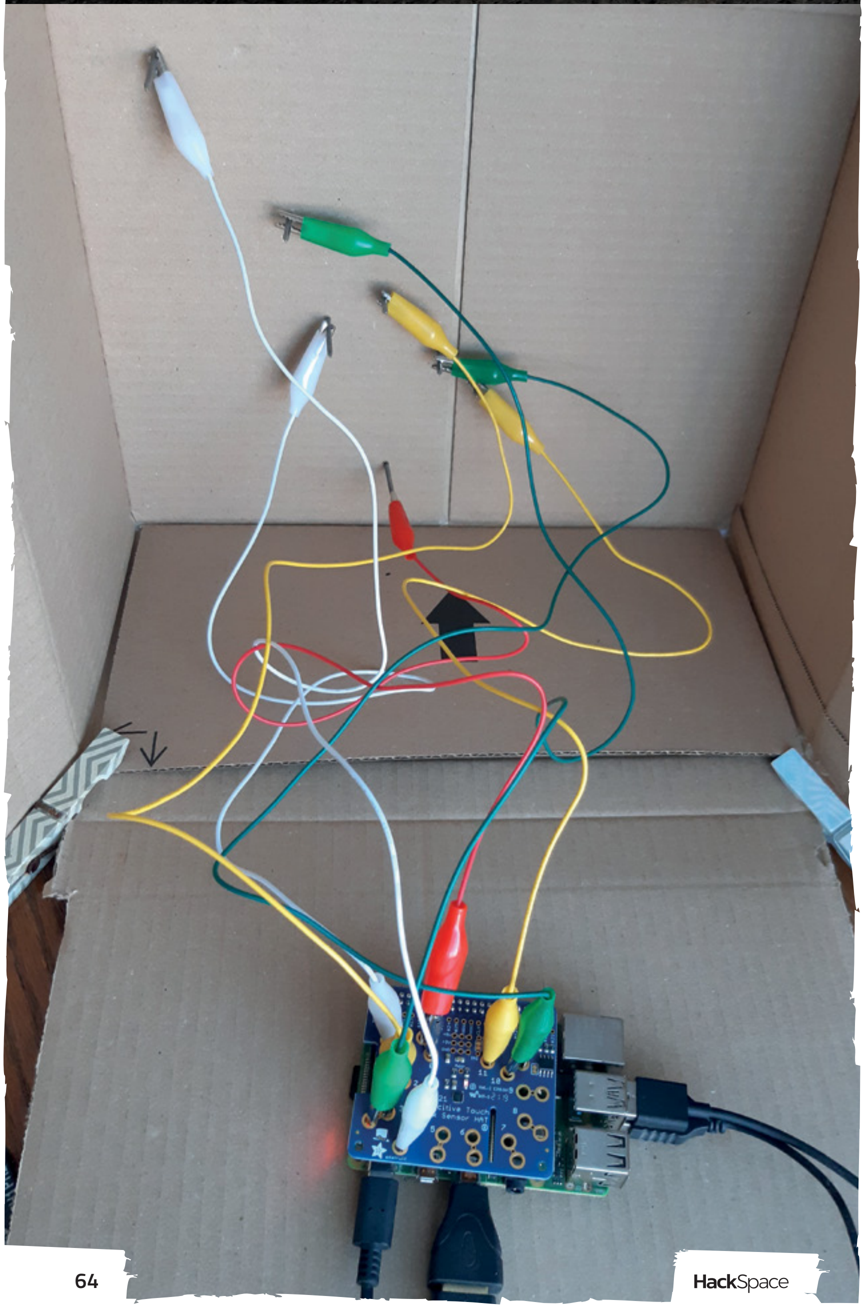


Above Raspberry Pi's inputs can't sense capacitance, so we used an Adafruit HAT

Left The Adafruit HAT we're using has twelve inputs, which coincidentally is the number of semitones in an octave

The map is far too big for the length of the wires we have

Below Soldering the HAT to the header is a quick job, and a test of consistency with the soldering iron



ABERDEENSHIRE
Ballater
Braemar
Mountains
KINROSS
Pitlochry
Dunkeld
ANGUS
Brechin
Montrose
Forfar
Coupar Angus
Perth
DUNDEE CITY
Arbroath
Cupar
St. Andrews
FIFE
Glenrothes
Kirkcaldy
Firth of Forth
Inchkeith
North Berwick
Edinburgh

Above Compiling the list of cities and bands was the hardest part

of our human skin into something that the computer can understand. For this purpose we've chosen the Adafruit Capacitive Touch HAT for Raspberry Pi MPR121, which is a snip at $14.95.

This HAT isn't the latest – it was first released in 2015 – but that's because the underlying technologies that it's built on have not changed since it was released. Raspberry Pis still have 40 GPIO pins, and still use Python. And human bodies, the last time we checked, still conduct electricity.

And so, we come to our first error: the first thing we did when we got the HAT was drop it onto the Raspberry Pi, eager to get started. This then crashed the Raspberry Pi. What you need to do, as we intuited, was to turn the Raspberry Pi off, then attach the HAT, then turn the Raspberry Pi back on. It sounds silly when you say it, but it was an honest mistake, and we believe in learning from those.

Second error: the map is far too big for the length of the wires we have. Let's scale this down, and focus our celebration of musical excellence on Scotland. We cut it to size, then stuck the map onto a cardboard box, and pushed nails through Aberdeen, Dundee, Dunfermline, Edinburgh, Dumfries, Kilmarnock, and East Kilbride, connecting the HAT to the back sharp bits of the nails on the inside of the box.

Now it's time to make sure that we're running the latest version of CircuitPython; Python may be as old as time, but CircuitPython is a relatively modern offshoot. Adafruit made the board, so we searched **circuitpython.org/downloads**, looking for the capacitive touch HAT Adafruit MPR121 (the

device's full name). It's not there. Never mind, Raspberry Pi 4 Model B is there, so that's what we used. This told us that the most recent version of CircuitPython for the Raspberry Pi 4 is 7.2.0 (as it is for almost everything).

We already had CircuitPython installed from an earlier project, so we just needed to get the libraries for the HAT with a quick command-line program. You can grab these with the following command (we used the Thonny code editor):

```
sudo pip3 install adafruit-circuitpython-mpr121
```

Adafruit has a chunk of sample code to get the MPR121 HAT working, which we copied from **hsmag.cc/CapacitiveTouchCode** (it's down at the bottom of that web page):

```
import time
import board
import busio
import adafruit_mpr121
i2c = busio.I2C(board.SCL, board.SDA)
mpr121 = adafruit_mpr121.MPR121(i2c)
```

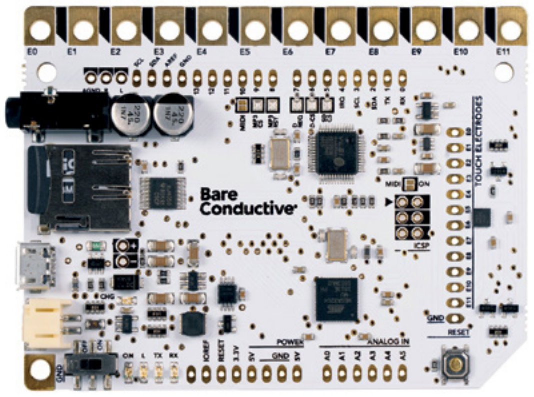
The key lines that we have to change are here:

```
# Loop forever testing each input and printing
when they're touched.
while True:
    # Loop through all 12 inputs (0-11).
    for i in range(12):
        # Call is_touched and pass it then number
of the input.  If it's touched
        # it will return True, otherwise it will
return False.
        if mpr121[i].value:
            print("Input {} touched!".format(i))
```

Rather than printing out 'Input 1 is touched', we want to play a song. There are many ways to do this, but as we're using the same Raspberry Pi that's hooked up to our work screen and speakers, we're going to use it to launch the trusty VLC media player to play the song.

We can use Python's OS module to run any commands we want on the system, so we just need to use these to launch the music player with the right song.

At least, that's the theory. In practice, we've been having some problems, though as always, we're sure that these are nothing more than precursors to future success. We'll let you know with a glorious video when it all comes together!



Above We had to space the nails out, so if there were two excellent bands from adjacent cities, we had to choose between them. Thanks to The Jesus And Mary Chain, East Kilbride got the nod ahead of Glasgow

Left Another option would have been to use Bare Conductive's touch sensitive music player

HackSpace
TECHNOLOGY IN YOUR HANDS

FORGE

HACK | MAKE | BUILD | CREATE

Improve your skills, learn something new, or just have fun tinkering – we hope you enjoy these hand-picked projects

PG 74
MONITOR PLANTS
Digital help for those lacking in green fingers

PG 78
PCB BUTTONS

Turn FR4 into an input device

PG 82
LOGIC CHIPS
Who needs programming when you've got solder?

PG 68
SCHOOL OF MAKING
Start your journey to craftsmanship with these essential skills

68 Pico Keyboard

PG 88
LASER CUTTER ENHANCEMENT
Upgrade your optics

PG 94

FOLDING METAL
Designing strong parts

Make a chord keyboard with Raspberry Pi Pico and CircuitPython

Make a keyboard you can use in the dark with one hand

Rob Miles

@robmiles

Rob Miles has been playing with hardware and software since almost before there was hardware and software. You can find out more about his so-called life at **robmiles.com** and follow him on Twitter at **@robmiles**

If you've ever wanted to be able to type without looking at the keyboard, this project is for you.

We're going to make a USB keyboard you can use to enter text quickly and unobtrusively, even in the dark. The keyboard has a key for each finger, and you type by pressing down 'chords' – combinations of different keys. There are lights in the keys to help you learn the combinations for each letter and a teaching game you can use to practice your typing skills. You can plug it into a USB port on a computer and use it as you would any other keyboard.

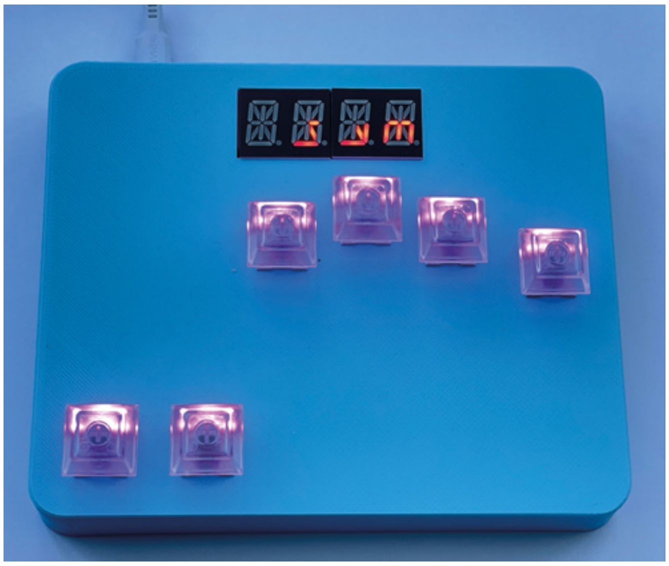


Figure 1 The keyboard can be used to enter text, numbers, and symbols. When entering symbols, the keys are lit up in magenta

THE PICO CHORD KEYBOARD

Figure 1 shows the device. Six illuminated keys are used to enter chords, and a four-character text display gives user feedback. The keyboard can be used to enter letters, numbers, and symbols. It has a teaching game that you can use to build your skills. It can also be used to type out its own documentation, in the form of a chart giving all the chord designs.

You can use any kind of switch in the keyboard, but this project uses the Adafruit NeoKey breakout board (**adafruit.com/product/4978**) and individual key switches.

The keyboard is powered by a Raspberry Pi Pico device running a CircuitPython program that scans the keys and sends them out over a USB connection to a computer.

Figure 2 shows a couple of Adafruit NeoKey breakout boards connected to a Raspberry Pi Pico. The NeoKey contains a socket for a CHERRY MX-compatible key switch. It also contains a NeoPixel that can be used to illuminate the key.

If you look carefully at **Figure 2**, you'll notice that there are two ground connections for each NeoKey. These are black wires at the bottom of the image. This is because a NeoKey board contains a diode that is connected in series with the key switch. This is useful if you want to use the boards in a keyboard matrix, but it slightly complicates our wiring because each key needs a ground connection for the power to the NeoPixel LED and another to bias the diode in the switch circuit.

Figure 3 shows the inside of the box. The keys are pushed into the outside of the case through square holes and the NeoKey boards are then plugged onto the back of the key inside the case. The Raspberry Pi Pico fits alongside the display to reduce the height

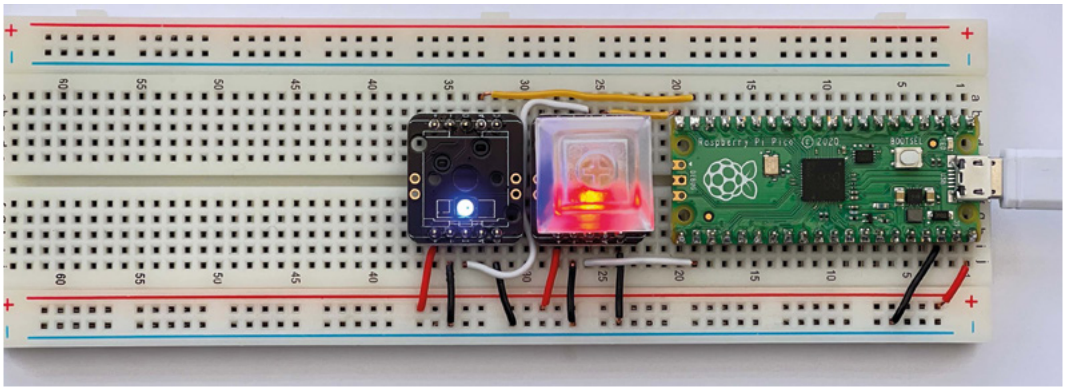


Figure 2 The key switch on the left has been removed to show the NeoKey breakout underneath. The white wires link the NeoPixels on each key and the yellow wires are used for the data signals from each key to the Pico. The pixels are connected to GPIO 17 on the Pico and the switches are connected to GPIO 14 and GPIO 15

of the case. The cables are colour-coded. Red and black wires provide the power, yellow wires transfer the signal for the NeoPixel, and the other colours are signals for the switches.

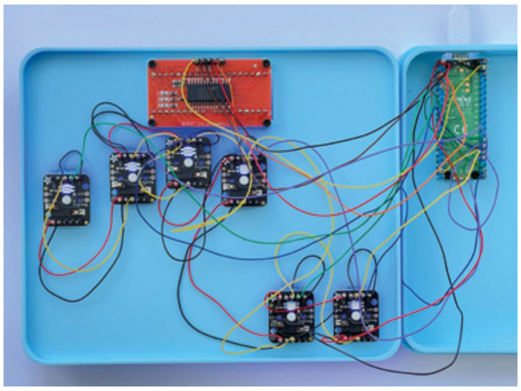
Figure 4 (overleaf) shows the complete circuit diagram for the keyboard. You can use different GPIO pins in your circuit, and even change the order of the LED colours by modifying the code that creates the **keyboard** object in the program.

```
keyboard = PicoChord (i2c_sda=board.GP0, i2c_
scl=board.GP1,
            pixel_pin=board.GP17,
            key_switches=[
                Switch(pin=board.
GP15,pixel=0,bit=1),  # control
                Switch(pin=board.
GP14,pixel=1,bit=2),  # thumb
                Switch(pin=board.
GP13,pixel=2,bit=4),  # index
                Switch(pin=board.
GP12,pixel=3,bit=8),  # middle
                Switch(pin=board.
GP11,pixel=4,bit=16), # ring
                Switch(pin=board.
GP10,pixel=5,bit=32)  # little
                ])
```

The code above creates an instance of the **PicoChord** class by calling the **constructor** method for that class. The constructor sets the GPIO connections used for the I2C connection to the text display board, the NeoPixel display, and the individual switches. You can use this code to help you wire up the components to Pico.

The CircuitPython program and all the required libraries are available on the GitHub site for this project which you can find at **hsmag.cc/ChordKeyboard**. →

Figure 3 This shows the underneath of the keyboard. The keys for the thumb are on the right



THE MICROWRITER AND THE AGENDA

The Microwriter was launched in 1978 at the very start of the personal computer revolution. It was one of the first-ever portable word processors. To aid portability, it used a chord keyboard. Text was entered by pressing down combinations of the six keys on top of the device. The concept was developed into the Microwriter AgendA personal organiser which was released in 1982 and had both letter keys and a chord keyboard for text entry.

The AgendA was very advanced for its time. It contained a large custom chip and was one of the first devices to use the I2C (inter-integrated circuit) bus to connect peripherals and memory cards. The author got one when it was first released and still has it, although its rechargeable batteries have long since died. The AgendA was not the success it deserved to be. Customers were wary of the chord keyboard and competitors from Psion with familiar QWERTY keyboards took over the personal organiser market. However, the chord entry keyboard lives on – search for 'cykey' to find out how to get one.

YOU'LL NEED

- **6 × key switches** The Kailh Mechanical Keyboard Switches work well. You can get them in 'clicky' or 'non-clicky' form
- **6 × transparent keycaps**
- **A four-character alphanumeric display with an HT16K33 backpack** Search for 'ht16k33 14 segment led'. Make sure that you get the 14-segment device as this can display text
- **A micro USB cable to link the Pico to the host**
- **Connecting wire (search for '30 awg wire wrap') which needs a wire wrap tool** (search for 'wire wrap tool')
- **A box** There is a 3D-printable design, or you can put your keyboard in any box you fancy
- **Screws** You'll need some screws sized M2 4 mm in length to fix things to the case (search for 'laptop screws')

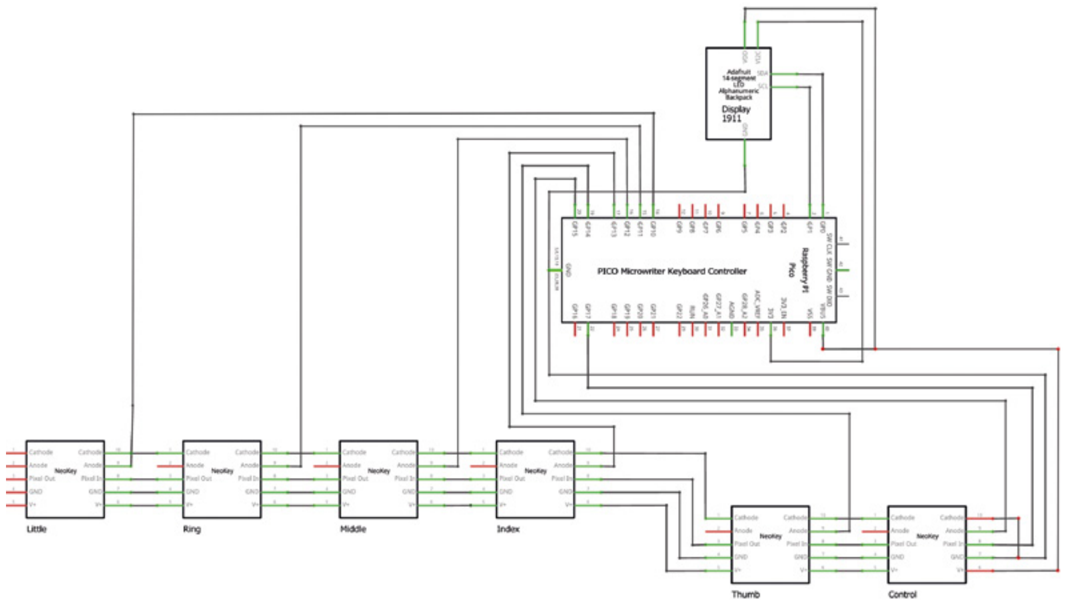
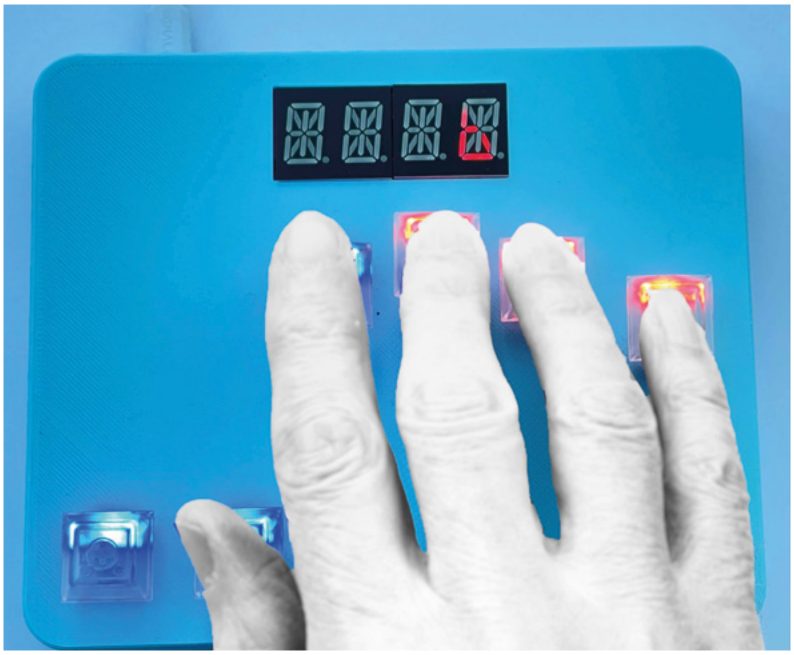

Adafruit 14-segment LED Alphanumeric Backpack
Display 1911
PICO Microwriter Keyboard Controller
Raspberry Pi Pico
NeoKey
Little
Ring
Middle
Index
Thumb
Control

Figure 4 If you don't want to use the NeoKey boards, you can connect switches directly between the GPIO pins on the Pico and ground. You could then use a separate string of NeoPixels for the key lights

Figure 5 The keys are blue to indicate that lower case text is being entered. The keys turn red when pressed. The text display is showing a 'b' which is represented by a chord made up of the three keys at the right-hand side of the keyboard

CHORD KEYBOARDS

The traditional QWERTY keyboard has been around for a very long time. But it does have its disadvantages. You need to use both hands to type quickly, and unless you are a skilled touch-typist, you will have to look at your hands while you're entering text. When you use a QWERTY keyboard, you usually press one key to enter a character. The word 'chord' is borrowed from music, when you play multiple notes together on a keyboard to make more interesting sounds. When you use a chord keyboard, you press a particular combination of keys to select a given character.



A chord keyboard can be used with one hand. Once you've learned the letter shapes, you can type without looking at the keyboard because your fingers never move off their 'home' keys. A chord can be much smaller than a QWERTY device, making it more portable.

Figure 5 shows a ghostly white hand entering text on the keyboard. Chord keyboards don't register characters when the keys are pressed. Characters are detected when the user takes their finger off the keys. This is because the keyboard can only detect that a complete chord has been entered when the user lifts their finger off one of the pressed keys. For example, you press three keys to enter a 'b', as shown in **Figure 5**. These keys also form part of the letter 'm', which is represented by four keys. The keyboard can only determine that the chord for a 'b' has been entered when one of the three keys in the chord is released. The user must then lift all their fingers before entering the next key. The good news is that with a bit of practice, you can enter text surprisingly quickly. The keyboard contains chords for text, numbers, and a range of symbols.

The key at the far left of the keyboard is the 'control' key. The thumb presses this key to enter chords that control the keyboard. Pressing the control key on its own will toggle the shift status of the keyboard so you can change between capital and lower case letters. When in shift mode, the keys light up yellow.

READING THE CHORDS

The keyboard converts chords into character codes to be sent over USB to the connected computer. The first step in the conversion is to represent every chord by a unique numeric value. We do this by mapping each key in the chord onto a particular bit in that number, as shown in **Figure 6**. For example, the letter 'b' that we typed in **Figure 5** would be represented by 8 + 16 + 32, which gives the value 56. Each time a key is pressed, the keyboard adds the 'bit' value of that key to a total that is stored in a variable called `bits`. When a key is released, the value in bits represents the chord that was entered. The program then uses a Python dictionary to look up the character that the chord represents.

```
self.text_decode = {
    12:'a', 56:'b', 10:'c', 14:'d', 4:'e', 30:'f',
48:'g', 34:'h',
    6:'i', 50:'j', 18:'k', 38:'l', 60:'m', 24:'n',
8:'o', 62:'p',
    40:'q', 22:'r', 16:'s', 20:'t', 32:'u', 36:'v',
54:'w', 58:'x',
    26:'y', 42:'z', 2:' ', 52:',',  28:'.'}
```

The code above shows the dictionary `text_decode` which is used to decode the keyboard chord values. The code for 'b' is clearly indeed 56. You could use this dictionary to work out the key combinations for all the letters. Some letters, for example 's', 'e', and 'u', are represented by a single key.

```
key = self.text_decode[bits]
```

The statement above uses the value in `bits` to access the `text_decode` dictionary and extract the character that the chord represents. The variable key would be set to the character that has been entered. The keyboard software contains two other dictionaries for symbols and command codes.

Once the keyboard has the key character, the next thing to do is send that character out from the keyboard over USB.

```
self.usb_layout.write(key)
```

The statement above writes out a string from the keyboard. The USB connection is managed by code from the usb_hid libraries in CircuitPython.

```
import usb_hid
from adafruit_hid.keyboard import Keyboard
# Import the keyboard layout description
from adafruit_hid.keyboard_layout_uk import
KeyboardLayoutUK

self.usb_kbd = Keyboard(usb_hid.devices)
# Set the required keyboard layout
self.usb_layout = KeyboardLayoutUK(self.usb_kbd)
```

The statements above import the required libraries and then create the keyboard connection for use by the program.

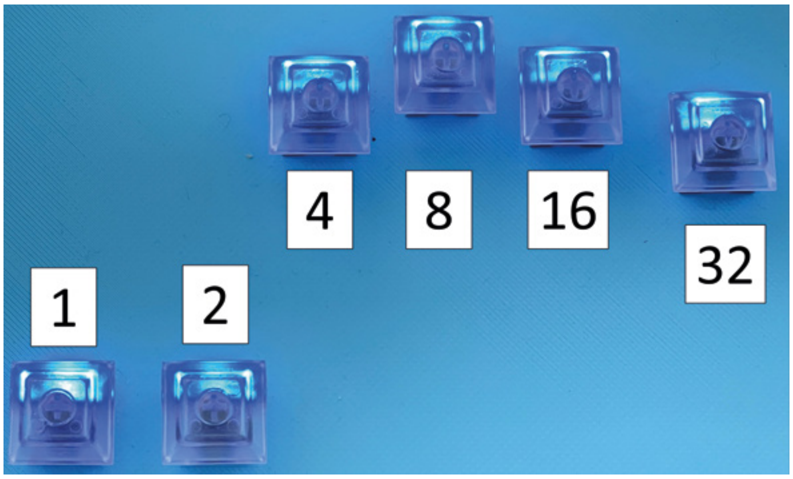


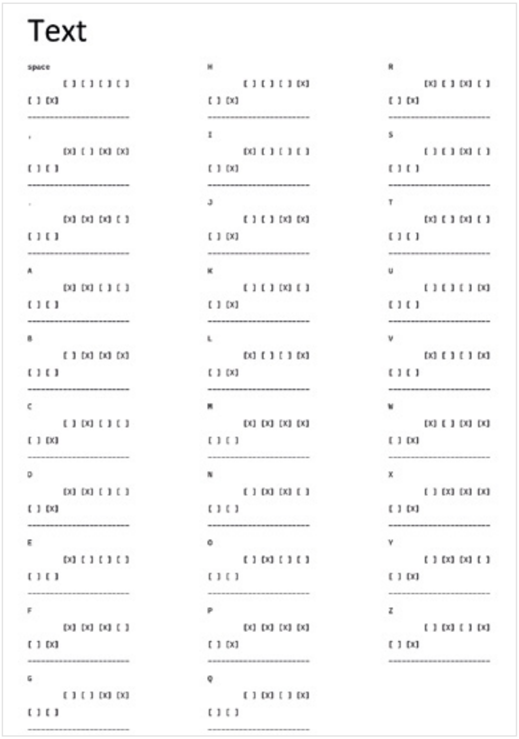
Figure 6 Each of the keys represents a different binary bit in the number that represents that chord value

GETTING HELP

The chord for 'b' is entered by pressing and releasing the rightmost three keys on the keyboard. We could work out the chords for the other characters by studying the `text_decode` dictionary above but what we really want is a document that gives the chords for every character and symbol. The best way to get this is to generate it automatically from the `text_decode` dictionary. This ensures the chord designs are only stored in one place, and if we change any of the chords or add new ones, the documentation updates automatically.

```
def print_key(self,symbol,bits):
    print("Printing:",symbol,bits)
    self.send_animated_text_to_keyboard(symbol)

    top_line = "        "
    bottom_line = ""
    for bit in (1,2,4,8,16,32):
        if (bits & bit) == 0:
            text="[ ] "
        else:
            text="[X] "
        if bit < 4:
            bottom_line = bottom_line + text
        else:
            top_line = top_line + text
    self.send_animated_text_to_keyboard(top_line)
    self.send_animated_text_to_keyboard(bottom_
line)
    self.send_animated_text_to_
keyboard("----------------------")
```

The `print_key` function is given a symbol and a bit pattern. It prints out a tiny description of the chord for that key. It tests each bit in the bits value and selects `[X]` or `[ ]` based on whether or not that key is used in the chord for that character. The key information is added to the top or bottom line of the description depending on which bit is being processed. →



Text

```
space
        [ ] [ ] [ ] [ ]
[ ] [X]
-----------------------
,
        [X] [ ] [X] [X]
[ ] [ ]
-----------------------
.
        [X] [X] [X] [ ]
[ ] [ ]
-----------------------
A
        [X] [X] [ ] [ ]
[ ] [ ]
-----------------------
B
        [ ] [X] [X] [X]
[ ] [ ]
-----------------------
C
        [ ] [X] [ ] [ ]
[ ] [X]
-----------------------
D
        [X] [X] [ ] [ ]
[ ] [X]
-----------------------
E
        [X] [ ] [ ] [ ]
[ ] [ ]
-----------------------
F
        [X] [X] [X] [ ]
[ ] [X]
-----------------------
G
        [ ] [ ] [X] [X]
[ ] [ ]
-----------------------
H
        [ ] [ ] [ ] [X]
[ ] [X]
-----------------------
I
        [X] [ ] [ ] [ ]
[ ] [X]
-----------------------
J
        [ ] [ ] [X] [X]
[ ] [X]
-----------------------
K
        [ ] [ ] [X] [ ]
[ ] [X]
-----------------------
L
        [X] [ ] [ ] [X]
[ ] [X]
-----------------------
M
        [X] [X] [X] [X]
[ ] [ ]
-----------------------
N
        [ ] [X] [X] [ ]
[ ] [ ]
-----------------------
O
        [ ] [X] [ ] [ ]
[ ] [ ]
-----------------------
P
        [X] [X] [X] [X]
[ ] [X]
-----------------------
Q
        [ ] [X] [ ] [X]
[ ] [ ]
-----------------------
R
        [X] [ ] [X] [ ]
[ ] [X]
-----------------------
S
        [ ] [ ] [X] [ ]
[ ] [ ]
-----------------------
T
        [X] [ ] [X] [ ]
[ ] [ ]
-----------------------
U
        [ ] [ ] [ ] [X]
[ ] [ ]
-----------------------
V
        [X] [ ] [ ] [X]
[ ] [ ]
-----------------------
W
        [X] [ ] [X] [X]
[ ] [X]
-----------------------
X
        [ ] [X] [X] [X]
[ ] [X]
-----------------------
Y
        [ ] [X] [X] [ ]
[ ] [X]
-----------------------
Z
        [ ] [X] [ ] [X]
[ ] [X]
-----------------------
```

Figure 7 The chord output was formatted into columns using a word processor

```
self.print_key('b', 56)
```

The statement above would print out the chord for the character 'b'.

```
b
        [ ] [X] [X] [X]
[ ] [ ]
-----------------------
```

Above is the description of the chord for 'b'. The top line gives the character and the next two show which keys to press. This text is transmitted from the keyboard using the `send_animated_text_to_keyboard` function. This sends characters to the host computer, but also displays them on the keys on the keyboard. This means that if you want a document for all the chords, you just open an empty document, put the text cursor inside the document, and press the Help command. You can then watch as the keyboard magically types its documentation for you.

Figure 7 gives the chord designs for text entry. There are also tables for number, symbol, and command modes. You can download them from the GitHub site for the project from this link: **hsmag.cc/ChordKeyboard**.

TAKING CONTROL

The keyboard uses control commands to set the input modes, trigger the help output, and play a training game that is used to learn the chords. Control chords are entered by pressing the leftmost key with your thumb and then typing the rest of the chord on the other keys:

```
del : Backspace and delete
        [X] [X] [ ] [ ]
[X] [ ]
-----------------------
```

This is the control chord that you can use to delete a character and move the cursor back. It is equivalent to pressing the **BACKSPACE** key. The control key is held down, along with the chord that makes up the letter 'd' (for delete). The control keys are decoded in a similar way as the text keys using a dictionary. The program looks up the bits value in the `command_actions` dictionary and returns an object that is used to process that command.

```
    1: ("caps", lambda x:self.do_toggle_caps_
lock(),"Toggle CAPS lock"),
    13:("del", lambda x:self.usb_kbd.send(Keycode.
BACKSPACE),"Backspace"),
    21:("text", lambda x:self.start_lower_case_
text(),"Set text mode (blue)"),
    25:("num", lambda x:self.start_number_
text(),"Set numeric mode (green)"),
    29:("for", lambda x:self.usb_kbd.send(Keycode.
RIGHT_ARROW),"Right"),
    44:("ret", lambda x:self.usb_kbd.send(Keycode.
ENTER),"Enter key"),
    57:("bak", lambda x:self.usb_kbd.send(Keycode.
LEFT_ARROW),"Left"),
    33:("Help", lambda x:self.start_help(), "Type
help information"),
    49:("Game", lambda x:self.start_game(), "Start
the game")
    }
```

The code above shows the `command_actions` dictionary. This holds command descriptions in the form of a Python 'tuple'. A tuple is a lump of data that contains multiple values. In the case of the command, the tuple contains three things. These are the text that is displayed for the command, a reference to the function that will deal with the command, and a string that is displayed in the Help

MICROWRITING AND THE MOVIES

One of the people behind the Microwriter was the movie director and inventor Cy Endfield who devised the key combinations for each character. Perhaps his most famous movie was *Zulu*, which was released in 1964 and starred Michael Caine.

for that command. The command functions are expressed as 'lambda' functions.

A lambda function is a lump of code with no name that can be dropped into a program anywhere Python expects a value. Some of the command functions just send a particular key from the keyboard (for example, the **DEL** key). Other command functions (for example, the Game function) call an internal function inside the keyboard class.

```
if bits in self.command_actions:
    # The dictionary contains a command for this
bit pattern
    # Get the command description tuple from the
dictionary
    command = self.command_actions[bits]
    # extract the name from the tuple
    name = command[0]
    # extract the function from the tuple
    function = command[1]
    # display the name
    print("Control:",name)
    self.display_text(name)
    # call the function
    function(self)
```

The code above shows how the `command_actions` dictionary is used to decode commands. The Python keyword 'in' allows the program to determine if there is an entry for the bits value in the `command_actions` dictionary. If there is, the command tuple is obtained from the dictionary. The first item (the item with the subscript 0) in the command tuple is the name which is printed on the console and displayed on the LEDs. The second item (the item with the subscript 1) is the function that is to be called to perform the selected command action.

If you find this talk of tuples and lambdas confusing, just remember that the program needs to store three items (name, function, and description) in the command description (that's what a tuple is for), and the command description needs to contain something to connect to the function that performs the command action (that's what the lambda is for).

GAMES FOR LEARNING

The keyboard contains a teaching game that you can use to learn the chords. Characters are displayed and you must enter the chord for that character. After a second, the keyboard displays the chord for that character on the keys. If you enter an incorrect chord, the game ends and your score is displayed. You get one point for every correct chord and five points for a correct chord entered before the help was displayed.

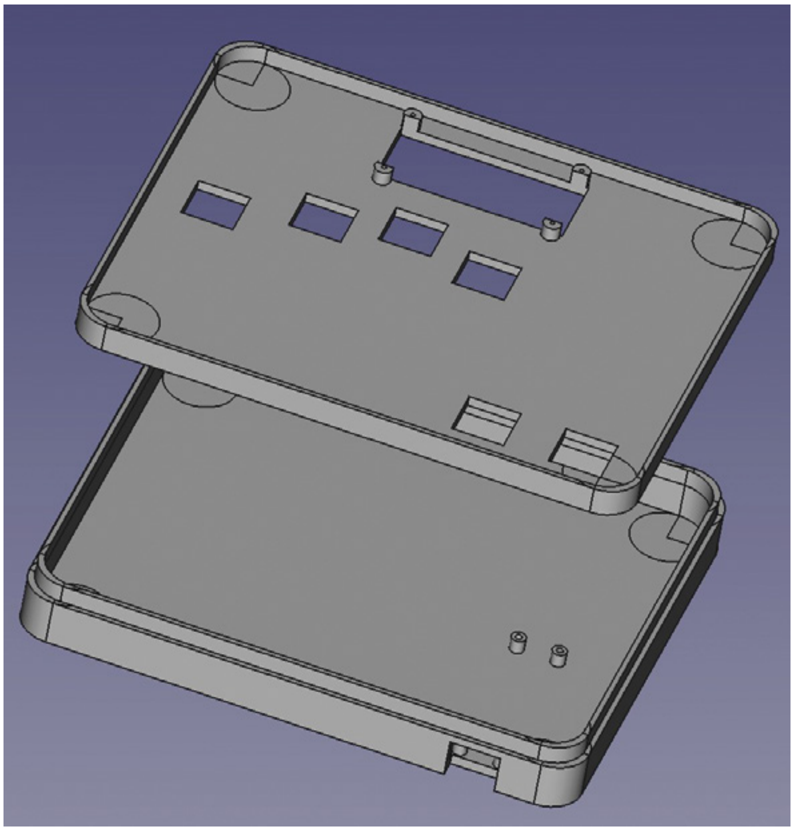


Figure 8
You can modify the spacing of the keys by editing the FreeCAD macro used to generate the case design

FUN WITH KEY CODES

You must have had the experience where the key that you press on your keyboard doesn't match the character that is displayed on the screen. You see it when you plug a keyboard from one locale (perhaps the UK) into a machine configured for another (perhaps the US). It happens because a USB keyboard sends out 'key code' values when keys are pressed. Each key code is mapped to a physical key on the keyboard. The keyboard uses the CircuitPython USB library to send key codes to the host computer. The CircuitPython library uses a lookup table to map characters sent from the Python program to USB key codes. The layout table is imported into the program and then used to send key codes.

```
# Import the keyboard layout description
from adafruit_hid.keyboard_layout_uk import
KeyboardLayoutUK

# Set the required keyboard layout
self.usb_layout = KeyboardLayoutUK(self.usb_kbd)
```

While chord keyboards might seem obscure, with a bit of practice, they can be quick and easy ways to enter text. You should now have the knowledge to build your own chord keyboard and get typing! □

Part 04!

Explore the sensory world: Make a plant monitor

MAKER

Phil King

Long-time contributor to *The MagPi*, Phil is a freelance writer and editor with a focus on technology.

@philkingeditor

Check the moisture level of your plant's soil and send an alert if it's too dry

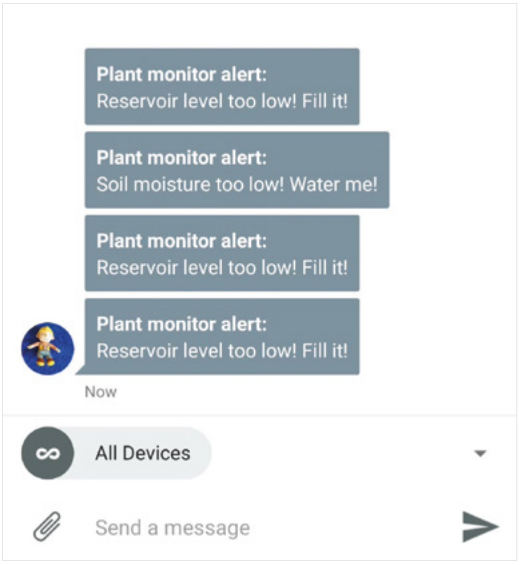
In this series, we are exploring some of the most commonly available sensors and their use cases. Previously, we have built a couple of alarms – one for fire and gas safety, the other for detecting intruders – and a basic weather station.

In the latter, we made use of an ADC to read the analogue output of a UV sensor. This time we'll be using an ADC to check the value from a moisture sensor placed in a plant pot. We'll also add a liquid level sensor to check the water level in a reservoir (saucer) holding the plant pot.

If the soil or water reservoir is dry, the monitor will send us a Pushbullet alert telling us to water the plant to keep it healthy.

You'll Need

- Moisture sensor **magpi.cc/moisturesensor**
- Liquid level sensor **magpi.cc/liquidlevel**
- MCP3008 ADC **magpi.cc/mcp3008**
- Jumper wires



Plant monitor alert:
Reservoir level too low! Fill it!

Plant monitor alert:
Soil moisture too low! Water me!

Plant monitor alert:
Reservoir level too low! Fill it!

Plant monitor alert:
Reservoir level too low! Fill it!

Now

Pushbullet notifications can be sent to a smartphone or the Chrome browser on another computer

01 Connect the ADC

Since the two sensors we'll be using for the plant monitor both output their readings as an analogue signal, we'll need to convert that to a digital value using an ADC (analogue-to-digital converter) chip. As before, we're using the MCP3008 ADC, which has eight input channels.

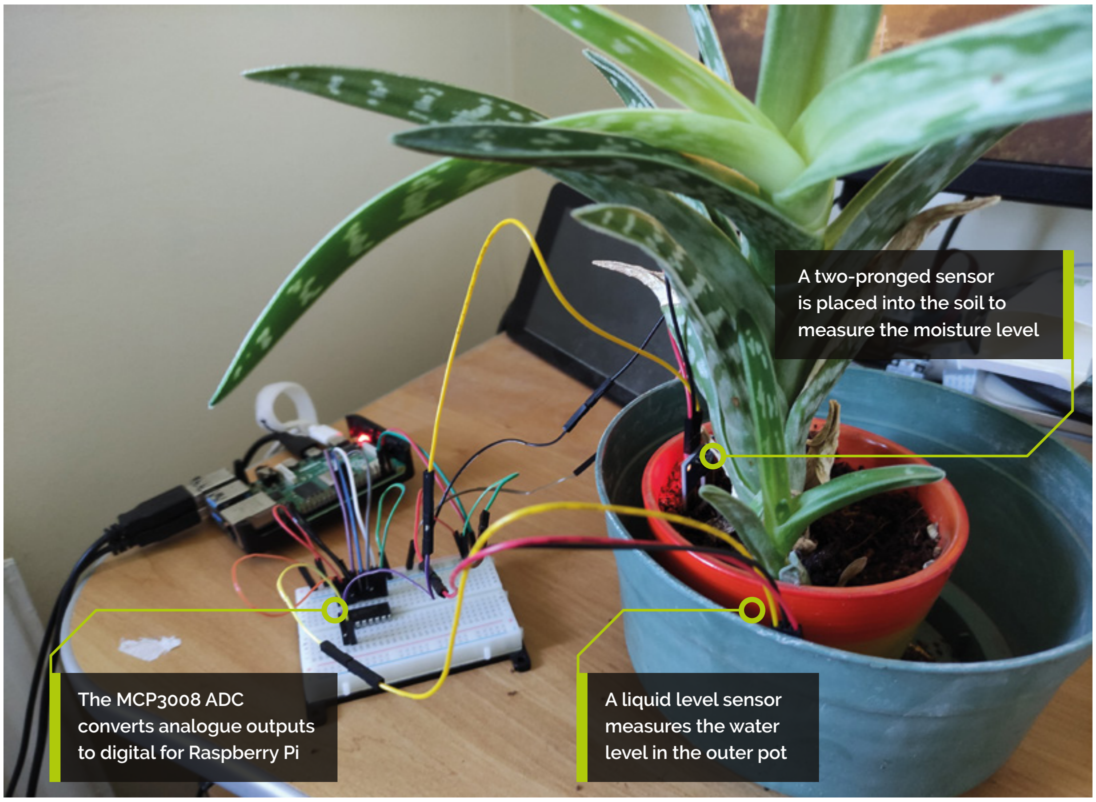
As the ADC chip makes use of the SPI interface, we'll need to enable SPI in the Raspberry Pi Configuration tool. It's also best to enable full SPI support in Python 3. To do so, open a Terminal window and enter:

```
sudo apt-get install python3-spidev
```

Now, with the power to Raspberry Pi turned off, it's time to connect our ADC – you may already have it set up from the last instalment in this series. Place the MCP3008 in the middle of the breadboard, straddling its central groove. Make sure it's the correct way round, as shown in the **Figure 1** wiring diagram, with the top of writing on top of the ADC nearest Raspberry Pi.

Now connect the jumper wires as in **Figure 1**. Two go to the '+' breadboard power rail, connected to a 3V3 pin; two others are connected to a GND pin via the '–' rail. The four middle legs of the ADC are connected to the SPI interface on Raspberry Pi: GPIO pins 8 (CE0), 10 (MOSI), 9 (MISO), and 11 (SCLK).

Double-check they are all connected to the correct pins otherwise it won't work properly and you may even damage the ADC chip or your Raspberry Pi.


A two-pronged sensor is placed into the soil to measure the moisture level
The MCP3008 ADC converts analogue outputs to digital for Raspberry Pi
A liquid level sensor measures the water level in the outer pot

THE MAGPI

This tutorial is from in The MagPi, the official Raspberry Pi magazine. Each issue includes a huge variety of projects, tutorials, tips and tricks to help you get the most out of your Raspberry Pi. Find out more at **magpi.cc**

" Double-check they are all connected to the correct pins otherwise it won't work properly "

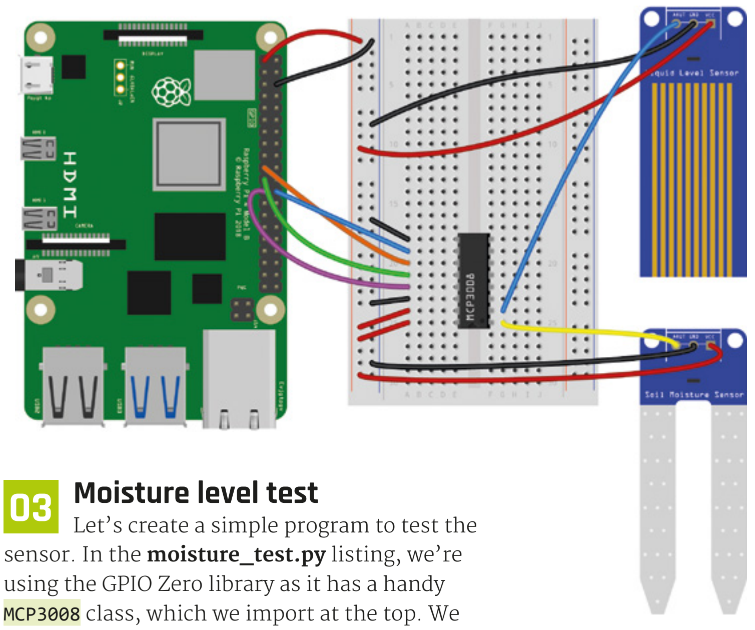
02 Connect the moisture sensor

With the ADC wired up correctly to Raspberry Pi, we can now add our moisture sensor to the setup. We're using one from the Waveshare Sensors Pack available in the UK from The Pi Hut (**magpi.cc/wavesensors**). The sensor is available separately, too, and there are also alternative soil moisture sensors you could use.

As in **Figure 1**, we connect the sensor's VCC pin to 3.3 V via the breadboard power rail, and its GND pin to GND on Raspberry Pi via the breadboard ground rail.

Since we want to place the sensor in a plant pot, we're using long female-to-female jumper wires (as supplied in the Waveshare Sensors Pack) to extend the distance from our Raspberry Pi and breadboard.

Finally, we connect the sensor's AOUT (analogue out) pin to the MCP3008's channel 0 pin, as shown in **Figure 1**. You could wire it to any of the eight channels on that side, but we're using this one in our code.



▲ **Figure 1** The wiring diagram for the plant monitor, including the liquid level sensor, soil moisture sensor, and ADC

03 Moisture level test

Let's create a simple program to test the sensor. In the **moisture_test.py** listing, we're using the GPIO Zero library as it has a handy `MCP3008` class, which we import at the top. We assign the moisture variable to the MCP3008's channel 0 to read the connected sensor's analogue output.

In a `while True:` loop, we multiply the digitally converted sensor output (which ranges from 0 to 1) by the 3300 maximum voltage (in microvolts) to get an accurate reading. In our `print` statement, the `%-3.2f` format parameter sets each output to a minimum three digits including two decimal →

Top Tip

Add a display

To see at a glance how your plant is doing, you could add a mini LCD display to your Raspberry Pi to show the sensor readings.

places. You can alter this to your preference. We add the `end = "\r"` parameter so that the message is always printed on the same line.

If the sensor is not placed in anything, the reading should be very low (ours was 1.61 mV). To test it, try holding it in your hand and the reading should rise to around 200 mV or more, depending on how damp your fingers are – they are acting as a conductor between the sensor's two prongs so that the current passes through from one to the other. The damper the material, the more conductive it is. This also applies to the soil in a plant pot.

If you dampen your fingers and try the test again, you should see the reading rise to over 800 mV. Be very careful not to get any of the electronics wet, though.

04 Plant pot moisture

Now we know that the moisture sensor is working correctly, let's try placing it in a plant pot. We put ours in fairly dry soil to take a reference reading. We will then be able to use this to trigger an alert when the soil becomes too dry and the plant needs watering.

Running the **moisture_test.py** program again, we found that the dry soil gave a reading well under 100 mV. When we watered the plant, it rose to over 1400 mV.

moisture_test.py

Language: **Python 3**

```
from gpiozero import MCP3008

moisture = MCP3008(0)

while True:
    print("Moisture: %-3.2f mV   " % (
    3300 * moisture.value), end = "\r")
```

liquid_test.py

Language: **Python 3**

```
from gpiozero import MCP3008

liquid = MCP3008(1)

while True:
    print("Liquid level: %-3.2f mV   " % (
    3300 * liquid.value), end = "\r")
```

05 Liquid level sensor

If you have your plant pot placed in a saucer containing a reservoir of water, you can also add a liquid level sensor to check the water level. This is an optional step, and probably a bit of overkill as the moisture sensor should suffice.

Our liquid level sensor is from the Waveshare Sensors Pack and also sold separately. It works in much the same way as the moisture sensor, with the water conducting a current between its metal strips. The higher the water level, the higher the voltage output: from around 0 V at 0 cm to 1.88 V at 4.8 cm. So we can use it to trigger an alert when the water level drops to near zero.

Wire it up as in **Figure 1**, with the AOUT pin connected to the channel 1 pin of the ADC. Test it out with the **liquid_test.py** code and immerse it a different levels in the water to see the voltage change. Again, be careful not to get water on your electronic connections or Raspberry Pi.

"We put ours in fairly dry soil to take a reference reading"

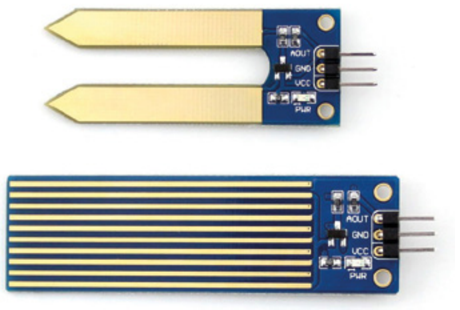
06 Push notifications

We will write a program to read the two sensors and trigger an alert when either the moisture or liquid level is too low.

While we could sound an alert with a buzzer, as for the alarms in parts 1 and 2 or this series, we thought it would be more useful to send a push notification to a phone or computer. For this, we're using Pushbullet, which offers a free service tier.

Go to **pushbullet.com** and sign up for an account. Then go to Settings > Account on the website and click Create Access Token. Copy this down, as you'll need to add it into the code.

We will also need to install the Python 3 library for Pushbullet. After making sure your Raspberry Pi is up to date, open a Terminal and enter: `sudo pip3 install pushbullet.py`.



The two sensors we're using: moisture (top) and liquid level (bottom)

07 Plant monitor code

In our final code, **plant_monitor.py**, we import the `pushbullet` library and assign the `pb` variable to our Pushbullet account – you will need to replace `Your Access Token` with the Pushbullet access token you obtained in the previous step.

You will also need the name of the Pushbullet-connected device that you want to send the push notification to. Your device names can be found either on the Pushbullet website, in Settings > Devices, or by using the line `print(pb.devices)`, as shown near the top of the code.

In the `alert` function, we set the `device` variable to the name of the desired device to send the notification to. We then send a push notification with the text 'Plant monitor: ' followed by the `message` string determined by the conditional statement in the `while True:` loop. We add a `sleep` of 60 seconds (or more if you want) so that multiple notifications aren't sent straight after each other.

In the `while True:` loop, we check whether the readings for moisture and liquid level (in mV) and set thresholds of 500 and 1000 respectively (alter them to your preference) under which an alert is triggered with the relevant message.

Taking it further

We now have a working plant monitor that alerts us when it needs watering. You take it a step further by creating a system to water the plant automatically when the soil becomes too dry. For this, you could use a solenoid valve to open and close off the water supply, or a water pump. You'll also need a relay board to control the solenoid or pump, since it will likely have a higher voltage and current than Raspberry Pi can output. ◻

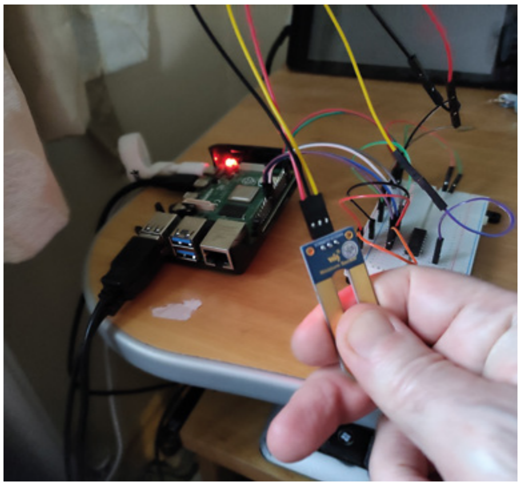
plant_monitor.py

Language: **Python 3**

DOWNLOAD THE FULL CODE:

magpi.cc/github

```
from gpiozero import MCP3008
from time import sleep
from pushbullet import Pushbullet

pb = Pushbullet("Your Access Token")
print(pb.devices)

message = ""
moisture = MCP3008(0)
liquid = MCP3008(1)

def alert():
    device = pb.get_device('Your Device')
    push = device.push_note("Plant monitor alert: ", message)
    sleep(60)

while True:
    moisture_mv = 3300 * moisture.value
    liquid_mv = 3300 * liquid.value
    print("Moisture: %-3.2f mV   " % moisture_mv,
"Liquid level: %-3.2f mV   " % liquid_mv, end = "\r")

    if moisture_mv < 500:
        message = "Soil moisture too low! Water me!"
        alert()
    elif liquid_mv < 1000:
        message = "Reservoir level too low! Fill it!"
        alert()
```



Testing the moisture sensor by holding it in the hand; current is conducted from one prong to the other

Top Tip

Watch it grow

Connect a Camera Module to your Raspberry Pi and take a photo of the plant every ten minutes, then combine them into a time-lapse video of it growing.

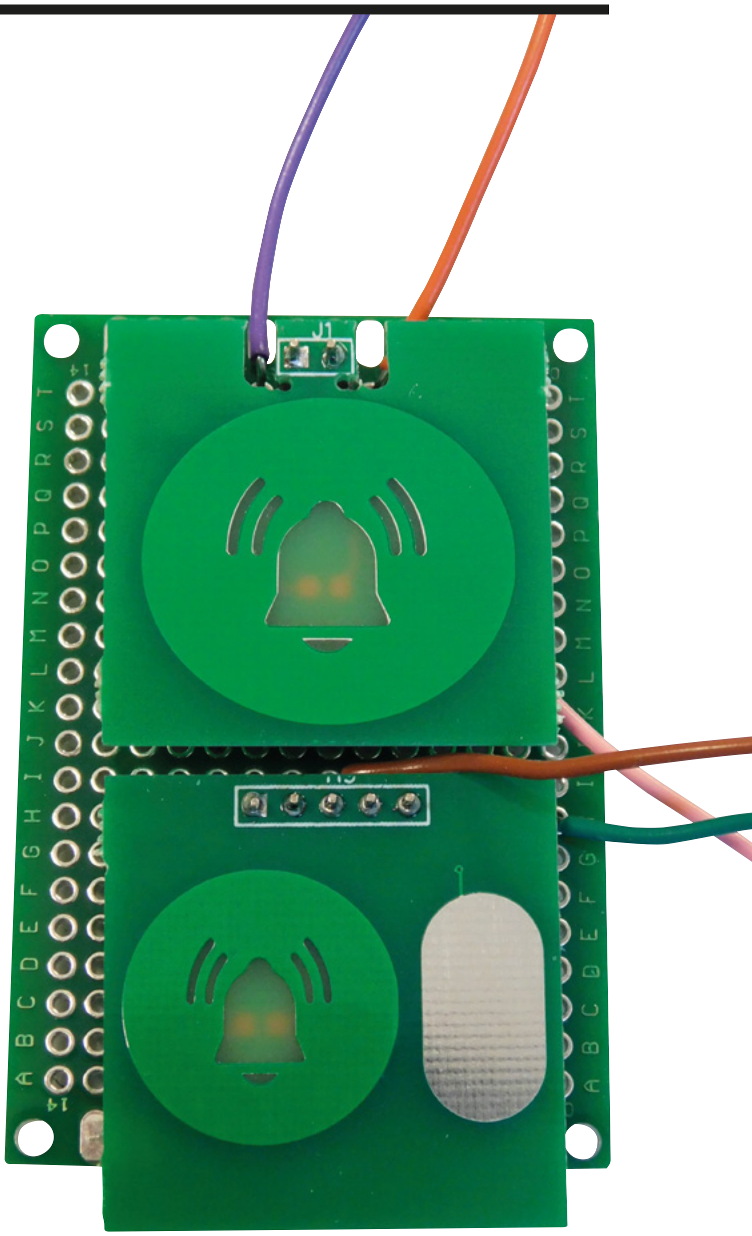
DIY Buttons with PCBs

Build user interfaces out of FR4

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.



We love the bare circuit board aesthetic. There's probably a word for it, but we're not quite sure what it is. It's not quite cyberpunk, but it's along those lines. It appeals to the part of us that instantly wants to pull something open to see what's inside because you can already see the insides – there's no need to unscrew the back and invariably lose a screw.

It's also a practical way of building things. Circuit boards are probably the cheapest material to get CNCed to your desired shape (as long as your desired shape is 2D), and if you need some electronics anyway, why not just work with everything in FR4?

Of course, if you're going to make things with circuit boards, you're going to need ways of interacting with the user from the circuit board. Sure, you can mount buttons and screens on the board, but in this article we're going to go a bit further and build them with the PCB itself in a couple of ways. We've already looked at reverse-mounting LEDs on PCBs and shining them through cut-outs in the copper layer. You can use this to light symbols and otherwise output information. Here, we're going to have a look at combining this with an input to create interactive buttons.

We're going to build a notification button in two ways. The idea behind this button is that it lights up to let you know about an event. When you've dealt with the event, you press the button and the light turns off again. However, you can use the basic idea of the light-up symbol-button however you like.

The two methods we're testing out are, firstly, using a PCB mounted over a physical button, so you can press the light itself and this will, in turn, press the button. Secondly, we'll use a touch-sensitive pad next to the light. This is a little simpler to create, but doesn't have the tactile feel of the first method.

"**The idea behind this button is that it lights up to** let you know about an event"

BENDY PCBS

For our first light-up button, we have a small PCB that has just the reverse-mounted LED and the cut-out in the copper layer. It also has two header pins that allow it to be mounted on a second PCB. This second PCB has a button mounted under the first

Above The two designs side by side. Both give us an indicator light and a button to press

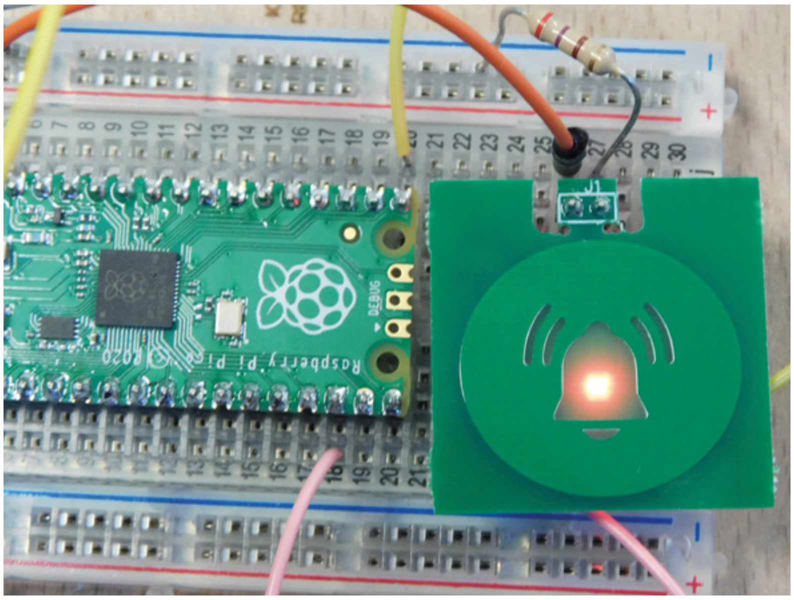
PCB so that when you press the first PCB, this PCB presses the button.

For this to work, we need the PCB to flex slightly, and this will put a strain on the header pins connecting the two PCBs. The more the PCB flexes, the less strain on the pins. One solution would be to use a very thin PCB, but we found that if the PCB was too thin, we didn't get a good diffusion on the light from the LED. Instead of making the PCB thinner, in the Z direction, we use cut-outs to make it narrower in the Y direction. As you can see in **Figure 1**, there are cut-outs to a small neck. This bends more easily than if the force was spread over the entire width of the PCB, and therefore should put less stress on the header pins. For the small amount of flex needed to press a button, this might be overkill, but it's easy to do.

You should be able to create a design like this in any EDA package. We used EasyEDA, and in the last issue we looked at how to create custom footprints using SVG Import. To create a shine-through area, the process is the same, but obviously you have to include a void in the copper layer to let the light shine through, and you have to match this void with a shape in the solder mask layer as this will also block the light. On the other side of the PCB to this, you want to place your LED. You can use a normal surface-mount LED footprint (we use 0805 because they're easy to work with), but don't forget to mount the LED upside down when you come to solder it, so that the light shines through.

You can then wire up the LED and button (on the underneath PCB – we used protoboard to test with) as you would any other LED and button, and they work as normal.

Having the large PCB to press gives a wonderful tactile feel that small buttons often lack.



TOUCHY-FEELY

We really like the click of a real button, but it's not always practical to mount a second PCB on top of your main PCB. As an alternative, you can use a touch-sensitive area. These are just areas of exposed metal; when you touch them, you create an electrical connection with your body and this changes the capacitance of the area.

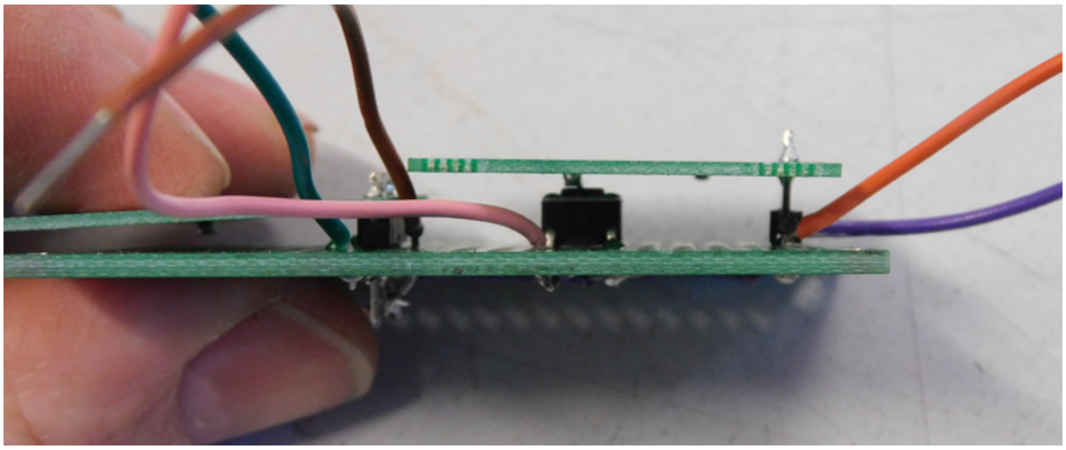
The only wiring your exposed metal area needs is a high-value (circa 1 MΩ) resistor to ground and a connection to a microcontroller I/O pin. With that in place, you can sense-touch with the CircuitPython TouchIO module (**hsmag.cc/CapTouch**).

Both of these methods have their pros and cons – buttons have a lovely clicky feel and the ability to press the actual notification light, while the touchpads are easier to implement.

However, both methods let us retain the PCB aesthetic and make it easy to build useful, practical designs out of PCBs.

Figure 1
The neck was made using a series of holes rather than the outline of the PCB. While functional, this isn't the neatest-looking way of doing it – in future, we'd use the board outline layer

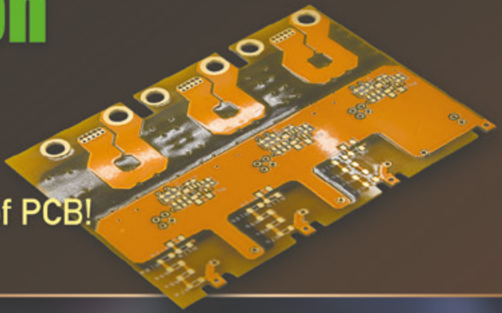
Below
Here you can see the button that is pressed when you press the top PCB



PCBWay

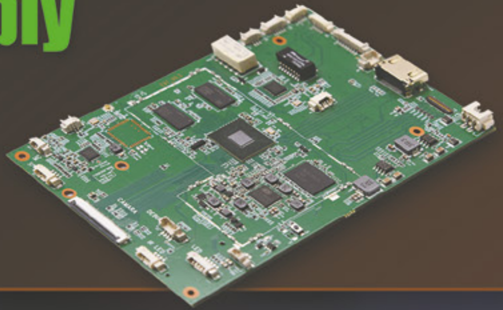
Advanced PCB Fabrication

Instant quote online & User-friendly web interface
Fast turnaround in 24 hours
Perfectly implement your idea with different types of PCB!



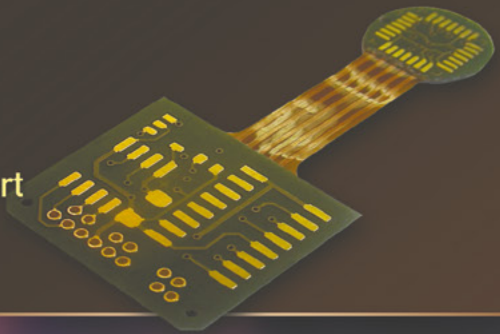
High-Quality PCB Assembly

Turnkey PCB Assembly Service
Start from only $30
Free stencil & Free shipping all over the world

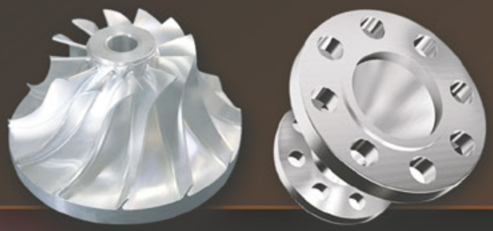


15% off for Flex PCB

Professional Flex/Rigid-Flex PCB.
Quality assurance with the inspection and test report
Make high-end products the easy way



3D printing & CNC machining service is also available now!



More information:
www.PCBWay.com

Phone: (0571) 8531 7603
Email: service@pcbway.com

Back to basics with logic chips

Even though single-board computers cost so little, small-scale logic chips are still very useful components, as Mike Bedford reveals in this hands-on guide

Mike Bedford

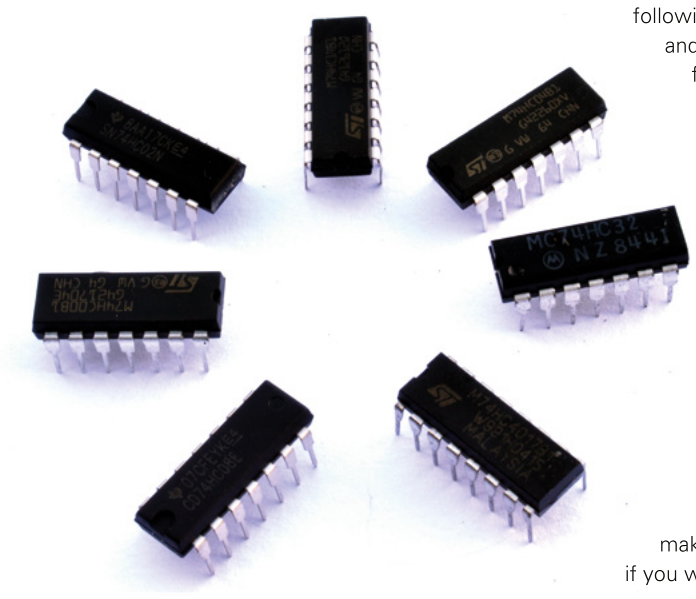
Despite loving all things digital, Mike admits to being a bit of a Luddite, vinyl records and all.

Back in the 1960s and early 1970s, computers had processors made from thousands of small logic chips. Then, in 1971, the microprocessor made its appearance and all that had changed. However, the microprocessor hasn't replaced these chips entirely – they are still alive and well and are very useful building blocks.

First, they can help us learn about logic theory. Second, they're useful for interfacing single-board computers (SBCs) to external devices. And third, a couple of logic chips can often provide a viable alternative to an SBC for simple projects. Here, we'll look at all these three themes and suggest practical ways of learning.

The 74 series was the first mainstream family of logic chips, and has probably been the most commonly used. Over the years, the original 74 series has spawned several variants providing improvements in their power consumption and interfacing capabilities. One of today's most popular is the 74HC series, and this is our subject here because it can be used with either the 5V or 3.3V supplies that are used in SBCs.

Before seeing how to use these chips, so you know what you're buying, we'll look at an example part number – SN74HC32N. The first letter or letters define the manufacturer, so you can ignore them. The 74HC specifies the family of chips. The following number specifies the family member and, in our example, the 32 chip contains four 2-input OR gates. The final letter or letters, and sometimes also a number, defines mainly the package type. N specifies a DIP package, and that's what you'll probably use because the alternatives are difficult to solder using hand tools, and can't be used on breadboards. Suppliers might just show an abbreviated part number like 74HC32, but the DIP package is easily identified in the accompanying photo.

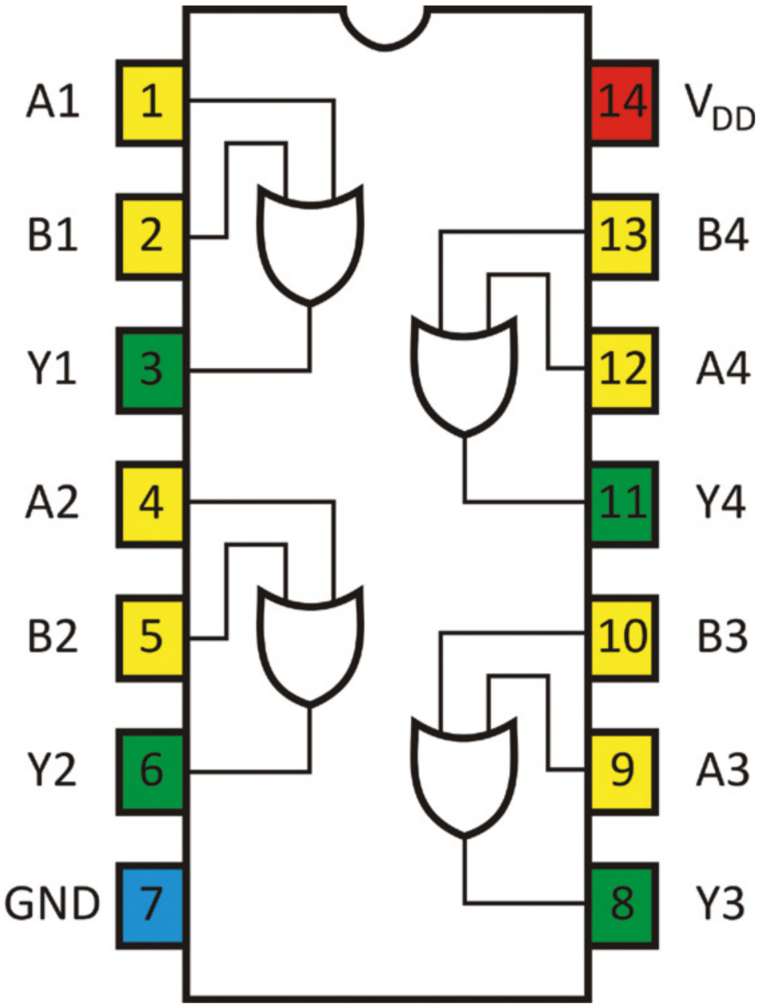


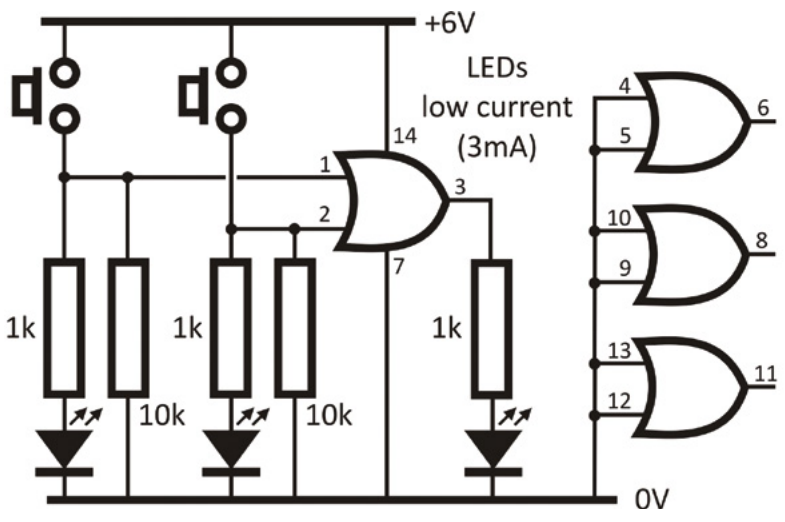
Right
Like the older 74 and 74LS devices, 74HC chips offer a huge range of functions, but they can be used with 3.3V or 5V supplies. We suggest using DIP package devices like those shown here

LEARN ABOUT LOGIC

Even though today's computers don't make extensive use of individual logic chips, if you want to go beyond being just a computer user, or perhaps a high-level language programmer, you'll probably want to understand logic. You could use a pencil and paper, but the hands-on approach is so much better. And while logic simulation software offers a practical way of learning, there's something special about using real hardware.

Here, we'll see how to learn about logic with a breadboard, a few 74HC series chips, and just a few additional components. To start, we'll describe exactly what to do but, after this initial exercise, we'll just suggest in broad terms what you could do next. We're starting with the 74HC32 and, although we'll show you the pin numbers, you might like to take a look at its datasheet, which you'll find online, because you'll need to be familiar with datasheets. Wire up the 74HC32 and the other components on a breadboard, as shown in the schematic. 74HC series chips don't have internal pull-ups or pull-downs, hence the 10 kΩ resistors on each input. Unused 74HC inputs should always be connected to either 0 V or the positive supply. Each 74HC series chip has a connection to 0 V and the positive supply, in this case via pins 7 and 14. Use four AA cells for that supply. The two LEDs connected to the push-buttons aren't essential but, if you put them next to the LED connected to the OR gate output, you can easily see the combination of the inputs and output. We used low-current LEDs, but if yours require more than 4 mA, you'll need a different value for the series current-limiting resistors. Now try pressing the push-buttons and here's what you'll see. If neither of the push-buttons is pressed, thereby ensuring a low-level signal (i.e. a logic 0) on both the OR gate inputs, the output will be a 0. However, if either or both the push-buttons are →

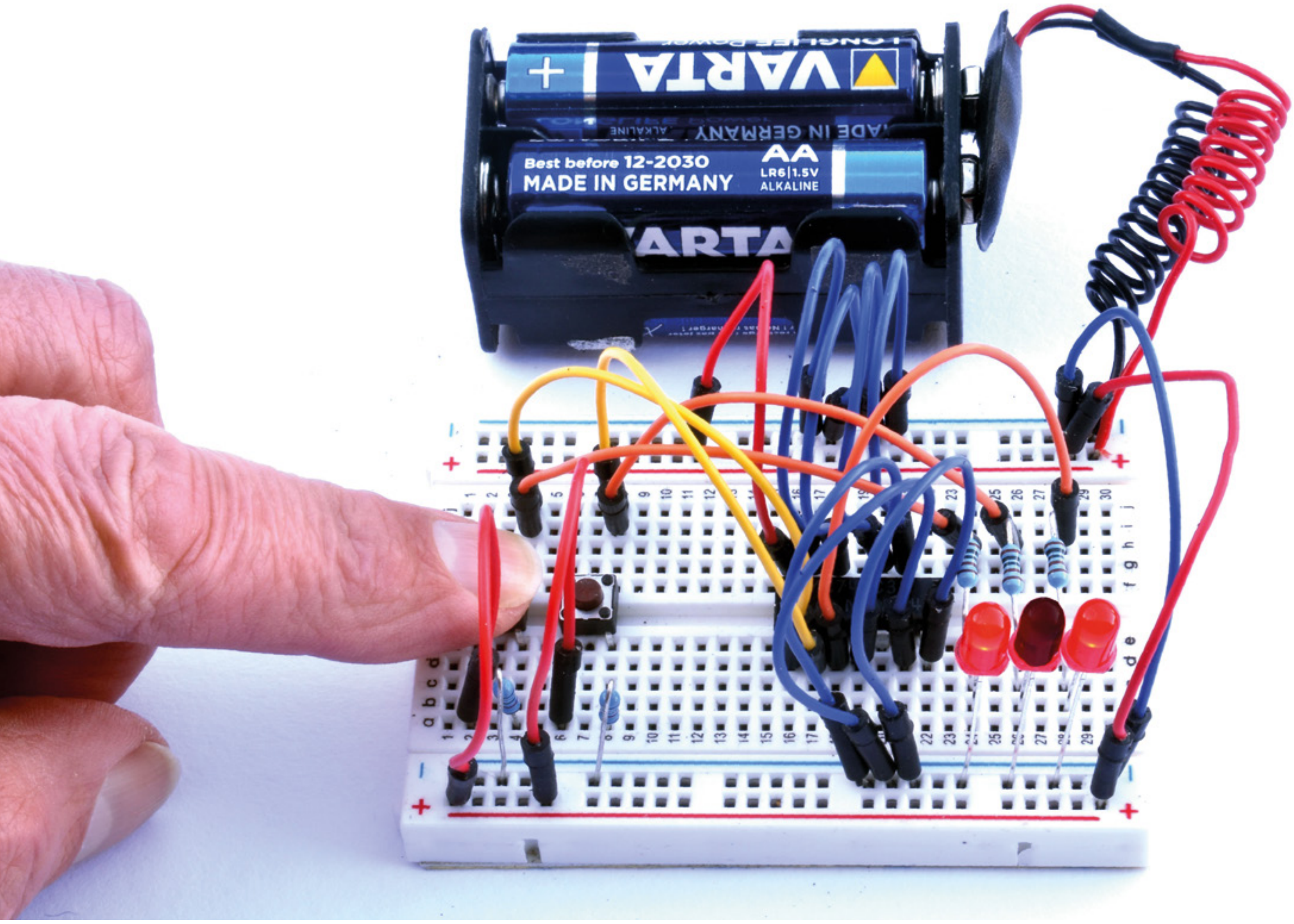




Inputs		Output
0	0	0
1	0	1
0	1	1
1	1	1

Above If you're using a particular 74HC series chip for the first time, you'll need to look at its datasheet which shows, among other things, the pinout, like this one for the 74HC32

Left Using a few LEDs and push-buttons on a breadboard, plus a 74HC chip or two, you can learn the basics of logic circuitry in a practical way. This circuit lets you learn about OR gates using a 74HC32



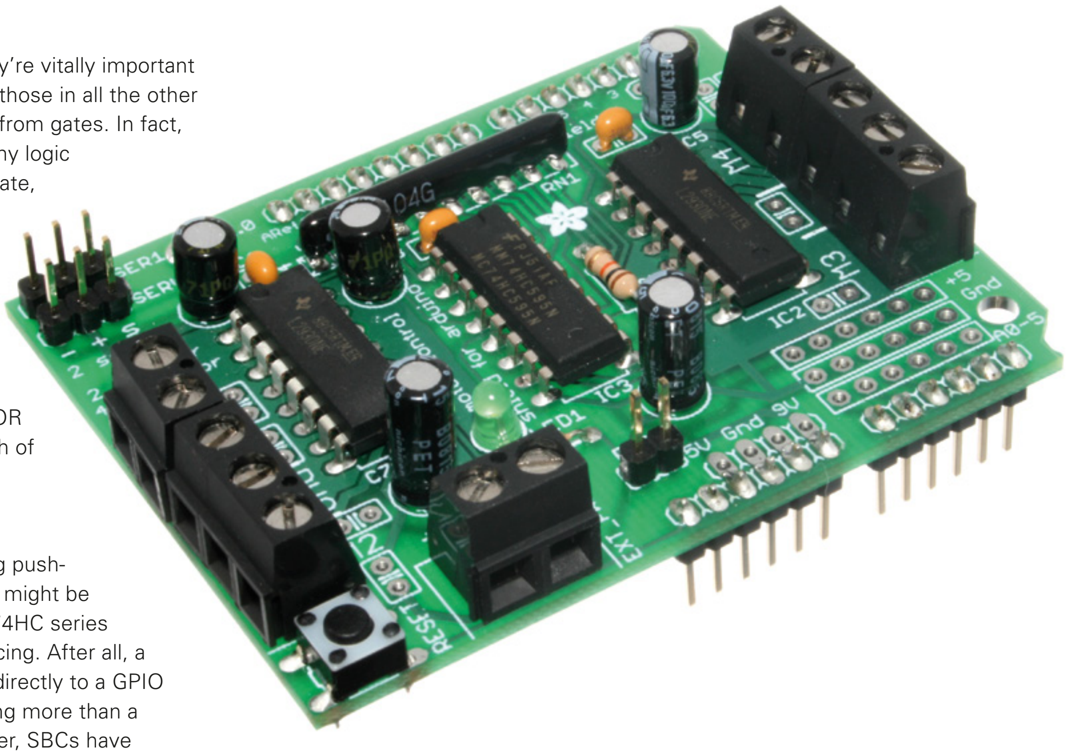
Above As you can see, you don't need to patch up many components to put a 74HC chip through its paces, and it's arguably a more enjoyable way of learning than using a logic simulator

pressed, thereby applying a logic 1 to one or other of the inputs, the output will be a 1. The OR designation is, therefore, fully descriptive and is illustrated by the truth table against the schematic. The 74HC32 contains 2-input OR gates, but gates can have any number of inputs. Although there are 74HC series OR gates with more inputs, you can make a 3-input gate from a couple of 2-input gates, or a 4-input gate with three 2-input gates. Making a bigger OR gate would be a good next exercise.

Next up, how about trying a 74HC08, which contains four 2-input AND gates? All of the gates' inputs and outputs are on the same pins as the equivalent gates in a 74HC32, so you can just change the 74HC32 to a 74HC08 on your breadboard. This time you'll find that the output is a 1 only if both the inputs are 1s. Moving on from OR gates and AND gates, the next obvious port of call is the NOR (NOT OR) gate or the NAND (NOT AND) gate. The appropriate chips are the 74HC02 and 74HC00 or, alternatively, you can make a NOR gate or a NAND gate by connecting an inverter (which changes a 0 to a 1 or vice versa – there are six in a 74HC04) to the output of an OR gate or an AND gate respectively.

"The secret is that the 74HC148 generates a binary number that **corresponds to which of its inputs is at logic 0**"

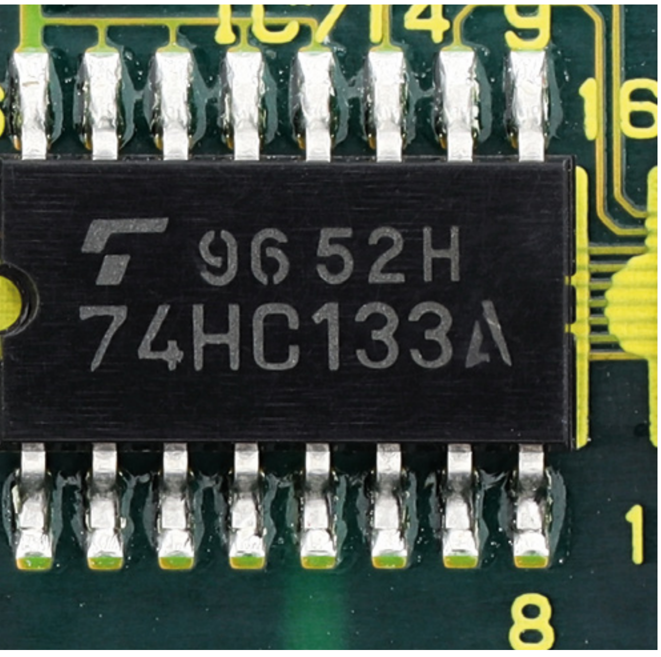
Gates are just a start, but they're vitally important because all logic functions, like those in all the other 74HC series chips, can be built from gates. In fact, it would be possible to create any logic function from just one type of gate, either NOR gates or NAND gates. For example, if you connect both inputs of either of these types of gate together, they'll work as inverters. And we'll leave you to discover what you get if you connect an inverter (or a NOR gate used as an inverter) to each of the inputs of a NOR gate.



SBC INTERFACING

If you're familiar with interfacing push-buttons and LEDs to SBCs, you might be puzzled at our suggestion that 74HC series chips are useful for SBC interfacing. After all, a push-button can be connected directly to a GPIO pin, and most LEDs need nothing more than a current-limiting resistor. However, SBCs have limited numbers of GPIO pins, and 74HC devices can be used to allow more external components to be connected. In other words, they can effectively increase the number of GPIOs.

Here's an example. In modern equipment, functions are usually selected either via a menu or by repeatedly pressing a push-button. But if you want to implement something with a retro interface, you might use a rotary switch so each position selects a different function. An 8-position switch, for example, is equivalent to eight push-buttons, of which only one can be pressed at once. Without doing something clever therefore, you'd need eight GPIO pins configured as inputs. That something clever is to use a 74HC148, which is called an 8-to-3 priority encoder, and which reduces the requirement for eight inputs to just three. The secret is that the 74HC148 generates a binary number that corresponds to which of its inputs is at logic 0. So, for example, input 0 gives outputs of 0, 0, and 0, input 1 gives outputs of 0, 0, and 1, through to input 8, which generates 1, 1, and 1.



One of the most commonly used chips in SBC interfacing is the 74HC595. It allows you to drive lots of outputs, such as LEDs, from just three GPIO pins. This chip is an 8-bit shift register which is really a serial to parallel converter. Each time the chip's clock input is toggled, each bit in its internal register moves along one position, and the value at its data input is latched into the first position. A third input is used to make all the register's bits available on the chip's eight outputs, which could be connected to LEDs. Multiple 74HC595s can be daisy-chained to provide more than eight outputs, while still only needing three GPIOs. →

Above
You can use 74HC series chips in your own SBC interfacing projects, and you'll also find them in ready-built add-ons like this motor shield from Adafruit
Photo: oomlout

Left
Surface-mount 74HC chips are very small, but they're harder to solder using hand tools than their DIP equivalents because their pins don't pass through holes on the board, and they're very close together
Photo: Mister rf.

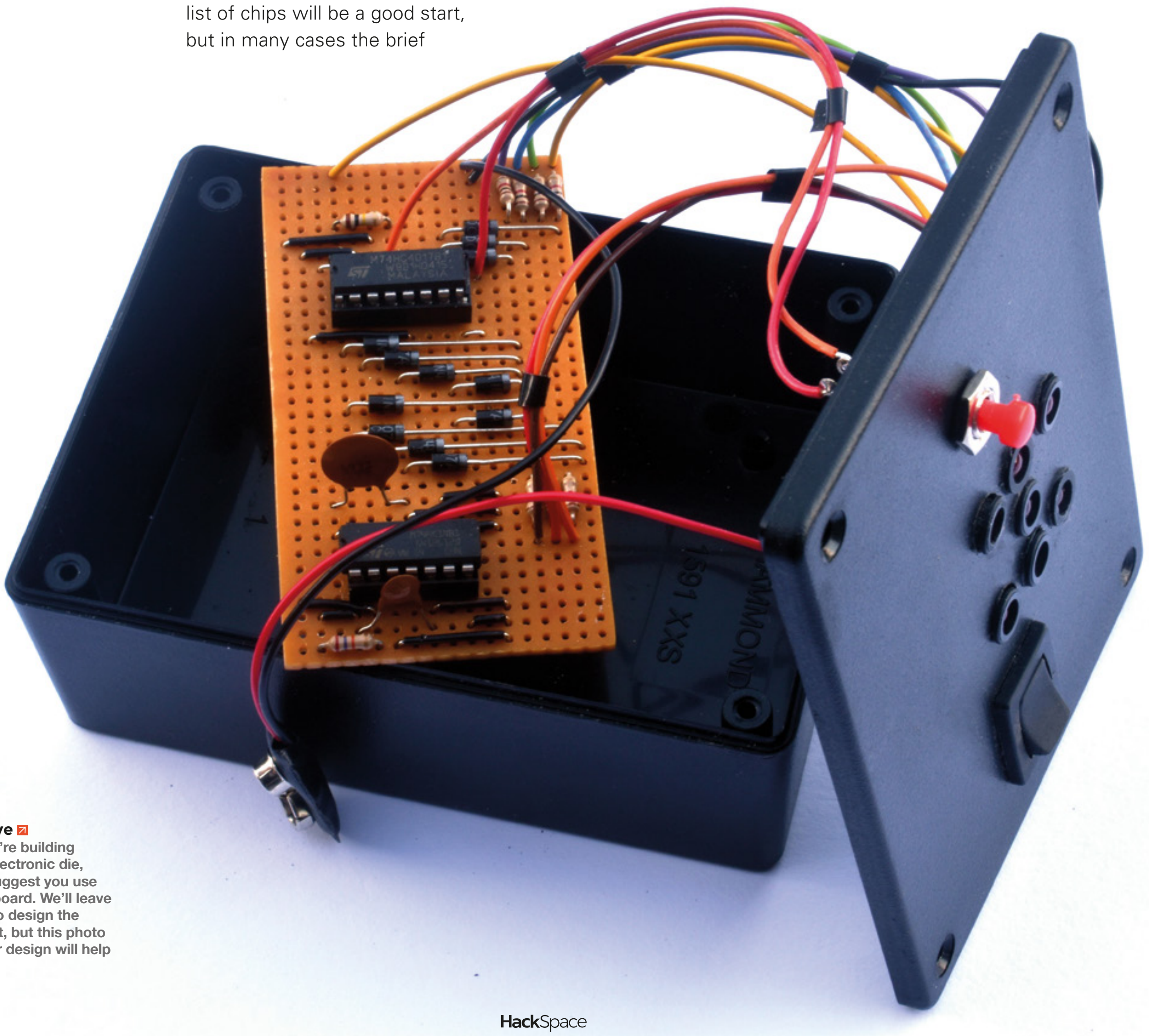
AN SBC ALTERNATIVE

Using 74HC series chips, just a few chips could be used for a project instead of an SBC. But with Raspberry Pi Zeros and Picos, and Arduinos, being so cheap, the question 'why?' might come to mind. There are several reasons. For a start, some of you will feel more comfortable soldering a few chips together than hacking code. Second, a couple of 74HC chips could take up less space than an SBC, especially if you use surface-mount components. Third, if well-designed, a circuit using logic chips could be less power-hungry than an SBC.

We started by using 74HC series chips to learn about logic, and designing a 74HC-based project is the ideal opportunity to learn more. As a first step, we suggest you get a feel for what's available in the 74HC family. Just looking through the list of chips will be a good start, but in many cases the brief descriptions will be meaningless, so be sure to look at the datasheets.

Eventually, you'll surely want to design your own circuits, but to start, we're providing a circuit for a simple 74HC-based project. It shows how something you might previously have thought needed an SBC can be done with just two chips. You might even find it useful. Our circuit (see the schematic) is an electronic die – just hold down the push-button and, when you're feeling lucky, let go and the LEDs will show what you threw.

The photo shows how it was implemented on a stripboard. The board isn't as small as it could have been with a PCB, although since you need four AA (or AAA) cells to power it, there's not much benefit in shrinking it.



Above
If you're building the electronic die, we suggest you use stripboard. We'll leave you to design the layout, but this photo of our design will help

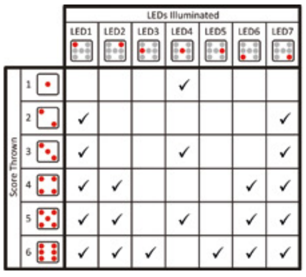
Here's how it works. We use one of the six inverters in a 74HC14 as an oscillator. The associated capacitor and resistor define the frequency, and we made it oscillate so quickly that it'll be impossible for you to stop it when you see a particular value on the LEDs. For use as an oscillator, you need to use the HC14, which has Schmitt trigger inputs, instead of an HC04 which has ordinary inputs. The oscillating signal from pin 2 connects to the clock input (pin 14) of the 74HC4017, which is a decade counter. It outputs a 1 on one of its ten outputs and, each time the clock toggles, the output increments. For example, if output 1 was high and the clock is toggled, output 2 becomes high instead. In our circuit, we've wired output 6 (pin 5) to the master reset input (pin 15) so the circuit cycles only between 0, 1, 2, 3, 4, and 5. We've attached a push-button to the clock enable input (pin 13) so the outputs only cycle while it's pressed, and settle on a final value when it's released. The diodes are used to implement OR gates, and take up less space than the several 74HC chips that you'd need otherwise. This works well if you're driving LEDs, but not if you want to connect the OR outputs to the inputs of other 74HC chips, because the necessary voltage that designates a logic 1 isn't guaranteed.

Who said you can't do anything without an SBC?

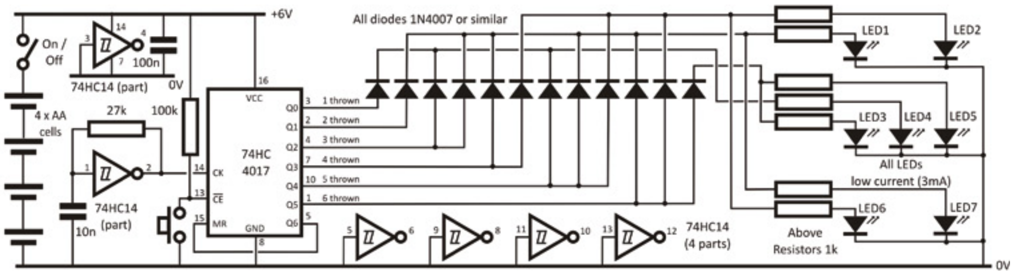
> Just hold down the push-button and, when you're feeling lucky, **let go and LEDs will show what you threw**

Above With the electronic die complete and built into a box, throws are generated at random each time you press and release the push-button

Below Two 74HC chips and a handful of passive components are all you need to implement an electronic die



		LEDs Illuminated						
		LED1	LED2	LED3	LED4	LED5	LED6	LED7
Score Thrown	1				✓			
	2	✓						✓
	3	✓			✓			✓
	4	✓	✓				✓	✓
	5	✓	✓		✓		✓	✓
	6	✓	✓	✓		✓	✓	✓



TUTORIAL

Moving beyond the K40 part 3

The laser and the optics

Dr Andrew Lewis

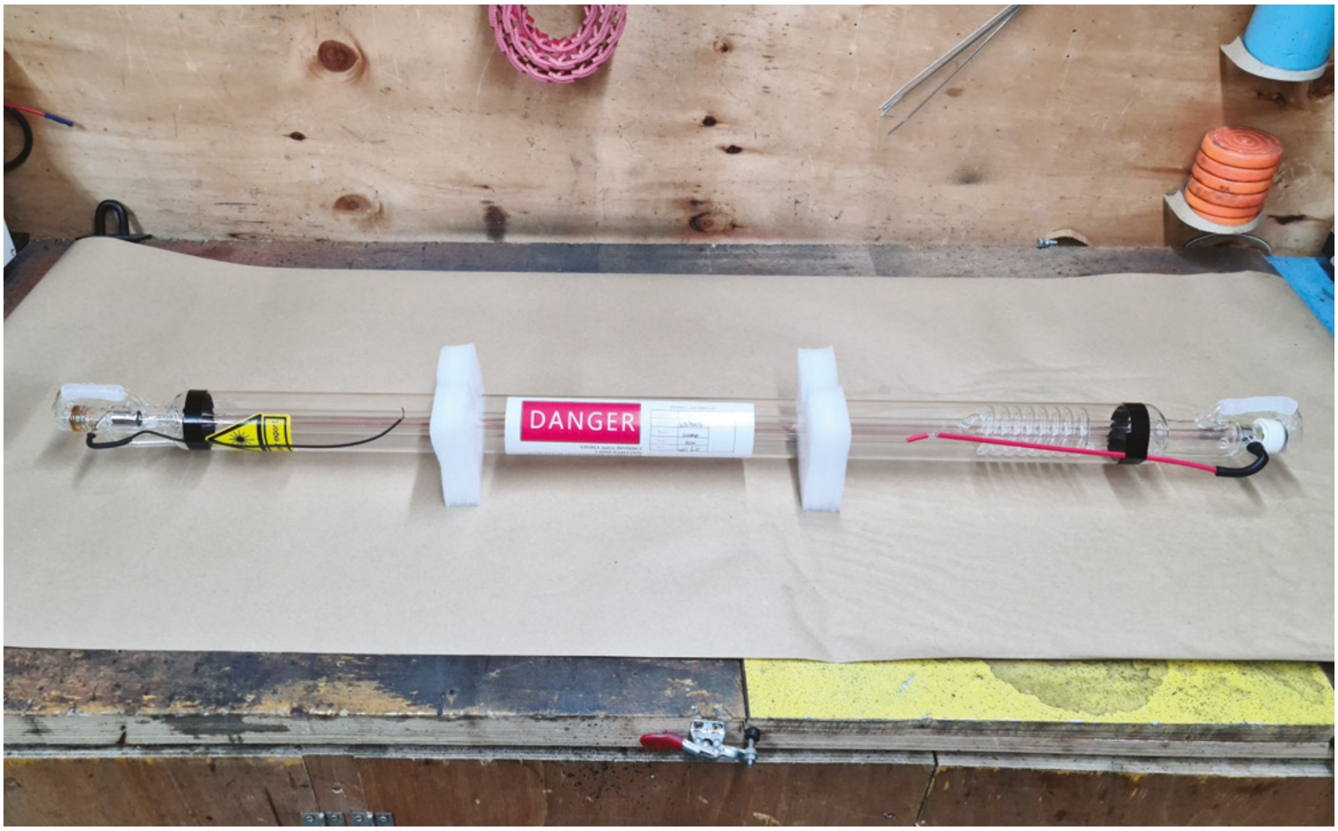
Dr Andrew Lewis is a specialist fabricator and maker, and is the owner of the Andrew Lewis Workshop.

The laser tube and power supply are the heart of the laser cutter. Considering the dangerous energy that they can produce, they are surprisingly delicate components that can be easily damaged. In this article, you'll find out more about the laser itself, the optics that reflect and focus the laser beam, and the special power supply that powers the laser tube.

Once you have a nice chassis and a moving gantry for your new laser cutter, you need to start thinking about the optics. Lasers like the K40 use a CO_2 laser tube to generate a laser beam. To all intents and purposes, the laser tube is like a very expensive light bulb. You connect power to it, and it emits a very specific sort of light. When operating correctly, a laser beam will be emitted from the end of the laser tube, and will get bounced around a series of mirrors and through the lens. The lens focuses the beam to a point a fraction of a millimetre at the surface of the material that you're trying to cut or engrave.

Like a light bulb, the laser tube has a rated power, and needs to receive a certain voltage and current in order to operate correctly. Unlike a light bulb, the laser is a greedy consumer and will happily draw enough current to destroy itself if you don't have a properly regulated laser supply attached to it. The laser will also overheat and can destroy itself very quickly if it isn't water-cooled. More specifically, a new laser tube will be rated to operate at a certain power level for a minimum number of hours, at a certain temperature. If the tube overheats, it will wear out more quickly, and if the tube runs above its rated power level, it will wear out more quickly. If you're thinking of reusing the laser tube from another machine, be aware that you might need to replace it sooner rather than later anyway. If you're getting a new tube, it might be worth considering an upgrade from the K40's standard power level of 30–40 watts. More powerful tubes can let the laser

Above
It's not too difficult to get your hands on laser reflectors and lenses these days, but which are best? Different lasers and different use cases benefit from different types of lens. In this image, you can see a typical selection of lenses and reflectors. This article will help to steer you in the right direction



ELECTRONICS

A large chunk of the electronics inside a laser cutter are dedicated to making sure that the tube doesn't just self-destruct when the machine is turned on. Most of the rest of the electronics that aren't used to control motors are dedicated to making sure that the other electronics are working properly so that the tube doesn't self-destruct if a pump fails or someone sends the wrong commands to the controller.

cutter work faster and produce a better cut in some materials. While more powerful tubes are more expensive, the limiting factor in the choice of tube will probably be its length. More powerful tubes are physically longer, and because the tubes need to be mounted horizontally with the coolant outlet at the top, things can quickly become unwieldy.

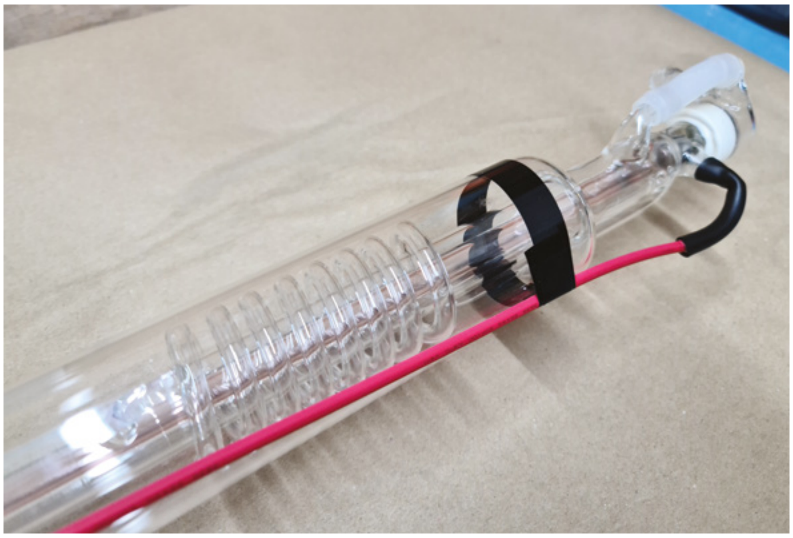
HIGH VOLTAGE – DO NOT LICK

The laser power supply looks a little bit like a PC power supply, but the output voltages and features it provides are wildly different. Most notably, the →

SAFETY FIRST

A laser cutter is a dangerous machine. Invisible laser radiation can disfigure or blind you, or someone else, permanently. Never operate the laser in an unsafe situation, and never let the machine run out of your control. Always wear laser safety goggles when working with the laser, and be aware of the risks from mechanical parts, high voltages, and reflective surfaces. Never leave the laser in a state where it could be activated accidentally by someone who does not understand the risks.

Above There are few things in this world more nerve-racking than accepting delivery of a laser tube from a courier and fitting it into a machine. The tube is made from thin glass, and has a few awkwardly shaped protuberances that can get broken very easily. Be extremely careful handling the tube, and use adjustable laser tube brackets to mount the tube into the chassis. It should be fairly obvious, but the mounting brackets should be made from plastic because 35kV of electricity has a nasty habit of arcing out whenever it gets the chance. For the same reason, the cooling fluid you use should be non-conductive



QUICK TIP

It's important to make sure your chassis and PSUs are properly grounded for safety reasons, and to help reduce the chance of electromagnetic interference affecting digital circuits.

main output of the laser supply goes up to 35,000 V, with a 5 V secondary output. The high voltage DC output connects to the laser tube, with the anode (+ve) connected at the end of the laser tube furthest from the laser output window. The cathode (-ve) connects at the opposite end of the tube, where the laser beam is emitted.

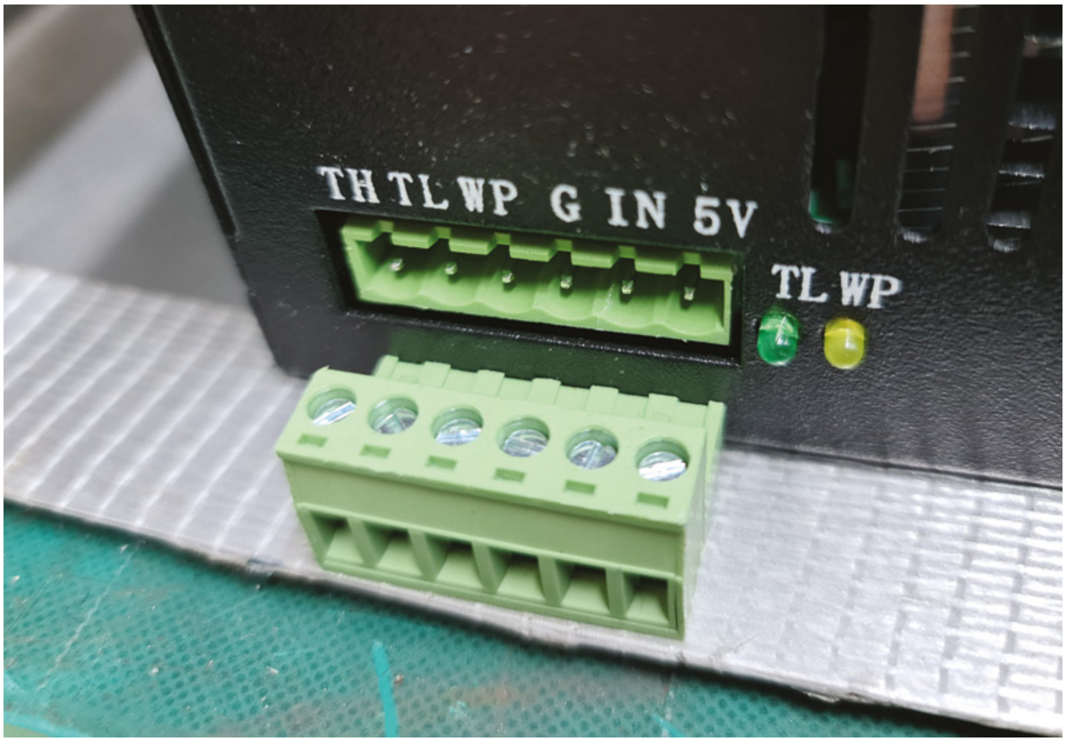
The 5 V secondary output is supplied to service a series of interlocks that will prevent the power supply from triggering the laser if they fail. The exact implementation of the interlocks will vary between power supplies, but will involve pulling a pin (often labelled 'I' for 'interlock', or 'WP' for 'water protect') on the power supply to 5 V or GND. Safety interlocks will simply break this connection to indicate a fault. Typically, a laser cutter should have (at the very least) interlocks on the chamber doors and on the coolant system so that the machine will not operate with the doors open, or if the cooling isn't running. The power supply in the K40 laser also has a 24 V output, which is used to power the motors directly. This isn't a great idea because the motors can overstress the power supply, and if you're reusing the K40 PSU, it's best to add a separate power supply for the motors.

If the interlocks are operating successfully, the laser will power up when the supply receives a signal on the trigger. Depending on the supply, the trigger might need to be pulled high or low, and the signal will be either analogue 0–5 V or digital PWM. Many supplies can handle both analogue or digital inputs from the same pin. Check the manuals and datasheets for your power supply, and apply the correct settings in your control software.

"Typically, a laser cutter should have **interlocks on the chamber doors and on the coolant system**"

Above The anode end of the laser tube is where the laser beam is born. You might be tempted to think that the glass coil around the end is part of the liquid cooling system, but it's actually a gas return tube, while the liquid coolant flows through the entire outer jacket of the tube

Right It's a good idea to fit a DC ammeter into the cathode line to keep an eye on how much power the laser tube is drawing – it's a low-cost safety feature that can save excess wear on your laser tube



Left
The connectors on this 60 W laser supply are labelled 'TH' and 'TL' for 'trigger high' and 'trigger low', which activate the laser on a high or low signal, respectively. 'WP' refers to 'water protect', which is sometimes called 'interlock' on some PSUs and is the signal loop that safety features are connected into. 'GND' is the low voltage signal ground, 'IN' is the input control signal, which controls the power of the laser and can be either an analogue 0–5V or digital PWM signal. The final 5V pin provides output power for the IN pin

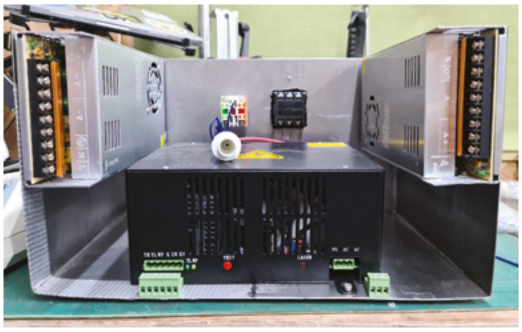
Below
It's particularly important to make sure that earth connections are made not just to the power supplies, but also to the chassis itself. Don't rely on proximity between parts to provide an effective earth, because paint and other coatings can prevent a proper contact

Note that CO_2 lasers don't have a smooth power response. It's likely that the laser won't produce any useful output at all until the power level hits somewhere around 10%

The laser tube is probably the largest component in the laser cutter, and it's far too big to consider moving it around on a gantry. It's much easier to reflect the laser beam around the inside of the laser chamber using a series of mirrors. Unlike regular mirrors, the mirrors in a laser cutter are tuned towards reflecting the invisible radiation emitted by the laser, and not visible light. Most laser cutters use a series of three (or occasionally four) mirrors to reflect the laser beam into the cutting chamber along the X and Y axes, before bouncing it down towards the lens and ultimately onto the item you're hoping to cut or engrave. All of these mirrors need to be aligned precisely so that the beam runs parallel to the axis of movement and doesn't stray outside the expected path. Misaligned mirrors reduce laser power and can cause stray beams to reflect inside the case. To make mirror alignment easier, lenses are held in adjustable brackets that can be finely positioned using tiny grub-screws. You can also use larger mirrors to give yourself more room for error, although the cost of reflectors rises quite quickly as the diameter of the reflector increases.

TIME TO REFLECT

Each time the laser bounces off a reflector, it loses some of its energy. While no mirror is a perfect →



Above
The laser PSU is quite large and heavy compared to the normal sort of 12/24V PSUs used in most 3D printers. You'll probably want to use some sort of simple rack system to keep the supplies neat and tidy. On top of the laser supply in this picture, you can see the high voltage cable and plug that go to the laser

QUICK TIP

If you do a single upgrade to a K40 laser, it should be the mirrors.

reflector, the stock reflectors provided with the K40 laser are absolutely awful. The K40 usually comes with a type of mirror called a K9, which is poor quality glass and a thin layer of gold. A better option is an Si mirror (silicon and gold), which has a good reflectance level, but can be quite delicate and difficult to clean. Cu (copper) and Mo (molybdenum) are good, but expensive, alternatives to gold. They are robust and can handle high-power lasers, but

"Planoconvex lenses have a better depth of field, **which gives straighter edges when cutting thick materials**"

they're not as reflective as gold-coated alternatives. In general, if you're using a tube less than 80 watts, Si mirrors are probably the best choice.

Laser lenses are available in a variety of sizes and focal lengths. It's very easy to over-tighten the lens mounting and damage the lens, so if you have a K40 laser, then you probably already have at least one broken planoconvex lens. The planoconvex lens has a flat side and a convex side, and is mounted into the laser cutter so that the flat side of the lens is closest to the object you are cutting. Planoconvex lenses have a better depth of field, which gives straighter edges when cutting thick materials. For greater precision when engraving, a meniscus lens (with one concave and one convex side) will produce a much tighter beam with more power at the focus point. The higher power also means it is possible to cut through material slightly more quickly with a meniscus lens, but it does this at the expense of accuracy in the vertical profile. Most lenses are made from zinc selenide, which has a high level of IR transmittance and can be cleaned when necessary with isopropyl alcohol. □

QUICK TIP

Remember that distance amplifies errors. The effect of a poorly aligned mirror will get more pronounced as you move away from the mirror, as will any vibration effects caused by motors or motion.

SHINING LIGHT

Clean optics mean better cuts. The frequency that you need to clean your laser optics can be reduced by having an efficient air assist and extraction system fitted to your laser cutter. These systems will be discussed in detail in a later article, but in the simplest terms, extracting smoke quickly and controlling where it is allowed to go will prevent particulate build-up on lenses and reflectors.

SAFETY IN ISOLATION

As with any large machine, you should have a way of easily stopping the laser cutter in an emergency. You should also consider what would happen if the power to the machine fails unexpectedly, because when the power is restored, you probably don't want the machine to start up again automatically. You can control this with a combination of an emergency stop button and a no-volt release (NVR) switch. The emergency stop button is a simple switch that usually has both normally open and normally closed terminals. On its own, the emergency stop will not isolate power from the machine, but it will cut power to the machine by interrupting the live wire. This is a significant distinction, because it means that even though the emergency stop is in effect, mains power can still get to the machine. An NVR switch will usually isolate both the live and neutral wires, making sure that no power gets into the machine. The NVR switch operates like a latching relay, and if power to the NVR switch is cut, the switch will release and break the contacts until the power button is pressed again. Wiring the emergency stop button before the NVR switch means that pressing the emergency stop will trigger the NVR switch and isolate the machine completely.

Building a sheet metal rover body shell

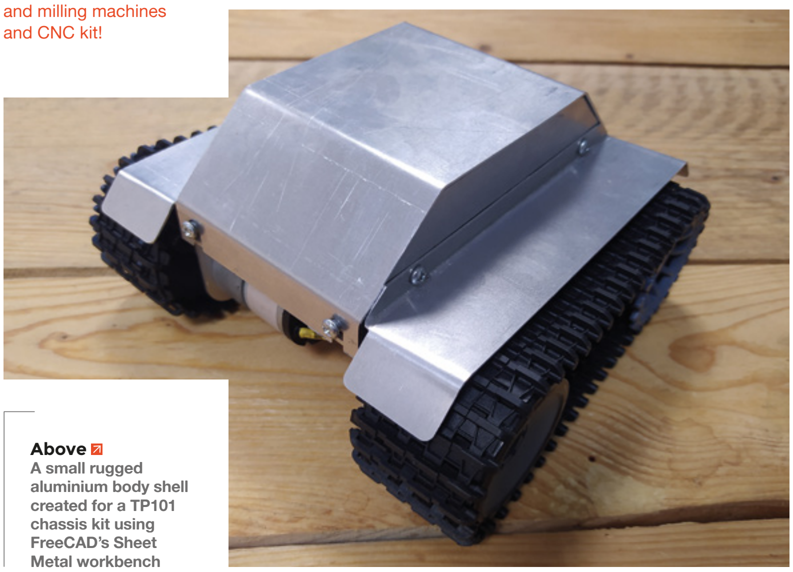
Putting together a range of skills in FreeCAD, CNC routing, and metalwork to make a metal rover body shell

Jo Hinchliffe

@concreted0g

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY space. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open-source. He also has a shed full of lathes and milling machines and CNC kit!

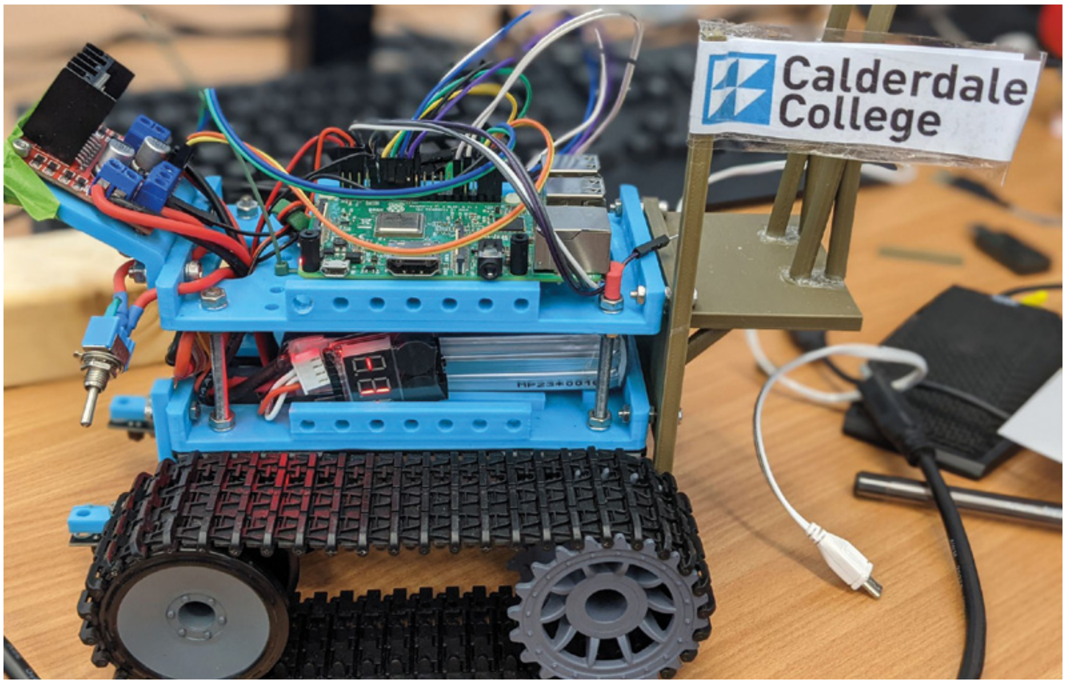
In one section of the FreeCAD tutorial series, we looked at a brilliant FreeCAD add-on: the Sheet Metal workbench. You might recall from issue 50 that it enables the dedicated 'CADista' to create a basic sheet of metal and then build a design by adding attached folded sections. Whilst amazing in itself, the real secret sauce of this workbench is that after you have designed something, you can then flatten the design. From this, you can then create CAM toolpaths, laser cutting files, and more. As I wrote that tutorial, I recalled a project from one of the early dedicated maker-type books from years ago. The first edition of *101 Spy Gadgets for the Evil Genius* had many cool ideas – in one, they took a rugged 4WD RC car, hacked it back to the bare chassis, and then created a sheet metal body shell for it in an attempt to make a rugged robot for covert surveillance missions! Ever since, I've wanted to build a metal-clad rover.



Above A small rugged aluminium body shell created for a TP101 chassis kit using FreeCAD's Sheet Metal workbench

I've previously designed the open-source modular tracked vehicle (MTV) robot which uses a cheap TP101 chassis, track, and motor kit as its base. It then adds a stack of reconfigurable 3D-printed parts to make a really modular platform for experiments. The original story with links to the MTV files can be found in issue 32. I experimented with numerous ways of rover control for the MTV, including direct radio control using hobby RC gear and brushed speed controllers, a hybrid system using an Arduino receiving the RC signals and using an L298N motor driver, as well as using a pair of micro:bits as a transmitter and receiver coupled with a cheap micro:bit motor-driving add-on board. Finally, I even wrote an article (issue 35) about controlling the MTV robot via dual-tone multi-frequency (DTMF) technology, opening up interesting ideas for controlling a robot from anywhere in the world with a telephone! I really like this chassis; it's cheap and rugged, and the tracks, although they take a bit of tinkering with, work very well. As an aside, it's great to see others using the MTV robot design; for example, Calderdale College used the design to compete successfully in the excellent Pi Wars competition, a set of challenges for Raspberry Pi-powered robots.

With all this in mind, I really wanted to make a robust shell for the TP101 chassis. A good starting point was to accurately measure and model the TP101 chassis and, in particular, the position and size of the available chassis mount points. I began by taking some accurate measurements and then drew the outline of the upper surface of the chassis. Making this rectangle the BaseBend object in the Sheet Metal workbench, I then added the folds to emulate the chassis sections where the motors and the guide wheels for the tracks mount, but stopped



Left
Lovely to see other people and organisations using the MTV robot design which also uses the TP01 chassis kit

Below
Creating an accurate model of the TP101 chassis and mount points in FreeCAD

short of modelling all the mount points for the motors and wheels.

I then did the longer task of drawing and constraining a sketch on the chassis surface which accurately modelled all the holes and slots that the TP101 chassis comes with. This is a useful job to do for designing all kinds of parts to this chassis, as well as the body shell I was aiming for in this project.

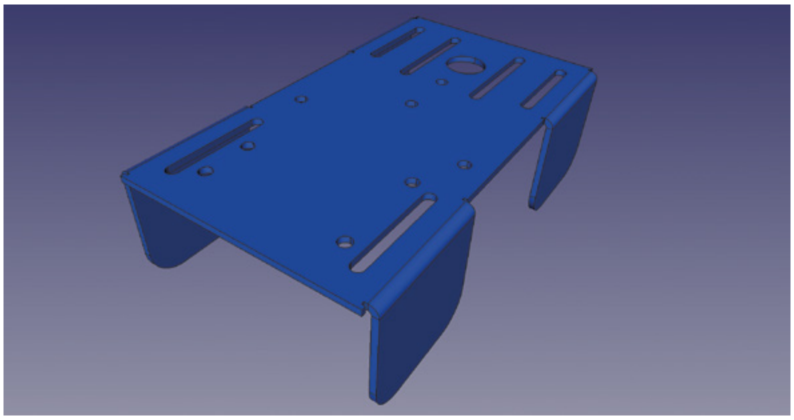
I was quite interested in creating a symmetrical body shell for this rover – this would reduce the number of parts I needed to model as I could use an Assembly workbench to add multiples of the same part from file to lay out the rover body shell. I kept the model of the chassis open in one project tab, then, in a new project, I began to model the sidewall. As background, I had a few thoughts on how I would rig the control system for this rover, and I gave a little thought to what components I might use. I am a fan of the cheap, but rugged, L298N motor driver modules which come with a fairly large heatsink on the board – these would probably be the tallest module I'd choose to go inside. With this in mind, I made a basic sketch of the sidewall and added dimensional constraints. Again, using the Sheet Metal workbench, I added the folded section underneath that would mount to the chassis, and also some small folded tabs that would serve to mount the roof of the body shell. Interestingly, those small tabs are a good example of something that is quite easily added in the Sheet Metal workbench that can be tricky to actually make, but more on that later.

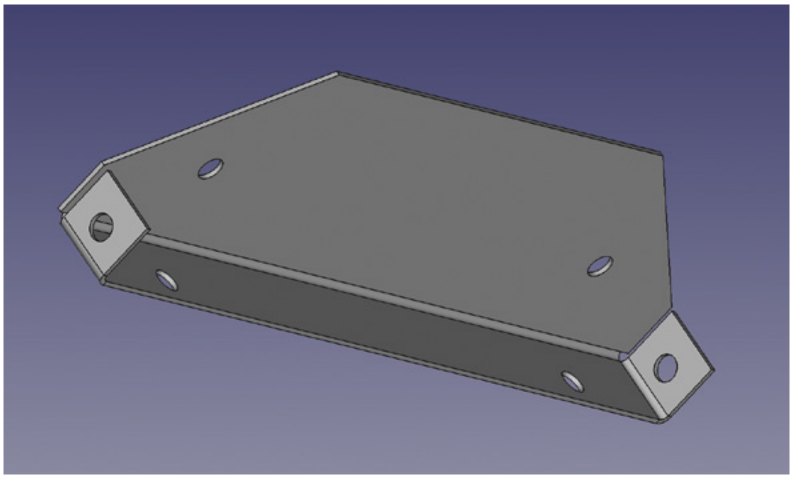
I added some mount holes to the project and saved the model file. I then opened a new FreeCAD project and used the A2plus workbench to create an assembly of the chassis and the sidewall. Of course, I could add two copies of the sidewall and constrain them into position on either side of the chassis – we covered the A2plus workbench in issue 43.

Next, I created some mudguards. I didn't want these to cover the full width of the tracks, but protrude around 30 mm to provide enough cover to stop mud/moisture being thrown up onto the body shell. This means any →

" I am a fan of the very cheap but rugged **L298N motor driver modules** "

QUICK TIP
All the FreeCAD techniques used in this build are covered in the FreeCAD tutorial series.





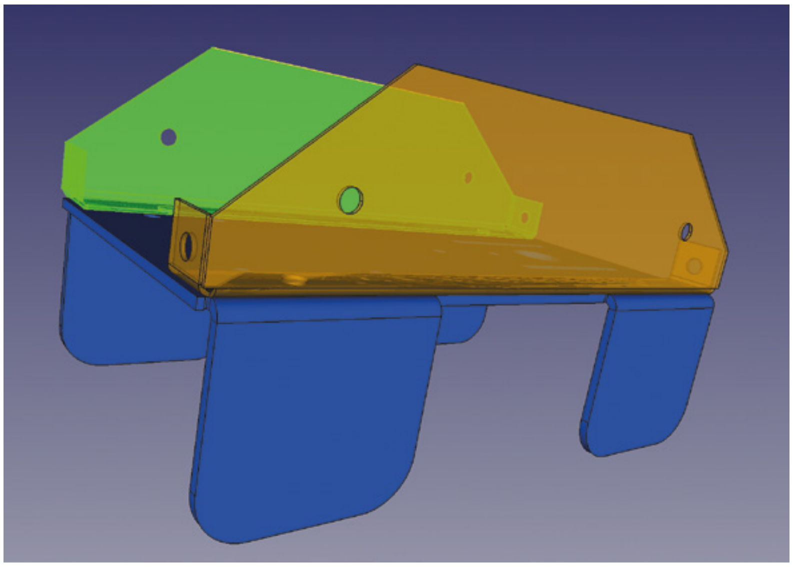
Above The Sheet Metal workbench makes complex parts really easy to achieve. This sidewall only has one basic sketch and then the folded sections are added

Below Using the A2plus workbench means that I could create an assembly of the chassis and panels, checking how things fit and look together

Right Here we can see both the 3D flattened object and the projected wire sketch of the object. These objects can be exported in a variety of file formats

external equipment, cameras, or sensors should be well protected. The other use is that the mudguards can provide a small platform for mounting objects onto directly. I drew the small section that attaches to the sidewall first, and then added the mudguard section as a fold. A good tip here was I drew the basic sketch for the part that attaches in the same file as the sidewall, importing useful edges from the sidewall object to snap the sketch too, then you can simply copy the sketch into a body in a new project to create a new part and delete the sketch in the sidewall project. To finish off the mudguard, I added some end folds at 45 degrees and jumped back to the Part Design workbench to add some fillets to round the edges and pocket the mounting holes.

Again, as running checks and to see the body shell develop, I pulled two copies of the mudguards into the project where I was creating an assembly of the parts. All that remained was to create a layout for what I called the 'canopy' – the roof section that covered the sidewalls. To do this, I returned to the sidewall sketch and made a copy of the base sketch in a new project. I then deleted most of the sidewall sketch constraints and constrained the length of the lines that followed where I wanted the roof to go. I then constrained those remaining lines as horizontal, copied the line, and created a rectangle. This meant that I had the rectangle for my BaseBend object, but also that I had geometry that I could import into further sketches to create the fold lines.



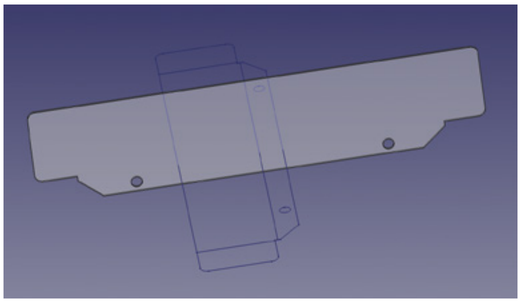
TO HAVE AND TO FOLD

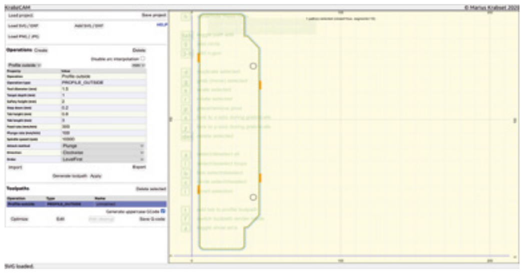
Once more, we covered this in the Sheet Metal workbench tutorial, but I then made a collection of four separate sketches attached to one face of the canopy object which were simple lines at the points I wanted to create the folds. Jumping momentarily back into the original sidewall project, I made some quick angle constraints (which created a duplicated constraint error) and then deleted them, just making a note of the angles I wanted to fold the canopy section too.

Again, a quick bit of folding in our canopy project, and the adding of some mount holes, and we could add the canopy into our assembly project to see the whole body shell created.

With the shell design complete and positioned in the assembly project, I then went back to each individual part project and used the Sheet Metal workbench tools to flatten each part. The flattening process creates both a flattened 3D part and a projected wire sketch of the part. I could have used the 3D model and moved to the Path workbench to create some CNC toolpaths, but I decided to export the wire sketches as flattened SVGs, and then I could make some changes in Inkscape, and then I could do a variety of processes with the files.

I know if you haven't got a shed full of tools and access to a CNC machine, reading about people making this stuff can be frustrating, and indeed I did CNC route the panels for my build. But in reality, these panels could be made in a variety of ways. Thin





aluminium sheet can definitely be cut well with a metal bandsaw, or you could even use a combination of hacksaw and fretsaw to cut these panels, perhaps finishing them to size with files. As such, it's useful to be able to make a template to use to create your cut-lines. The simplest approach is to print the flattened SVG onto paper and glue the paper to your aluminium sheet – this gives you a nice set of lines to follow, and then you can soak and wash the paper away. Another approach, if you have access to a laser cutter, is to laser-cut some templates. As an experiment, I laser-cut some really accurate templates out of leftover corrugated cardboard – the nice thing about this approach is that I set the fold lines to be included as a small incision onto the cardboard to again aid marking up on metal.

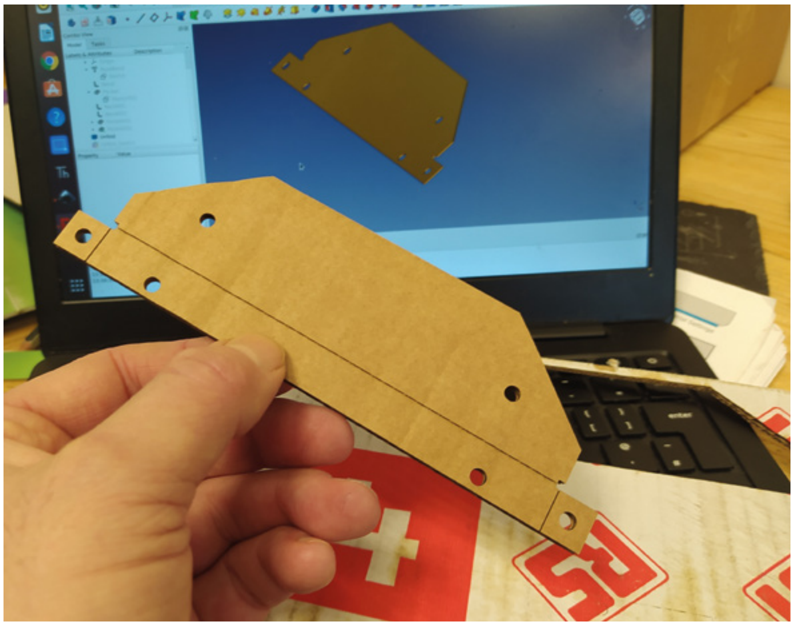
As an aside, it made me think that a possible approach could be to create folded card stock parts and cover them with fibreglass and epoxy for strong and lightweight panels.

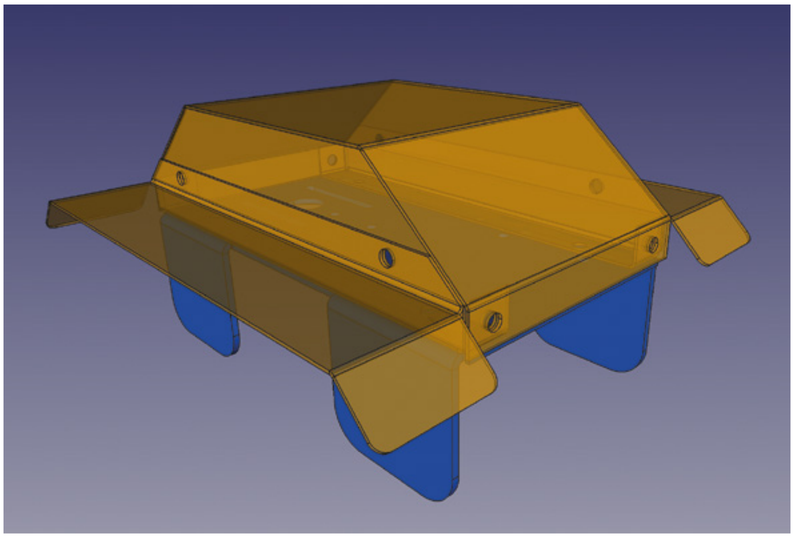
" Cutting aluminium on my CNC3040 is **pretty straightforward if you go nice and slow** "

I made no real change to the SVGs for the sidewall and the mudguard designs and pulled the SVGs into KrabzCAM to create some toolpaths for my CNC router. We looked at the free and open-source KrabzCAM software as part of the mini-series we wrote on the cheap CNC3018 machine back in issue 43. KrabzCAM is very useful for this type of 2D CAM work and makes setting up toolpaths really simple. Cutting aluminium on my CNC3040 is pretty straightforward if you go nice and slow with low feed rates. I ran a small 1.5 mm end mill cutter at 350 mm per minute and a step down of 0.25 mm per pass, and I'd designed the body shell to use some 0.8 mm aluminium stock I had in the pile.

As I don't tend to skim my waste boards, there's often a little variance in height across a piece of work stock, so I tend to cut well through the material to compensate. I set all my toolpaths to cut to 1 mm deep to ensure it cut through. I added some work holding tabs to the toolpaths to keep the part in place and then set about machining. I tend to flood the material with a puddle of cutting fluid which acts to cool the tool tip, but also, the liquid tends to capture the aluminium chips and stops them from flying everywhere. So, with a good dollop of fluid added, I set about machining panels.

I wanted to cut and assemble the side panels and mudguard sections before finishing the canopy cover, as I wondered if there might need to be some tweaks. Having cut the mudguards and side panels, I went to the shed and dug out my three-in-one metal folding, rolling, and cutting machine. We reviewed this machine from Warco in issue 36. →

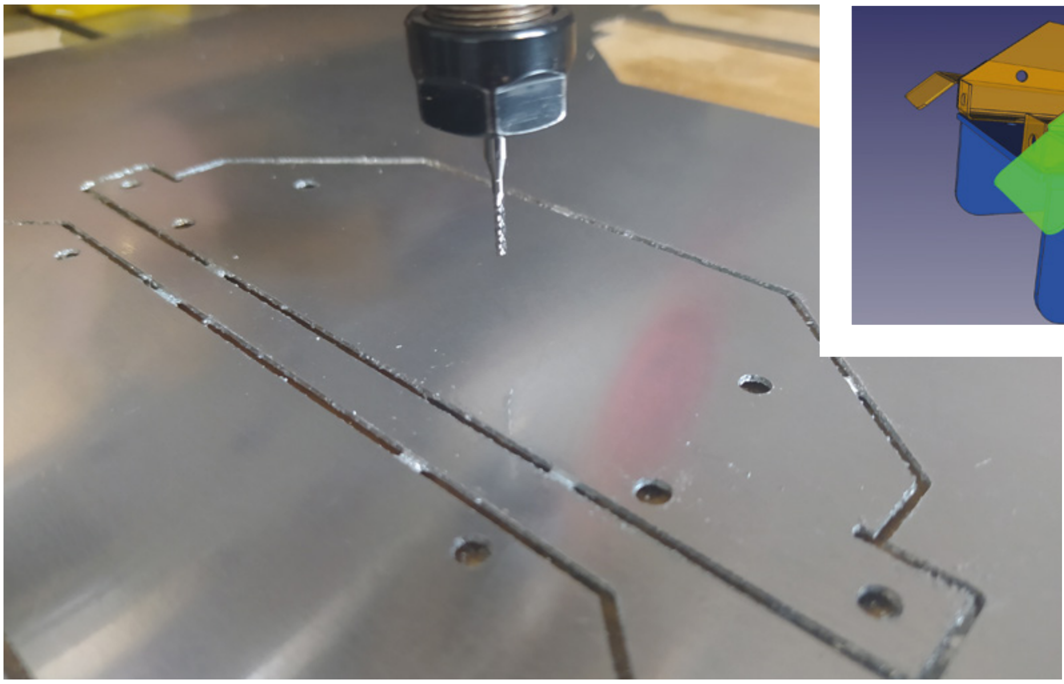


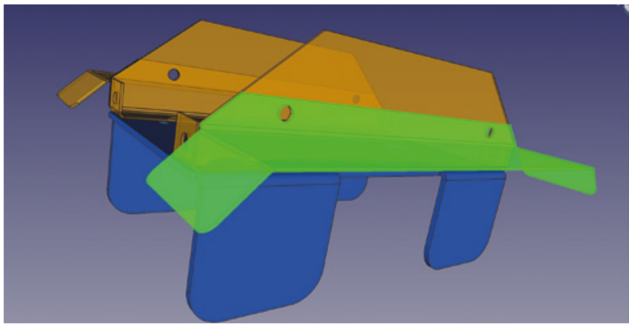


Above Laser-cutting cardboard templates or simple printing onto paper or card is a great option for template making for those without access to a CNC

Left Importing SVG and creating 2D toolpaths is pretty straightforward in the free and open-source KrabzCAM

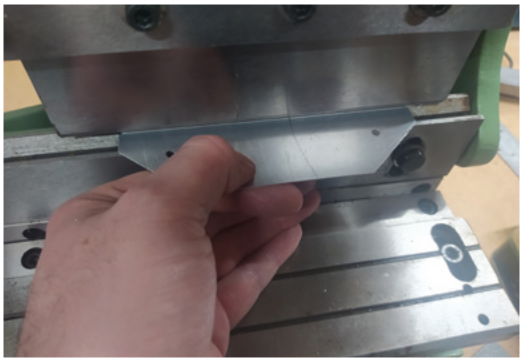
Below With the canopy modelled, everything could be assembled and checked out





Above Making the design symmetrical meant I could use multiple copies of single components to assemble the 3D design

Left The ball screw version of the CNC3040 router does an OK job of routing thin aluminium sheets



Above First assembly of a mudguard and a sidewall section in the real world!

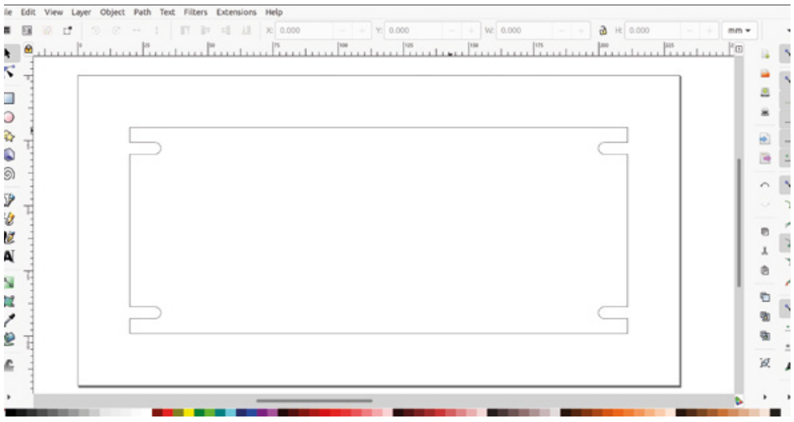
Right Using the press brake section of the 3-in-1 Warco Formit metal working machine

Although it's an expensive tool for occasional use, we found it does make sharper and neater folds than other methods. If you don't have the space for a tool such as this, there are other options. There are folding accessories that act in a similar manner to this that clip onto a bench vice, but you can also use a vice on its own to create folds. In fact, on occasion, it was easier to use a small vice than it was to use the folding machine for some of the smaller bends in this project.

My main advice about creating the folds using the three-in-one machine is to make sure that your workpiece is in the correct position, because if you make even a slight bend in the wrong place, it's incredibly difficult to rectify. It's quite a Zen activity – you need to mindfully manipulate the part into the correct place, but then be quite assertive and confident to make the fold in one neat move!

ORDER, ORDER

I tended to use the larger machine to make the longer folds in the parts, but doing so often means you can't then fit the panel back into the folding apparatus to create the smaller folds. A good example of this was mentioned earlier – the internal small tabs that end up at 90 degrees to the sidewall are quite difficult to make once the long fold in the sidewall has been created. You might think you could use the open edge of the folding wedge, but the angle of the sidewall clashes with the apparatus during the fold. So I added these folds by pinching the work in a tiny vice and bending the panel by hand. This works, but creates a wider radius fold and is less consistent – but it's good enough.



It's lovely to assemble CNC-cut panels. Fitting the folded sidewall with a folded mudguard section, it's extremely satisfying to see the holes align and the angled edges of the mudguard follow the line of the sidewall so perfectly!

Once I had the sidewalls and the mudguards assembled, I considered the canopy. I was quite happy with the idea of it but I made a change to the mount holes. I realised that it would be easier to mount if we had slots rather than holes, as there was slight variance in the small folded tabs that had the receiving holes having made them with a vice. Instead of remodelling the piece in FreeCAD, it seemed much simpler to make this simple adjustment to the exported SVG in Inkscape. The other slight bonus being that, using slots instead of holes, the entire canopy cover panel could be cut in one outside profile operation instead of four-hole operations and then a cut out. Again, I set up the toolpaths in KrabzCAM and then cut the canopy in around ten minutes on my CNC router. Once again, prior to folding, the part needs a little de-burring and any of the tab remnants need removing from the sides of the panel.

It's pretty tricky to get the folds in the correct place for the canopy. I revisited the FreeCAD sketches and wrote down the coordinates for the fold points and marked these onto the canopy by scribing a small line. It's also tricky to get the folds to the correct angle. If I thought hard enough, I could probably work out some kind of 3D-printed guide block, but I ended up doing this by eye. I'd make a fold, then offer up the folded piece to the sidewalls to check angles and then readjust. I got it as close as I could and then test-fitted the canopy. It fits quite well, but it takes some adjustment to get it so that there are very few gaps around the close line where the canopy meets the sidewall. For the next revision, when I have more of an idea of the internal structure and running gear, I have a couple of other ideas. I'll either redesign the canopy to be more modular, with two angled end pieces more permanently in place and a flat plate to close the roof, or I'll make the canopy sit slightly inside the sidewalls, as I think it would be easier to create a flush finish this way.

You might also be wondering how I managed to use the nuts and bolts that go into the canopy slots as you can't access the internal side to tighten the nut. I used an old trick where I created a captive nut on the inside panels by using a small amount of epoxy glue to attach the nut in place!

As body shells go, I am really pleased with this project, and it's certainly proved that the FreeCAD Sheet Metal workbench is very capable at designing real-world objects. I also think the results look pretty cool. With a few tweaks and some more building, I should have a very capable and rugged little rover platform.

If you are interested in looking at the design, I have added a stack of files to this repository: **hsmag.cc/SheetMetalRover**.

The separate panel FreeCAD files and the assembled model are there, as well as the SVG files. I've stopped short of supplying the G-codes because everyone's CNC routers and tooling are different. Hopefully, this little project has inspired you to explore metalwork for robot rovers and perhaps to look at the FreeCAD Sheet Metal workbench! □

Left
Once the side walls were in place, I epoxied captive nuts in position to be able to bolt the canopy cover on

Above
Adjusting an SVG in Inkscape prior to creating the toolpath was easier than remodelling in FreeCAD

Below
Some of the smaller bends, which are added after the longer folds, are morae easily achieved using a small bench vice

HackSpace
TECHNOLOGY IN YOUR HANDS

FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG 102

BEST OF BREED

Add-ons for your micro:bit

PG 108

PRUSA MINI

The small printer with big dreams

PG 112

RP2040 STAMP

Build your own RP2040 PCBs

ONLY THE BEST

Micro:bit revisited

Back to the bits

By Marc de Vinck @devinck

It has been a while since we originally looked at the micro:bit ecosystem – a lot has happened since then, including a new V2 micro:bit with upgraded features, such as a faster processor and a much-needed microphone and speaker, among other things. Last year, we looked at making music with the micro:bit (**hsmag.cc/issue46**), but this capable little board can do a lot more than just make music.

> This capable little board **can do a lot more than just make music**

In this Best of Breed, I'll be looking at an eclectic mix of useful accessories for your micro:bit. You have a micro:bit right? If not, don't despair. Although they are much harder to find – which I assume is because of the ongoing supply chain issues that we are all experiencing – I was able to find a few stores online that have them in stock. If you already have a micro:bit, the accessory market seems pretty well-stocked at most online retailers. So, let's take a look at some of the available accessories for expanding your micro:bit.

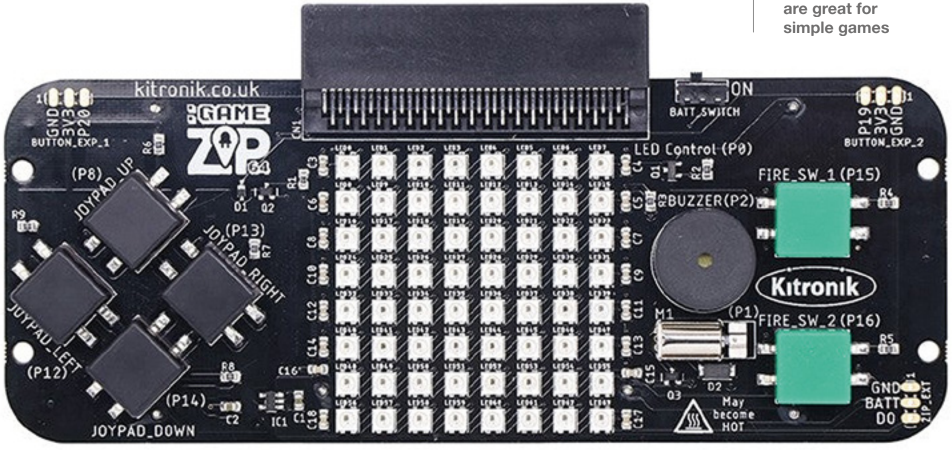
GAME ZIP 64 vs KittenBot JoyFrog

KITRONIC $31.14 | pimoroni.com

KITTENBOT $29.99 | kittenbot.cc

If you want to get into gaming with your micro:bit, the Kitronic :GAME ZIP 64 is a must-have accessory in your toolbox. No, it won't replace the latest and greatest console gaming system, but it's not supposed to, and it's still an excellent platform for games. At the centre of the GAME ZIP 64 is an 8×8 individually addressable full-colour matrix of LEDs. Couple that LED display with four directional buttons, two fire buttons, haptic feedback, additional breakout connections, and you've got a fun controller for your micro:bit.

All those components are fully programmable via the micro:bit, and it's compatible with either V1 or V2. Just keep in mind the micro:bit is not included. The GAME ZIP 64 has a lot to offer, and it's one of my favourite micro:bit accessories. Head over to the Pimoroni website to learn more about this board, available code, and even download the files needed to 3D-print or laser-cut a custom case.



Below RGB LED matrices are great for simple games

The KittenBot JoyFrog is another interesting micro:bit gaming accessory. You can easily use the board in conjunction with its online Kittenblock software (a fork of Scratch 3.0) when plugged directly into the computer. But what I find more interesting is when you plug a micro:bit into the JoyFrog using it as an expansion board. Now, it becomes a cute and very functional interface for your next project.

The JoyFrog features a two-axis joystick with a button, four additional D-pad buttons, an infrared emitter and receiver, eight edge connectors, a buzzer, selectable dip switches, and an audio jack. That's a nice collection of additional inputs and outputs for your micro:bit. →

Below Add lots of inputs to your micro:bit

VERDICT

GAME ZIP 64

A great gaming accessory.

10/10

KittenBot JoyFrog

If you like cute things, then this is for you!

8/10

scroll:bit micro:bit Kit

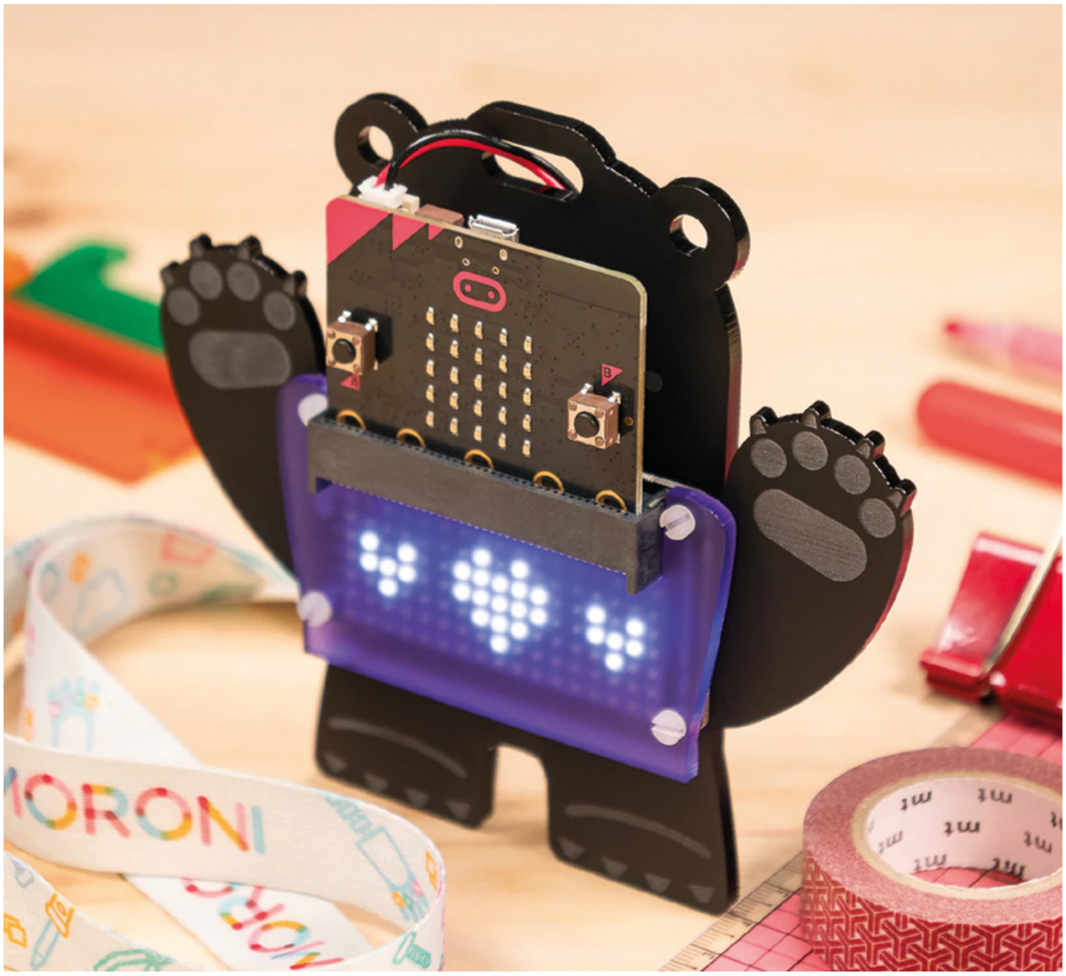
PIMORONI ◈ $46.90 | pimoroni.com

With the scroll:bit micro:bit Kit from Pimoroni, you can build and program your very own wearable bear-shaped badge with integrated LED matrix.

The kit contains everything you need, including all the acrylic pieces, a lanyard, battery box, micro USB cable, stickers, and even the micro:bit V2 board. All you need to do is follow the simple instructions and get connected to your computer to start writing code.

After assembling the kit (no soldering required), you'll learn how to display words or images that scroll by, write code to become a fortune-teller, and create a tilt-controlled ghost animation.

It's a fun kit and makes for a great interactive badge or simple display for your workbench or desk, and the design is sure to make people notice.



Left ◈
Cuteness can be an important feature in making electronics seem accessible

VERDICT

scroll:bit micro:bit Kit

Great-looking and very functional.

9/10

Smart Agriculture Kit

ELECFREAKS ◈ $59.90 | elecfreaks.com

The Smart Agriculture Kit for micro:bit is a well-thought-out collection of sensors and components that will get you up and running with your smart gardening project in no time. At the heart of the kit is the IoT:bit, this allows you to quickly add all the additional sensors found in the kit to the board, and also connect up your micro:bit to the system.

The kit includes an RGB LED, motion sensor, humidity and temperature sensors, an ultrasonic sensor, soil moisture sensor, water level sensor, a servo, and all the wires needed to hook everything up. It's a comprehensive starter kit for anyone wanting to get into automation and gardening. The website gives a few examples of what could be created, including an automatic watering system, temperature sensing, crop growth measurements, and more. Be sure to head over to the website to learn more and visit the extensive documentation and example files.

Left ◈
Electrify your garden

VERDICT

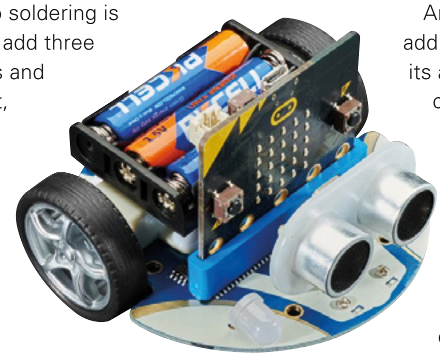
Smart Agriculture Kit

A great automation starter kit.

9/10

Smart Car Cutebot Robot

ADAFRUIT ◈ $29.95 | adafruit.com



The Cutebot is a fully assembled and ready-to-go robot for your micro:bit. No soldering is required, just add three AAA batteries and your micro:bit, and you're ready to start coding. The bot features two high-speed, metal-geared N20 motors and rubber tires, making for a zippy little Cutebot. It also features an ultrasonic distance sensor, two RGB LED headlights, two LED underlights, an active buzzer, and line tracking abilities.

Another nice feature is the ability to add additional servos or sensors with its available header pins. It's a very capable and expandable little robot at a great price point. If you already own a micro:bit, and love little bots, this is something you should have on your wish list. Head over to the product page for more details and links to 15 projects, with code, ready to go. →

Left ◩
Turn micro:bit into motor:bit

VERDICT

Smart Car Cutebot Robot

A great value bot.

10/10

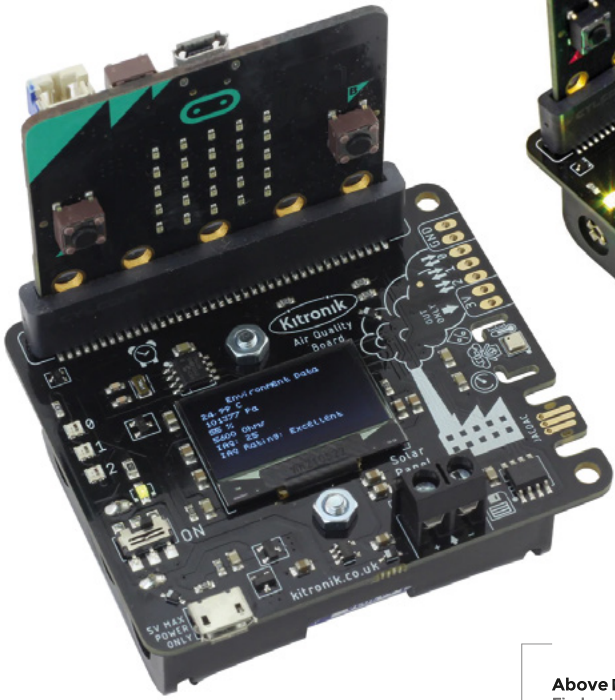
BEST OF BREED

Kitronik Air Quality and Environmental Board

KITRONIK $40.20 pimoroni.com

Air quality monitoring has become a very popular project, and there are a lot of different kits out there on the market. However, very few are as compact and versatile as the Kitronik Air Quality and Environmental Board. The board packs a lot of sensors into a tiny form factor. You can measure temperature, pressure, humidity, eCO_2, and calculate an air quality index. The board also features a black and white 128 × 64 OLED display and three status LEDs. You'll also find a real-time clock module and 1Mb of onboard EPROM memory allowing for long-term storage of the data you collect.

To get up and running, all you need to do is add your own micro:bit and three AA batteries. And if you want to be a little more environmentally friendly, you can add three rechargeable batteries and a solar cell using the onboard connections. The Pimoroni website has links for learning how to use all the sensors and more. Head over to their site for more details!

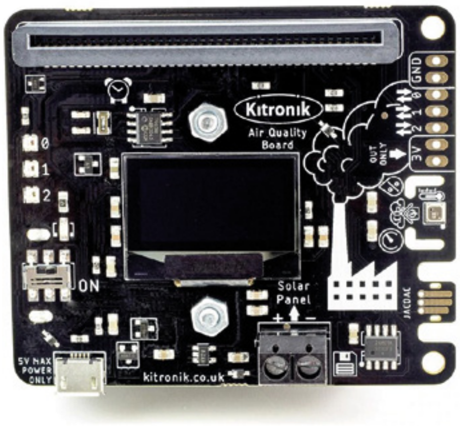


VERDICT

Kitronik Air Quality and Environmental Board

Compact air quality sensing.

9/10



Above Find out what's in your air

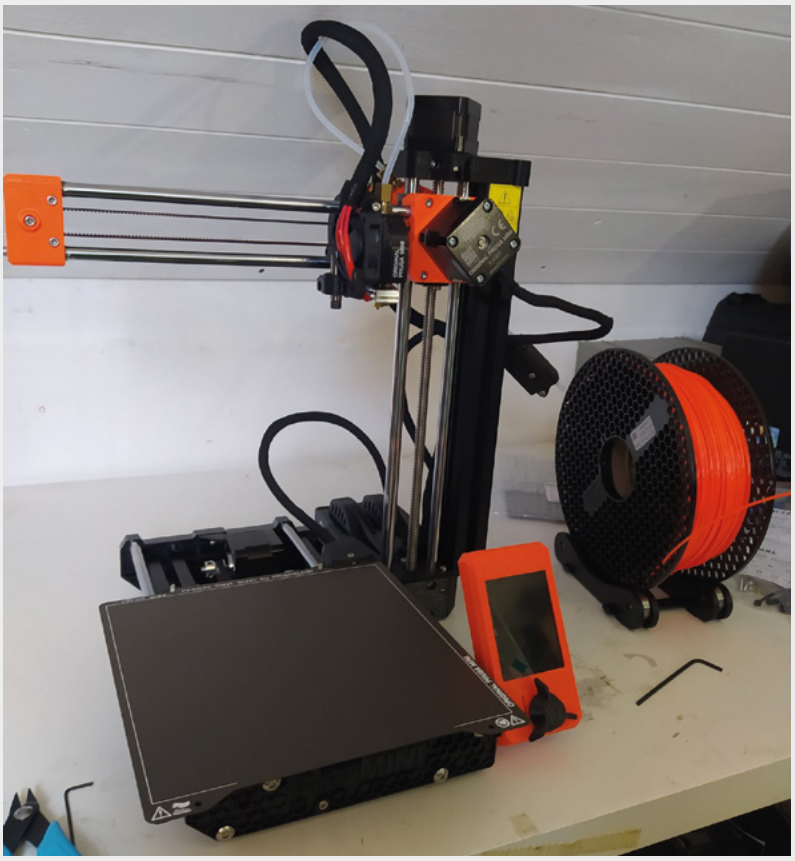
Prusa MINI+

The tiniest Prusa gets an upgrade

PRUSA PRINTERS ◆ £375 | prusa3d.com

By **Jo Hinchliffe** @concreted0g

Below ◆ The semi-assembled version of the Prusa MINI+ is really quick to assemble and calibrate



Prusa 3D printers are both extremely well-known and well regarded in the 3D community. The Prusa i3 design is an absolute classic open-source design that led to many clones and versions based on it. For many though, the flagship Prusa i3 is perhaps at a price point that makes it less likely to be a choice for a first printer, and therefore, many will go with a more budget option. Many of these machines are extremely capable, but are perhaps are less refined than the Prusa, requiring more work to set up and calibrate, and often requiring modification to get good results.

The Prusa MINI+, at £375 (semi-assembled version), sits in the middle of these two tiers. It's a lot more affordable than the larger Prusa i3 but, as we'll see in this review, lacks none of the refinement of its larger counterparts. In fact, in some ways, the Prusa MINI+ exceeds its larger Prusa sibling. We wanted to take a look at this mid-priced home 3D printer.

You can tell from the packaging that a Prusa product has been well thought about, and the MINI+ makes no exceptions to this rule, arriving in excellent packaging. It's available in two formats; the kit version which is slightly cheaper but you need to assemble every component, or the semi-assembled option that only has a very small amount of building to be done to complete the unit. We went for the latter and, opening the box, you see that the MINI+ is in two parts; the base which has the Y axis and heat bed, and the X and Z axis assembly which has the control box attached.

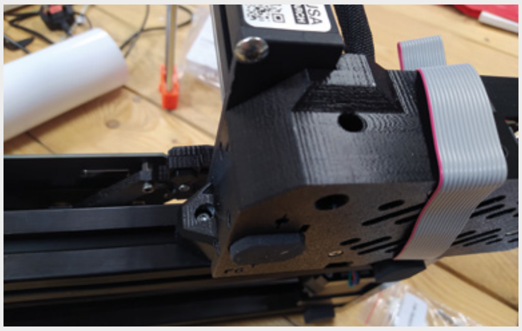
There are numerous accessories included, such as tools, lubricants, a USB drive, and a filament reel holder, and we opted for the package that included both the filament sensor unit and the smooth and the textured build plates. The other item of note is the instruction manual which is double-sided, with one side being the assembly and calibration instructions, and the other side being a book with general approaches to 3D printing.

The assembly of the Prusa MINI+ is a pretty short list of tasks. It consists of adding foam feet to the base, removing the control box lid, attaching the XZ gantry to the base using three supplied bolts, adding the LCD and the filament sensor, and connecting a few well-marked cables. We've actually assembled two of these units now – the first one we very carefully took our time over, and it took around an hour, and the second probably closer to 35 minutes.

The Prusa MINI+ has a build area of 180 mm cubed, which is actually comparable to many larger machines but is smaller than Prusa's i3 offerings. Hardware-wise, it has Prusa's new 'Buddy' motherboard which sports a 32-bit processor and TMC2209 stepper motor drivers running Prusa's own custom firmware. The LCD is colour and the interface is incredibly well laid out. The MINI+ uses a USB pen drive instead of an SD card and, when you insert the USB drive with a new model on it, the LCD renders a 3D image of the new file and, in one click, you can set the file to print. Before each print starts, the MINI+ is fitted with the excellent SuperPINDA probe technology which automatically does a multipoint probe of the build area to ensure consistency in printing the first layer.

After assembly at first boot, the machine automatically takes you through some calibration tests. It moves the axes to their limits and tests the nozzle and heated bed. It then jumps to suggesting you load some filament and perform the Z height/first layer calibration. Just to reiterate, you don't need to read up on how these work – all the information and instructions are delivered to you on the LCD screen in simple terms. The Z axis calibration consists of the nozzle printing a small test pattern, while you use the input dial to move the Z axis up and down in tiny increments until the test print is stuck to the build area and you are extruding a just-right first layer. For both the machines we've assembled, we needed to run this job twice, with the first pass getting it close and the second pass getting it perfect. The MINI+ also comes with a small amount of Prusament PLA filament to get you started.

Below Putting together the semi-assembled version of the MINI+ is really straightforward. The XZ axis attached to the Y axis with only three bolts



One of the great things about having a Prusa is that you are not only buying a machine that has had a huge amount of research and development, but you also get access to the Prusa community. This community is useful for questions, but it's also this community that has brought a lot of data back into Prusa products. A good example of this is using the recommended PrusaSlicer. Of course, the Prusa Slic3r software has built-in support for the MINI+, but it also has printing profiles for a huge range of filaments. Using Prusament PLA or PETG is going to yield great results, but there are many filaments from all manufacturers included, as well as generic profiles for budget no-brand filaments. In use, we have found most filaments work extremely well, but we have only experimented with PLA and PETG so far, but have used a range of Prusa and non-Prusa filaments – all with great results. If, like us, some of your other 3D printers are older and cheaper machines, you will be amazed at how quiet the Prusa MINI+ is – it's certainly usable in small accommodation without disturbing others. □

Left On first boot, the Prusa MINI+ performs a stack of self-tests and calibrations, which it guides the user through

Below A nice included accessory is the filament holder, which is adjustable for different width filament spools

VERDICT

This is an excellent mid-price printer, a great first choice of printer if it's in budget, and a workhorse machine for veteran printers.

10/10

Solder Party RP2040 Stamp

Stripping a development board back to its bare essentials

SOLDER PARTY $12 | solder.party

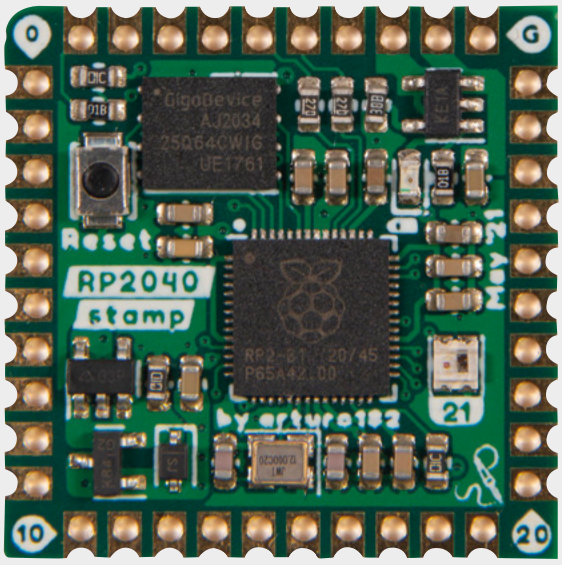
By **Ben Everard** @ben_everard

We live in a world with a dizzying array of different development boards. For almost any microcontroller and form factor, there are several to choose from, with only minor differences. In such a crowded marketplace, it's rare for a board to stand out as unusual or unique, but the RP2040 Stamp does. It's got the core of what you'd expect on a development board – a microcontroller, memory (8MB), and the various external components needed to make everything work – it's even got a few bonus bits such as a reset button, LiPo charger, and WS2812B RGB LED.

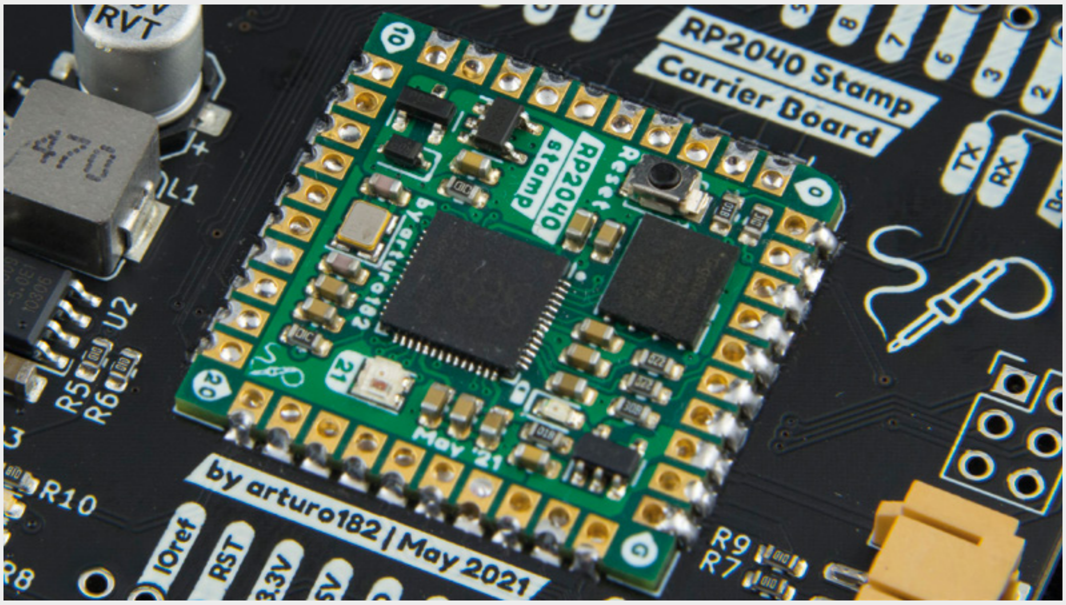
The most obvious thing that's missing, though, is a USB port. Instead, the USB – like all the GPIOs and power connections – are broken out to header pins going round all four sides of the carrier board. Also a little unusually, the pins are 2 mm spacing, not the 2.54 mm spacing of most breadboards and protoboards. Both of these things help keep the board as small as possible – it's just 25 mm by 25 mm. While there are smaller dev boards (such as those in the XIAO form factor), none that we're aware of have as much broken out.

The RP2040 Stamp isn't really a development board in the conventional sense. Instead, it's a module that's easy to mount on custom-designed PCBs. It's got castellated holes and nothing mounted on the rear, so surface-mounting is easy, even with a soldering iron. There are also holes for header pins, if you'd rather mount it this way.

All this means that if you want to create a custom project based on RP2040, you can use a Stamp and all the hard work – the more complex routing and small components with fiddly soldering – is done for you. At the same time, you don't have the constraints that come with a typical dev board (such as having to mount the board by the edge to expose the USB). This lets you concentrate on the bits that make your project uniquely yours, and you don't have to fuss about the boilerplate work of getting the



Right Almost everything you need for an RP2040 project in 25 mm × 25 mm



Left The Stamp soldered in place on the carrier board

Below The carrier board is both great for testing your ideas and a reference design for supporting circuitry

microcontroller running. There is a trade-off to this, and basically, it's a trade-off between unit cost and time. Doing everything from scratch will be a bit cheaper (especially on larger runs), but using the Stamp will save you time. For most of the projects we make, this is a trade-off that's well worth making.

Another advantage of having everything on a standard module is that the software setup only has to be done once, so there's a CircuitPython version, and board definitions for the Pico SDK already created.

What takes a project like this and turns it from something niche into something really useful is the non-hardware bits that go together to make it easy to use. That's what makes it approachable for people with intermediate electronics skills, who would get the most out of this.

It's here that Solder Party has really helped its board stand out. To start with, there are footprints for EAGLE, KiCad, and EasyEDA, which should cover most hobbyists. There is also a reference design for the additional hardware in the form of the carrier board schematics. This shows you exactly how to hook up things like USB and LiPo ports. These are not complex, but having them explicitly laid out gives us much more confidence when designing our own PCBs.

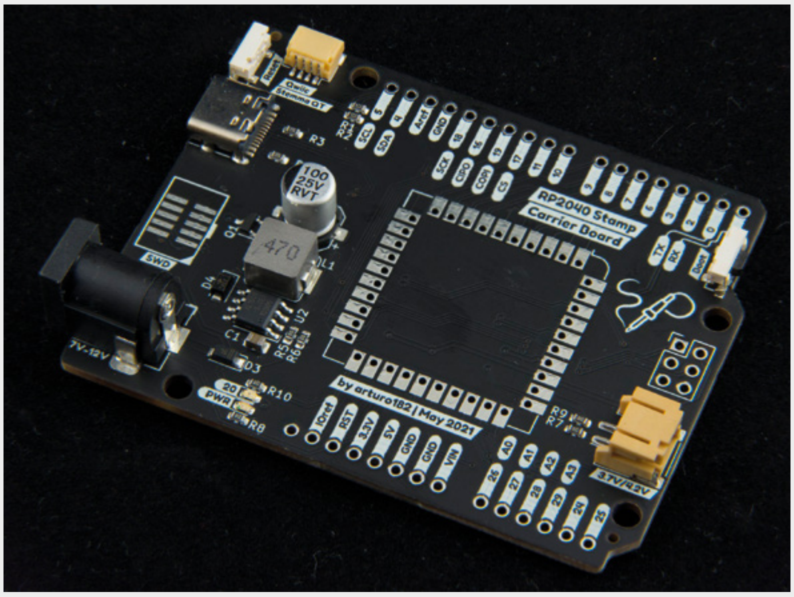
We were sent two RP2040 Stamps to test out. The first we've been testing out using the carrier board. The second, we're putting in a design – we're still waiting on boards to come back from the manufacturer, so stay tuned for that in a future issue.

The RP2040 Stamp fills a very important niche for makers. By breaking everything out to headers, it gives you a lot of freedom for designing the PCB that will hold it, but, by including all the components needed to get the microcontroller running, it does a lot of the hard work. It fills the space between creating a shield for a more general-purpose dev board and creating a completely custom PCB using the bare microcontroller. As PCB design becomes more popular and accessible, this is a more important niche. ◻

VERDICT

Build your own RP2040-powered PCBs easily and cheaply.

9/10



next month

issue #54

ON SALE 21 APRIL

CLASSIC COMPUTERS

ALSO

→ RASPBERRY PI PICO

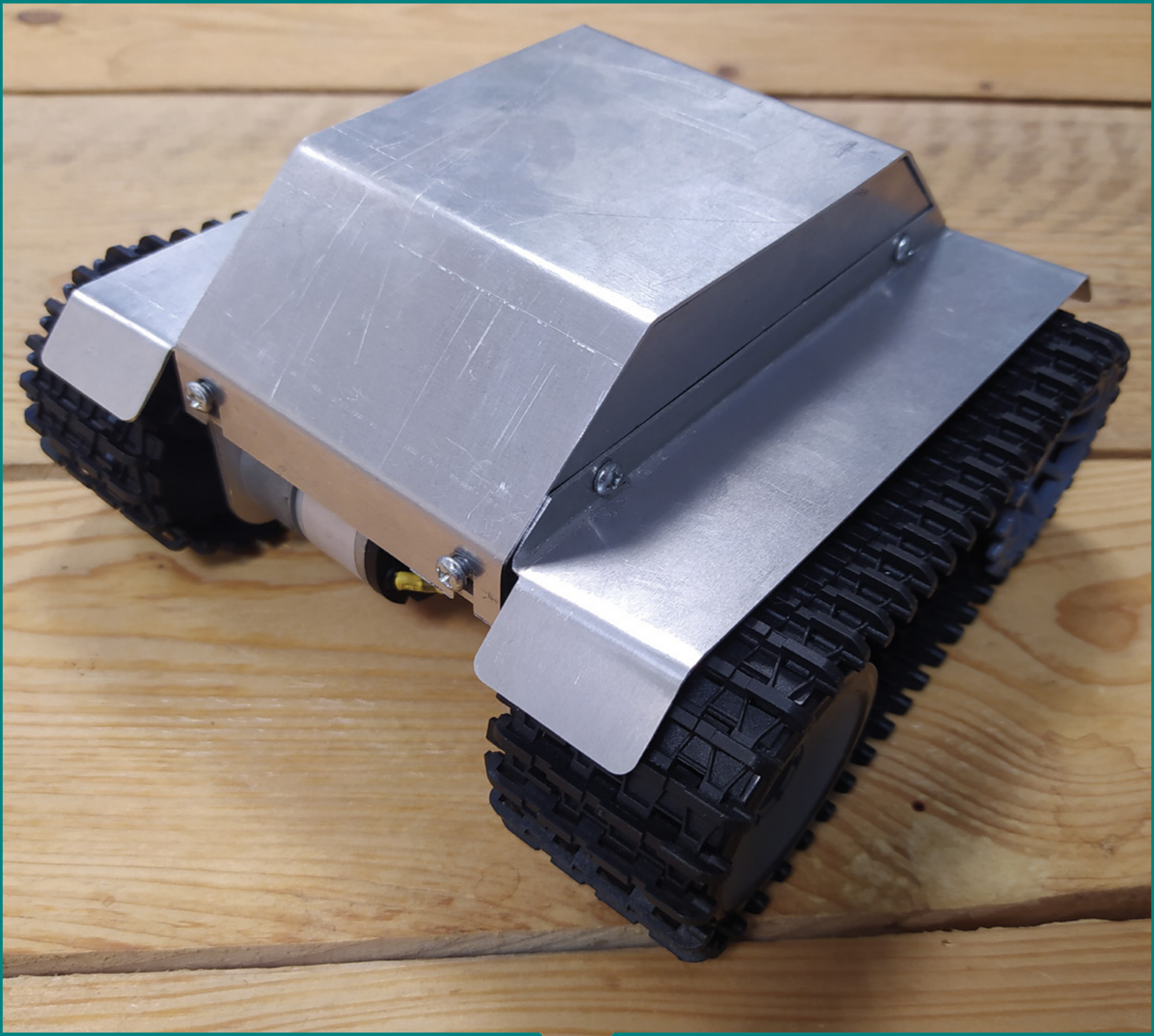
→ DIY MUSIC SYNTHESIZER

→ GARDENING

→ AND MUCH MORE

DON'T MISS OUT

hsmag.cc/subscribe



3D design with FreeCAD

3D design is an essential maker skill, and modern tools let you design for almost any medium including 3D printing, CNC work, and folding sheets. There are many different software options, but here at HackSpace magazine, we're huge fans of FreeCAD. As the name suggests, this is free software that you can use (and contribute to).

In our forthcoming anthology, we'll bring together Jo Hinchliffe's tutorial series to guide you through getting started. Keep an eye on our website – **hsmag.cc** – for details of the release.

HS