

MAKE | BUILD | HACK | CREATE

# HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc

July 2020

Issue #32

MICROPYTHON  
ON ESP32 **PLUS**  
DIY MOBILE  
PHONE +  
3D PRINTER  
NOZZLES



1GB  
8GB  
RASPBERRY PI 4

## EXTREME BUILDS

Meet the makers going higher, further, and faster



How one hacker built  
his own METAL BAND

ROBOT  
GUITAR



July 2020  
Issue #32 £6



CABLE TIES FIBREGLASS BUILD A CLOCK SPARKLES

**FREE  
SHIPPING**

ON ORDERS OVER  
£33 OR \$50 USD\*



*let's make  
something!*



DIGIKEY.CO.UK

0800 587 0991  
**DIGIKEY.CO.UK**



9 MILLION+ PRODUCTS ONLINE | 1,000+ INDUSTRY-LEADING SUPPLIERS | 100% FRANCHISED DISTRIBUTOR

\*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. © 2020 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA

 ECIA MEMBER  
Supporting The Authorized Channel



# Welcome to HackSpace magazine

The only way to know what the limits are is to push something until it hits them. That's what the makers we're looking at in this issue are doing. By looking at what it takes to make something perform in extreme environments, we can learn more about the best ways of making things work in more sedate places.

If you're looking to send your own creation to the extremes, we also look at a whole heap of techniques and parts to help

**If you're looking to send your own creation to the extremes,** we also look at a whole heap of techniques and parts to help you do it

you do it, from new, high-power control boards (in the form of Raspberry Pi 4 8GB and Arduino Portenta), strong and lightweight construction methods such as fibreglass, and pushing your 3D printer to its limit with different nozzle selections. Whatever you're planning on building, we're here to help.

## BEN EVERARD

**Editor** [ben.everard@raspberrypi.org](mailto:ben.everard@raspberrypi.org)

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

get in touch at  
[hsmag.cc/hello](http://hsmag.cc/hello)

## GET IN TOUCH

- [hackspace@raspberrypi.org](mailto:hackspace@raspberrypi.org)
- [hackspacemag](http://hackspacemag)
- [hackspacemag](http://hackspacemag)

## ONLINE

- [hsmag.cc](http://hsmag.cc)

Available on the  
App Store

GET ON  
Google Play



# Contents

106



06

## SPARK

- 06 **Top Projects**  
Creativity in physical form
- 18 **Objet 3d'art**  
Great Scott! It's a 3D-printed DeLorean!
- 20 **Meet the Maker: Evil Mad Scientist**  
Inside the secret hollowed-out volcano lair
- 26 **Columns**  
Can you teach yourself to be creative?
- 28 **Letters**  
What do you want to see more of? Let us know!

31

## LENS

- 32 **Extreme Builds**  
Projects taken to the limit
- 46 **How I Made: Robot guitar**  
Move over Tom Morello: the machine has won
- 52 **In the workshop: Raspberry Pi camera**  
Turning a sensor and a lens into something useful
- 54 **Interview: Gina Häußge**  
3D printing is fun because of one person. Thanks, Gina!
- 62 **Improviser's Toolbox** Cable ties  
Snip, weave, and melt your way to maker excellence
- 66 **Raspberry Pi 4 8GB**  
Twice the RAM and now a 64-bit OS for Raspberry Pi

Cover Feature



32

Tutorial

## Photogrammetry



82

Bring real objects into 3D design software



52



18

**School of making****Fibreglass**

76

Make moulds for building your next boat/rocket



80

**How I made****Robot guitar**

100

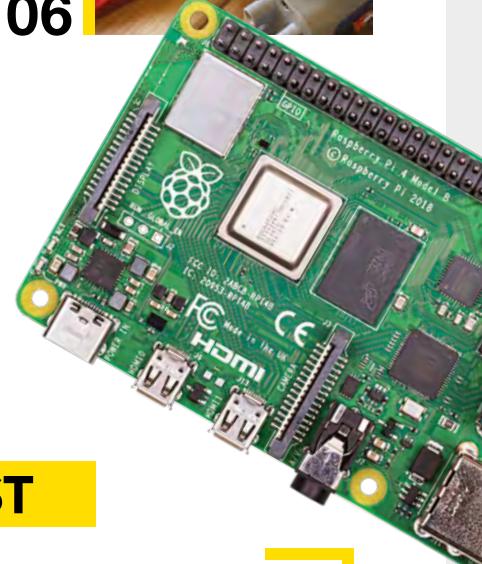


46

Musicians, replace your feeble flesh fingers with PCBs and solenoids



06



71

**FORGE****72 SoM MicroPython**

Control devices from the internet

**76 SoM Fibreglass**

3D-print moulds for light, flexible builds

**80 Tutorial MetalFil**

Printing in metal and stone filament

**82 Tutorial Photogrammetry**

Scan real objects with your phone

**88 Tutorial 3D photography**

Combine two images for stereo vision!

**94 Tutorial Build a robot**

Upgrade a cheap chassis into a unique build

99

**FIELD TEST****100 Best of Breed**

The best clock kits

**106 Can I Hack It?**

Obsolete electronics put to good use

**108 Review E3D Nozzle Fun Pack**

Upgrade your printer for more versatility

**110 Review Ringo mobile phone**

DIY mobile computing

**112 Review Arduino Portenta H7**

Two cores for extra power

66

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

# Flux capacitor

By Hojadurdy Durdygulyjov

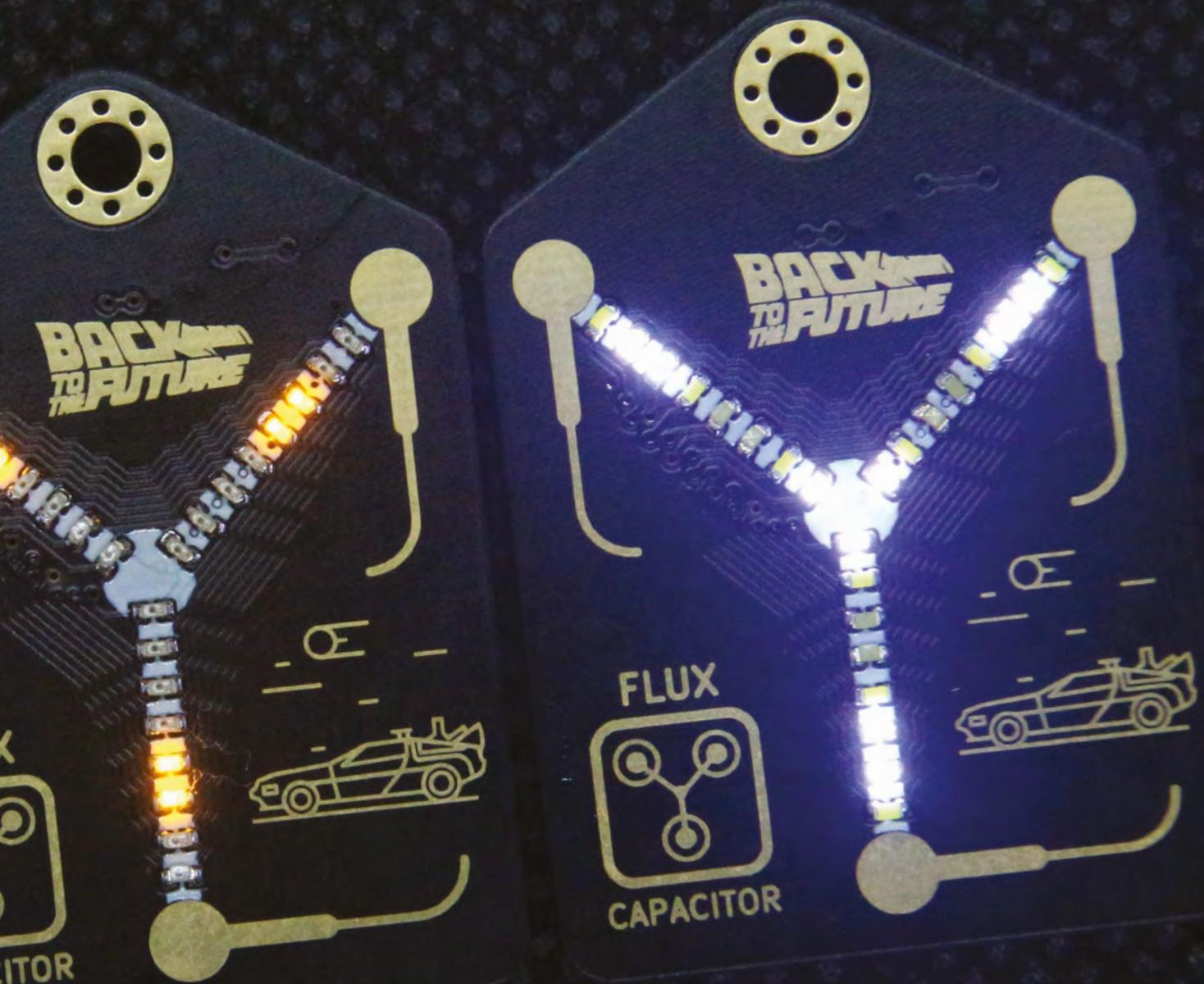


twitter.com/Hojadurdy

**T**echnically, this coin cell-powered badge is just a few surface mount LEDs on a PCB, plus an ATtiny44A to control the animation. But combined with the artwork on the silkscreen and the placement of the traces, it becomes a dynamic homage to the greatest film of 1985: *Back to the Future* (there must be something about this film that strikes a chord with makers, as we've another *Back to the Future*-inspired build on page 18). □

**Right**

Maker Hojadurdy Durdygulyjov is an electronic engineer by day; time traveller by night



# Wireless tin can telephone

By Geoff McIntyre

 [hsmag.cc/016OF6](https://hsmag.cc/016OF6)

We've all run a length of string between two tin cans at some point. It's cheap fun, and gives a good introduction to sound waves. Geoff McIntyre has upgraded this classic experiment for the electronic age.

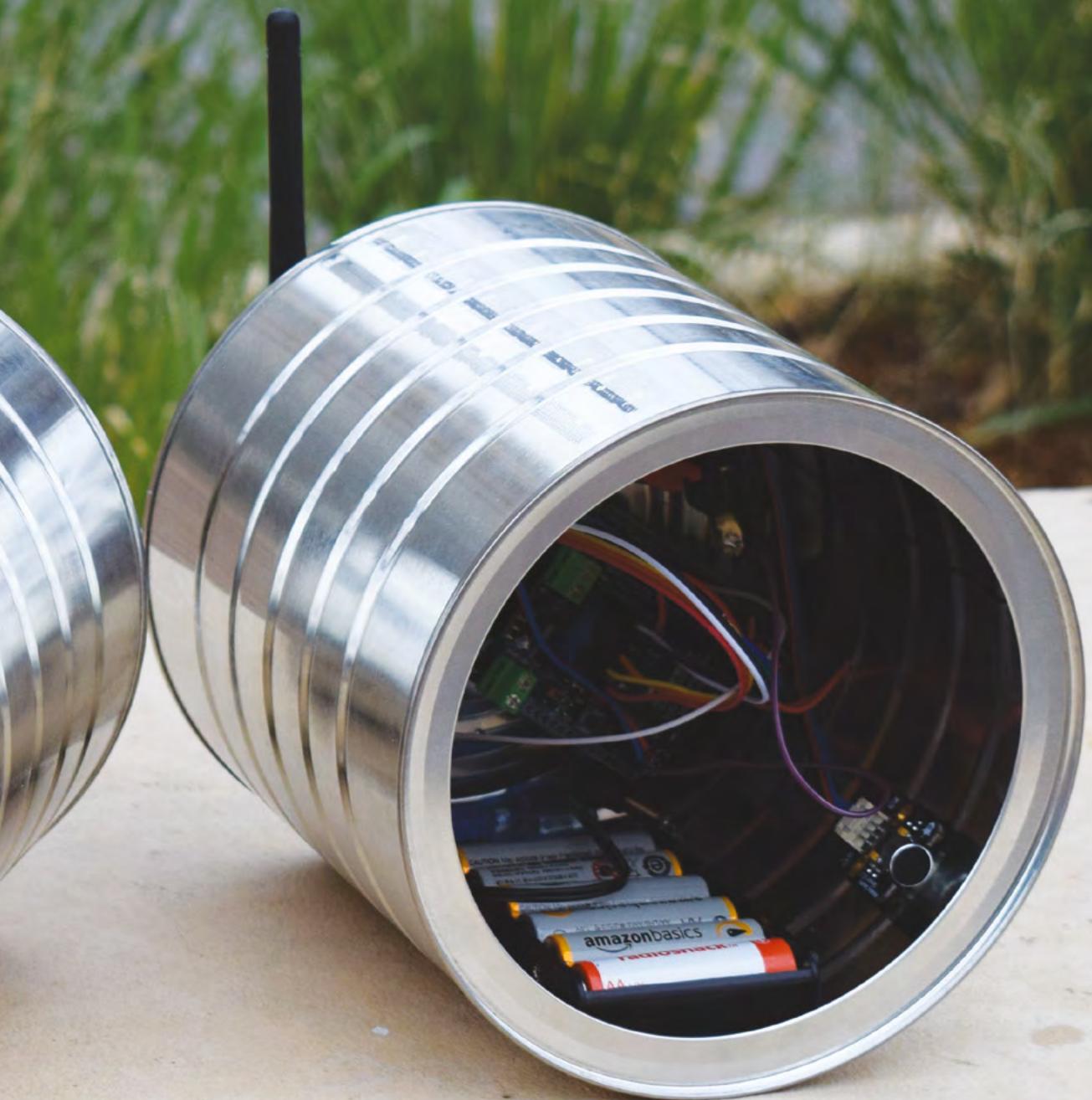
His creation uses a microphone input plugged into the analogue input of an Arduino Uno to take in sound, which it transmits with an nRF24L01+ wireless transceiver. At the other end, there's an LM386 amplifier and a small speaker.

Geoff's instructions are beautifully clear, and he's even included a banana to give a rough sense of scale. □

#### Right

Geoff's creation shows how few components you need to make a functioning comms device





# Lego coin sorter

By AnhPhu Nguyen

 [hsmag.cc/NFqi6p](https://hsmag.cc/NFqi6p)

M

**ake, fail, repeat: that's the mantra of successful hardware developers, and that's what makes**

**Lego so useful.** You can take it apart and put it back together without destroying anything, until you get your build just right.

That's how AnhPhu Nguyen built this automatic coin sorter. There are no optical scanners to measure the diameter of the coin passing through it, nor are there scales to weigh them. This is just clever hardware engineering, relying on the speed of the coins as they roll down a slope (the angle of which took some adjusting) to guide them to the right diameter hole. No computer required! □



**Right** □

We like this project for its simplicity: it's completely analogue



# Cat wheel speedometer

By Shawn Nunley



[hsmag.cc/k4BLWM](http://hsmag.cc/k4BLWM)

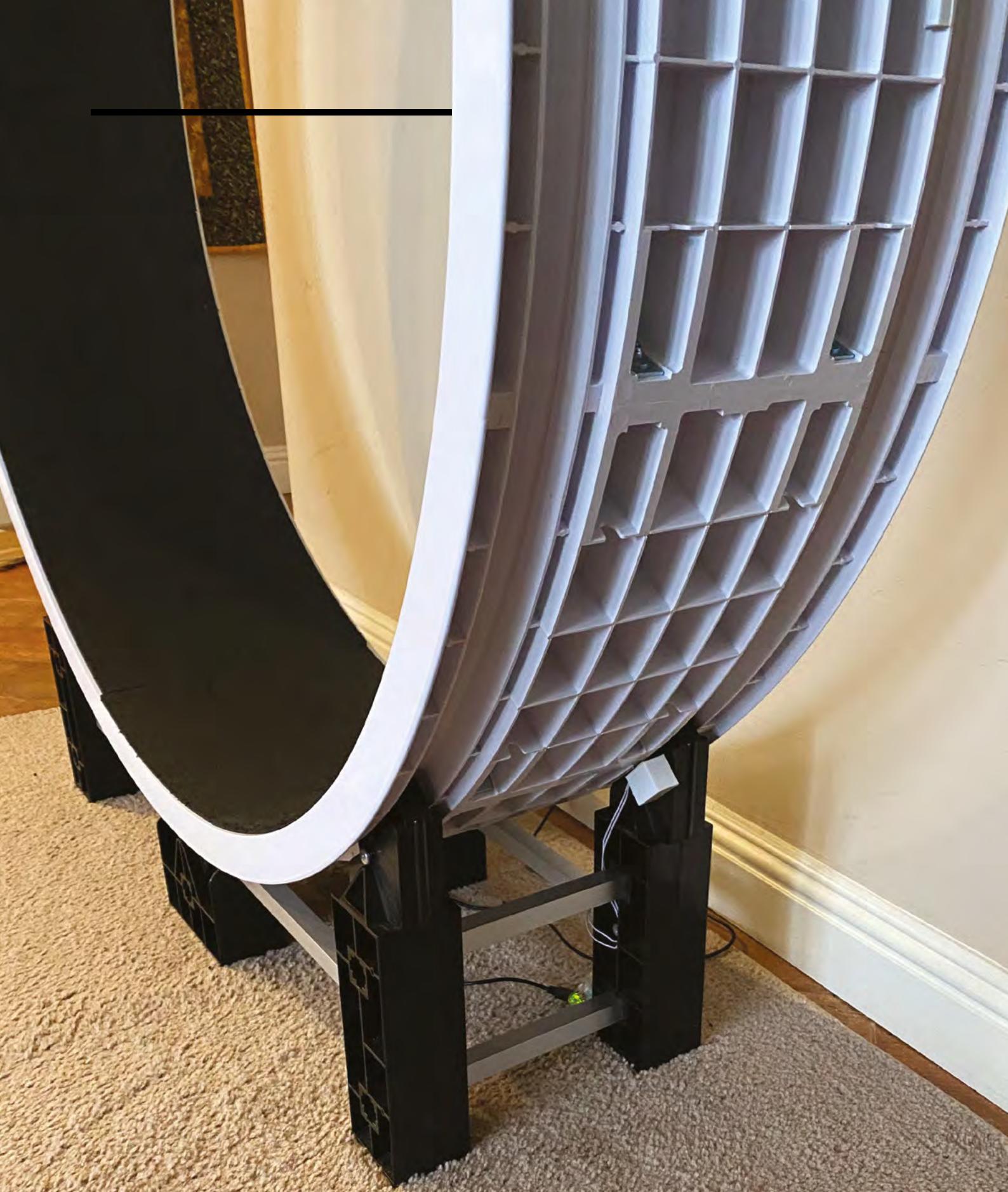
This speedometer/odometer for cats uses a Raspberry Pi Zero W and six magnetic sensors, which each record when 2.095 feet of wheel travel has gone by. Its maker, Shawn Nunley, attached it to an existing cat exercise wheel when he grew curious about how much exercise his cats were getting.

Apart from the technical challenges, the thing that struck us about this build is that the cats are willing to co-operate and actually use the thing. Shawn explains it like this:

"These are Bengal cats. Mine is actually a G2 Bengal which is to say he's two generations away from Asian leopard cat. Since they have quite a bit of wild blood, they are way more active than normal cats. The wheel is really so he has an outlet to spend his energy, and thank God he uses it. My first Bengal cat did not have a wheel. Bad things happen when your Bengal has unspent energy!" □

**Right** □

Shawn's cats do about 4–6 miles per night on this wheel, and reach a top speed of 14 mph



# Components pendant

By Razvan Coloja



[hsmag.cc/bXOXBJ](http://hsmag.cc/bXOXBJ)

The psychological benefits of switching off at the end of the day by making something are well-known. So it's great to see that these pendants, made from dead electronic components, are made as a form of stress relief by a real-life psychiatrist and Linux geek, Razvan Coloja. They're small, nicely made, and look a good bit more comfortable than some of the bare PCB jewellery we've seen out there. □



**Left** □  
Check out more of  
Razvan's work on  
his Tindie store





# Giant Battleships

By Dan Aldred

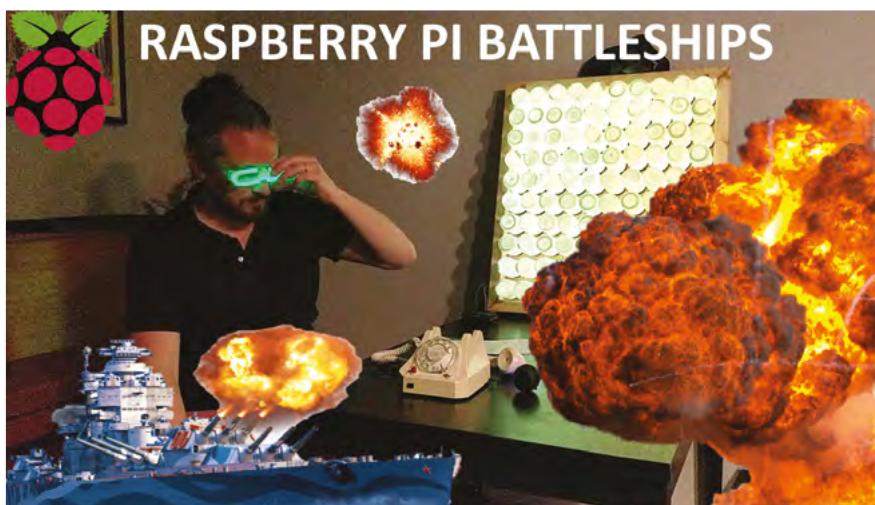
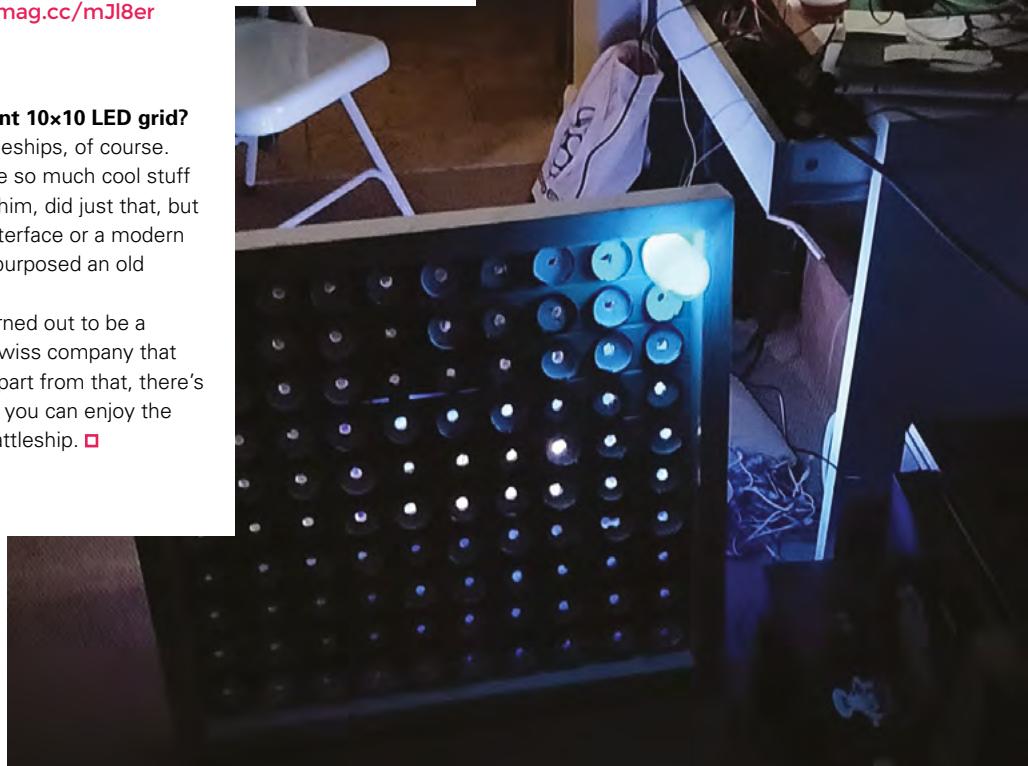
 [hsmag.cc/mJl8er](http://hsmag.cc/mJl8er)

**W**

hat do you do with a giant 10x10 LED grid?

Turn it into a game of Battleships, of course. Dan Aldred, who has made so much cool stuff we're a little bit in awe of him, did just that, but rather than using a web interface or a modern push-button device, he repurposed an old rotary phone as the game controller.

Getting hold of a schematic for the phone turned out to be a challenge – it turns out that it was made by a Swiss company that specialised in WWII-era surveillance devices. Apart from that, there's a Raspberry Pi, some Python, and a speaker so you can enjoy the sound of victory as you sink your opponent's battleship. □





**Left ♦**  
If you're going to do anything with bakelite (which is what the phone case is made from), make sure you take precautions – the old stuff was made with arsenic and asbestos



# Objet 3d'art

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

**G**reat Scott! Here's a 3D-printed radio-controlled 1982 Delorean DMC-12, created by Brett Turnage in homage to one of the greatest films ever made – *Back to the Future*. It's incredibly detailed – there are 69 separate STL files to print out and assemble, including the badge and the essential Mr. Fusion portable reactor that made its appearance in the final scene of the film.

The detail isn't just cosmetic: Brett recommends that functional parts, such as suspension and servo mounts, are printed with an infill of 80–100% for strength, and he's made engineering adjustments in areas such as the door hinges to allow for the fact that printed PLA isn't as strong as the original car's stainless steel.

The print is made functional with a Tamiya F104 Pro II R/C car kit, and a few extra parts such as a motor, battery, and car controller. Brett's even added a length of electroluminescent wire to give it the full 'time travel' effect. You can watch it in action here:

↗ [hsmag.cc/cwojUe](http://hsmag.cc/cwojUe) ↘

**Right** ♦  
A Delorean wouldn't be a Delorean without functional gull-wing doors





# Meet The Maker: Evil Mad Scientist

Making inspiration for current  
and future mad scientists



**R**egular HackSpace magazine readers will know the name **Evil Mad Scientist**. We featured their EggBot kit last issue in our roundup of beginner's kits, and the AxiDraw pen plotter is all over the internet – at least, it's all over the hashtag #plottertwitter.

They're also keen supporters of open-source hardware. We caught up with 50% of Evil Mad Scientist, Lenore Erdman, to find out what made them become accidental entrepreneurs.

"We started Evil Mad Scientist accidentally. We did not mean to start a business. We went to the very first Maker Faire with our project and people said 'Ooh, how d'you do that? I want to do that!' So, we

**What motivates us is that people are interested in and enjoy what we do, and that's still what drives me**

started making kits to make it feasible for other people to do projects like ours. Every time we would do a kit, we would bring money back into the next round of the kits. It grew very gradually, and now it's our full-time job. It's been a slow-going, organic, interesting journey.

"That first project was our interactive LED dining table. It had 400 LEDs and a connected series of nodes that had a light sensor on them. When something would change over that sensor, it would

change which LEDs were on and how bright they were. For example, when you pass the salt, or move the napkin, or pick up a drink, a sensor would send a message to its neighbour node – I've changed, do you want to change? That would trigger a ripple of changes throughout the table.

"The interactive LED panels that we made for that were large – initially the PCBs were 12 by 14 inches; now the ones we make are 12x12 inches, and they have 80 LEDs on a panel. And they're completely analogue – there's a damped signal, so they'll come on and ripple and fade out. And they're pretty. People still get them. Coffee tables, bars, nightclubs. There's a museum in Texas that has a wall of them, and there's a museum in Australia that made an archway that you could walk through with them.

"It took a few years for me to go full-time, and Windell [Windell Oskay, co-founder of Evil Mad Scientist] went full-time a year or two after that. The years blur together after a while! We've been doing this since 2007. It was several years before I went full-time.

"I had no dream of entrepreneurship. What motivates us is that people are interested in and enjoy what we do, and that's still what drives me. People love it; they get so much out of it. It's really rewarding to see how much people get out of the projects that we do.

#### FINDING YOUR COMMUNITY

"We sell components, we sell kits, and we sell plotters. The components tend to be purchased by ➤

**Above ♦**

The EggBot has put open source hardware into some surprising hands – and there's now an ostrich-sized version



**Above** EMS's deconstructed 555 timer shows exactly what goes into an integrated circuit

educators. LEDs for classroom use, pager motors for making art bots.

"They get used, for instance, by model train enthusiasts who want to make their trains more realistic and who want to put LEDs into their trains, but who find it hard to shop for LEDs at a traditional electronics store because there isn't information, or someone to contact about how to do that. Well, I have a really good article about what resistor you

problems that you and I don't know exist. They're looking at very narrow problems in fields that we would never think about our thing being used there.

"For example, a group was studying coral reproduction, which is tied to the phase of the moon. And so, they wanted to shift the length of the month: what happens if you have 30 days instead of 28 in the cycle? They used some of our LED boards to control the light source to the coral over the course of the experiment – I would never have imagined someone ever needing to selectively light a tank of coral!

"One interesting use case for our stuff that we helped out with is a group that's using the AxiDraw as an XY stage. Instead of using it for drawing, they're using it to make a light source and a sensor on the head of the AxiDraw, and they have it mounted on a tripod. What they're doing is material analysis of artwork. They're scanning the artwork to find out what's in the pigments.

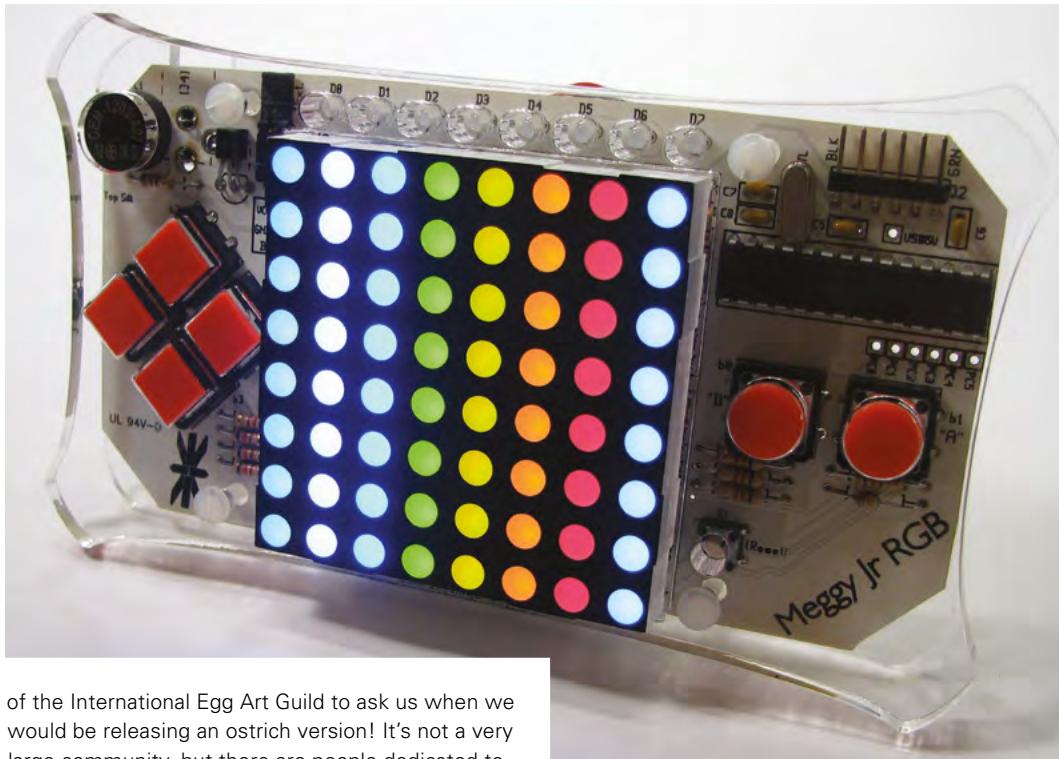
"The reason they didn't use any existing scanners is that they needed it to be portable – they can't move the artworks. Most collections are not going to let you move a painting, still less a mosaic on the floor. So if they can mount their XY scanner on a tripod, they can make it portable

"Here's another great niche. When we started making the EggBot, we were selling it to a maker crowd, thinking that robotics hobbyists were getting to try out some art. We were contacted by members

**“This is one of the beautiful things about open-source hardware – when you document your hardware well, people can use it for other things”**

should use with your LED if you're using an AA battery, for example ([hsmag.cc/MFcmTe](http://hsmag.cc/MFcmTe)). We have niche cases like that where it's a hobby that's not necessarily electronics-related, but somebody wants to do something with LEDs or electronic components.

"This is one of the beautiful things about open-source hardware – when you document your hardware well, people can use it for other things. Scientists are always looking for solutions to



of the International Egg Art Guild to ask us when we would be releasing an ostrich version! It's not a very large community, but there are people dedicated to making art with eggs.

"AxiDraw has gotten us several new niches, including people sending direct mail. They want a letter, or the address on an envelope, to look handwritten. This is a large group of customers, and some of them are doing interesting things, often retail- or marketing-related.

"And that lets us make machines that artists love. #plottertwitter is a joy, and much of #plottertwitter is AxiDraw. There are also people using vintage plotters which are wonderful machines. We've been doing plotter projects since the very beginning of →

#### **Below** ♦

The project that launched a company – Evil Mad Scientist's interactive LED dining table



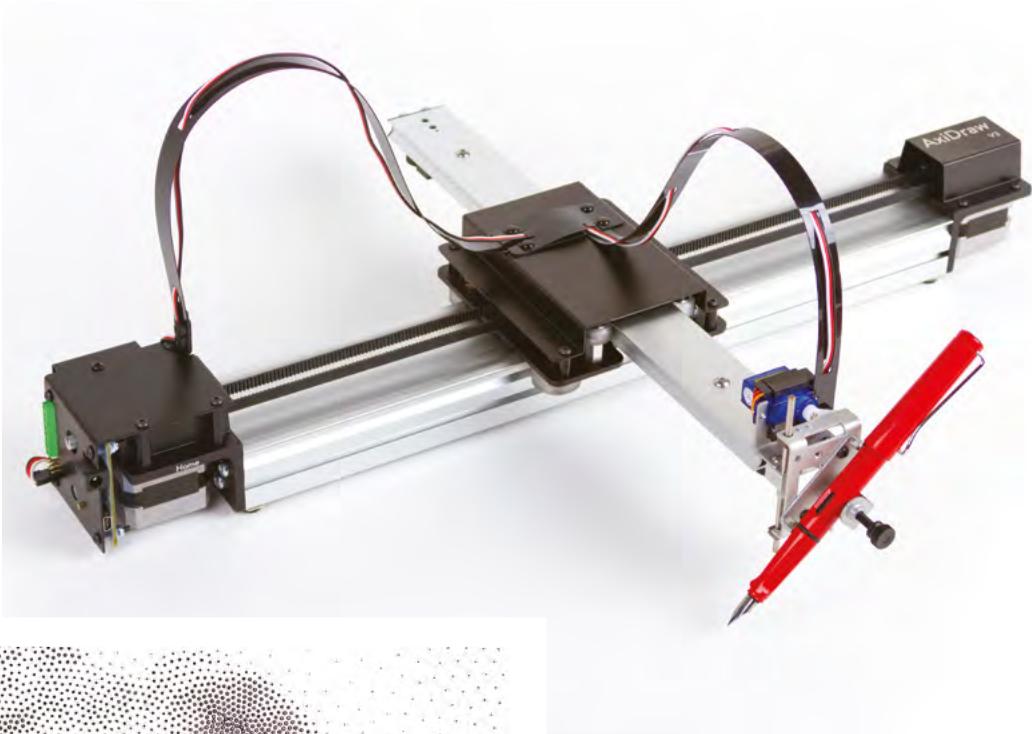
#### **Left** ♦

The Meggy Jr RGB is certified open source hardware, so you're free to pull it apart, copy and modify it to your heart's content

#### **Below** ♦

The Digi-Comp II – a gravity powered mechanical computer, seen here in its gigantic form

## REGULAR



**Right** ♦  
The AxiDraw, a USB,  
computer controlled  
pen plotter...

**Below** ♦  
... has spawned a  
flourishing community  
of plotter artists



Evil Mad Scientist, whether as side projects or related to other research projects.

"We've been inspired by pen plotters for a long time, and the vintage ones are incredible hardware. It's interesting to watch these communities of plotter artists that have grown up. Most of them will recommend the AxiDraw as a start machine, but for people with a lower budget and more time on their hands, they'll recommend the vintage plotters, with the caveat that you'll be in for a lot of work in terms of connectivity, software, and all those issues. There's no tech support, for example."

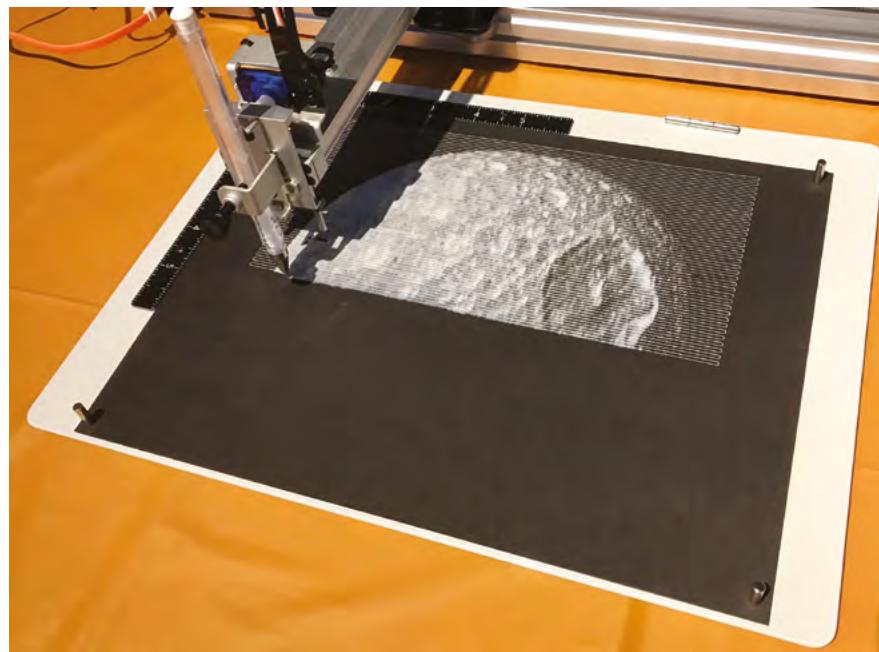
### NEW HORIZONS

"We're always working on new AxiDraw-related things. Whether it's software or hardware, there are some advances we'd like to make on the hardware. One of the difficult software projects related to handwriting is stroke reconstruction. If you're generating strokes, if you start with a font, that's relatively easy. But if you want to reproduce something that already exists, that's actually quite difficult: knowing which stroke to draw first, what direction to do it. For example, if you think about the letter X, you and I know that it's two lines that cross. The computer does not necessarily know that. Is it two Vs connected?"

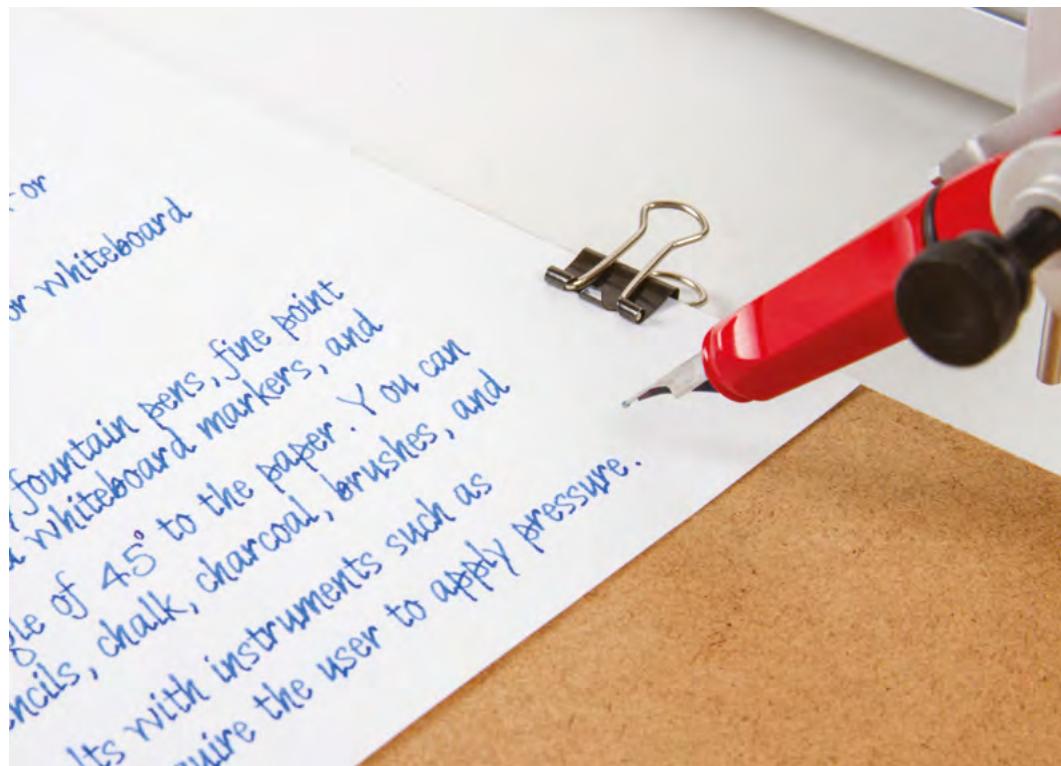
"People will say 'I have hand-drawn artwork – can I get AxiDraw to draw it?' Well, did you draw it on the computer? Did you record those strokes in a way that AxiDraw can do? There are tools for vectorising scanned artwork, but they're limited. They don't necessarily know the intention of the artist.

"One of the questions I get a lot is: 'how should I get started?' And it's a really tough question. For most people, getting started shouldn't necessarily be an open-ended kit. It should be something that's going to capture their interest, so they should have a goal in mind, or an area that they want to focus on. The most successful introductions to electronics happen not because you want to learn electronics but because you wanted to do A Thing: you wanted to light up the epaulettes on your costume; you wanted to light up the tinder-box on your train engine; you wanted something to make a noise when something happened.

"So, the most successful learners usually have a goal in mind. And a lot of our projects are like that. If you have a kit and you build it, you end up with something that does something. If it's a clock, if it's a robot eye... sometimes that's just enough to get someone hooked and bold enough to try another thing." □



**The most successful learners usually have a goal in mind. And a lot of our projects are like that**



**Left** ♦  
The machines can now forge handwriting. What could go wrong?

# Finding new ideas

Are you like me, and don't feel 'naturally creative'?



**Lucy Rogers**

@DrLucyRogers

Lucy is a maker, an engineer, and a problem-solver. She is adept at bringing ideas to life. She is one of the cheerleaders for the maker industry, and is Maker-in-Chief for the Guild of Makers: [guildofmakers.org](http://guildofmakers.org)

**F**or most of us, creativity is defined as being able to paint, draw, or write stories. I challenge that. Creativity is the ability to IMAGINE new things and to act on those thoughts.

Creativity is not innate. It's not binary – it's not that you have it, or you don't. We have tendencies to do certain things, but creativity is a skill. And as a skill, it can be learnt, and it improves with practice.

Here are seven tricks that have helped my creativity:

**1) Permission is granted.** This was one of the most difficult things to get my head around. Perhaps you think you 'should' be doing something else, or that you're not creative/good enough.

Do you give yourself permission to read a book, watch TV, go on social media? Then give yourself permission to be creative for fun.

**2) Play, fun, fiddle.** Find something and pick it up. Touch it. Hold it in a different way than normal. For example, if it's a pen, grab it with your non-writing hand. Take that thing – pen, paper clip, cat – and imagine it's an aeroplane. 'Fly' it around. By playing, by having fun, and by being silly, we unlock parts of our imagination that our 'adult' brains have tidied away.

**3) Prolific and limitless.** When I need to solve a problem, I make as long a list as possible of ways it could be solved – and then keep going, because sometimes it's the ridiculous ideas that lead to a better understanding of the problem.

**4) Copy, copy, create.** I am never going to be able to emulate my creative heroes well enough to be mistaken for them, and I don't try and pass my work off as anyone else's – so it's not plagiarism! By copying, you practice and, with practice, you will begin to find your own 'voice', your own style. And if you copy a few people, you won't be the next Escher or the next Heath Robinson, but you will be the first YOU.

**5) Change your perspective.** Do you remember looking out of an aeroplane window and thinking the towns below look so different? Use a phone camera and zoom in, and see things from a different perspective. Get down and see things from an ant-eye view.

Looking differently can change your mindset and jump you in to different ways of thinking.

**6) Critical eye.** Do you evaluate things? Do you think 'ooh that's good', or 'ooh that's bad', and then

wonder what was it about it that was good or bad? It's the 'because' that will help you develop.

And finally,

**7) FAIL, learn, and try.** Are you scared of failing because you'll look silly? Imagine if a toddler did that. 'Oh, I won't bother with this walking thing any more because I fell over last time.'

Try something that really isn't you – I've tried ballet, (and discovered I liked it!), I've learnt how to use TikTok (and discovered I liked it), I tried to learn to fly (and discovered I got airsick.)

Creativity is the ability to IMAGINE new things, and to act on those thoughts. □

**Sometimes it's the  
ridiculous ideas  
that lead to a better  
understanding of  
the problem**

# The art of PCBs

When form is as important as function



**Drew Fustini**

@pdp7

Drew Fustini is a hardware designer and embedded Linux developer. He is the Vice President of the Open Source Hardware Association, and a board member of the BeagleBoard.org Foundation. Drew designs circuit boards for OSH Park, a PCB manufacturing service, and maintains the Adafruit BeagleBone Python library.

We live in an age where it's easier and cheaper than ever to create printed circuit boards (PCBs), and the hardware

**hacking community has embraced them as a way to practise and show off their technical and creative talents.**

If designing and making your own circuit board is an electronics rite of passage for many hardware hackers, then DEF CON's Badgelife has got to be our Olympics.

At DEF CON, the iconic Las Vegas hacker conference that has been running since 1993, your

conference badge is not just proof you've bought a ticket, it's a PCB with electronic components that forms part of a conference-wide puzzle. Cracking this puzzle is a social and technical

feat rewarded with the coveted Black Badge, which gives you free entry to DEF CON for life, plus a big bucket of hacker kudos points. Inspired by these DEF CON badges, groups and individuals in the hardware hacking community started creating their own badges, creating what is now known as Badgelife. Sophi Kravitz made a fantastic short documentary on Badgelife for Hackaday that you can watch at [hsmag.cc/YPHvYr](http://hsmag.cc/YPHvYr).

Badgelife is a great representation of the community of hardware hackers and programmers who design and make electronic conference badges as a form

of artistic expression – for example, the iconic Bender Badges from ANDIXOR, and the work of Twinkle Twinkie. Searching the internet or social media for Badgelife, especially during a big electronics conference, will provide many more awesome examples of these artistic PCB badges and badge add-ons, often referred to as SAOs.

SAOs follow a community standard for connecting badge add-ons, providing power, ground, and an I<sup>2</sup>C bus, with more recent connectors also providing GPIO. Making these little add-ons proved to be much more accessible – in terms of

time, money, and knowledge required – than full-blown badge design, so the number of people making their own blinking, bleeping, blooping, and sensing PCB art soared.

But PCB art isn't just limited to badges and SAOs. Boldport has been making stunning DIY circuit art kits for years, Andy Sowa has transformed a series of photographs into PCBs, and I've seen some incredible art making use of the different textures and colours of fibreglass, metal traces, silkscreen, and solder mask that you find on the different layers of a PCB. If you want to explore these techniques more, I wrote a blog post a little while back with examples and links to some of my favourites: [hsmag.cc/3pR1NH](http://hsmag.cc/3pR1NH).

There is a whole world of PCB art out there to discover! □

**The number of people making their own blinking, bleeping, blooping, and sensing PCB art soared**

# 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](http://hsmag.cc/hello)



### OPEN ACCESS

I saw on Twitter that iFixit has released a database of repair manuals for medical equipment... then I opened HackSpace magazine, and found an interview with iFixit's founder talking about the same thing. It's amazing that on the one hand you have makers who are designing face shields and masks and making those designs open for the whole world to use, and on the other hand you've got companies who won't even share a repair manual. I know whose side I'm on.

**Deborah**

Ohio

**Ben says:** The global effort to make PPE has been incredible to see from our vantage point. We've heard more on that from our interview subject this issue, Gina Häußge, who said she felt guilty at first for not making more PPE, even while she was debugging the software that made it possible in the first place. If more companies took a leaf out of her book, we'd be in a much better place.

### BOXING

Thanks for pointing me in the direction of Ultimate Box Maker [HS31]. Since I quit smoking, I was running out of cigar boxes, so I needed a new source of project enclosures. I can't believe the makers give it away for free.

**Mike Schultz**

Maryland

**Ben says:** Ultimate Box Maker is a great find. I used it this month to make an enclosure for a camera. The best bit is that you can modify the design file – for example, to make a mounting area stronger, or to add extra holes to accept bolts.



## MATHS

Thanks for Gareth Branwyn's maker maths article, specifically the bit on calculating how much to charge. I've always struggled with this. The dilemma is that you want the work, but you don't want to work for free. I always get the fear that I'm charging too much, or not enough. I don't think that will ever go away, but at least now I know that there are other ways of working out a price, other than making a wild guess.

**Robert Hogan**

Birmingham

**Ben says:** A friend of a friend used to make cabinets, until he was so busy he had no time to make cabinets any more. He put his price up to something crazily high, so that he'd only take on work if it was really going to be worth it, thinking that nobody would bother him any more... and he's never been busier. Pricing is weird. Just remember that if you're making something unique, you should be getting paid more than if you were stamping it out in a factory.



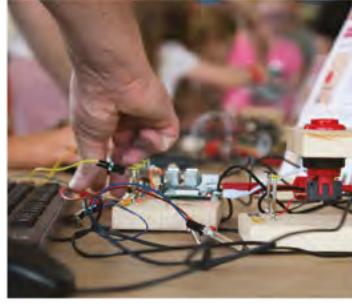
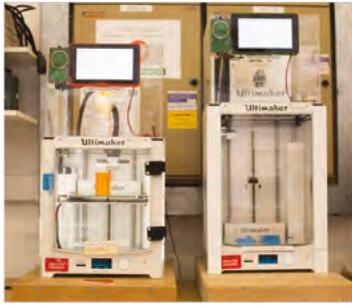
## GARDENING

Hey, how about a how-to on smart gardening? There's loads to monitor in my garden: soil pH levels, moisture, temperature, sunlight, humidity... I just need someone to push me in the right direction to save me doing all the steps that don't work. Help me out!

**James Campbell**

Peterborough

**Ben says:** It's been a weird few months, I'll give you that, culminating in my potatoes getting frost-damaged in May. We don't have any plans for a garden system at the moment, though we will be looking at setting up a camera trap next issue, to keep an eye on the creatures that visit your gardens. Hopefully we'll find more than rats and pigeons. If you're after a weather monitor, then Pimoroni's Enviro+ (now available in Feather- and Raspberry Pi-compatible form) will get you most of the way there – you just need to add a soil moisture input, and there are a few of those on the market.



# Build a Makerspace for Young People

Join our **free online training course** on **makerspace design** to get expert advice for setting up a makerspace in your school or community.

Sign up today: [rpf.io/makerspace](https://rpf.io/makerspace)



Raspberry Pi

The Raspberry Pi Foundation. UK registered charity 1129409

#RPiLearn

# LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG  
**46**

## HOW I MADE: ROBOT GUITAR



Pluck the strings of your guitar with a MIDI-controlled machine

PG  
**52**

## RASPBERRY PI CAMERA



Add components to a Raspberry Pi to make a cheap digital camera

PG  
**54**



## INTERVIEW: GINA HÄUSSGE

The creator and curator of OctoPrint on the hows, whys, and whats of 3D printing

PG  
**32**

## EXTREME BUILDS

Explore the world with homebuilt tech

PG  
**62**

## IMPROVISER'S TOOLBOX

Turn cable ties into something less useful, but more decorative

PG  
**66**

## RASPBERRY PI 8GB



The world's favourite computer gets twice the RAM and a 64-bit OS



# EXTREME BUILDS

Pushing the DIY envelope

**T**hinking out of the box comes naturally to our maker geeks, which is how they come up with ingenious builds that blow us away. On the other side of the scale of creativity are enormous and involved builds that will leave most of us perplexed by their complexity. This feature is a celebration of such magnificent hacks that are extreme either in their approach, or their build process, or are meant to be operated under extreme conditions – from underwater to outer space.

Many of the extreme projects in the next few pages take a lot of planning and require a huge number of components. Some of them are also fairly time-consuming, taking up to a year or more to come to fruition. These makers create for the love of making and don't really mind spending the time working on and, equally importantly, documenting their extreme builds.

While we encourage you to follow their lead and take their projects to your workshop, please make sure you take adequate safety precautions. Most of these projects involve the use of extremely dangerous and unforgiving tools that should never be operated without their respective mandated safety gear. →

# WATER

**20,000 leagues under the sea...give or take a few**



David Coleman hopes to one day add a grabber arm to his ROV to make good use of the unused radio control channels and an unused cable pair in the tether



The images will have to be cropped around the edges, so it's best to use the periscope with a full-frame camera

**B**y some estimates, we have only explored less than 10% of our oceans. That's mostly because, despite the abundance of water on our planet (and in our bodies), it remains one of the most inhospitable environments for humans.

But don't let that deter you from taking a look at what lurks underneath the waves.

That's exactly what Alex, of the 'I did a thing' YouTube channel, thought after he installed a frog pond in his yard. In his bid to keep an eye on his amphibian tenants, he hacked together a simple periscope ([hsmag.cc/6IZvE8](http://hsmag.cc/6IZvE8)) that only cost him around AU\$10 (about £5.50). He first epoxied a round mirror at a 45-degree angle inside a PVC elbow. One end of the elbow was then sealed with glass from a picture frame, and the other was connected to a PVC pipe using PVC cement. If you're replicating the build, make sure the PVC pipe is big enough to fit around

your camera lens. That's it – you've got yourself a periscope that lets you see and take pictures underwater. To shoot, just put your camera lens inside the PVC pipe and submerge the other end into the water. While you can't dive with this periscope, it works great for peeking inside shallow water.

If you do want to take some photos on your next dive, then you can adapt Bobby\_M's instructions to build yourself an underwater housing for your camera ([hsmag.cc/nrm9bK](http://hsmag.cc/nrm9bK)). He too uses a PVC pipe to house his camera, and suggests you stay clear of the cellular-core pipes and instead use solid-core PVC that can take the pressure. It's a good idea to take your camera with you when you go to the plumbing store to make sure it fits inside completely. Then get 10 feet of that pipe, and the other components that Bobby lists, and assemble them into a watertight container using some screws and PVC cement.



Some of the air from the propeller inflates the skirt when the engine is started. Don't worry if the engine stops midstream, since there's enough foam to keep it buoyant



Bobby suggests you put masking tape on the surfaces that aren't to be glued, to save yourself unnecessary headaches in cleaning up

We also love Fred Fourie's Raspberry Pi-powered autonomous underwater recording device, dubbed PipeCam, which we've covered previously (issue 27, page 45).

## DIVING ROBOT

If you want more manoeuvrability while exploring the depths from a safe and comfortable distance, then you need David Colemans's remotely operated underwater vehicle ([hsmag.cc/z25DQp](http://hsmag.cc/z25DQp)).

The submersible ROV has a sturdy PVC frame with six thrusters that propel it through water pretty much like a drone. It also has depth and heading hold to enable you to control it with ease. The build uses several 3D-printed components, including the propellers that are all controlled via an Arduino Mega. The ROV uses a 120-degree FPV camera that's hooked up to the base station via a 50" long Ethernet cable stuffed inside a length of polypropylene rope. The camera transmits images to a small display, together with some monitoring data to the base station that uses an Arduino Uno and an OSD shield.

## HOVER AND OUT

From underwater to slightly over it. Roland MacDonald is a retired engineer who was looking for a project he could build with his grandson. The duo decided to build a hovercraft ([hsmag.cc/LfAhNd](http://hsmag.cc/LfAhNd)) after watching one on TV, because of course that's what someone who builds kit planes and restores Cessnas would do.

After doing some research, Roland decided to purchase the plans for Universal Hovercraft's UH-10F hovercraft, along with the hardware kit that included

the all-important skirt, and the drive kit that contains the propeller. The other materials for building the hull, such as several sheets of polystyrene foam, and plywood, were sourced from a hardware store.

Roland's hovercraft is powered by an old 10 HP Briggs horizontal shaft engine that he scavenged from an old lawn-mower. The craft worked without any issues, but he would have liked to install a quieter exhaust, since the current one makes quite a racket.

## CAPTAIN NEMO

Talking of crafts, Justin Beckerman's fully functional submarine ([hsmag.cc/40NdOk](http://hsmag.cc/40NdOk)) that can safely carry one person to a depth of 30 feet surely takes the cake. Justin's submarine, which is made primarily with a large piece of drainage pipe and many other repurposed and recycled parts, took him all of six months to put together and test.

Christened the Nautilus, Justin's submarine operates pretty much like its life-size cousin. It has ballast tanks to maintain its depth and balance, as well as several emergency systems, including backup batteries and breathing apparatus. Justin says his submarine moves along at a leisurely 1.5 miles per hour, and can remain submerged for a couple of hours. →



The submarine has almost a mile of wiring that runs between the four motors and four batteries, as well as to several pumps and dials

# LAND

## Break new ground

Make sure you connect the motor in the right polarity, or it might go in reverse



**H**aving evolved on land, we've mastered the art of navigating terra firma, but some makers have taken it to the extreme. If you love skateboarding, you can use Chitlange Sahas's all-terrain electric longboard ([hsmag.cc/2TI1IN](http://hsmag.cc/2TI1IN)) to cruise around town in style. Dubbed Torque, the board can go up to 15 mph, and uses a high-power Li-ion battery that can take you up to 20 miles on a full charge.

The deck of the board is made with three stacked sheets of 5 mm-thick birch plywood. Chitlange has a good amount of detail to help you cut, sand,

and glue the boards together. You'll then need to cut the board to shape it like a longboard. You can either use Chitlange's template, or use the Adobe Illustrator file if you want to resize his design as per your requirements.

Besides the board, the project has mechanical and electronic aspects, both of which are equally involved. Designing and attaching the trucks to control the movement of the longboard will require careful planning and execution. The board has four 10-inch pneumatic wheels and big spring



suspensions to help you ride across any terrain. Then there are the electronics, which include a 250W 2750 rpm motor mounted to the board. Its speed is controlled with a small 3D-printed remote joystick that has an RF module, an Arduino Nano, and a couple of 3V Li-ion batteries. The good thing about the build is that Chitlange has described the process in great detail, which makes duplicating his efforts rather straightforward.

## E-PEDALS

If you want to go a bit faster, you can build yourself a 40 mph electric bicycle ([hsmag.cc/TLP2Me](https://hsmag.cc/TLP2Me)). While most e-bikes fall in the 15–28 mph range, Micah Toll builds something that'll allow him to keep up with traffic. The most important factor when building a fast e-bike is to choose a sturdy and well-built donor bike. Micah doesn't recommend using a cheap department store bike, because it might not be safe at high speeds. He's using a Motobecane Jubilee FS bike that retails for around \$600 (£490), although you can even use a pre-owned downhill bike as they are built to be very strong.

You'll also need adequate braking to stop at the speeds your e-bike will be doing. Micah suggests using Shimano hydraulic brakes, if your donor bike doesn't come with them. Also, make sure the bike has at least a 19" frame to be able to fit the battery. He's using a 48V 15 Ah Samsung battery that's part of a 1500 W e-bike conversion kit. In addition to the battery, the kit has all the bolt-on parts you'll need, including the hub motor, controller, throttle, brake

levers, pedal assist sensor, gear free-wheel, custom torque arms, and the cycle analyst meter. Assembling the e-bike is fairly straightforward once you have sourced all the components.

## CONVERSION FACTOR

Ben Nelson is a self-professed tinkerer who has already created himself an e-bike. So, he stepped up the game and upcycled a gasoline motorcycle into an electric one! Ben has taken a 1981 Kawasaki KZ440, and converted it to run on the powerful ETEK motor by Briggs & Stratton ([hsmag.cc/ToWDsk](https://hsmag.cc/ToWDsk)). While the donor bike and the motor are the two major components of the build, the project needs quite a lot of other bits and pieces.

The good thing about Ben's Instructable is that, while it's mostly about converting his Kawasaki, he shares lots of general guidelines and suggestions to help you adapt his plans for your own bike. His detailed suggestions on hunting for a donor bike are particularly useful. Since you'll be swapping out most of its mechanical components for electric ones, you can save quite a bit of money by hunting for a bike with a bad engine and transmission. He also shares some tips for removing the engine to make room for the electric motor and batteries.

Needless to say, you shouldn't attempt this build unless you have a fair bit of experience repairing or maintaining motorcycles. The build also involves rather a lot of metalwork as you grind, weld, and paint a fair few mountings and racks. In addition to the written instructions, Ben has documented the entire build process in a series of videos ([hsmag.cc/nsLsnE](https://hsmag.cc/nsLsnE)) that have a total runtime of over 2.5 hours. →

**Ben's bike is geared to 45 mph, and takes ten hours to a full charge that's good for anywhere between 23–32 miles**



**When building an e-bike, make sure you comply with local laws and regulations.**





The engine Tate had was from a manual transmission, but you can save yourself some hassle by finding one with an automatic transmission

# SNOWAZ

**Break the ice**

**W**hen life gives you snow, make a luge track. At least, that's what Ontario resident Steve Falk made of the proverbial phrase. Come wintertime, Steve landscapes his 80x60 foot back yard into a luge track ([hsmag.cc/MnTje3](http://hsmag.cc/MnTje3)) that's won approval from professional skiers. If you've got a large back yard and a thick cover of snow, you can use Steve's useful tips to create your own Olympic playground with little more than a shovel, a wheelbarrow, and some kids in a flying saucer sled.

## SKI SADDLE

If luging isn't your thing, perhaps you can use one of these other modes of navigating the extreme terrain. A keen skier, Instructables user melonpeel wanted himself a ski-bike, and though these things aren't a novelty, they still cost quite a lot. So, he decided to build one himself that'd be easy to fabricate and





**Although Josh had removed the suspension from the sled, he put them back on after bumping down the hill on his first run**

wouldn't break the bank ([hsmag.cc/uKUBZJ](https://hsmag.cc/uKUBZJ)). He's used an old BMX bike, but suggests you can use a mountain bike as well.

Start by removing the wheels, the axles from the wheels, along with the cranks, and chain. You'll then need to create a standing platform using a set of large flatland-style alloy pegs and an extra 3/8" BMX axle. These pegs go in the bike's original bottom bracket cups. If you're using a geared donor bike, you'll need to add spacers to account for the extra gap in the rear axle for the gears.

Be extra careful while creating the wooden platform and the hinges that go on them, which you'll then use to mount the skis. Line the skis on the struts, and make sure you take the right measurements before drilling holes. The wooden platform was at an 80-degree angle and was kept at the same height as the wheel axle just so the bike would feel natural, which is a nice suggestion.

## SNOW MUCH FUN

If you've got kids, a ski sled would make for a better ride on the white stuff. Josh Charles has created ski sleds before, but his earlier one wasn't very manoeuvrable. He recently updated the design of his Ski Sled ([hsmag.cc/AZUvoc](https://hsmag.cc/AZUvoc)), which is inspired by the classic Flexible Flyer sled. The sled is built around a central box frame, made with aluminium box tubes and also has red steel runners, a wooden deck, and a handlebar up front that can be used to steer the sled – either with the hands while sitting, or the feet when lying down.

The frame has four cross-bars, two of which have the support towers that connect to the skis. Josh says these ski supports were also the trickiest to fabricate since they are to be cut at an angle. A couple of pivots on the skis allow them to be steered via the handlebar. In terms of safety, Josh has also added a couple of stops to the skis to prevent them from flipping over. The skis are also configured in a negative camber configuration for better cornering.

## FRANKENSTANK

If you have a craving for something a little more extreme, how does a tracked, all-terrain tank ([hsmag.cc/9mBezO](https://hsmag.cc/9mBezO)) grab you? Tate's been meaning to build one for several years, and finally decided to build one almost entirely from scrap. The design is inspired by the world's fastest tracked vehicle, named Ripsaw. The project took about a year, from

sourcing the components to applying the final licks of paint.

Tate says the construction wasn't difficult as such, but the process was tedious, thanks primarily to the sheer number of parts that needed to be sourced and fabricated. Tate didn't document the build process during construction, since he wasn't planning to share it with anyone. Yet the details that he's shared post-construction are very adaptable. In fact, Tate suggests you watch his video, and swap out any of the parts for something that's more readily available to you.

Of all the components, constructing the tracks took the longest amount of time. This, when all the components for the tracks were store-bought. The 53 feet of tracks needs 106 plates, and you'll also need to weld several angled rods to keep the tracks from rolling out from underneath as you drive.

The tank is powered by a leaky 3.9 litre V6 from a 1992 Dodge Dakota that Tate got for free. He didn't even build a custom housing for it, and mounted it, along with its original frame, into the tank. →



**Steve says that building a luge track is a constant juggle between sculpting, testing, and tinkering to find the perfect balance between fun and safety**

# AIR & SPACE

## Up, up, and away

**S**pace is the ultimate extreme environment, and one we have only just begun exploring. To get a taste for gathering data from up above, several high school students built a low-altitude, remote sensing platform that carries an Arduino-based weather monitoring device up to a height of 500 feet with the help of a kite ([hsmag.cc/zRli1Y](http://hsmag.cc/zRli1Y)).

The Instructable, by their science teacher Mr Delemeester, explains the entire process in great detail. He has listed the various SparkFun components that were used to construct the weather instrument, along with details on its assembly and programming for collecting and storing the data. It weighed about 90 grams and was mounted on a device known as the AeroPod. The design of the AeroPod is patented by NASA, which licenses it for free to educational institutions. While they applied for the licence, the students reverse-engineered it, based on a video with the help of the school's industrial arts teacher.

The students worked in multiple teams and came up with four designs for the AeroPod, built using materials in their school workshop. The AeroPod that worked the best was then tethered to a 7-foot kite that required winds between 5 and 15 miles per hour to take off. The students connected the AeroPod, together with the monitoring device, to the kite when it reached a height of about 250 feet. The students let the kite rise up to 500 feet, and pulled it down after waiting for some time for the Arduino-based device to collect the weather data. →



Copenhagen Suborbitals are dedicated to putting an amateur in space. In doing this, they are designing and building a rocket from scratch.



Make sure you stop by Axel's YouTube channel to see his \$10,000 multirotor flying machine in action

## LED ZEPPELIN

If you need to go higher, you can use Aaron Price's high-altitude weather balloon ([hsmag.cc/PqoxF0](http://hsmag.cc/PqoxF0)). Inspired by a Socratic seminar about space, Aaron decided to build himself a high-altitude balloon (HAB) that can reach heights of over 80,000 feet.

There are two aspects to a HAB flight – the balloon and the data logger. It's best to buy the balloons either from speciality stores near you, or from online retailers. You'll also need to arrange helium to inflate the balloons. The non-flammable gas is popularly used for welding, and shouldn't be too hard to find.

The other important aspect is the data logger. Aaron's data logger is built around an Arduino Nano and captures altitude, temperature, wind speeds, ascent and descent rates, latitude, longitude, time, and date and stores them on a microSD card. It also uses a perfboard for increased durability. Aaron suggests you use the genuine Arduino, as many of the cheap clones will probably not work in the freezing temperatures at those heights (the coldest temperature recorded on his flight was -58°F / -50°C).

Follow his Instructable for a complete listing of the parts, along with the wiring and the code for the data logger. He has also done a nice job of analysing his flight that reached a staggering 91,087 feet. His logger is easy to replicate and can be modified as per your own requirements. For instance, Aaron suggests you can easily add extra sensors to his design without much trouble.

## FLY BY WIRE

If launching balloons doesn't catch your fancy, and you have no regard for your personal safety, Swedish engineer Axel Borg has just the thing for you. He's taken some common household items and assembled himself a functional helicopter ([hsmag.cc/0MVEBp](http://hsmag.cc/0MVEBp)).

After going through several designs, he decided the best approach to get off the ground was to use multiple rotors and design a craft that would take off and land vertically. The first version of his flying contraption relied on a two-stroke petrol engine to power the eight heavy-duty propellers. These were mounted on a tubular frame, with a seat in between for the dare-devil, who then controlled the craft using an RC controller.

The craft, though successful, had lots of drawbacks. It wasn't just hugely complex, it was powered by a flammable liquid. On top of it all, it wasn't very stable, as you can make out from the videos, and lacked the flight characteristics and features Axel was looking for in such a vehicle. It was also slow to respond to throttle inputs, and the power wasn't delivered linearly. He, in fact, wrecked it after crashing it into some trees, though he was thankfully flying it remotely at the time.

So, he set about improving the design over the next two years. This time around, he swapped the petrol-powered propellers with electric motors. The latest version of the aircraft now features four circular rotor-mounts that hold 19 propellers each, for a total of 76 propellers for the entire aircraft.

Aaron placed the data logger in a lunchbox along with a GoPro: HERO Session camera to take these stunning images





Before you blast off into space, it'll be a good idea to enlist the help of Pacific Spaceflight, which will help you train for outer space with Dr Smith's spacesuit

## SHOOT FOR THE MOON

Planning to go a little higher, into space, perhaps? Just because our intrepid makers didn't make it into the UK Space Agency, doesn't mean they can't fulfil their dreams of blasting into orbit.

The volunteers at Copenhagen Suborbitals are working to make that dream a reality and put a human being into space ([copenhagensuborbitals.com](http://copenhagensuborbitals.com)). Based in Copenhagen, the non-profit, crowdfunded project came about in 2008 and, since then, has designed and tested several rockets and prototypes of the space capsule that'll take an amateur astronaut on a suborbital ride to space. After all these years of testing, the group has finally started the construction of the Spica rocket that'll use liquid oxygen and ethanol to catapult their space capsule over 100km above Earth.

Copenhagen Suborbitals already have the record for launching the most powerful rocket by amateurs. But it'll be eclipsed by Spica, which, along with the capsule, has a combined height of 13 metres and a lift-off weight of 4000kg. The group have published extensive details about their rockets, space capsules, and other components that are critical for the mission, which make for an interesting read for budding physicists and amateur astronomers.

## SUIT UP

But you can't just waltz into space in your pyjamas. To make sure you're in proper gear to take on the hostile environment, Dr Cameron Smith, who's been dreaming about space exploration since he was a child, has designed a fully functional spacesuit ([hsmag.cc/q6zMsr](http://hsmag.cc/q6zMsr)). Dr Smith's suit has had several upgrades since his first one that was used in the capsule integration and altitude chamber tests by Copenhagen Suborbitals back in 2013.

The suits maintain appropriate pressure, deliver breathable air, and regulate the pilot's temperature, and have also gone through various tests, including skydiving stability tests and underwater leak detection.

Dr Smith's first suits were made by modifying old scuba-diving suits, though now he and his team at Pacific Spaceflight make them from scratch. The good thing about the project is that Dr Smith plans to release the design of the completed suit under an open-source licence.

You can still read several biomedical research papers on all the tests that the suits have been put through, and follow the test videos on his YouTube channel. □

# SUBSCRIBE TODAY FROM ONLY £5



## SAVE UP TO 35%

Subscribe today and get:

- ▶ **FREE delivery**  
Get it fast and for FREE
- ▶ **Exclusive offers**  
Great gifts, offers, and discounts
- ▶ **Great savings**  
Save up to 35% compared to stores

Subscribe online: [hsmag.cc/subscribe](http://hsmag.cc/subscribe)

# SUBSCRIBE TODAY

**Subscribe for 12 months**

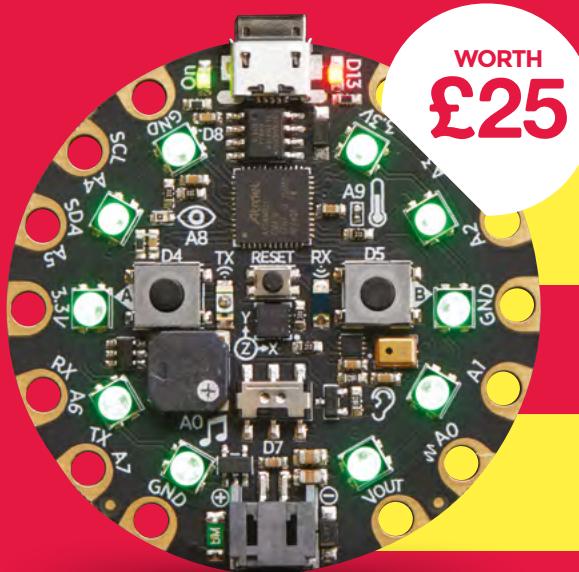
**£55** (UK)    **£90** (USA)

**£80** (EU)    **£90** (Rest of World)

Free Circuit Playground Express with 12-month upfront subscription only (no Circuit Playground Express with rolling monthly subscription)

**Rolling monthly subscription**

- ▶ **Low monthly cost** (from £5)
- ▶ **Cancel at any time**
- ▶ **Free delivery to your door**
- ▶ **Available worldwide**



# FREE!

Adafruit Circuit  
Playground Express

**With your 12-month print subscription**

This is a limited offer. Offer subject to change or withdrawal at any time.

# SUBSCRIBE on app stores

From **£2.29**



Buy now: [hsmag.cc/subscribe](http://hsmag.cc/subscribe)



# How I Made A ROBOT GUITAR

Build a guitar that plays itself!

By Step Tranovich

**always wanted to  
be a metal  
musician when I  
was growing up.**

But life is what happens while you're busy making other plans, so I got distracted by college, job, etc. I was missing the sound of angry guitars in my life, and made the commitment to myself that now would be the time to learn the guitar properly – then everything changed. Because of a rare neurological illness, I lost the use of my hands (lest you worry too much at this point, I am slowly recovering).

If I couldn't play the guitar myself, I was going to do the next best thing. I decided to build a robot to do it for me! I was even more excited to make music via a robot because it combined my love for electronics with my love for music.

Very early on, I had to decide between whether to build a robot that plucked the



strings, or one that also pressed down frets as a guitarist normally would with the fingers of their left hand. Eventually, I decided not to do the latter, for two reasons. Number one is that it makes it more versatile – other people might have wanted to play it so they can use one hand for fretting and have the robot pluck the strings.

The other reason was one of cost. Motors aren't cheap and with the money I'd saved, I could build more guitars. The nucleus of my robot metal band was formed!

Mechanically, this robot uses six solenoids on each guitar – one to pluck each string. I chose them specifically because they were the ones that I could find cheap on Alibaba, but could also find with two-day shipping on Amazon. I could get one quickly, make sure it worked, and then order the rest –



**Left ♦**  
The transparent plate is just there for protection – it's not integral to the build

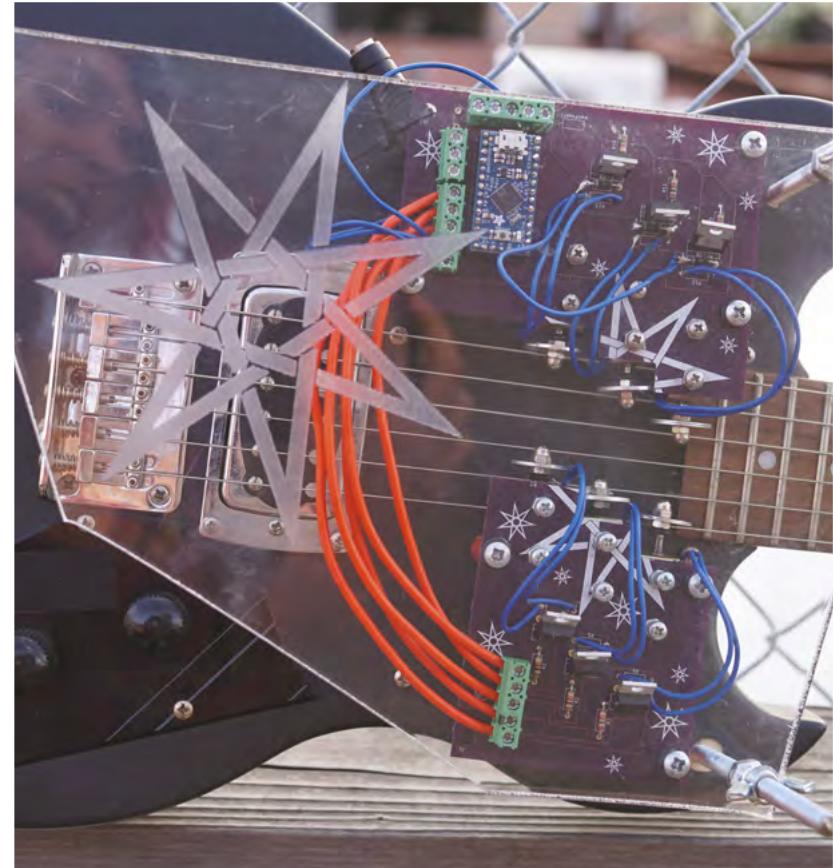
**Below ♦**  
The PCB is split in two, with three solenoids on each PCB

prototyping with the one I had while the others were shipping from China.

Each solenoid is driven by a transistor that in turn is controlled by an Adafruit ItsyBitsy (which essentially is a smaller format Arduino Leonardo). I needed to use the ItsyBitsy because I wanted to access the USB port, which you can't do with an Arduino Uno. This access to the USB port meant that I could use it as a MIDI device. You specifically need to be able to mess with how the USB port is read in order for it to be read as a native MIDI device, so that I can go into a program and simply say 'connect MIDI device' and the Arduino Leonardo pops up. It's literally this easy to plug into software as any MIDI device that you can buy off the shelf.

My favourite thing about this project is that it's a bunch of really simple pieces that are put together in a cohesive way to build something awesome. Getting a single solenoid to fire is a pretty simple electronics trick. Getting something to read MIDI is a pretty simple trick. Getting that MIDI to the solenoids is easy. The sweetest part of this project is that it does so much by really simple, thoughtful engineering.

The firmware is mostly a MIDI read library connected to a case statement – you get this note, play this string. It's →





**Below** ◊  
The PCB and electronics suspended above the body of the guitar in version 2

less than 50 lines of code. And because of that it never breaks, because it's not over-complicated.

Everything is mounted on to two PCBs, attached with a bunch of wires running between them.

The PCBs themselves are structural rather than just there to house all the components. They're the actual physical structure of the piece holding the motors on and bolting into the guitar. Normally you'd have a PCB that was a fraction of this size mounted on to a piece of wood or acrylic or something to screw everything on to.

With the first guitar, when figuring out how to mount it, I just screwed it right in. It's not suitable for your \$2000 Gibson, but this is a cheap guitar I got from Craigslist. I might even have got a discount because the person who sold it was intrigued by the idea of a robot guitar!

I got the PCB from OSH Park. I had in the design files the holes for the screws and the holes to mount the solenoids.

As I was designing it, I had a few friends who were nice enough to use a computer for me while I was designing the software. I had three different people helping at different times, none of whom had used PCB software before.

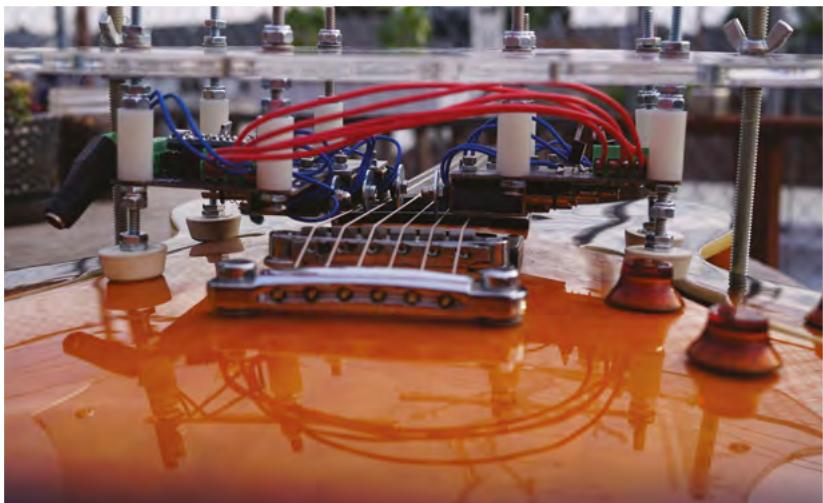
They sat there and clicked all the buttons to put the PCB design together so it could

go and get fabbed. It was literally a case of "press that button, now that button, now that button."

I have an early version of this PCB where it's still attached to the grid it was milled from; you can see all the holes and the traces, it just needs to be cut out. It came back as one solid slice of PCB. That was not what I was going to for – it turns out I used a tool that OSH Park didn't support, but they were super rad and printed me another batch. If I ever need to, apparently, I can take a Dremel to this and cut it apart, and it'll be fine, but with my hand situation, there's no way I was going to do that.

This thing can play roughly 900 beats per minute (bpm). Most songs are around 120bpm, so it's unnecessarily fast. 900bpm is faster than even the fastest heavy metal song – at least until I get out there!

We had it working, and I was pushing it to the limit. Normally when it gets too fast, the solenoids stay on because the spring doesn't have time to pull it back so it can hit again (it's a physical limitation of how fast the spring moves). But it kept running to this point where instead of staying on, the thing would just cut out. And not when it got super-high either: I had tested individual solenoids enough so that I knew that speed wasn't the issue. I had calculated the power supply of it only needing 6 amps at max because each one





Above: V2 looks messier, but it doesn't require you to drill into your guitar

takes 12V and 1 amp at fire. I had a 6 amp power supply, so figured everything should be fine, since  $1 \times 6$  is 6. Easy maths.

That was my physics minor ruining my engineering degree, because physicists like precision, whereas engineers would rather calculate it, then double it, because electronics are weird.

I wasn't sure whether it was a software issue or what, so I ended up taking the power supply of a desktop computer

is always hot. You connect to ground to complete the circuit and when the transistor says go to ground, it fires. What was happening was: when it went on, it started grounding through the guitar wire. Somehow the string itself was getting a charge, which it shouldn't have, and it went and hit the guitar wire and was grounding through the guitar and staying on.

There are five other solenoids and I wasn't getting the same problem with

## Most songs are around 120 bpm. 900 bpm is faster than even the fastest heavy metal song – at least until I get out there!

and plugged that straight in. This power supply could handle 10 amps, and then it worked great.

The weirdest problem I had was: there was a point where the second solenoid on the A string of the guitar would fire and stay on. The way the circuit works, the pin

those, so it took me a long while to figure it out. It was only happening to that one.

It took me about three months between problem and solution (including two months of procrastination). With my hand issues, taking the board off and cleaning under the solenoid was the last thing I

wanted to do. It was extremely hand-intensive, and so I wanted to try everything else before I resorted to that.

I had to tell my friends how to deconstruct this thing. We took it, cleaned it, put it back on, and it worked.

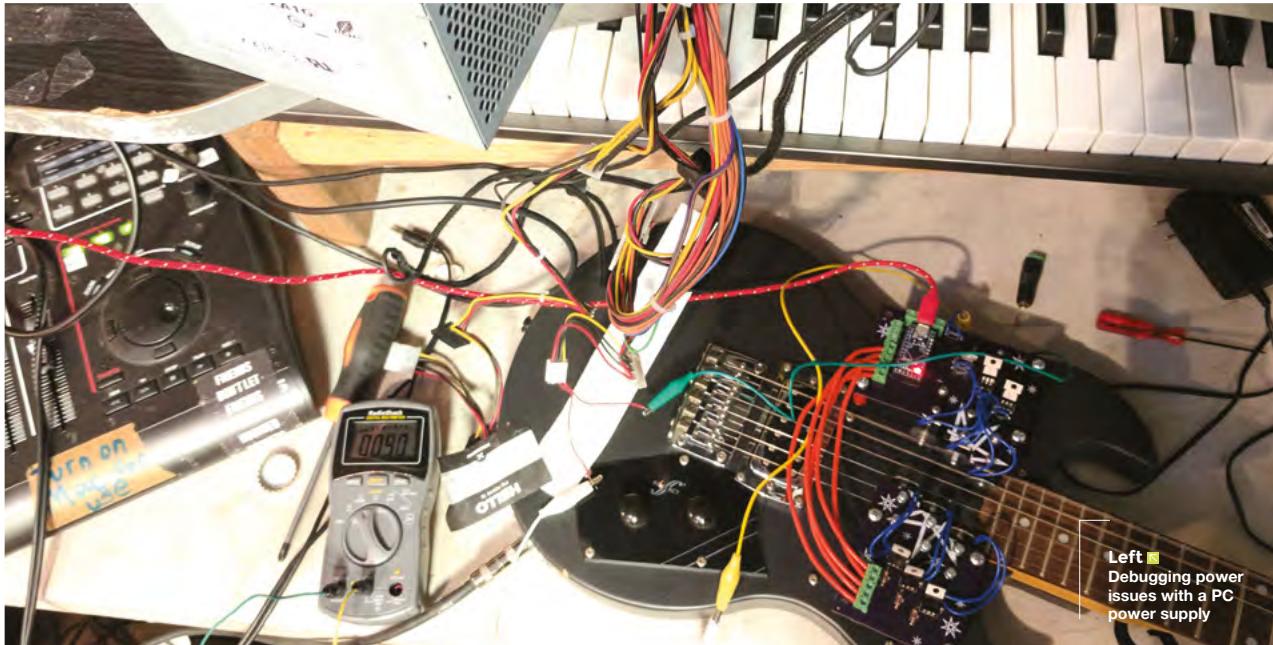
It turned out that there was just a little piece of metal fibre that got caught under →

### Tunings

The guitar is tuned to A, B, C, D#, F, F#. It's almost a Lydian scale, but with the third moved up a half-step in order to get a tritone into the scale, so you get that nice spooky Black Sabbath, *Purple Haze* interval in it. It has a minor chord in it, a major chord in it, tritones in it, so it's both emotionally versatile while also sounding properly gross for heavy metal. It's my own little sweet creation. I've never seen it used anywhere else, but that being said, people have played music forever, so I'm sure somebody has done this somewhere.



**Above** As it accepts MIDI, Step's guitar can be played by preprogrammed data or live through a keyboard



the solenoid, and somehow was connecting to the actual encasing rather than the wire.

The funniest part of that story was actually before when I was showing this to a couple of friends. I was telling them about it, looking at them as I plugged it in, and I heard them say, "UNPLUG IT." I didn't ask what the problem was; I just did it. And they were like, "Oh, this LED on the board was just flashing in a weird way, it didn't look right."

**Below** ♦  
The robot metal band is taking shape!



I looked at the board and said, "There's no LED there. What?" We plugged it back in, and there was a piece between two pins that was arcing. That's how we were able to find the little piece of scrap that was causing the short circuit.

#### NEXT STEPS INC V2

A robotic playing system that necessitates drilling holes into your priceless guitar is only going to be of limited use, so I updated the design in version two. There's a back plate and a front plate – the guitar is gripped by a silicon mat on the back plate, and feet attached to the front plate.

screws on the front plate how high the PCB is hanging off the guitar. Different guitars have different string heights and angles relative to the body, and in addition to that, the area where the neck meets the body varies on different styles of guitar. This updated attachment system means that I can use V2 on any guitar. I've even attached it to a hollow body guitar to demonstrate this – screwing into a hollow body wouldn't work, because you're screwing into air. But this guitar works beautifully, and now I can do duets with them, or a backup guitar for when I'm playing live.

## A robotic playing system that necessitates drilling holes into your priceless guitar is only going to be of limited use

Instead of being screwed into the guitar, the PCB is suspended from the front plate. The reason I did that was that one of the biggest difficulties with the first one was getting the solenoids to hit in just the right way. It was hard to adjust the board where the solenoids were, because it was hard-bolted to the thing. And so I made this suspension system so I can adjust with the

So what now? Great things, that's what! The guitars are just step one in the robot band I'm building – you can keep an eye on my progress on Twitter at **@RobotMetalBand**, and on Patreon (search for robotmetalband). I've recently uploaded my first piece of music, a guitar cover of Steve Reich's 1967 work *Piano Phase*, and I'm working on a full album. □

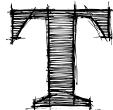
# IN THE WORKSHOP: A Raspberry Pi Camera

Turning the High Quality Camera  
into a portable camera

By Ben Everard



**Right ↗**  
The final  
camera isn't  
much to  
look at, but  
it works!



**The Raspberry Pi High Quality Camera came out last month.** You can read our full review in issue 31 but, in case you're unaware, it's a 12MP camera for Raspberry Pi boards that takes C- and CS-mount lenses. This means you can switch out the lens to one suitable to your application, as you might on a DSLR camera. While Raspberry Pi doesn't make any official lenses, there are two recommended ones available from most stockists: a general-purpose security camera lens and a telephoto lens.

After we'd put the camera through its paces, we decided to look at making a 'regular' camera – the sort you could carry around and take snaps on. There are a few bits missing from a bare Raspberry Pi and High Quality Camera setup that are needed to bring it all together:

- A case to protect the electronics while out and about, and to mount the Camera Module so it's held securely
- A way of triggering it to take pictures
- A screen for previewing the image – this is particularly important with the telephoto lens, as it needs focusing
- A power supply to keep it running when out and about



We looked at the case first. None of the off-the-shelf cases were suitable because our case needed a mount for the camera. As we reviewed the Ultimate Box Maker 3D modelling OpenSCAD script last issue, we opted to use that to make the case. It creates a four-part box, with PCB mounts that can hold your Raspberry Pi in place. You can also add holes in the front panel, and we added holes for the power lead and camera ribbon cable. This wasn't quite everything we needed though, so we exported the STL files and took them into Tinkercad, where we added a mount for the camera. This was just a square to make the case slightly thicker at that point with some 2.2mm holes (for M2.5 screws). We also added an M6 threaded bolt (by importing this STL model from Thingiverse ([hsmag.cc/DXCKq4](http://hsmag.cc/DXCKq4)) which can attach to a standard tripod screw.

I made a few mistakes with this case. The mounting holes for the PCB don't quite line up properly, as apparently I'm careless with callipers. The tripod mount was also a mistake – I should have mounted the camera lower and used the mounting bolt on the lens mount. The lens weight is significantly more than the weight of the rest of the camera, so putting the bolt where I did puts more stress on the tripod.

Neither of these deficiencies stops the camera working though, so rather than waste more plastic on another revision, I'm going with it. I fully expect to find further problems as I go along, so at some point in the future, I might bundle them all up into a new revision.

## CHEESE

Once Raspberry Pi and camera were mounted, I needed a way of taking pictures. My first thought was to have a button and TFT screen, and I started rummaging around my spare parts. I do have a few TFT screens that should work with Raspberry Pi. However, it dawned on me that I carry a far higher-resolution screen with me at all times – my phone. Why not use my phone screen to preview the image?

There's a fairly comprehensive web app for controlling Raspberry Pi cameras called RPi Cam

Control, available from [hsmag.cc/EYIGyV](http://hsmag.cc/EYIGyV). With this, I just need to connect my Raspberry Pi to the WiFi network, and I can control the camera through the web app. It's got a preview section and the ability to take photos, as well as more advanced features such as the ability to take time-lapses. It's not the most mobile-friendly interface, but if this camera proves useful, I'll be able to update the interface in the future.

There is a slight downside in that I need my phone to be connected to the same WiFi network as the camera. In the house, that's not an issue, but if I take my camera out, I'll need to set up a WiFi network on Raspberry Pi that I can connect my phone to.

I have been wondering a bit about adding something to the case to slot my phone into, so it's held in place like the screen on a regular digital camera. This is something I might look at in the future, but in the meantime, a couple of rubber bands do the job.

The battery is the final piece of the puzzle. There are roughly two options for powering a Raspberry Pi via batteries: USB battery packs, and LiPo adapters. This is one of those areas where 'use what you've got' is a good maxim. I had a spare suitable LiPo, so I ordered a Pimoroni LiPo SHIM to power my Raspberry Pi from it. There's one slight complication: while you can shut down Raspberry Pi from the web app, you can't restart it if needed. I put an in-line LiPo switch between the battery and the SHIM. Double-clicking this restarts Raspberry Pi and also makes it easy to access the LiPo battery connection to recharge it between uses, as the LiPo SHIM doesn't include charging circuitry. I taped the LiPo to the side of the case using duct tape, but if I did another version, I'd consider other ways of mounting it.

There we have it: my personal Raspberry Pi camera. Like many things I write about in this section, it's not a final product, but one step in the evolution. If it proves useful, you'll see some further iterations of it here. □

**Above** The SHIM includes the ability to shut down the Raspberry Pi if the battery gets too low

**Below** Put together a Raspberry Pi, a High Quality Camera, a case, and a battery, and you have a portable camera



HackSpace magazine meets...

# Gina Häußge

If your 3D printer works, it's because of one person...

**T**here's something enchanting about watching a 3D printer lay down hot plastic. Seeing an object take shape before your eyes is utterly compelling, which is perhaps why we love watching 3D printing time-lapse videos so much.

Despite this, it would be impractical and inefficient to sit and watch every time you sent a print job through. That's why we should all be grateful for OctoPrint. This free, open-source software monitors your 3D printer for you, keeping you from wasting plastic and ensuring that you can go about your business without fearing for your latest build.

OctoPrint is the creation of Gina Häußge. We enjoyed a socially distant chat with her about the challenges of running an open-source project, making, and what it's like to have a small project become huge. →



Above   
Gina Häußge, creator and  
maintainer of OctoPrint

## INTERVIEW

**HackSpace** Most people who have used a 3D printer will have heard of OctoPrint, but for the benefit of those who haven't, what is it?

**Gina Häußge** Somebody once called it a baby monitor for your 3D printer. I really like this description. It's pretty much a combination of a baby monitor and a remote control, because it allows you to go through any web browser on your network and monitor what your printer is currently up to, how much the current job has progressed. If you have a webcam set up, it can show you the print itself, so you can see that everything is working correctly, it's still on the bed, and all that.

It also offers a plug-in interface so that it can be expanded with various features and functionality, and people have written a ton of integrations with notification systems. And all of this runs on pretty much any system that runs Python. I have to say Python, not MicroPython, the full version. Usually Linux, and the most common use case is to run it on a Raspberry Pi, and this is also how I originally set it out to work.

Most people think it only runs on a Raspberry Pi, but no. It will run on any old laptop that you still have lying around. It's cross-platform, so you don't need to buy a Raspberry Pi if you have another machine that will fit the bill.

**HS** How long have you been working on it?

**GH** I originally sat down to write it over my Christmas break in 2012, because I had got my first 3D printer back then. It was sitting in my office producing fumes and noise for hours on end, which was annoying when trying to work, or game, or anything else.

I thought there must be a solution involving attaching one of these nifty new Raspberry Pis that had just come out. Someone must have written

something, right? I browsed around the internet, realised that the closest thing to what I was looking for treated the printer as a black box – to fire job data at it and hope that it gets it right. That was not what I wanted; I wanted this feedback channel. I wanted to see what was happening; I wanted to monitor the temperatures; I wanted to monitor the job progress.

The very first version back then was a plug-in for Cura, before Cura even supported plug-ins. After my Christmas break, I went, OK, it's doing everything I wanted it to do; back to work at my normal regular job. And then it exploded. I started getting emails, issue reports, and feature requests from all over the world. 'Can you make it also do this?' 'Hey, I have this other printer with this slightly different firmware that behaves like this; can you adapt it so that it

It's cross-platform, so you don't need to buy a Raspberry Pi if you have another machine that will fit the bill

works with this?'. 'Can you remove it from Cura, and have it so it works standalone?' Suddenly I had this huge open-source project on my hands. I didn't do any kind of promotion for it or anything like that. I just posted about it in a Google+ community, of all things, and from there it grew by word of mouth.

A year or so later, I reduced my regular job to 80%, to have one day a week for OctoPrint, but that didn't suffice either with everything that was going on. Then I had the opportunity to go full-time, sponsored by a single company who also made 3D printers, and they ran out of money in 2016. That was when I turned to crowdfunding, which has been the mode of operation ever since. Around

95% of everything that is done on OctoPrint is run by me, and I work on it full-time now. Since 2014.

A lot of the stuff that I have been adding over the years, for instance, the plug-in system itself, would not have been possible as a pet side project, not with a day job.

**HS** What are you working on at the moment?

**GH** In March just gone, I released the next big version, to make OctoPrint Python 3-compatible, because at the start of the year Python was deemed end of life, so I had to do something. The problem is that there's a flourishing plug-in ecosystem written in Python 2, so for now, I'm stuck with having to support both, and trying to motivate the plug-in maintainers to also migrate,

which is a ton of fun actually. I wrote a migration guide, tracking in the plug-in repository how many plugs are compatible. Newly registered plug-ins have to be compatible too.

**HS** Do you have any idea how many people use OctoPrint?

**GH** Nine months, a year ago, I introduced usage tracking. It's my own bundled plug-in that ships with OctoPrint that does anonymous user tracking through my own platform, so no GDPR issues should arise there. And what this shows me is that, over the course of the last seven days, I saw 66,000 instances, and the last 30 days, I saw 91,000 instances.

But that's only those who have opted into the usage tracking, which obviously is only a fraction. I have no idea about the fraction – whether the real number is five times, ten times higher, I've no way of knowing.

When I did the most recent big update, I got some statistics back from piwheels [a Python package repository]. They saw a spike in repositories that were being pulled from their index, which →



Above ♦  
If you're the sole  
maintainer of essential  
software, you deserve  
decent kit

corresponded to dependencies that the new version of OctoPrint depends on, and the spike that they saw corresponded with the day that I rolled out the new version. Based on that, it looks like there's probably ten times as many instances out there. I didn't expect that. So the total number of users could be 700,000, it could be over a million, I have no idea. But based on these piwheels stats, it's in that ballpark.

**HS** And are you seeing a growth in those figures?

**GH** Yes. Especially now, with the pandemic going on. If you had asked me three or four months ago, just when the pandemic started, I would have told you more like 60,000 per 30 days. So I saw a significant increase. I also saw a significant usage increase in the last couple of weeks.

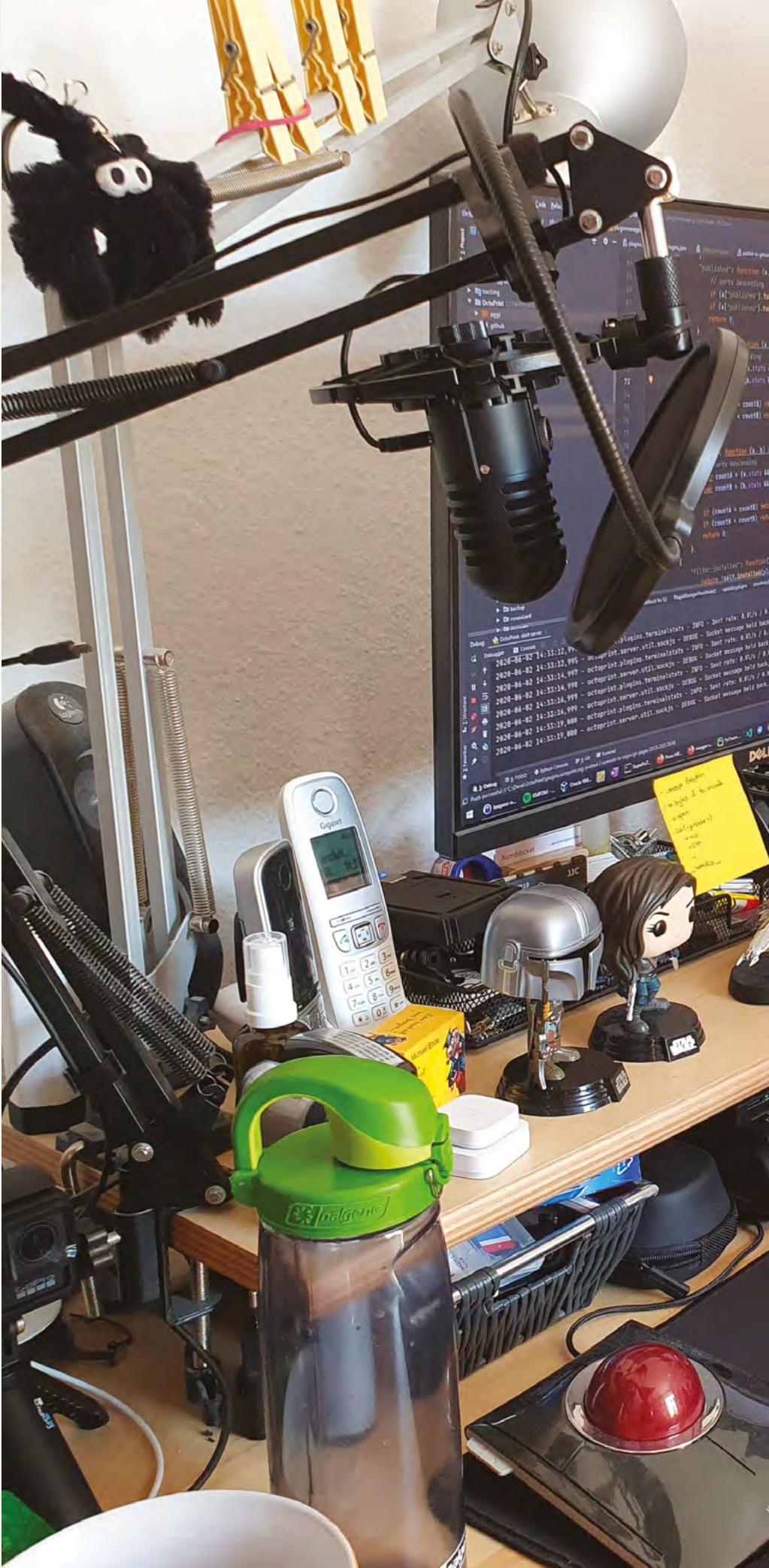
I also saw a significant increase in support overheads in the last couple of weeks, which was absolutely insane. It was like everyone and their mother wanted to know something from me, writing me emails, opening tickets and all that, and this influx of people has not stopped yet. At first I thought, well I'll just go into crunch mode and weather this out, but that didn't work out. I had to find new ways to cope in order to keep this sustainable.

**HS** You can't have crunch mode for three months.

**GH** I mean it's OK for four weeks or so, but then you start to notice side effects on your own well-being. It's not a good idea. I'm in for the long haul.

**HS** Wanting a feedback channel instead of just firing off commands that work silently makes a lot of sense.

**GH** It's not like a paper printer where you fire and forget, so treating it as a black box, where you don't get anything back on status and all that, is bound to be trouble. This is a complicated ➤





Above  
The tools of the trade – note the 3D printed dinosaur, for quality control purposes



Above ▶  
Imagine having to physically check the progress of every print job. That's what life was like before OctoPrint...

machine where a lot of stuff can go wrong, so it makes sense to have a feedback channel – at least that was my intuition back then, and evidently, a lot of people thought the same.

**HS You must have saved people countless hours and hours of wasted time, filament, and energy.**

**GH** I've also heard that I've saved at least one marriage! Someone wrote me an email a couple of years ago thanking me because the person had a new printer in their garage and was constantly monitoring it, sitting in front of it. Apparently the wife and kids were not too thrilled by this. They installed OctoPrint, and since then they've been happy again.

**HS What sort of things do you make yourself?**

**GH** Mostly functional stuff, to be honest. And the funny thing is that ever since I started working on OctoPrint, I haven't been printing as much, apart from calibration cubes to check that I've not broken anything with the most recent set of changes.

I do most of the tests against a virtual printer. But when I do print, it's usually something functional. I have some little dinosaurs and fun things around the house, but most of the stuff I made is either functional or something nerdy like a Mass Effect pistol or something like that.

**HS Do you get much feedback from users about the things they make?**

**GH** Earlier this year we created the official Discord server, and on this channel, we also have a show and tell – people post what they create with OctoPrint, or for OctoPrint; people write custom user interfaces and so forth. That's been very popular, so much so that we had to step in and create a separate channel for the NSFW parts that people print.

Yesterday someone posted a Zelda Master Sword that they printed. There's a nice healthy mixture of functional and fun stuff. Raspberry Pi cases, keycaps for keyboards.

What I really saw during the beginning of the pandemic, the middle of March when the lockdowns started all around the globe, I saw in my stats that people were doing significantly more print jobs. A lot of these, based on the feedback that I've got, were face shields. There was a ton of effort there, and I was able to see it happening, which was kind of cool. Also, it helped me get over the fact that I didn't find the time to participate myself in the PPE printing because of having to debug stuff all the time, take support questions, and all that.

There are a ton of PPE designs now, and some local hubs around here made an ultra-fast printable design. There

I realise that, and I do my best, but I often get people who come on the forums and say 'I don't understand that, can someone write a better tutorial?' I'm not the right person to do that, because I'm simply too close to the code. With regard to bug reports, those are a bit of a mixed bag. Some people understand that you need logs and you need reproduction steps and all that. And then you have people who open a ticket on GitHub and say, 'It doesn't work. Fix it.'

What doesn't work? When doesn't it work? Can I please see how it doesn't work? I try to combat that with a bot that checks to see if a template has been filled in with certain key phrases in the issue.

That has helped a bit, but it's still very frustrating when I see people who apparently have a problem, and I would like to solve their problem, but they don't give me anything to help them.

But overall, I would say that considering how many users are out there and how few there are of these cases there are in comparison, I would say yes, the community is great. They've also been supporting me financially enough that for the past four years now I've been able to do this full-time, purely crowdfunded. Which is also not something that is that common in open source.

I often go on Reddit or read Twitter mentions of OctoPrint or something, and there are so many people out there who think there is a huge team behind OctoPrint and two or three companies that fund it. No – it's more for less just me, sorry!

When people learn that (and also that I happen to be female, that boggles some minds), it helps them to moderate their expectations a bit. If you think that it's a huge company with a staff of 200 people who do nothing but code on OctoPrint, then it's easier to rant about it and demand that it does other stuff than it is to write a plug-in to add the functionality you want, or to contribute documentation. But no – it's more or less is just me. □

There are so many people out there who think there is a huge team behind OctoPrint and two or three companies that fund it

was a huge manufacturing spree, which now has been replaced, in part at least, by injection moulds which have since been created. But to cover this immediate disparity between supply and demand – that's one thing that 3D printing is good at.

**HS Do you find that the OctoPrint community is supportive with things like documentation and bug reports?**

**GH** It's tricky. The thing is that most of the docs are also written by me, and of course that's a bit of a problem because I'm so deep into the code that it's tricky to write stuff in a way that's understandable for the pure beginner.



Batten down the hatches



**Mayank Sharma**

@geekybodhi

Mayank is a Padawan maker with an irrational fear of drills. He likes to replicate electronic builds, and gets a kick out of hacking everyday objects creatively.

**A**long with duct tape, cable ties are one of the handiest things that you're likely to find in the toolbox of every maker. Like all good tools, cable ties have a pretty simple and straightforward design that makes

them very easy to fasten, but usually hard to release. In its most basic form, a cable tie is little more than a small strip of plastic that's got a row of teeth. At one end of the strip, the plastic is moulded into a point, and at the other end, a small square ratchet holds a lip which is designed to catch the teeth.

A patent for the cable tie was submitted in 1958 by Maurus C Logan who worked at the Thomas & Betts electrical company. He invented the tie for the purpose of organising electrical wiring, specifically in aircraft. Before the invention of cable ties, a nylon cord was the popular choice for grouping bundles

the teeth of the strap. Back then, the device was marketed under the name Ty-Rap, which is how it's still referred to in some parts of the world.

The popularity of the simple but highly effective tool can be gauged from the fact that several billion of them are produced around the world every year. The standard cable ties are made from a nylon polyamide, which makes them suitable for both indoor and outdoor use. Cable ties are available in various colours, sizes, and strengths. They are moulded from small nylon pellets, which are first heated until they turn to liquid. This heated liquid nylon is then injected into cavities to mould them. The manufacturers usually have several moulding machines, each with different-sized cavities to create ties of different sizes and width. If you're looking for ties for critical applications, make sure you check their tensile strength which is measured in pounds. A cable tie tensioning device is often used to apply a cable tie with a specific degree of tension. The tool will also cut off the extra tail in order to avoid a sharp edge.

While they are still most popular for bundling groups of wire, cable ties also make for reliable emergency fasteners. In fact, they are a life hacker's favourite weapon of choice for all kinds of emergency rescue and repair tasks. Thanks to their simple design and malleable nature, they have also won the approval of the maker community.

**"THEY ARE A LIFE HACKER'S FAVOURITE WEAPON OF CHOICE FOR ALL KINDS OF EMERGENCY RESCUE AND REPAIR TASKS"**

of wire. On a tour of the Boeing factory, Maurus noticed that not only was this process tedious, it was also hard on the hands of the workers. Maurus's original design had a metal ratchet head to engage

# COILING BASKETS

**W**

ith a background in manufacturing, Brian Jewett looked at making coiled baskets as a wonderful outlet for upcycling. Coiling basically involves using anything long and

flexible that's then built up and lashed together to sculpt any object, like a basket. Brian adapted the process and instead of using a string, used some PVC tubing and cable ties. The process isn't really complicated, and has remained unchanged since it was originally developed by ancient Egyptians. You'll first need something to wrap the tubing around. Brian likes using valve handles and drain covers, but you can use anything that's big and rigid enough to let you wrap the tubing around it. The wrapped tubing

**"BRIAN HAS DONE A NICE JOB OF ILLUSTRATING THE PROCEDURE, SO YOU CAN FOLLOW HIS BUILD"**

is anchored to the centrepiece with cable ties. When you get to the starting point, continue wrapping the coil over it while ensuring that the ties this time around are fastened adjacent to the first ones. Brian has done a nice job of illustrating the procedure, so you can follow his build while just glancing at the pictures. Brian notes that the advantage of using cable ties is that you can control the shape of the basket simply by regulating the angle of the ties. →

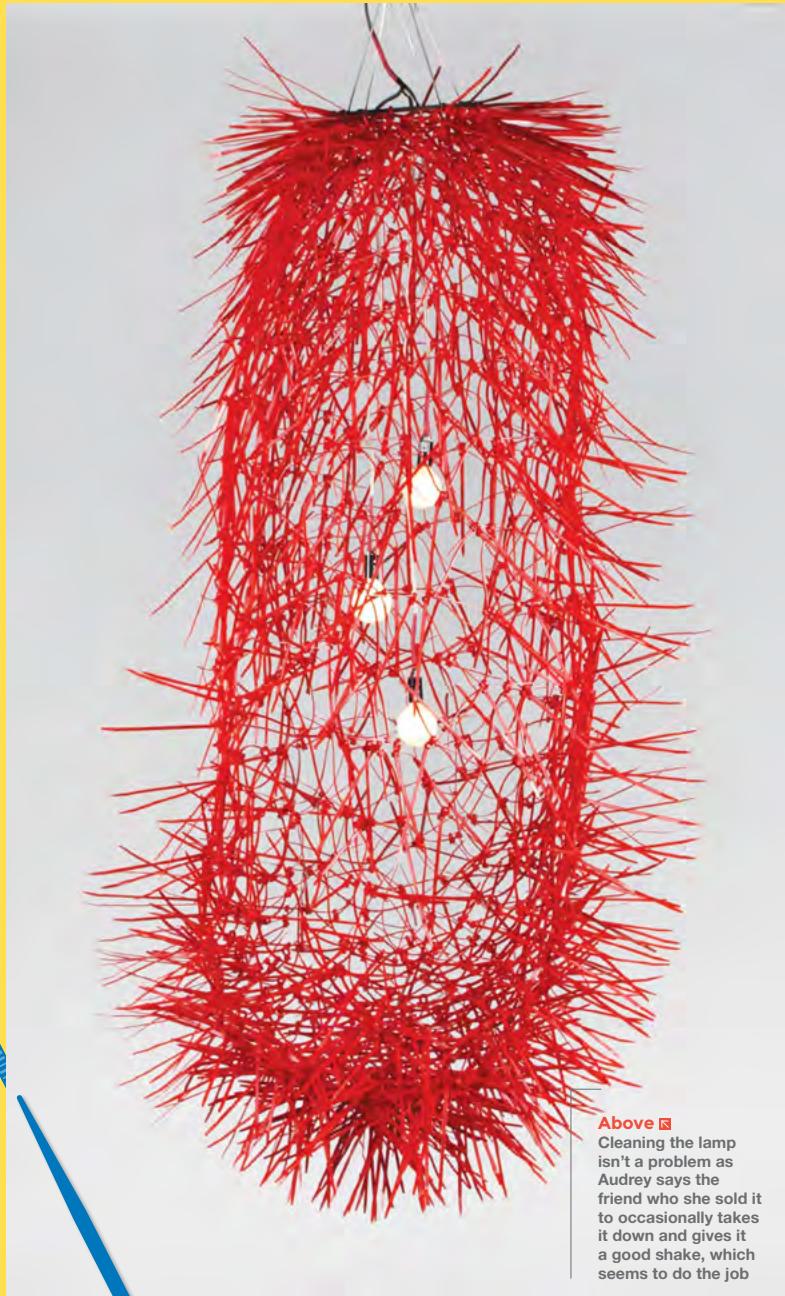


**Project Maker**  
**BRIAN JEWETT**

**Project Link**  
[hsmag.cc/FFpwYP](http://hsmag.cc/FFpwYP)

**Left ↗**  
You can make some interesting designs by changing the direction of the ties, or by trimming them

# LAMP-SHADE



**Project Maker**  
**AUDREY LEE LOVE**

**Project Link**  
[hsmag.cc/Z35PXp](http://hsmag.cc/Z35PXp)

**A**

Audrey is an artist and a former member of the Instructables content and community team. An art installation made her explore the use of cable ties as a design material, and she wrestled with about 1200 of them to create a hanging lamp-shade. The first part of the build involved creating several rings of 8, 12, and 16 cable ties. These were then grouped to create the two hemispheres that she then connected with some more cable ties to create the final lamp. Although it might sound simple, creating the sheer number of rings will take quite a while: "If I ever made an error

**"THE LIGHTS IN THE SHADE ARE MOUNTED ON A LASER-CUT PLATE THAT'S CONNECTED TO THE LAMP"**

in my pattern, I was able to gently pry the tab [of the cable tie] open to correct my mistake." However, Audrey suggests that if you do reopen a cable tie, make sure you don't use it for critical applications as its tab closure has been compromised. The lights in the shade are mounted on a laser-cut plate that's connected to the lamp with, what else, but cable ties. You can follow Audrey's lead and use candelabra bulbs, or swap them with something else. But be mindful of the heat they produce, to ensure the bulbs don't melt the cable ties.

# FRUIT BOWL

**I**

If you don't have the time or the energy to craft thousands of cable ties into a sculpture, you can start with something more manageable.

The mystery Instructable maker Craft Within Reach fabricates a unique fruit bowl in under two hours with about 150 cable ties. Describing the procedure in words would just make the process sound a lot more cumbersome than it really is. Basically, you need to arrange the cable ties into the shape of a flower. She looped 6" and 8" cable ties inside one another, arranged in the shape of a flower. She's created a total of six flowers, with one made solely with the larger cable ties which acts as the base of the fruit bowl. Again, follow her Instructable to tie the other five flowers to the base to complete the design. It could be hard keeping the ties at the right angle, and you'll need to tighten some of them as you shape the bowl.



# DODECAHEDRON BALL

**A**

Though a software engineer by profession, Dr Alejandro Erickson loves creating complex geometric shapes in real life. In this Instructable, he's used 60 small cable ties to create a dodecahedron ball. He's also painted the ties in five different colours and arranged them to show off its icosahedral symmetry. In simpler terms, you can view 60 permutations of the five colours by rotating the ball about a pair of opposite faces. The construction is thankfully simpler to follow than the mathematics it depicts. You'll need to connect three cable ties in a loop to create a triangle, several of which are then connected to form a circle. The final ball has twelve such circles, each made of five triangles, and each pair of adjacent triangles is attached at two places to form a diamond between them. Dr Erickson does a nice job of illustrating the steps, including colouring the ties to show off the mathematical symmetry. □



**Project Maker**  
**CRAFT WITHIN REACH**

**Project Link**  
[hsmag.cc/kACgnx](http://hsmag.cc/kACgnx)

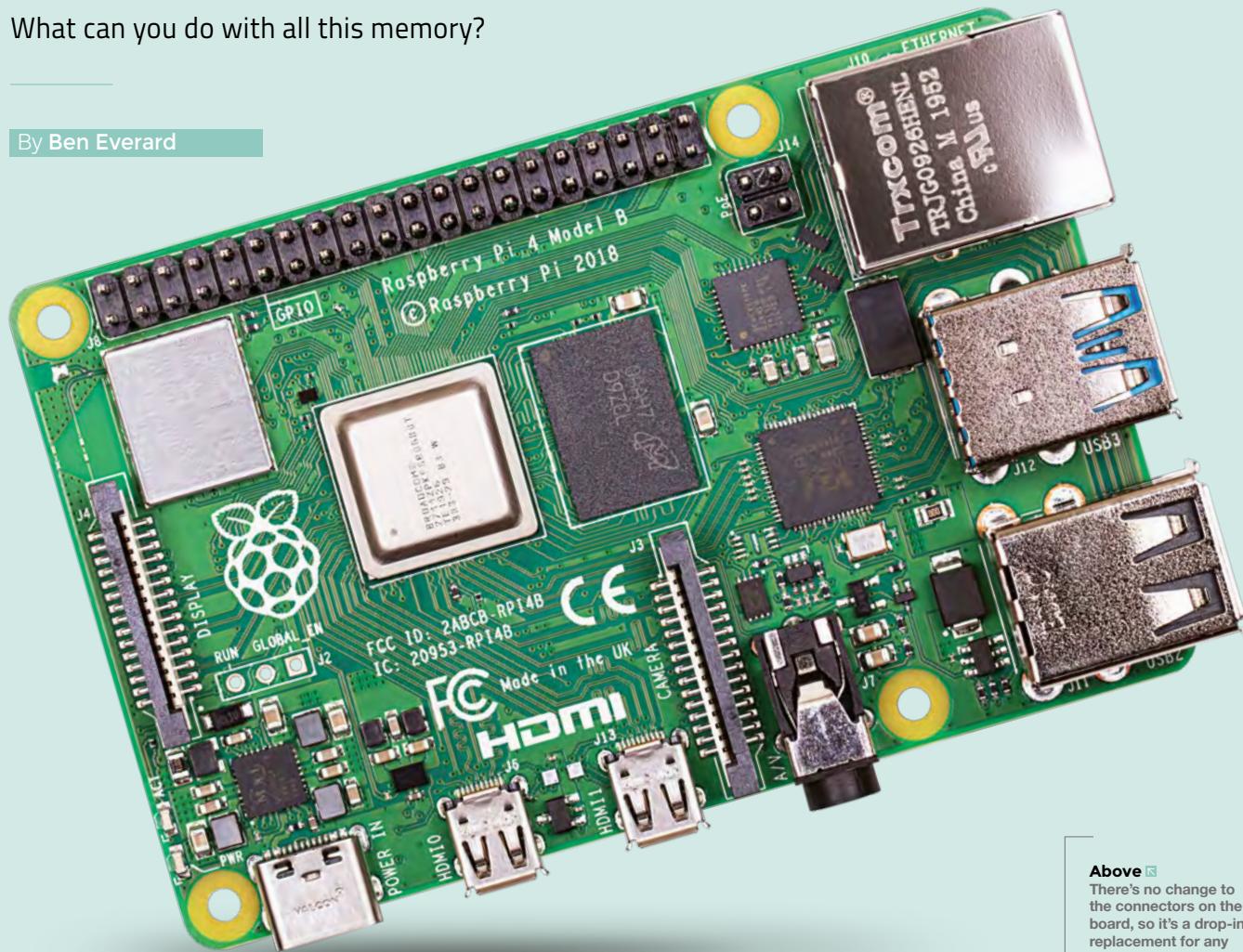
**Project Maker**  
**DR ALEJANDRO ERICKSON**

**Project Link**  
[hsmag.cc/AL7xfV](http://hsmag.cc/AL7xfV)

# Raspberry Pi 4 8GB

What can you do with all this memory?

By Ben Everard



**Above**

There's no change to the connectors on the board, so it's a drop-in replacement for any other Raspberry Pi 4

# R

**raspberry Pi has released a new version of its Model 4B board, increasing the maximum RAM from 4GB to 8GB.** This new board costs just \$75, and is already available from all the usual Raspberry Pi retailers. This is eight times more memory than was available on the top end Raspberry Pi model just a year ago.

When buying a Raspberry Pi 4 now, you have a choice of 2GB (35\$), 4GB (\$55), or 8GB (\$75). So, which should you choose?

There's no one right answer, and it depends on what you want to use the machine for. Adding more memory to a computer doesn't magically make it faster, but not having enough memory can cause problems in a number of ways:

- Some applications simply won't run – or won't be able to open large files – if you don't have enough memory
- Swapping (see 'Swapping memory' box overleaf) can slow down applications. This is most noticeable when switching applications or changing browser tabs
- Not having enough disk cache (see 'RAM disk' box) can slow down disk cache

Deciding on the amount of memory you need can be a bit of a challenge (and involve a little guess-work). There's no such problem as having too much memory, but at the same time, there's no point in buying memory you don't need.

## Deciding on the amount of memory you need can be a bit of a challenge

The biggest memory-hogging program most people use is a web browser. If you use your Raspberry Pi as a desktop, and have a number of tabs open, you'll probably be familiar with waiting for swap (see 'Swapping memory' box overleaf). Having a high-memory Raspberry Pi can help if you're a tab-addict.

Beyond this, the 8GB model is probably most relevant for specialist uses that require processing large amounts of data: machine learning, large databases, and video processing can all need significant amounts of RAM.

### OPERATING SYSTEM

Alongside this new version, there's a shiny new operating system, Raspberry Pi OS. This is the new name for what Raspberry Pi →

## RAM DISK

One way of making use of extra RAM is creating a RAM disk. This gives you storage that acts like a hard drive, but with two key differences. Firstly, it's very fast, and secondly, the contents are wiped every time the computer is rebooted. This second aspect makes it quite a specialised type of storage that's only suited to some applications. For anything that involves intermediate processing of data, it can be particularly useful. Final data that's needed, and should be kept, can then be saved to permanent storage once it's finished.

The process to create a RAM disk is quite straightforward and should work the same on any Linux-based operating system (such as Raspberry Pi OS). There's a unified file structure, so every file – regardless of what physical device it's stored on – is in some path away from the root directory, which is denoted as '/'.

For historical reasons, file systems that change rapidly are often located in `/var`, and here we'll create a directory for our RAM disk with the command:

```
sudo mkdir /var/ram
```

Information about different file systems is stored in the file `/etc/fstab`. You'll need to open this with 'sudo' permissions and edit it. You can do this with the nano text editor with:

```
sudo nano /etc/fstab
```

Be a little careful not to change any existing lines in here, as making a mistake could mean things go strange. We'll create a new line that just contains:

```
tmpfs /var/ram tmpfs nodev,nosuid,size=1G 0 0
```

This uses the `tmpfs` file system type (which is a RAM disk) to create a 1GB file system at `/var/ram`. You then need to mount the file system with:

```
sudo mount -a
```

You can now save data in `/var/ram`.

Let's take a look at just how fast this new drive is. We used the `dd` command to copy 105MB of data to files both on the RAM disk and on an SD card and here are the results:

```
>dd if=/dev/zero of=/var/ram/testfile bs=1M count=100
```

```
104857600 bytes (105 MB, 100 MiB) copied, 0.439727 s, 238 MB/s
```

```
>dd if=/dev/zero of=/home/pi/testfile bs=1M count=100
104857600 bytes (105 MB, 100 MiB) copied, 2.46455 s, 42.5 MB/s
```

The RAM disk took the data at a rate of 238MBps, while the SD managed just 42.5MBps. That's over five times faster for writing data.

### DISK CACHE

You might look at 8GB and think that if you're not using 8GB of RAM, then there's no point in having it. This isn't completely true, as there are a few ways Linux (the kernel under Raspberry Pi OS) uses memory. The most obvious is the amount each program uses. You can see this using the program `top`. If you open a terminal and enter `top`, you'll see various memory stats at the top of the page.

In the upper section of the terminal, you'll see the total amount of memory on the machine, and the amount used. The second amount is the total used by different applications (you can see which applications are using how much in the list of processes).

You'll often find that the amount allocated to applications is a lot less than the total memory, but you may also find that you have very little free memory. The reason for this is the right-most memory item: 'buff/cache'. When you have memory not currently needed by applications, Linux tries to make use of it in buffers and cache. The largest of these is the disk cache.

Reading data from disks (or SD cards) is, in computing terms, slow. RAM, on the other hand, is fast. It turns out that it's quite common for users and applications to read the same data more than once, so when you read data from the disk, this goes into the disk cache. If you need to read it again, it can grab the copy from RAM rather than going back to disk. However, if an application needs to use more memory, it can simply overwrite the disk cache without risk of losing data or a program caching, so it can also be considered 'free' memory that's available for applications.

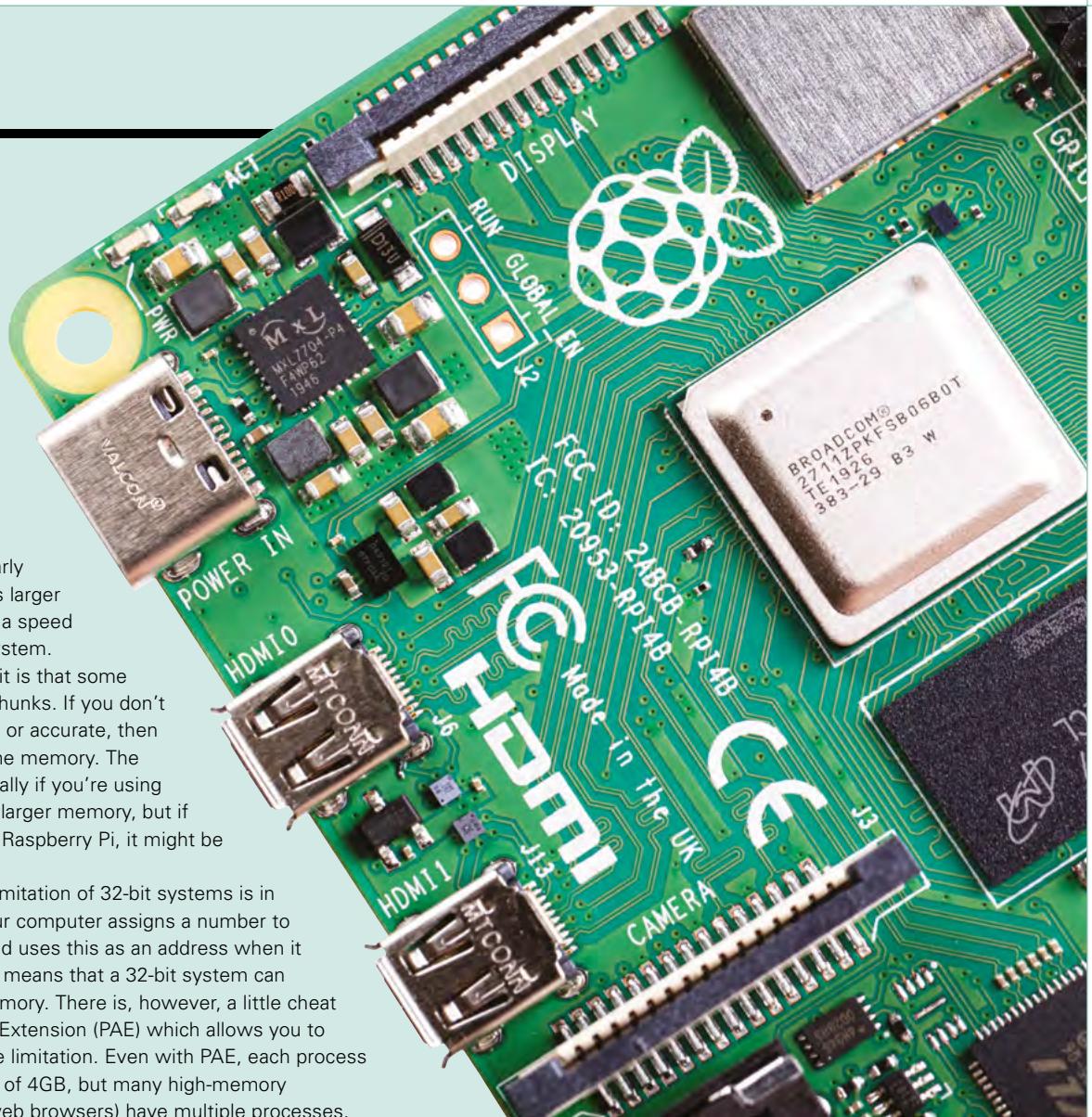


**Above** The new (black) RAM chip sits next to the main system on a chip

used to slightly inaccurately call its Raspbian image, recognising that Raspbian is an independent open-source project on which its images are based. Raspberry Pi OS comes in two flavours, 32-bit (built on top of Raspbian, as before) and a 64-bit version built on top of the Debian arm64 port. The 64-bit version is currently in public beta testing.

There's been a continuous narrative in computing for at least the past 40 years that the more 'bits' a computer has, the better. The first home computers were 8-bit machines; these were then superseded by 16-bit machines and so on, until the modern 64-bit machines.

Put simply, the number of bits is the amount of data that a computer processes at any one time. A 32-bit machine (or 32-bit operating system) lets the computer deal with a binary number 32 digits long. This number can either be a whole number (known as an integer) which in decimal will be between -2,147,483,648 and 2,147,483,647, or a number with a decimal point (known as floating point) which can be between + or  $-3.4028235 \times 10^{38}$ .



**Right** More RAM needs more power, so there's a slightly different power supply on the 8GB board

These are pretty big numbers to be working with, but if you're regularly working with data that is larger than this, you might get a speed increase with a 64-bit system.

The downside of 64-bit is that some data is stored in 64-bit chunks. If you don't need numbers this large or accurate, then you end up wasting some memory. The effect isn't huge, especially if you're using one of the variants with larger memory, but if you're on a 1GB or 2GB Raspberry Pi, it might be large enough to notice.

Perhaps the biggest limitation of 32-bit systems is in addressing memory. Your computer assigns a number to each byte of memory and uses this as an address when it needs to look it up. This means that a 32-bit system can only address 4GB of memory. There is, however, a little cheat called Physical Address Extension (PAE) which allows you to go beyond this, with one limitation. Even with PAE, each process is limited to a maximum of 4GB, but many high-memory applications (including web browsers) have multiple processes.

That's a huge amount of caveats and points to consider. In truth, for most uses, there's very little difference between 32- and 64-bit operating systems, and for now, we'd recommend most users stick with the tried and tested 32-bit version. However, if you know that your workload is sensitive to the word length, or if you have a single process needing a lot of memory, then it may be worth trying out the 64-bit version. □

#### Below

Our 8GB Raspberry Pi before it got pushed to its limits. As you can see, it's currently using just 116.4MB of RAM (with 565.8MB of buffer) and no swap space

```
pi@raspberrypi ~
pi@raspberrypi ~ % free -m
total: 116.4 used: 116.4 free: 565.8 buffers: 565.8
Swap: 1899.0 total: 1899.0 free: 1899.0 available: 1899.0
pi@raspberrypi ~ % top
Tasks: 11 total, 0 running, 10 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.1 us, 1.3 sy, 0.0 nt, 98.5 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
Mem: 116.4 total, 116.4 free, 116.4 used, 565.8 buff/cache
Swap: 1899.0 total, 1899.0 free, 0.0 used, 2473.0 avail Mem

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	SIZE	TIME+COMMAND
658	www-data	20	0	38224	2504	1816	S	3.6	0.0	151:41,59 raspimjpeg
1	root	1	-19	0	0	0	S	1.0	0.0	40:51,49 vcsnq-flop/0
26334	pi	20	0	10188	2812	2449	R	0.7	0.0	0:00,34 top
419	root	20	0	7808	2718	2510	S	0.0	0.0	0:00,00 /sbin/init
1	root	20	0	33836	5168	6500	S	0.0	0.1	0:14,92 systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00,59 kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00,00 rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00,00 rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00,00 rcu_bh
6	root	20	0	0	0	0	S	0.0	0.0	0:07,16 ksoftirqd/0
7	root	20	0	0	0	0	T	0.0	0.0	0:00,09 rcu_sched
11	root	20	0	0	0	0	I	0.0	0.0	0:00,00 rcu_bh
12	root	20	0	0	0	0	S	0.0	0.0	0:00,02 migration/0
13	root	20	0	0	0	0	S	0.0	0.0	0:00,00 migration/1
14	root	20	0	0	0	0	S	0.0	0.0	0:00,00 migration/2
15	root	20	0	0	0	0	S	0.0	0.0	0:00,06 migration/1
16	root	20	0	0	0	0	S	0.0	0.0	0:22,64 ksoftirqd/1
19	root	20	0	0	0	0	S	0.0	0.0	0:00,00 cpupdf/2
20	root	20	0	0	0	0	S	0.0	0.0	0:00,00 cpupdf/3
23	root	20	0	0	0	0	S	0.0	0.0	0:21,93 ksoftirqd/2
28	root	20	0	0	0	0	S	0.0	0.0	0:00,00 cpupdf/3
29	root	20	0	0	0	0	S	0.0	0.0	0:00,05 migration/3
30	root	20	0	0	0	0	S	0.0	0.0	0:20,24 ksoftirqd/3

## SWAPPING MEMORY

All computers have a limited amount of RAM, yet we as users often want to run a lot of programs – often far more than will fit in RAM at any one point. The solution to this is swap (similar to what Windows terms virtual memory). This is where your operating system keeps an eye on what RAM you're actually using and when you need more memory than you have, it takes a chunk of memory that you haven't used recently and moves it to disk (or SD card). This is known as 'swapping out'. When you need that memory, it reloads it from disk to memory.

All this should happen seamlessly without interaction from the user, but since disk is much slower than memory, you'll notice it happening. It's most noticeable when you switch between applications – sometimes this is really quick, but sometimes there's a noticeable delay as the operating system shuffles memory around. Older readers may remember computers with audible hard drives that would start chunking away when switching applications.

It's perfectly normal to use a little swap, and it can keep your machine running smoothly even with lots of applications running. However, if you're finding that your current Raspberry Pi is slowing you down as you're constantly waiting for memory to swap in and out, you might want to consider a model with more memory.

You can view how much swap you have and how much is currently used using the 'top' command. In a terminal window, enter `top`, and in the header, you should see data for total, free, and used swap.



**The**  
**MagPi**

**HackSpace**  
TECHNOLOGY IN YOUR HANDS

**CUSTOM PC**

# 3 ISSUES FOR £10



# FREE BOOK



**[hsmag.cc/hsbook](http://hsmag.cc/hsbook)**

Subscribe to The MagPi, HackSpace magazine, or Custom PC. Your first three issues for £10, then our great value rolling subscription afterwards. Includes a free voucher for one of five fantastic books at [store.raspberrypi.org/collections/latest-bookazines](http://store.raspberrypi.org/collections/latest-bookazines)  
UK only. Free delivery on everything.

# FORCE

HACK | MAKE | BUILD | CREATE

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

PG  
**80**



## METALFIL

Printing in shiny metal  
(well, partly metal)

PG  
**82**



## 3D MODELLING

Create printable designs using  
your camera

PG  
**88**

## 3D PHOTOGRAPHY

Add depth to your images  
by using two cameras

PG  
**72**

## SCHOOL OF MAKING

Start your journey to craftsmanship  
with these essential skills

**72** MicroPython

**76** Fibreglass

PG  
**94**

## BUILD A ROBOT

Take a cheap chassis and  
create your own rover

# Control devices from the internet using MicroPython

Control your world using MicroPython, Adafruit IO, and an ESP



**Ben Everard**

@ben\_everard

Ben loves cutting stuff, any stuff. There's no longer a shelf to store these tools on (it's now two shelves), and the door's in danger.

The **ESP series of microcontrollers are powerful and have a lot of features, but a lot of the time, we only need two features: the WiFi and a digital pin.** In this project, we'll take a look at how to set up your device so you can control this one digital pin from the internet. You can use this to drive a relay, or with a little more coding, control almost any object that you can hack. Whether you want to turn the lights on, set your sprinklers going, or anything else, you can use this technique.



**Above**   
You should be able  
to use any ESP  
device, but we used  
this LOLIN32

Before we get to the hardware and coding, though, we need to think about the data flow. The ESP device will connect to a WiFi network, and using this it can pull data from the internet. However, we want to be able to send data to it, and this is a little tricky. There are ways of setting up home internet connections so that they can receive data like this, but they can be a bit hard to set up and also can lead to security problems. Instead, we're going to use Adafruit IO. This is an online service that we can send data to, and our device can read data from. In this case, our phone (or computer) will send a message to Adafruit IO to turn the IO pin on, and our ESP device will keep checking the service until it has a new message. At this point, it can check the message and either turn the pin on or off.

The first thing we need to do is set up Adafruit IO. You can sign up for a free account at [io.adafruit.com](http://io.adafruit.com). There you need to create three things: a dashboard, a block, and a feed – you can do them in that order. On the main page, click on Actions > Create a Dashboard. On the new dashboard, click on the plus sign in

The screenshot shows a user interface for Adafruit IO. At the top, there's a navigation bar with links for Shop, Learn, Blog, Forums, Videos, Adabox, and IO. The IO link is highlighted in blue. On the right of the bar, it says "ben\_ev" with a dropdown arrow. Below the bar, there's a header with "Profile", "Feeds", "Dashboards", "Triggers", "Services", and "Adafruit IO Key". Under "Dashboards", it says "ben\_ev / Dashboards / test dash". The main area is titled "test dash" and contains a single block labeled "light" with a red "OFF" button. To the right of the dashboard are several small, colorful icons. At the bottom of the screen, there's a footer with "Help" (Get Help, Quick Guides), "Explore" (Learn, IO Plus), and a quote: "If you have knowledge, let others light their candles at it" - Margaret Fuller.

the top right, then Create a New Block. We're going to use a toggle as this works best with on/off data, so select that, then on the next page, enter a new feed name and hit create. We called ours 'onoff', but you can call it whatever you want.

That's all the setup. On the dashboard, you can now click on the new toggle, and it'll slide between on and off. You won't be able to see it, but it'll be saving a stream of data in the back end that's going on and off.

## HARDWARE

You can use any hardware that supports MicroPython and has a network connection for this, but we've used a LOLIN32 ESP32 dev board.

The first step is to get MicroPython onto the device. This differs a little depending on what hardware you

**On the dashboard, you can now click on the new toggle, and it'll slide between on and off**

have, but for ESP devices, it's all broadly similar. You will need a program called esptool – this is a Python program, so you'll first have to make sure you've got Python (2.7, 3.4, or newer) installed. We used Anaconda on Windows to do this (follow the instructions at [anaconda.com](https://anaconda.com)), but if you install Python through a different process, it should also work.

Once you've got that, you'll need to get to a terminal – this is Anaconda Prompt if you'd used Anaconda. Enter the following to install esptool:

```
pip install esptool
```

You should now have the program installed so that you can run it with the command 'esptool' (the documentation suggests that the command should be **esptool1.py**, but we found that it didn't work with the **.py** at the end).

Now you need to download and flash MicroPython to the device. You can download this from [micropython.org/download/#esp32](https://micropython.org/download/#esp32). The latest at the time of writing was v1.12.

Before we can flash the firmware, you need to connect the board to your computer via USB. This should create a serial connection that you can use. →

## HOW TO USE THIS

You can use this technique to control almost any hackable electronics. For something simple, you could just use the IO pin to drive them. By connecting this pin to a suitable relay board, you could turn almost anything on or off via the internet. This will even work with high-voltage appliances, but make sure you understand how to do this safely if you want to go down this route.

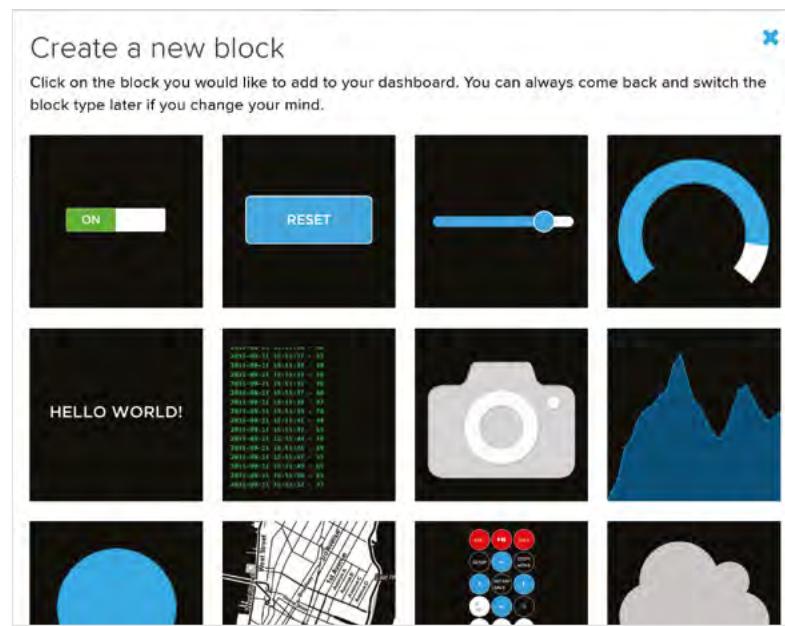
You could change the callback function to interface with some hardware in any way you like.

You also don't have to send an on/off as we did. Adafruit IO also has a slider that can send a varying number over MQTT which could, for example, be used to dim your light rather than turn it on and off.

**Left** ◆  
Our dashboard has a toggle switch that lets us flick the light on and off

# Control devices from the internet using MicroPython

## SCHOOL OF MAKING



**Above** There are loads of different types of blocks we can add to our dashboard, but we'll use a toggle

**Below** The Adafruit IO help system has examples for a range of languages

Finding the address of the serial connection can be a bit tricky. On Windows, you can use the command mode. Type it into your terminal before plugging in the board, then run it again after, and you should find that a new COM port turns up the second time. In our case, it was COM33, but yours will likely be different. In Linux, you can do a similar thing by looking at how the output of ls /dev looks (the new port should be something like /dev/ttyUSB0).

With that, we can flash our new firmware. There are two commands: one to remove old firmware, and one to flash the new firmware.

```
esptool.py --chip esp32 --port COM33 erase_flash  
esptool.py --chip esp32 --port COM33 --baud 460800  
write_flash -z 0x1000 esp32-20190125-v1.10.bin
```

A screenshot of a web browser displaying the Adafruit IO API documentation at https://io.adafruit.com/api/docs/#about-the-api-docs. The page features a large Adafruit logo on the left and a sidebar with links to "Adafruit IO HTTP API", "About the API Docs" (which is highlighted in blue), "Authentication", and "Client Libraries". The main content area is titled "About the API Docs" and contains text about the API being hosted on GitHub Pages and available on GitHub. It also includes a link to file issues and information on reporting bugs or broken links. Below this, there's a section for feedback and a link to the Adafruit Discord server.

In both cases, you'll need to change the COM33 port to what is correct for your machine, and you'll need to update the file name to the one you downloaded (you'll also need to include a path to the file if it's not in the current directory).

Pewh, that's a lot of setting up, but we're almost there now. The only thing we need is a program to send code to MicroPython on the new board. For this we'll use ampy, which you can get by typing the following into your Anaconda (or other) terminal:

```
pip install adafruit-ampy
```

Before diving into our code properly, it's a good idea to run a quick test to make sure all the setup of the device we've just done has worked properly.

Open a text file and enter the following:

```
print('Hello world! I can count to 10:')
```

```
for i in range(1,11):  
    print(i)
```

Save this as **count.py** in the same folder that your terminal is in, then run the following:

```
ampy --port COM33 put count.py  
ampy --port COM33 run count.py
```

If everything's gone to plan, you should see the numbers 1 to 10 appear on the screen.

### MQTT

Adafruit IO is a front end for an MQTT, which is a protocol for sending messages. An MQTT server has one or more topics (these correspond to feeds in Adafruit IO), and these are streams of messages. Each client can publish messages to a topic or subscribe to it to receive messages that other clients publish to it.

In our system, we want the toggle to publish messages to the topic (which it already does – Adafruit IO takes care of this for us), and we want our ESP32 to subscribe to the topic to get information from the toggle switch.

To make this a little easier, we will use the umqtt.robust library. This handles the nitty-gritty and lets us concentrate on dealing with our messages. You can download the full code from [hsmag.cc/pyL1Zp](https://hsmag.cc/pyL1Zp), but the important bits are as follows.

At the top, you'll need to give the code login details for Adafruit IO and your WiFi network. For Adafruit IO, just click on the Adafruit IO Key button and copy the values across.

First, you need to connect to the MQTT server and subscribe to the topic. This is done with:

```

random_num = int.from_bytes(os.urandom(3),
'little')
mqtt_client_id = bytes('client_'+str(random_num),
'utf-8')

client = MQTTClient(client_id=mqtt_client_id,
server=ADAFRUIT_IO_URL,
user=ADAFRUIT_USERNAME,
password=ADAFRUIT_IO_KEY,
ssl=False)

client.connect()

mqtt_feedname = bytes('{:s}/feeds/{:s}'.format(ADAFRUIT_USERNAME, ADAFRUIT_IO_FEEDNAME), 'utf-8')
client.set_callback(cb)

client.subscribe(mqtt_feedname)

while True:
    client.wait_msg()

```

You might notice the `set_callback` line here. This tells the library that we want it to run the `cb` function when it gets a message on the topic we're subscribed to. This function is:

```

def cb(topic, msg):
    global p5
    print('Received Data: Topic = {}, Msg = {}'.format(topic, msg))

```

```

if msg == b"ON": p5.value(0)
else: p5.value(1)

```

As you can see, this `callback` function has to take two parameters, one for the topic and one for the message. We can ignore the topic because we're only subscribed to one topic, but we can use the message value to know if we want the pin to turn on or off. The LOLIN32 has an LED on pin 5, but it's wired so that it lights up when the pin is low, so – rather confusingly – it'll turn on when the toggle switch is off.

There are two ways of running the code. You can either run it as we did with the count example above – this is good for testing, but it won't run the code if you restart the device. Alternatively, you can call the code file `main.py`. If you do this, it will automatically run the code each time you reboot the board, so it should run permanently. □

## SECURITY

We have built our little application with no security whatsoever. We've turned off SSL to make it easier to connect. This means that an attacker could see that we're turning something on and off, and even potentially turn the thing on or off themselves.

Whether or not this is a problem depends very much on what you are doing. For more details on how to secure this with SSL, take a look at the Adafruit IO documentation at [hsmag.cc/5PvnNh](https://hsmag.cc/5PvnNh).

## Choose feed

**Toggle:** A toggle button is useful if you have an ON or OFF type of state. You can configure what values are sent on press and release.

If you have lot of feeds, you may want to use the search field. You can also create a feed quickly below.

Group / Feed	Last value	Recorded
<input type="checkbox"/> freeHeap	99392	7 days
<input type="checkbox"/> Kingswoodten	4378.1	12 months
<input type="checkbox"/> Kingswoodtwofive	1679.9	12 months
<input checked="" type="checkbox"/> onoff	OFF	7 days
<input type="checkbox"/> Welcome Feed	78	about 2 mo...

Enter new feed name:

[Create](#)

[Previous step](#) [Next step](#)

# Moulding fibreglass with 3D printing experiments

Replicate 3D-printed parts with lighter, stronger materials



**Figure 1** A collection of small parts and 3D-printed moulds, some single-piece moulds, some multiple, and some with bolts allowing easier alignment

Jo Hinchliffe

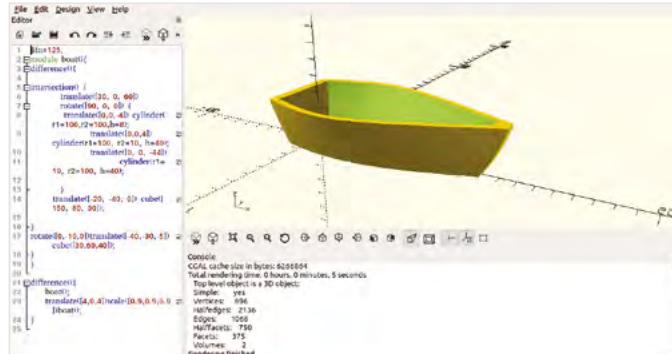
@concreted0g

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!

We fell into experimenting with using 3D printing to help create moulds for fibreglass work through rocketry when we needed a small nose cone that could potentially stand up to the forces on a small experimental rocket planning to go faster than the speed of sound. We wanted to explore these ideas further and came up with the idea of trying to make a simple, small model boat hull from fibreglass. We modelled a simple flat-bottomed boat in the script-based 3D modelling platform OpenSCAD (**Figure 2**). It's a simple design with a flat

bottom, similar to a rowing boat. We made a test print of the boat model in PLA with quite thick 4 mm walls printed with 1.8 mm wall thickness and an infill of 15%. The result is quite sturdy – it allowed us to view the model and check how it floated.

We tried different types of mould for our hull design, all aiming to create a mould with the outer dimensions of the resulting hull matching the size of the 3D-modelled version. Our main consideration for this simple one-piece mould was to not have any parts of the model undercutting itself to increase the chance of the fibreglass releasing from the mould. Other considerations included that the rearmost panel of the boat, the transom (where one might

**Figure 2**

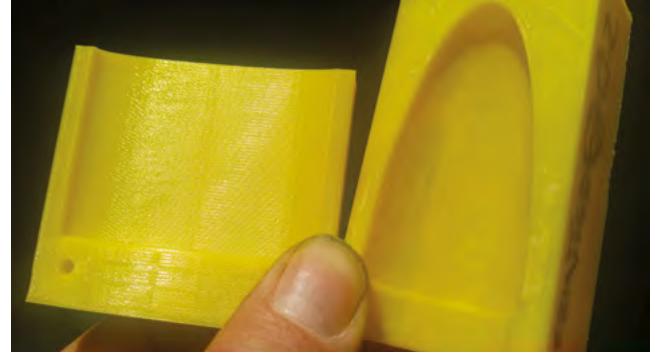
We modelled our simple boat in OpenSCAD – a free and open-source script-based CAD program

imagine a small outboard motor would go), is angled at around 10 degrees off vertical. Similarly, the sides and the front of the boat are designed with their release in mind. It's certainly possible for fibreglass to pick up extremely small details from moulds, and trying to print with as high a resolution as possible is a good starting point for a better-finished part. The orientation of layer lines in the 3D print can create very different results. In **Figure 3**, we can see that a small mould has been printed horizontally, meaning the layer lines create quite large contours within the mould section – these marks were clearly visible in objects made from this mould. We can also see another mould which has been printed vertically – dramatically improving the surface finish.

For our boat, we first printed some bulky moulds by subtracting the boat model from a large cube and then subtracting further cubes to make it taper roughly with the shape of the boat, to help reduce the print time a little. The rationale for this was that a firm, solid mould could be levered against more heavily to help try and release the model if needed. However, after some experimentation, we realised

## FIBRE CHOICES

Fibreglass materials are sold with many varieties of cloths, rovings, and matting. For our experiments, we tried different weights of cloth, all of which were considered lightweight. We have a range that is from 40 g/m<sup>2</sup> though to 195 g/m<sup>2</sup>. We found that the extremely lightweight cloth was very difficult to use – any light touch would simply drag the material into a lump. Most of the items in the images, including our boat hull, are made with single or multiple layers of the 195 gm material.



that it would be better to try a thinner and more flexible mould to allow the fibreglass to pop off and out when flexed (**Figure 4** overleaf).

We found our thinner, more flexible mould was good enough with just a light sanding inside after printing. For really high-end results, we see that many others experimenting online with these techniques may sand, fill, re-sand, and spray-paint to get the mould surface to a really high finish. In **Figure 1**, the small nose cone two-part mould has been printed at a high resolution and further sanded,

**Many others experimenting online with these techniques may sand, fill, re-sand, and spray-paint**

and this has achieved a really good surface finish on completed parts.

Depending on the geometry of your design, it may be easier to try and cut full patterns of fibreglass cloth that fold to fit the shape; or indeed it may be simpler to overlay strips of cloth on the various surfaces. For our boat hull and other experiments, we weren't too concerned with the inside of the object being uniform, so it wasn't an issue if we wanted to overlap cloth. For the boat hull, we created a rough paper pattern by cutting pieces of paper to size for each internal surface and taping them together into a pattern that could lie flat and be traced onto the fibreglass cloth.

## PLEEEEASE RELEASE ME!

Before we start mixing epoxy, we need to treat our mould with some form of release agent. A release agent is a product that will help stop the epoxy permanently sticking to the mould and help it to →

**Figure 3**

On the left, we see a small mould printed flat on the machine bed. On the right, a mould printed vertically, resulting in a much better finish and closer layer lines in the mould sections

## YOU'LL NEED

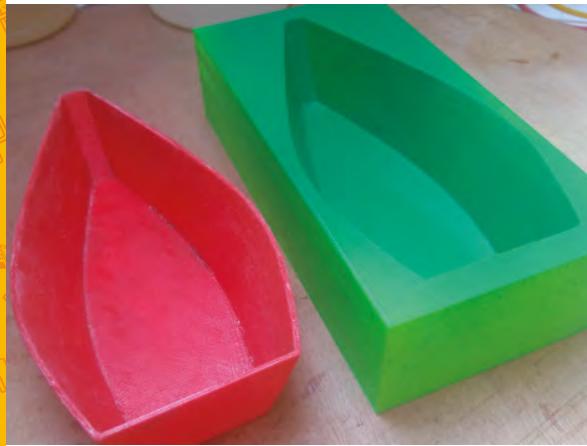
- ◆ 3D printer
- ◆ Some lightweight/medium weight fibreglass cloth
- ◆ Epoxy resin
- ◆ Shoe polish (or proper mould release wax)

## QUICK TIP

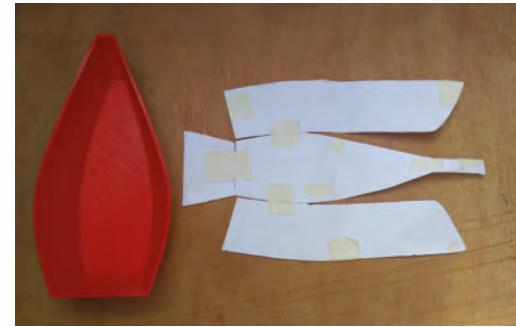
When working with resin and cutting fibreglass cloth, it's important to shield your eyes and lungs from the fine cloth filaments. Masks and eye protection should certainly be used, and we also used a workshop vacuum to clean the area and ourselves of debris after each work session.

# Moulding fibreglass with 3D printing experiments

## SCHOOL OF MAKING



**Figure 4** Our more solid mould in green, and our more flexible mould in red



**Figure 5** Cutting one-piece patterns for our boat hull experiment



**Figure 6** Using a small set of digital scales to accurately weigh out the correct ratio of resin and hardening agent

come out. Epoxy doesn't stick to PLA very well anyway, so our mould material is a good starting point; however, it's definitely not enough on its own. Proper fibreglass release agents that are sold are often hard wax, but we were keen to have a go with some less professional options we had lying around to see what worked. We'd seen online that some people use glycerine as a release agent. We had a bottle of glycerine from another project (homemade bubble mixture!), so we brushed some into a mould, and yes, it completely failed to release! Similarly, we had seen people successfully using candle wax as a release by simply rubbing a hard candle over the mould surface. While this may have worked, we found it very tricky to get good coverage, particularly into corners using this method, so it was abandoned. A popular online choice that we didn't try is car wax, but it makes sense as it's a similar consistency to what we did end up using – neutral-coloured shoe polish!

We found shoe polish to be excellent, as a little warmed between the fingers can be spread in a very fine even layer, and can even be pushed into sharp corners using a small, stiff paintbrush. Care needs to be taken not to allow any to build up in a particular corner, as that obviously changes the

mould geometry, but we found this a cheap and workable solution.

For epoxy, we are using some laminating epoxy that takes a long time to fully cure (around 24 hours). There are online retailers selling epoxy – we found a merchant that could supply smaller amounts of this resin at a reasonable price.

Our resin mixture is a 2:1 ratio of resin to hardener given as weight; as such, we used a small set of digital scales to weigh out a small amount of resin, and then half of that amount in hardener. While we laid up a few other small part experiments at the same time, we estimate we used only around 9 grams of mixed resin for the boat hull (**Figure 6**).

Previously for larger items, we have laid out our cut cloth pieces onto a piece of plastic film or greaseproof paper, and brushed epoxy onto the cloth to wet it before applying it to the mould/form. For these smaller projects, we had seen others applying a brushed layer of epoxy directly to the mould surface and then adding the dry cloth before brushing further epoxy through. Using this approach we made sure to add enough epoxy to totally 'wet out' the cloth, which essentially is the point that the cloth becomes fully soaked, and in the case of these lightweight cloths, becomes transparent (**Figure 7**).

Using a small brush to apply the epoxy, it's also possible to gently push the cloth into place and work it into quite tight corners. However, these are also potential points where air bubbles can be introduced, and it can be frustrating ensuring the cloth doesn't fold or tuck around a tight corner. In future experiments, we would try to avoid tight corners and use fillets and chamfers if possible. Having laid up our boat hull, we left it somewhere safe to cure overnight. After a few hours of curing,

### QUICK TIP

You might notice that, before adding paint, our fibreglass hull had lots of black lines in it! This is due to the ink marks we made drawing around the pattern onto the cloth!

### LAYER UP

For our boat lay up, we could add as many layers of epoxy-soaked cloth as we wanted all in the same session. We ended up using around three layers of 195 g/m<sup>2</sup> cloth on top of each other, and then added some small strips along the internal edges to stiffen the final hull. Adding all the layers at once means that we end up with a single, thicker skin that has no real discernible layers as the epoxy cures at the same rate throughout the piece – so it's as solidly bonded as its chemistry allows. Sometimes, and with some designs, we might need to let a section cure and then add another layer of fibreglass. We approached some of the tiny fibreglass nose cones this way, as it was easier to lay up each half of a nose cone, then, once cured, trim the pieces, join the moulds, and then apply another layer of epoxy and fibreglass cloth onto the inside. While we feel this would be strong enough, it certainly is going to not be as strong as we are creating a more mechanical bond between layers.



**Figure 7** A freshly 'wetted out' piece of 195g cloth in a small mould. Note the cloth becomes transparent when enough resin is applied



**Figure 8** Our boat hull demoulded, and ready to be trimmed and sanded to final size

the fibreglass goes through a 'leather' stage, where the epoxy is much less sticky but not fully cured – some people advocate this is a good time to trim off any excess cloth; however, as our hull was quite thin, we left this until we felt it had cured more fully.

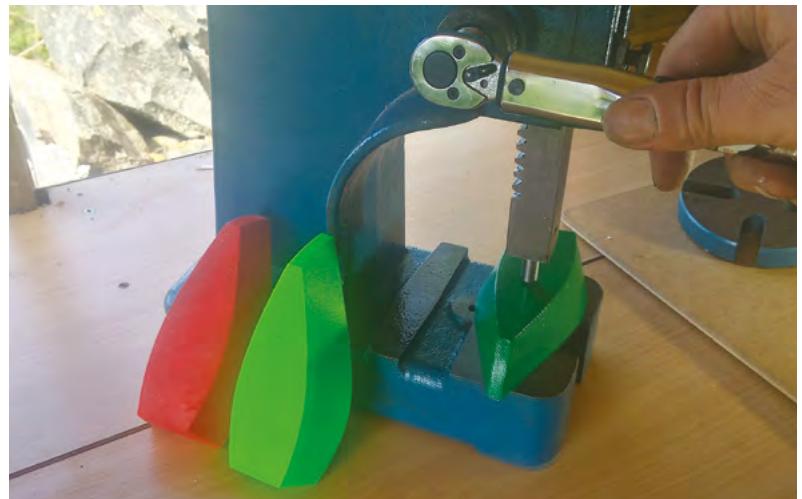
After around 15 hours, we demoulded our boat hull (**Figure 8**). Flexing the mould allowed the fibreglass to 'pop' out of the mould slightly in places, and then we used a thin, small scraper tool to ease into the gaps and gently move the fibreglass and remove it fully. Having demoulded, we inspected the outer surface – it was an OK finish but had indeed picked up every tiny detail of the mould. While it was good enough for our experiments, it's fair to say that more time invested in finishing the mould surface is time well spent in terms of the surface finish of the object.

We could tell that our boat wasn't fully cured after 15 hours – it continued to harden for around another day. Having cleaned up the moulded hull a little, we gave it a quick coat of primer and a lick of green paint (**Figure 9**).

It's certainly watertight and, comparatively, the fibreglass version is lighter than the 3D-printed version which had the thicker hull walls – the fibreglass hull weighed 21 grams, while the printed hull weighed 30 grams. In issue 24, we explored a modification to our arbor press that allows us to



**Figure 9** The fibreglass hull trimmed and with a quick coat of paint



repeatedly apply a similar force to an object. We used that to make some crude strength comparisons between the lighter fibreglass hull, the thicker 3D-printed hull, and the red mould hull we printed using just a solid thick perimeter.

The results revealed that all of the hulls exhibited some deflection and flexibility, but to a lesser extent on our fibreglass model. We could intuitively tell that the thicker 3D-printed hull would reach its elastic limit quicker than the fibreglass, and it would crack or lose layer adhesion. The single perimeter wall printed mould was similar, but had more flex than either the fibreglass or the thicker printed hull – and again, we could tell that it was closer to cracking, while the fibreglass hull was still flexing and returning to shape. We can imagine building a hull this way – with some internally added fibreglass/G10 bulkheads added afterwards it could make a very tough item indeed. □

**Figure 10** Using a modified arbor press and a torque wrench to do some comparative force testing

**QUICK TIP**  
We didn't cover this in this article, but another avenue to explore is vacuum moulding to pull composites securely onto moulds. Plus, there are other variants, such as inflating balloons inside moulds to push composite lay-ups into place.

# Printing with metal

Getting a natural look on your 3D prints



**Ben Everard**

@ben\_everard

Ben loves cutting stuff, any stuff. There's no longer a shelf to store these tools on (it's now two shelves), and the door's in danger.



# W

ood-filled filament (which we looked at in issue 29) is the most popular type of filled PLA, but it's not the only option to give your prints a less plasticky look.

This issue, we're going to take a look at a couple of other options: metalfil and stonefil. These are, as you might expect, PLA with added metal or stone. These additives can lead to really good-looking prints with some of the visual appeal of another material.

For this article, we tested bronze filament and marble filament. Both can create great-looking prints, but neither is the same as the original metal.

Bronze filament retains the colour and lustre of the original metal, but doesn't have the same shine that you can get from pure metal.

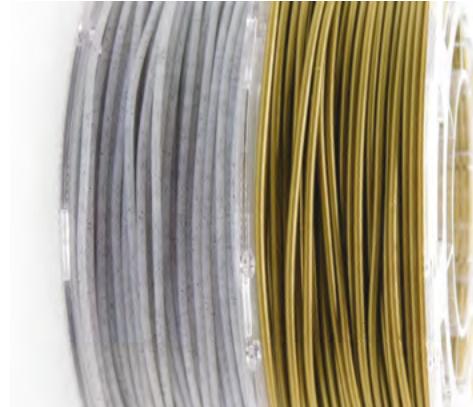
The end effect from marble filament isn't really much like marble at all. It doesn't have the sweeping patterns caused by mineral impurities. Instead, it has dots of black in a slightly off-white, translucent plastic. Dots at different depths show up as different shades of grey, which does give a pleasant natural stone effect.

## PRINTING PROBLEMS

While these filaments are mostly PLA (with a few bits added in for effect), they do print a little differently from pure plastic.

### Left

The marble filament gives a certain fossil-ness to this dinosaur



The particles in them are much harder than plastics, so you need to use a wear-resistant nozzle. The most common of these nozzles are the hardened steel nozzles, but you can also get more exotic options, such as those made from ruby.

You'll need to adjust your slicer settings. While these filaments are mostly made from PLA, they do need to be printed a little differently. There are four things that you might need to adjust:

**While these filaments are mostly PLA, they do print a little differently from pure plastic**

- **Temperature** You'll probably find that you need to increase the temperature by 5/10 degrees (but no hotter than 220°C), compared to regular PLA.
- **Flow-rate (or extrusion multiplier)** Increasing this pushes more filament out of the nozzle. Going to 1.1 or even higher can improve prints.
- **Fan** Turning down the fan helps layers stick together better.
- **Print speed** As more delicate than regular PLA, you may find that you have to slow the speed.

You should also be aware that most filaments are sold by weight, and filled filaments are typically heavier than regular PLA, so you'll get less volume for a given weight. □



## UNCLOGGING

These materials are quite prone to clogging the printer nozzle. Most printers come with a 0.4mm nozzle, and this should work with most filled filaments, but a 0.5 mm or larger nozzle will be less likely to block (you will need to adjust your printer settings accordingly).

If you do block your nozzle, the first thing you can try to unblock it is a cold pull. Check your printer's documentation for how to do this, as it can be a little different depending on how the extruder is set up, but basically, the approach is to use filament to stick to any clogs and pull them out of the top of the extruder.

You'll need some non-filled filament, and you insert it into your extruder as normal. The filament will melt around any small blockages in the nozzle. Once it's in, you then drop the hot-end temperature down to about 120 degrees. At this point, the filament will still be pliable, but will be fairly solid, and any particles blocking the nozzle should be stuck to the filament. You can then pull the filament out of the top of the extruder.

You may need to repeat this procedure a few times to get all the clogs out. White filament is particularly good for this, as this will show up any bits of clogs, so you can repeat the procedure until there are no discoloured bits on the filament, post cold pull.

**Above** □  
Small rolls let you get a feel for the filament but aren't too expensive

**Left** □  
The dark shine of bronze can give your makes a regal glow

**Left** ♦  
MetalFil is a bit brittle (even more so than regular PLA), but we tested it with a bag clip, and so far, it's been working fine

# Photogrammetry with your phone

Use any camera as a 3D scanner using open-source software



Glenn Horan

• @BatGlenn13

Glenn is a software developer by day and a 3D printing enthusiast by night. Rumour has it that building and troubleshooting cheap Prusa clones is what caused him to go bald before his years.

## D

ecent 3D scanners typically cost anywhere from hundreds of pounds up to tens of thousands of pounds for professional units. They can produce accurate digital representations of real-world objects, but they're not the only way of doing this. In this tutorial, we'll walk you through how to get 3D models from physical objects using something most of us carry around with us every day – our phones. Theoretically, the scale of the models can be anything from a small object, such as a battery or USB pen, up to the size of large buildings, assuming you can get far enough away to get a decent selection of photographs (and have a powerful enough PC). We're going to run you through the entire workflow from photos to 3D print using a statue as an example. We'll be using Meshroom, which is fully open-source, and is set up out of the box for use by beginners to photogrammetry.

## MESHROOM

Meshroom is open-source 3D reconstruction software based on an application called AliceVision. AliceVision had its inception in 2010 as part of a collaborative project between several European universities and industries. At its core, it uses a pipeline of tasks to convert a number of 2D images into useful 3D data (more on the pipeline later). There's other photogrammetry software out there, some with lower spec PC requirements. However, we chose Meshroom because it's 100% free. There are no restrictions on the number of photos per project, for example, and has an interface designed for beginners to pick up easily while having the complexity available for those who wish to dig a little deeper into photogrammetry. You can learn more about the project here: [alicevision.org/#about](http://alicevision.org/#about).

Firstly, we need to install Meshroom from the official website [alicevision.org/#meshroom](http://alicevision.org/#meshroom). You'll find links for Windows and Linux installations, along with some extra instructions for certain distributions of Linux. The official Meshroom website states that you will need a PC with an Nvidia GPU with a 'compute capability' of more than 2.0 (for reference, GPUs from up to ten years ago meet this requirement, but you can check your compute capability score here [hsmag.cc/UhRHXY](http://hsmag.cc/UhRHXY)) and a chunky 32GB of RAM. We got by just fine with 16GB of RAM, and some steps can be skipped if you don't have an Nvidia GPU. However, the quality of your 3D model will not be as good. You will also need to install Microsoft Visual C++ Redistributable package from this link: [hsmag.cc/wDCYak](http://hsmag.cc/wDCYak) (see **Figure 1**). This file contains dependencies required for Meshroom (and other programs written in Visual Studio) to run. To open Meshroom, unzip the file you downloaded from their site and click on the **Meshroom.exe** file. Once it has booted up, you should be greeted with their pleasantly minimal UI. However, if you have any issues, check out the troubleshooting page on their GitHub site.

Before you get out there and start taking hundreds of photos of inanimate objects, we recommend taking our chosen photogrammetry software, Meshroom, for a spin with their sample

### The latest supported Visual C++ downloads

[Apollo for Visual Studio 2019](#) [Visual Studio 2019](#) [Visual Studio 2015](#) [More](#)

#### Notice

Some of the downloads that are mentioned in this article are currently available on [MyVisualStudio.com](#). Make sure to log in by using a Visual Studio Subscriber account so that you can access the download links.

If you are asked for credentials, use your existing Visual Studio subscriber account or create a free account by selecting "Create a new Microsoft account".

#### Summary

This article lists the download links for the latest versions of Microsoft Visual C++.

#### Visual Studio 2015, 2017 and 2019

Downloads the Microsoft Visual C++ Redistributable for Visual Studio 2015, 2017 and 2019. The following updates are the latest supported Visual C++ redistributable packages for Visual Studio 2015, 2017 and 2019. Included is a baseline version of the Universal C Runtime (see MSDN for details):

- [x86\\_vc\\_redist.x64.exe](#)
- [x64\\_vc\\_redist.x64.msi](#)
- [ARM64\\_vc\\_redist.arm64.exe](#)

dataset to ensure that your PC can cope with the demands of this resource-intensive process. You'll also get a feel for how long the process takes and get exposed to the quality of photos you'll need to use for your project. The dataset can be found here [hsmag.cc/PL0TgK](https://hsmag.cc/PL0TgK). If you're unfamiliar with GitHub, all you need to do is click on the green 'Clone or download' button and then 'Download ZIP'. Unzip the file, open the full folder, then select and drag all of the photos into the left sidebar in Meshroom. Next, click on File then Save As to save your project.



Once it has booted up,  
you should be greeted  
with their pleasantly  
minimal UI

We recommend creating a new folder to save your project, as Meshroom will create some folders at this location each step of the way – this is where you'll find your finished 3D model file once everything is completed.

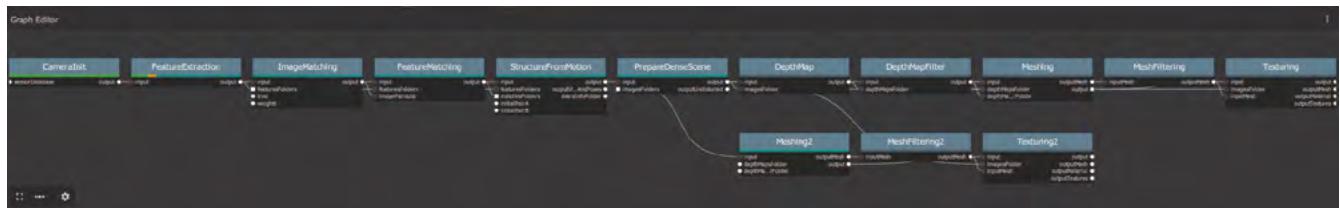
### NO NVIDIA? NO PROBLEM!

At this stage, we can just click Start and let everything run for a couple of hours, then come back to our 3D model of an anthropomorphised tree. ➤

**Figure 1** It's not immediately obvious where you need to click to download with Microsoft Visual C++ Redistributable

## YOU'LL NEED

- ◆ Camera
- ◆ PC with decent specs (16GB of RAM and Nvidia GPU recommended)
- ◆ 3D printer



**Figure 2** Clicking on a node will bring up its associated parameters, and give you the option to view the log of the work being completed

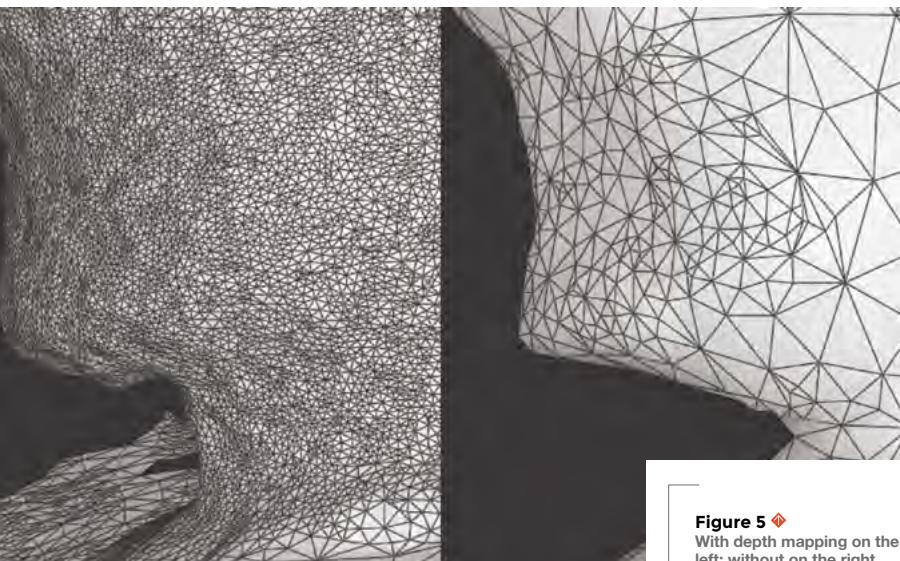
## TUTORIAL

**Figure 3**

Our sample tree model with all layers showing. You can click on the individual cameras surrounding the model to have that image show up in the image viewer pane

**Figure 4**

So far in our experiments, statues seem to work best – photogrammetry works well with large, textured subjects



**Figure 5**

With depth mapping on the left; without on the right

## PHOTOGRAPHY 101

The choice of camera you use will, of course, affect the quality of your scan, but while a market-leading digital SLR is better, modern smartphones are capable of producing images that are more than good enough quality for photogrammetry. As a general rule, try to keep everything in focus (the bokeh effect looks great on social media, but not so much for photogrammetry) and make sure nothing in your model is overexposed, or those areas will certainly end up as gaps in your final model. Contrary to other 3D scanning techniques, photogrammetry works best with a feature-filled background, as this allows the software to pick up frames of reference when trying to find out where different features exist in 3D space. Don't have too many moving subjects in the background, though the odd car moving by shouldn't be too detrimental. Lighting, as with any photography, is very important – if taking photos of a subject outdoors, then cloudy days work best as there won't be any solid shadows to throw off the software. Walk in circles taking plenty of overlapping photos (more is better here, anything from 30 to 300+, depending on the target) and try to get different heights, angles, and distances from the subject.

However, we're going to use a feature from the documentation that will give us a rough model in around 20–30 minutes, rather than several hours. This is our recommended workflow when working with your own photos, as it saves several hours of waiting to find out whether or not your photos were successful. To do this, we need to dig in a little to the beating heart of Meshroom: the graph editor. This is a visualisation of each of the steps that are taken during the photogrammetry process and is updated in real time to show which steps have completed, which are in progress and, if applicable, which steps have failed by displaying a green, orange, or red bar respectively. A detailed description of each of these steps isn't necessary for this tutorial, but if you're interested, you can find a great overview here: [alicevision.org/#photogrammetry](http://alicevision.org/#photogrammetry).

Right-click on the DepthMap node, and click on the fast-forward symbol beside Duplicate Node. This will copy the entire pipeline from this point onwards and give us a draft quality pipeline to work in. Depth mapping is the step in the 3D modelling process that an Nvidia GPU is required for and is easily the most resource- and time-intensive part, so we're going to remove it from our draft workflow. Right-click on DepthMap2, then select remove node, and do the same with 'DepthMapFilter2'. Next, click and drag from the input circle (not output) on the PrepareDenseScene node to the input circle on the

Meshing2 node, and finally connect the output circle of PrepareDenseScene to the imagesFolder circle in the Texturing2 node. Your pipeline should be the same as shown in **Figure 2** overleaf. Now we're ready to get started, right-click on the Texturing2 node and click on Compute to get the ball rolling.

When everything has finished, you can double-click on certain nodes to see their results. Double-click on StructureFromMotion, Meshing2, MeshFiltering2, and finally Texturing2 to see them in the 3D viewer (**Figure 3**). If you have an Nvidia GPU, you can now proceed to testing the depth mapping stages. Right-click and select Compute on the Texturing node, and leave things to run for an hour or two. If everything has worked up until this point, then you're ready to get started with your own photos – we used a piece of ominous modern art that was within range of this author's daily exercise during lockdown (**Figure 4**).

### GIGO – GARBAGE IN, GARBAGE OUT

Once you have all your photos uploaded to your computer, just follow the same process we used for our tree example. After the feature extraction step, you can quickly review the images on the left-hand side of your screen and check if there are any photos the software wasn't able to use (they will have a red symbol in the top right of the thumbnail). You can also click on the three dots in the bottom right of the image viewer to check how many 'features' were picked out from any of your photos to help narrow down the types of images that work best. The Meshroom documentation suggests that if you have issues, you can increase the number of features and feature types that are extracted. However, we found that this massively increases the amount of time the workflow takes, and if you haven't got a decent low-resolution mesh already, then it won't help massively. If you have a complete scan at the end of the draft workflow and you have an Nvidia GPU, we highly recommend running the depth mapping steps. Check out the difference in resolution in **Figure 5**.

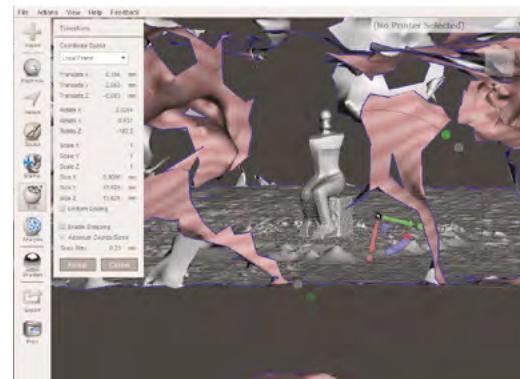
### SCULPT FICTION

So you're looking at a great model in the Meshroom 3D viewer, now what? Before we can 3D-print our new model, we'll need to clean it up using mesh editing/sculpting software. We're going to use Autodesk Meshmixer (download it for free at [meshmixer.com](http://meshmixer.com)), but if you've used similar software before, feel free to use whatever you're most comfortable with.

Open Meshmixer, click import, then navigate to the folder where you saved your Meshroom project. In the MeshroomCache folder, you'll find folders for each of the nodes in our pipeline. You'll find OBJ files in each of the steps that output a 3D model, but we'll want to lift the model from the MeshFiltering folder. There may be multiple folders in here named with a long ID number – one for each time you've run the MeshFiltering step. In each of these folders, you'll find an OBJ mesh file, which is what we want to import. You can also use the OBJ file from the Texturing folder, but we found the untextured model is easier to work with at this stage. The model is going to import in a fairly random orientation, so the first thing we need to do is transform it into an upright position. Click on Edit in the Meshmixer toolbar on the left of the UI, followed by Transform to overlay the transform tool wheel →

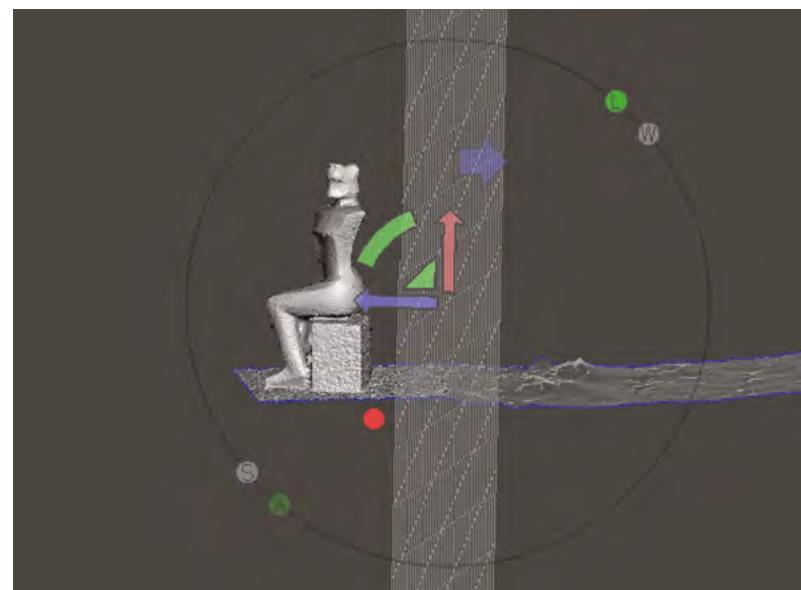
### QUICK TIP

If you have a good digital SLR, you can better fine-tune the settings for your input photos.



**Figure 6**

When transforming the model, align it as if the grid was your build plate to make your life easier when it comes to slicing the model



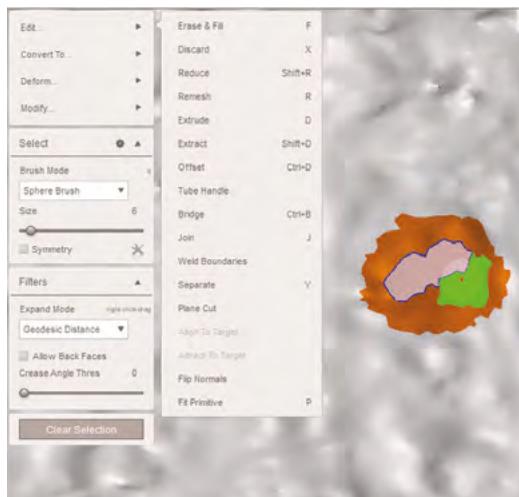
**Figure 7**

When plane cutting, you can click the thicker blue arrow above the tool wheel to toggle which side of the plane you want to keep and which will be discarded

### PICK YOUR SUBJECT WELL

With the right subjects, photogrammetry is incredibly impressive. The detail that can be achieved with just a phone, especially when using depth mapping is incredible. However, we had many frustrating hours trying to get models of human subjects to work (something the author's patient wife can attest to). From what we can tell, there were two issues. Firstly, the subject has to stay prohibitively still for the duration of the time that they're being photographed. This is much harder than it first seems, and you won't realise they've moved until you've uploaded the photos and gone through the whole workflow. Secondly, human faces just aren't textured enough to be adequately picked up by the software (even those among us with beards). This is why dedicated photogrammetry rigs exist that take photos from multiple angles, all at once, using multiple cameras. Some even project a pattern onto the subject's face so that the software can better pick up all of their features. In the future, we want to experiment with face paint, or even glitter, to add a texture to faces to see if that works any better (and if not, we're sure it'll make for some funny photos).

**Figure 8** You can also use the brush to select and delete areas around the base of your model that were hard to delete using the plane cut



At this stage, it's important not to brush away any of the details of your model that we worked so hard to extract

over our 3D model. Adjust the position of the model using the green, blue, and red curved lines by clicking on them and dragging. Then use the red, blue, and green arrows to move the model up and down relative to the grid (**Figure 6**). When you're happy with the positioning, click Accept at the bottom of the Transform menu.

#### QUICK TIP

In Meshmixer hold the **ALT** key then click and drag with your mouse to move your viewing angle around, or click your mouse wheel, then drag to pan around your model.

#### ALL ABOUT THE BASE

Next, we're going to cut around our statue to get rid of all the surrounding objects and give it a nice flat base. Click on Edit again followed by Plane Cut. You'll notice a second grid has appeared in our viewing pane, along with a tool wheel similar to the one we saw in the Transform step. Use the tool wheel to move this new plane around to where we

want to cut our model. You can move the plane so that it cuts along any axis if there are other sections you want to discard. When you're happy with your selection, click on Accept in the Plane Cut menu. Repeat this with each plane around the model until you just have the area you're interested in (**Figure 7**). You might notice some small fragments are floating around your model. We're going to remove those by again clicking Edit, followed by Separate Shells. An object browser will appear with a list of all your fragments. Select the fragmented shells in turn and click on the bin icon in the bottom right of the object browser to be left with your model.

Next up, we're going to plug any holes in your model that aren't supposed to be there. First, zoom in on any of the holes you wish to fill in. Click on Select from the toolbar on the left, then adjust the brush size to one that you're happy with and draw around the edges of the hole. The area should turn orange as you select it (**Figure 8**). When you've selected all around the hole, release the mouse button and press **F** on your keyboard to fill the area. If you wish to delete a selection, you can follow this same process and then press **X**. You can also make the model solid at this stage by clicking on it, followed by Edit, then Make Solid.

Optionally, you can smooth out any areas on the surface of your model by selecting Sculpt from the toolbar on the left, then clicking on Brushes, and selecting the RobustSmooth brush. At this stage, it's important not to brush away any of the details of your model that we worked so hard to extract – use the strength slider to reduce/increase the smoothing effect.

Finally, click on File, then Export and save your model in STL format, and now you have your final model ready for 3D printing. If the model is tiny or huge when you bring it into your slicer, simply adjust the scale until you're happy with the size. Check out our finished printed model in **Figure 9**. □



**Figure 9** □  
Marble filament was  
the only choice for  
replicating this statue

# Make a 3D camera



PJ Evans

MAKER

PJ is a writer, software engineer, 3D photography enthusiast, and organiser of the Milton Keynes Raspberry Jam.

@mrpjevans

## You'll Need

- ▶ 2 x Raspberry Pi Zero W  
[magpi.cc/pizerow](http://magpi.cc/pizerow)
- ▶ 2 x High Quality Camera  
[magpi.cc/hqcamera](http://magpi.cc/hqcamera)
- ▶ 2 x 6 mm or 16 mm lens  
[magpi.cc/hqlens](http://magpi.cc/hqlens)
- ▶ 2 x Official Raspberry Pi Zero Case  
[magpi.cc/pizeroCASE](http://magpi.cc/pizeroCASE)
- ▶ Google Cardboard (optional)  
[magpi.cc/cardboard](http://magpi.cc/cardboard)
- ▶ 3D-printed mounting plate (optional)  
[magpi.cc/hq3dmount](http://magpi.cc/hq3dmount)
- ▶ 8 x M 6 mm bolts and nuts (optional)

Add some depth to your photography by taking true 3D pictures with the new High Quality Camera

**W**ith the release of the High Quality Camera, photography using Raspberry Pi has been taken to a whole new level.

A 12.3 megapixel sensor and support for multiple lenses has already had enthusiasts experimenting with new types of photography previously out of their reach. Adding a Raspberry Pi to the mix allows for many possibilities, such as time-lapses or connection to sensors. Here, we're going to use two cameras together to create amazing 3D photos that can be viewed in a variety of ways, even without glasses.

## 01 Prepare Raspberry Pi

To create 3D photographs, we need to be able to take two photos simultaneously, about 5cm apart. These images can then be processed into a variety of different formats such as parallel view, cross-view, or anaglyph (when you wear red/green glasses). As this is a 3D project, it's not surprising to learn that we'll need two of everything. As we can only attach one camera per Raspberry Pi Zero W, we'll need to prepare a left and right computer. Start by installing Raspbian Lite as normal on both computers and updating everything with `sudo apt update && sudo apt upgrade`. Make sure both computers are connected to WiFi before proceeding.



▲ Here's an example of what can be produced. This is a cross-view image, so, if you can, cross your eyes together until they settle on a central image. Don't strain if your eyes feel tired!

## 02 Know the difference

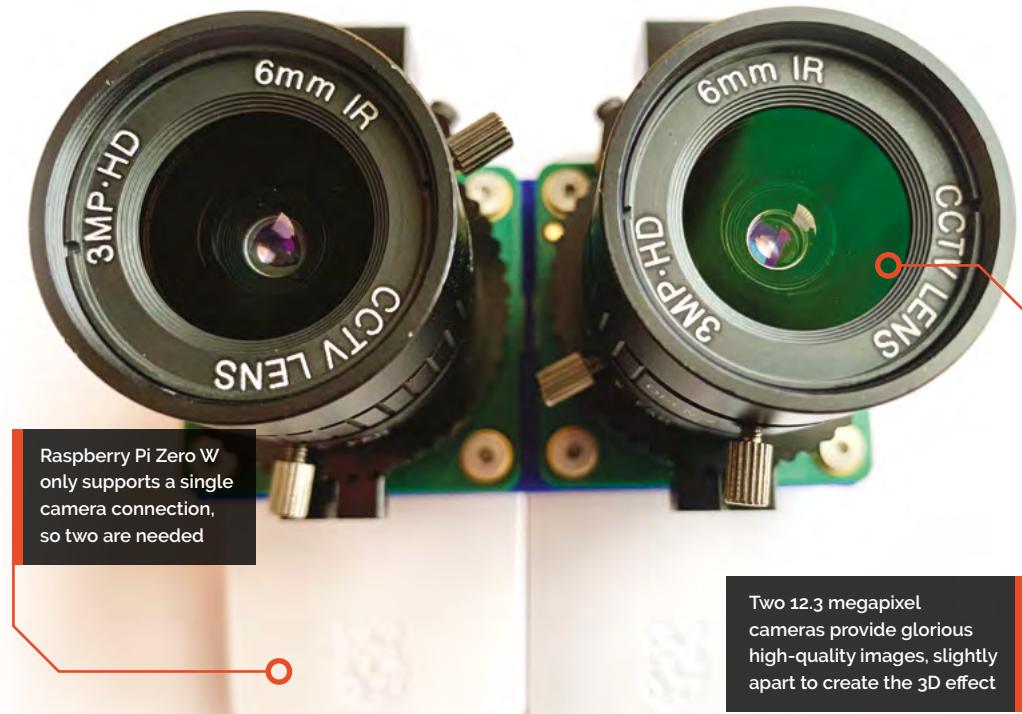
Choose one computer for the left camera and the other for the right. From the command line, run `sudo raspi-config` and go to Networking Options > Hostname. Change the name from 'raspberrypi' to 'leftcam' and 'rightcam' on each respective Raspberry Pi Zero. Also in raspi-config, make sure SSH is enabled on each (Preferences > Interfaces).

After this, leave the configuration utility and shut both computers down. Now is a very good time to attach the short camera cable that was supplied with the case to each Raspberry Pi Zero and thread it through the slot on the rear of the case and insert both computers. Add the cover and label each case as 'left' or 'right'.

## 03 Power sharing

As we have two Raspberry Pi Zero boards, we need two power supplies, right? Well, we can pull a little trick so that only one is required. With some wire, solder a 5V line and a ground (GND) line from one GPIO to the equivalent on the other (see **Figure 1** diagram). This 'power rail' allows the second Raspberry Pi Zero W to pull power from the one connected to a USB power supply. Just remember to use a suitable power supply with enough amperes to power both. We found





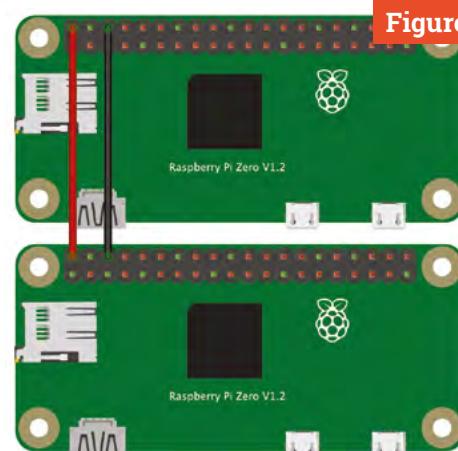
the official micro USB supply worked well. Check, check, and double-check before following this step. Soldering to the wrong connectors could permanently damage your devices.

#### 04 Attach the cameras

It's really important that the two cameras are lined up together and not at odd angles. We've provided a STL file for 3D-printing a mounting plate that holds them perfectly in place. You don't have to use it, but if you do have access to a 3D printer, it'll make life easier. Connect the short ribbon cables provided with the cases to each camera, then attach the cameras to the mounting plate side-by-side using the nuts and bolts. Be very careful not to bend or tear the ribbon cables. Finally, flip the cameras over so the plate is resting on the case lids and affix them with some sticky pads.

#### 05 Camera testing

Before going any further, test that both cameras are working as expected. Carefully attach the lenses to each camera board (if you're using the mounting plate, watch out for the control levers hitting each other). Use SSH to log in remotely to your left camera (`ssh pi@leftcam.local`) and at the command line, enter this:



`raspistill -o test.jpg`

After a few seconds, an image file will be created in your current directory. Transfer it back to your computer and have a look. Chances are it'll be blurry – but so long as an image was taken, we're all good. Repeat this test on the right-hand camera.

#### 06 Streaming for two

If you've been wondering how one camera is going to get its image to the other, the answer is by setting up a HTTP-based stream on each

#### 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](http://magpi.cc)

Figure 1

◀ Figure 1 Here, a 5V GPIO pin and GND pin are connected to their equivalents on the other GPIO so one Raspberry Pi Zero W provides power for the other

#### Top Tip

##### Parallel

Parallel (left/right) is needed for devices like Google Cardboard. Without it, you can get the effect by looking into the distance until the image merges.

**Top Tip****Cross-view**

Some people struggle to see parallel images. Cross-view images (right/left) create the same effect by crossing your eyes to create a central image.

camera. This will turn each Raspberry Pi Zero W into a streaming webcam and we can then view both images from a further website we'll install later. The following steps need to be followed on each Raspberry Pi Zero W. Start by installing some libraries we need:

```
sudo apt install cmake libjpeg8-dev git
```

Now we'll get and build the MJPEG streaming software:

```
git clone https://github.com/jacksonliam/
mjpg-streamer.git
cd mjpg-streamer/mjpg-streamer-
experimental
make
sudo make install
```

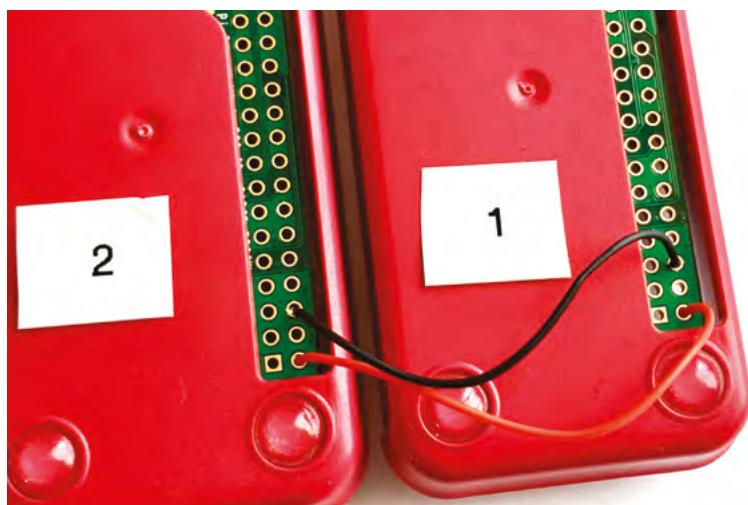
**07 First test images**

To run the package we've just installed, enter the following commands on each Raspberry Pi Zero W:

```
cd ~/mjpg-streamer/mjpg-streamer-
experimental
export LD_LIBRARY_PATH=.
./mjpg_streamer -o "output_http.so -w ./www"
-i "input_raspicam.so"
```

There is now a web server running on each device. Have a look by visiting <http://leftcam.local:8080/> and <http://rightcam.local:8080/>. Each will have a fun little website where you can view static and

▼ So we're not troubled by multiple power supplies, we can power one Raspberry Pi Zero from the other



video feeds from each camera. When you're done, you can stop each server by entering **CTRL+C** in the Terminal.

**08 Bring them together**

To view both images on the same page and generate 3D images, we need a further web service that takes a feed from both sites we've just installed. Create a directory called **3dcamera** in your home directory, then create two files: **3dcamera.py** and **control.html**. Enter both code listings (or download them from [magpi.cc/3dcameragit](https://magpi.cc/3dcameragit)) and save. This is a very simple web server and an HTML page that will display both images on a single page and, with a simple click, create and download a parallel-eye image.

Make sure both MJPEG streamers are running and then start the additional server on leftcam only:

```
python3 3dcamera.py
```

You should be able to access the site at <http://leftcam.local:8081> and be able to see a video stream of each device.

**09 View your image**

Take an image by holding the cameras steady and clicking 'Snap!' on the website. The dual image will be downloaded to your computer. If you have a Google Cardboard kit or one of the widely available mobile phone VR headsets, transfer your parallel image to your mobile phone and then view it in glorious three dimensions. If not, many people can see the image by focusing their eyes 'beyond' the two images until they merge into one. If you're struggling with this, reverse the two captured images as noted in the code to create a cross-view image.

**10 Take it further**

This is just a starting point for your adventures in 3D photography. We haven't touched on creating anaglyphs or streaming 3D video. Check out the GitHub repo ([magpi.cc/3dcameragit](https://magpi.cc/3dcameragit)) for a more advanced version that allows you to set the type of image to generate and adds a few more features. What's the most creative thing you can do with your 3D camera? **M**

# 3dcamera.py

**DOWNLOAD  
THE FULL CODE:**



[magpi.cc/3dcameragit](http://magpi.cc/3dcameragit)

► Language: Python 3

```

001. import os
002. import urllib.request
003. from http.server import SimpleHTTPRequestHandler,
    HTTPServer
004. from PIL import Image
005. from io import BytesIO
006.
007. port = 8081
008. control_html = os.path.dirname(os.path.realpath(
    __file__)) + '/control.html'
009. # Reverse the URLs to create cross-view images instead
    of parallel
010. left_camera =
    "http://leftcam.local:8080/?action=snapshot"
011. right_camera =
    "http://rightcam.local:8080/?action=snapshot"
012.
013.
014. def process_image():
015.     left_image = urllib.request.urlopen(left_camera)
016.     right_image = urllib.request.urlopen(right_camera)
017.     images = [Image.open(x) for x in [left_image,
    right_image]]
018.
019.     widths, heights = zip(*[i.size for i in images])
020.     total_width = sum(widths)
021.     max_height = max(heights)
022.
023.     side_by_side_image = Image.new('RGB', (
    total_width, max_height))
024.
025.     x_offset = 0
026.     for image in images:
027.         side_by_side_image.paste(image, (x_offset, 0))
028.         x_offset += image.size[0]
029.
```

```

030.     image_buffer = BytesIO()
031.     side_by_side_image.save(image_buffer,
    format='JPEG')
032.     image_data = image_buffer.getvalue()
033.
034.     return image_data
035.
036.
037. class handle_request(SimpleHTTPRequestHandler):
038.     def do_GET(self):
039.         print('Sending control HTML')
040.         f = open(control_html, 'rb')
041.         self.send_response(200)
042.         self.send_header('Content-type', 'text/html')
043.         self.end_headers()
044.         self.copyfile(f, self.wfile)
045.
046.     def do_POST(self):
047.         new_image = process_image()
048.         self.send_response(200)
049.         self.send_header('Content-type', 'image/jpeg')
050.         self.send_header('Content-disposition',
    'attachment; filename="3d.
051. jpg"')
052.         self.end_headers()
053.         self.wfile.write(new_image)
054.
055.
056.     httpd = HTTPServer(("", port), handle_request)
057.     try:
058.         httpd.serve_forever()
059.     except KeyboardInterrupt:
060.         pass
061.
062.     httpd.server_close()
```

# control.html

► Language: HTML

```

001. <!DOCTYPE html>
002. <html>
003.   <head>
004.     <style>
005.       .viewfinder {
006.         display: flex;
007.       }
008.     </style>
009.   </head>
010.   <body>
011.     <h1>3D Camera</h1>
```

```

012.     <div class="viewfinder">
013.       
014.       
015.     </div>
016.     <form method="post" action="/">
017.       <button>Snap!</button>
018.     </form>
019.   </body>
020. </html>
```

CODE  
THE  
CLASSICS  
VOLUME 1

# CODE THE CLASSICS

VOLUME 1



Brimble  
Crookes  
Gillett  
Malone  
Tracey  
Upton<sup>2</sup>



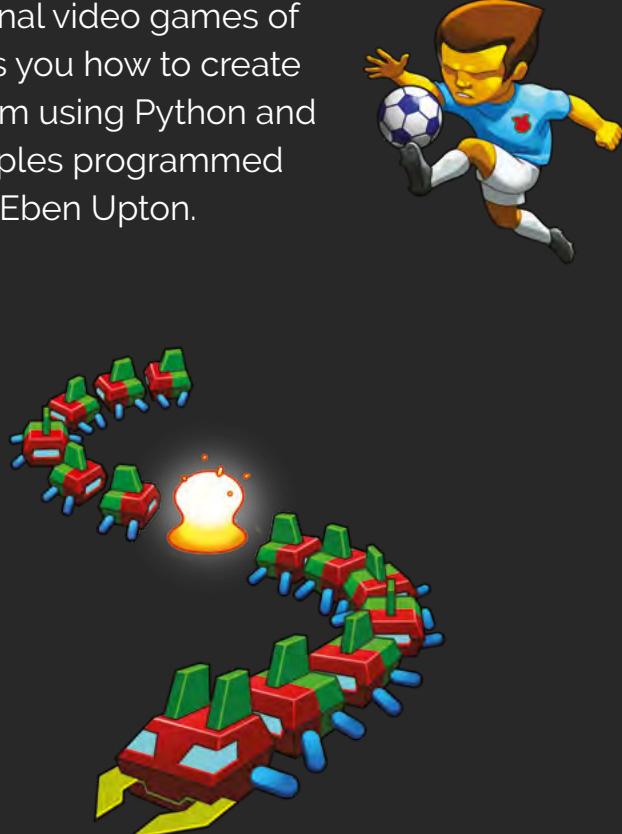


# CODE THE CLASSICS

VOLUME 1

This stunning 224-page hardback book not only tells the stories of some of the seminal video games of the 1970s and 1980s, but shows you how to create your own games inspired by them using Python and Pygame Zero, following examples programmed by Raspberry Pi founder Eben Upton.

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



Available now [hsmag.cc/store](http://hsmag.cc/store)

# Designing and building a modular tracked vehicle

Using an affordable, off-the-shelf, tracked robot chassis, let's build a 3D-printable modular tracked vehicle (MTV) that's easy to attach future upgrades to



**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!



**Above** Our completed modular tracked vehicle design and build

**A**t around £20 to £25, this chassis kit is often described online as a 'tank crawler chassis'. It arrives with a metal chassis, two geared DC motors labelled for 350 rpm at 12V, a set of drive wheels with some drive hub mounts, two non-driven/idler wheels, and some tracks. Also included is a bag of assorted nuts, bolts and washers, some tools, and a battery box designed to take two 18650 cells. Noticeably, having bought two of these kits, no instructions are supplied for the chassis kit, so we had to use some



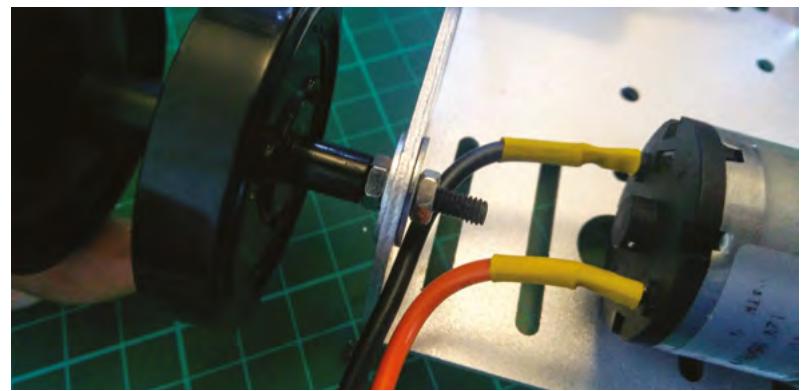
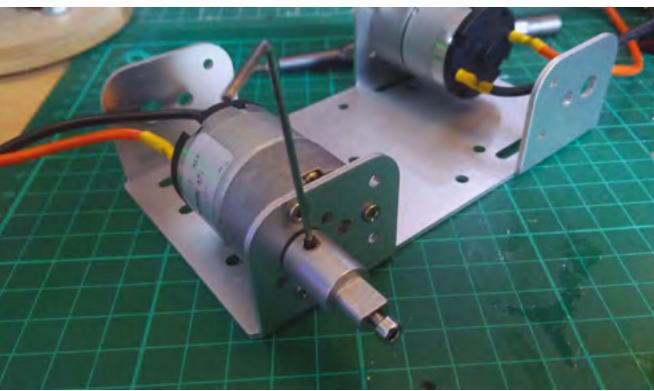
trial and error to explore the best way to assemble it. You can download the designs for our upgrades from [hsmag.cc/mtv](http://hsmag.cc/mtv).

Looking at the metal chassis, each side has two mounts: one with nine holes, and one with five holes. We realised that the nine-hole mount was the only one that had the correct holes to mount the motors; this meant that the motors are diagonally opposite each other, in turn meaning that there are some issues around symmetry in the build already.

We attached the two motors to the chassis with three of the M3 screws each, and then, using the small grub screws and the supplied Allen key, we attached the aluminium hub adapters onto the shaft of the motors. Next, we added the driven wheels, which were a good fit, onto the hub adapters, keyed into position with a flat section, and pulled onto the hub adapter with an M3 bolt (**Figure 1**).

## MAKING TRACKS

The idler wheels are interesting in that it's not particularly clear into which hole on the chassis mounts they are designed to go. There are a couple of candidate holes. The first we tried is the hole that is closest to the diameter of the long semi-threaded bolt that acts as an axle for the idler wheel. We tried this first using a nut and washer on both the outside and the inside of the chassis mount to create a fixed axle, carefully locking the nuts in a position that allowed the idler wheel to spin, but not be so loose as to be able to move too much up and down the axle. Skipping ahead slightly, once we had added both idler wheels using these holes and added the tracks, we realised that although the tracks were parallel to the chassis sides, they were offset front to back. So looking from

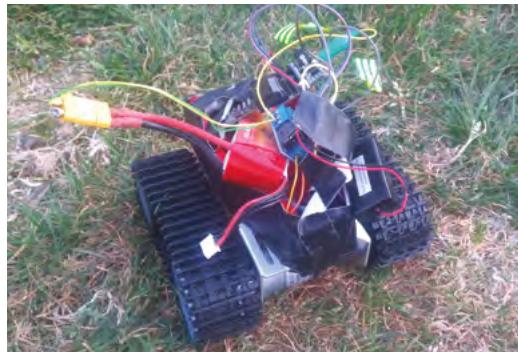


above, one set of tracks was around 10 mm extended beyond one end of the chassis, while the other set of tracks was extended the same amount off the other end. It definitely looked odd, and we worried that it might affect the tracking of the vehicle.

### LET'S COBBLE AND TEST!

We cobbled together some drive electronics and gaffer-taped them on to test (**Figure 2**). While we were still unhappy with the way that the tracks looked, it did track correctly and drove well. However, we decided to strip the tracks and idler wheels off and tried to reconfigure them. We ended up using the much larger holes on the five-hole mount tabs, and discovered we could get the tracks to align not perfectly, but much better. We also realised that clamping the axle bolt using the large washers through the larger hole meant that we could slacken the idler wheel axle, and it could move within the larger hole to allow the tracks to be loosened for removal without having to split the tracks (**Figure 3**).

As we mentioned earlier, once we had the tracks and motors in place, we soldered some wires to the motors (**Figure 4** overleaf) and quickly cobbled together some running gear, a micro:bit in an RKub2 breakout board, an L298N motor driver, and a 3S 2200 mAh LiPo battery. We used a slightly modified version (**Figure 5** overleaf) of the micro:bit code that we used to control the :MOVE mini MK2 robot in our *Working with micro:bit radios* article in issue 30.



While this looked like an abomination, it allowed us to test the chassis to see how it ran – it also helped us consider the design. We realised that this chassis was very capable and can move quite quickly and accurately – it is also pretty good across some quite rough terrain and can certainly traverse short grass, gravel, and more. Another useful thing we realised was that if the centre of gravity wasn't particularly →

### MAKING TRACKS

Whichever option you choose in terms of the idler wheel axle position, you'll have to shorten the supplied tracks by removing some links. To do this, we need to remove the pins between some links. For this, you will need a thin pin-type tool. We found a tool originally designed to be an insert pin to release a mobile phone SIM card cover that worked – a drawing-pin with the point filed off would also work. Once we discovered that one end of the track link internal pins was knurled and one wasn't, we could remove pins fairly easily – but until that point, this was a very frustrating task!

The tip we can share is that, as you can see in the image below, if you push your tool against the track pin from the side where the pin ends on the inside of the track and the pin link is lower than the edge of the track, you are pushing the pin out in such a way that the short knurled (and therefore slightly wider) section of the pin is pushed out at the other end first. Trying to push the thicker knurled end through the track link holes doesn't work. We removed one pin and then wrapped the track around the idler and powered wheel to work out how many links to remove, and then we removed them. Once we were happy with the track lengths, we pushed the pin back in.



**Figure 1** The motor and hub mounted. The chassis mounts feature lots of holes, some of which are used, and some of which aren't

**Figure 2** A horribly cobbled and gaffer-taped ten-minute prototype was put together to allow a test-drive of the chassis which informed our design ideas for the modular mount system

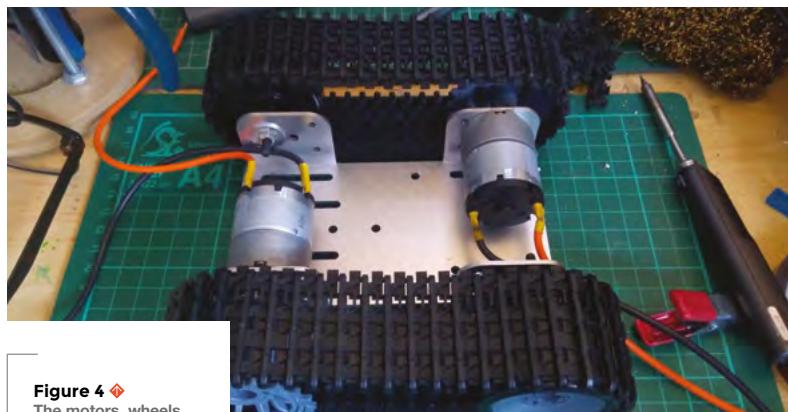
**Figure 3** The idler wheel mount with the axle clamped into the larger of the available holes using the four larger washers

### YOU'LL NEED

- ❖ A 'tank crawler chassis kit'
- ❖ Soldering iron and some wire
- ❖ Access to a 3D printer
- ❖ Some M4 threaded bar
- ❖ Some M4 and M3 nuts and bolts
- ❖ L298N motor driver and a micro:bit

# Designing and building a modular tracked vehicle

## TUTORIAL



**Figure 4**

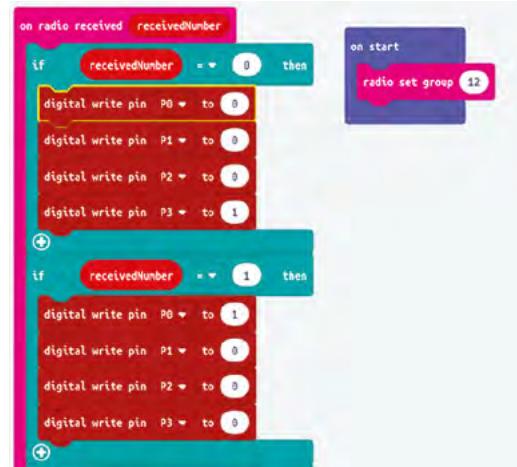
The motors, wheels, and tracks mounted, and wires soldered to the motors

**Figure 5**

We used the code featured in the *Working with micro:bit radios* article, replacing the servo blocks with 'digital write' pin blocks. Pins 0, 1 and 2, 3 are attached to the motor control pins on the L298N – writing one pin high and one low makes the motor rotate one way or the other

central, it was prone to tipping front to back when going over an incline. As the 2200 mAh battery is by far the heaviest component, we realised our design had to hold this centrally in place, and as close to the chassis as possible.

Measuring up the dimensions of the chassis and noting the positions of the slots, we used FreeCAD to design the lower tray first. We added our 10 mm-spaced hole rails on the front and the edges. We also added some raised ridges inside to ensure our 2200 mAh cell stayed in a precise central position. We



decided that the end of the chassis with the larger circular hole was going to be the back, and that we would run the motor wiring up through that hole. Therefore, we decided to leave a larger area around the hole and under the area of the battery open to allow wires to be routed through (**Figure 6**). After printing, we attached the lower deck using some short lengths of M4 threaded bar we cut to around 50 mm, with a nut under the chassis and a nut on top of the lower deck.

The upper deck is the same overall dimensions as the lower deck, and also has the side mount 10mm-holed rails in the same relative position to the lower one, but is peppered with more 10mm-spaced holes. We also designed a larger slot into each end of this deck to allow for the routing of cables. The gaps between the side rails and the front and back rails on both decks were also for if we needed to route cables around the outside of the decks, but we might get rid of these in a future revision as we haven't used them. When fitting the upper deck (**Figure 7**), we used a couple more nuts as this allowed us to experiment with different heights for this deck. However, we might replace this system with

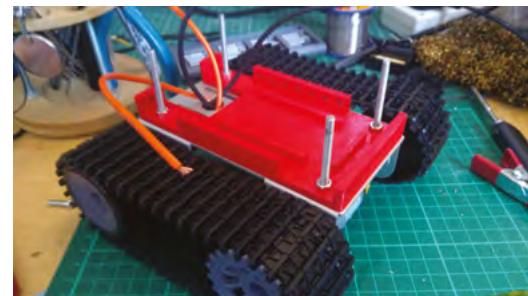
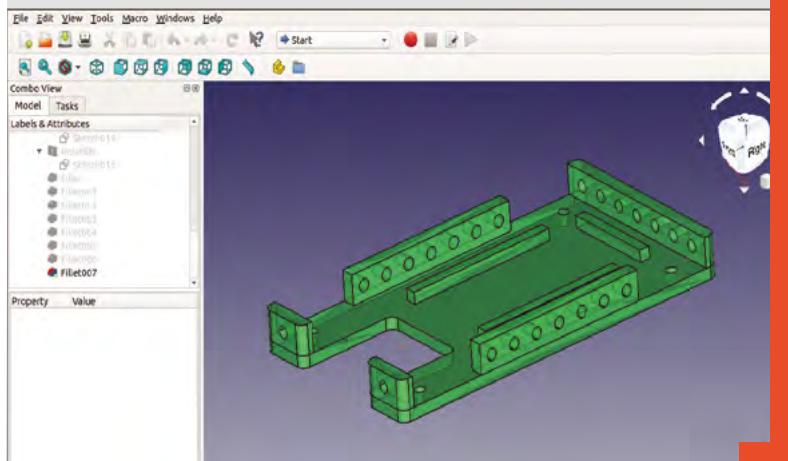


**Figure 6**

Our lower deck design, printed and bolted onto our chassis

## MODULAR TRACKED VEHICLE

Thinking about our design, we wanted to create some upper sections that mount onto the chassis that could hold the electronics, but that could grow and change, and we could add and remove components and experiments. Realising this, the modular tracked vehicle project was born! We decided that we would include a lot of mounting holes and that, wherever possible, they would be spaced at 10 mm between centres. This means that when designing additional modules, we can add 10 mm spaced mount holes and know we can mount it in a variety of positions. We settled on the hole diameters to be roughly equivalent to M4 – we actually drew most of the holes in FreeCAD to be 4.2 mm diameter. The reason for this is that then they can be used as a through-hole for an M4 bolt, but also this is a good diameter to use if we wanted to use a brass thermal insert nut with an internal M3 thread. This gives us options to be able to still use the 4.2 mm holes, even if the back of the hole is covered or inaccessible.

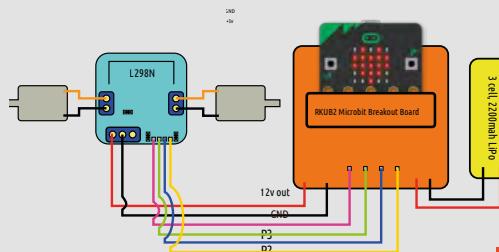


## WIRING

We are more focused on sharing the mechanical design aspects in this tutorial, and we may well change the control system in the future. The micro:bit and L298N motor controller setup is pretty basic. There's no proportional control for the motors – they are simply switched on or off. That said, the image to the right shows the main connections, and there are plenty of tutorials online to show you how to use the micro:bit and the L298N, such as this great primer by Les Pounder: [hsmag.cc/UVBUTz](http://hsmag.cc/UVBUTz).

We used the RKub2 breakout for the micro:bit because it can be powered directly from a three-cell LiPo battery – it regulates power to the micro:bit and offers a 12 V, 5 V, and 3.3 V output. This is useful because we ran our motors and L298N off the 12-volt (or in our case 11.4 V) rail, and the LED

headlights were connected via a 150 ohm resistor to the 5 V rail. It also breaks out all the micro:bit pins as we needed one more than the three available if attaching directly to the micro:bit.



some cut lengths of aluminium tube slid over the 4mm threaded rods to act as standoff spacers, as this would make it quicker to disassemble and reassemble.

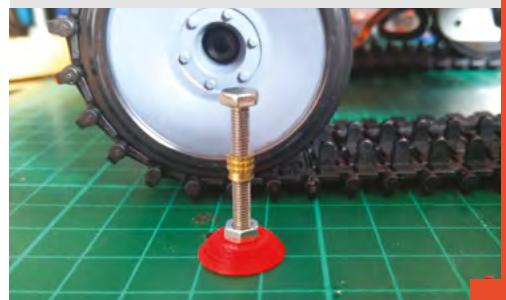
The first add-on modules we made for the MTV consisted of a couple of bolt-on platforms that could hold the electronics in place. For now, we have built this version using the micro:bit and L298N motor driver, and so we designed with those in mind. We wanted to see what it was like to mount something at an angle, so the first module we designed was a 45-degree mounting plate, into which we added the mounting holes for the L298N – this was attached to the back of the upper deck. We mounted the micro:bit and RKub2 breakout panel onto a flat module panel that could be bolted flat onto the upper deck, which is a simple solution. We could have mounted it to the side rails, or indeed created an L-bracket mount to mount it rising up vertically. The RKub2 board has the frustration that when soldered together, the mounting holes on the PCB are all partially obscured by the components, so we ended up using some double-sided tape to stick the board to the mounting panel.

## LET'S ACCESSORISE!

Next, we wanted to design some more accessories to explore how we might add modules to the MTV. First up, we added some LED headlights consisting of a 3D-printed mount that could contain a 3 mm LED, which we used a spot of hot glue to retain into the print. We then designed and printed a holed 45-degree L-bracket that could be used to side mount accessories, and then a similar bracket with no holes. The no-holed 45-degree bracket was used because we had some Lego-compatible tape that we added, meaning that Lego or Lego-compatible items could be built onto it. Finally, while we haven't had a chance to wire this up and write some code for it yet, we designed a mount to be able to add an HC-SR04 ultrasonic sensor. □

## HOLDING THE BATTERY

We 3D-printed a small pressing foot that could be threaded onto an M3 bolt and held in position with a locking nut that would act to clamp the LiPo battery down into its bay. We used our thermal insert rig (a modified soldering iron rigged onto a Dremel drill press accessory) to heat and push an M3 thermal insert into a central hole on the upper deck. Assembling the clamp through the threaded insert, we now had a nice clamp that held the LiPo in place securely.



**Figure 7** Top deck fitted after printing

**Below** Lots of accessories added to the upper and lower deck, but still plenty of room to mount more



DON'T MISS THE BRAND NEW ISSUE!

YOUR OFFICIAL RASPBERRY PI MAGAZINE

# The MagPi

Issue 95 | July 2020 | magpi.cc

The official Raspberry Pi magazine

## BUILD A RASPBERRY PI 4 GAMES CONSOLE

Play classic arcade games with RetroPie

**SUBSCRIBE FROM JUST £5**

- FREE! 3 issues for the price of one
- FREE! Delivery to your door
- NO OBLIGATION! Leave any time

Complete

**FREE PI ZERO W STARTER KIT\***

With your 12-month subscription to the print magazine  
[magpi.cc/12months](http://magpi.cc/12months)

\* While stocks last

**Buy online: [store.rpipress.cc](http://store.rpipress.cc)**

# FIELD TEST

## HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG  
**106**

### CAN I HACK A JOGGLER?



Revitalise some old electronics

## REVIEWS

**108 Nozzle Fun Pack**

Upgrade your 3D printer

**110 Ringo mobile phone**

DIY mobile computing

**112 Arduino Portenta H7**

Two cores for extra power



PG  
**100**

## BEST OF BREED

Build your own clock kits



Clock kits that every DIY enthusiast will love to show off

---

BEST OF BREED

---

ONLY THE  
BEST

# Clock kits that every DIY enthusiast will love to show off

Clock kits that make for a great time. </end pun>

---

By Marc de Vinck

 @devinck

We really like kits that have a long life even after you build them. There is nothing wrong with just enjoying the process of building, but when a kit has a use afterwards, it makes it that much better. We also have a fascination with clocks, both analogue and digital. So, in this Best of Breed, we'll be looking at some of my favourite electronic clock kits.

Very few kits have such a lasting usefulness compared to clocks. It's something everyone needs and can use on a daily basis. And even if you don't need a clock, build one and give it as a gift, as we have in the past. We assure you, the recipient will love it!

Another nice theme among clock kits is the availability of enclosures. So many kits don't have any safe or aesthetically pleasing way to display the final build. But that's not the case with many of these kits. Just solder, build, and display!



# Alpha Clock Five, White Edition vs Game Timer Pro

**EVIL MAD SCIENTIST** ◇ \$175 | [shop.evilmadscientist.com](http://shop.evilmadscientist.com)

**NOOTROPIC DESIGN** ◇ \$54.95 | [nootropicdesign.com](http://nootropicdesign.com)

The **Alpha Clock Five** is a larger than expected and a beautifully designed desk clock kit from Evil Mad Scientist. Each digit of the clock is 2.3" (5.8 cm) tall, and it's available in several different LED colours, including white, red, and blue. Each kit also includes a ChronoDot real-time clock module, which makes this a very reliable and accurate alarm clock.

And as you would expect, all the basic functions are covered in the Alpha Clock Five – including an alarm, multiple alarm tones, a snooze function, and month/day display – housed in an acrylic case with laser-engraved buttons. Also included is a universal-input power supply that will work with worldwide voltages. A nice complete kit!

If you're on a budget, or you want to design your own enclosure, you can also opt for the basic version,

which does not include the laser-cut case, but features everything else included in the standard variety. The Alpha Clock Five clock kit does not require any programming – Evil Mad Scientist did that part for you. But if you like to tinker, the kit is open source and all the necessary code and files are provided to customise it to your heart's content.

**Below** ◇ You can display a wide range of characters on this display



**P**ress the red button and let the countdown begin! Which wire will you cut? The Game Timer Pro from Nootropic Design is definitely the most interesting clock in this roundup. This scary-looking, bomb-like contraption is the quintessential 'which wire do you cut?' bomb defuser device that you have seen in countless movies. The Game Timer Pro can be used as a prop for films, a fun novelty countdown clock, or what I think is the most interesting use, as a component for an escape room. This version is not a kit: it comes fully assembled. However, you'll most likely want to include some kind of additional prop to attach it to for added authenticity.



Beyond the basic countdown clock, the Game Timer Pro can also be controlled with an infrared remote controller, or a keypad add-on package that allows you to disarm the timer by entering a four-digit code to defuse it. And if you really want to build it yourself, check out the Defusable Clock Kit, which is very similar to the Game Timer Pro, but you have to solder everything.

**Disclaimer:** Please be responsible! This kit could certainly scare people if used out of context. Shooting a film or building an escape room? Then this is for you! Using it to scare someone? Not a good idea.

**Left** ◇ 'Game' is printed on the silk screen to help avoid any confusion

## VERDICT

**Alpha Clock Five, White Edition**

A big display, and fun to assemble.

**9 /10**

**Game Timer Pro**

A fun addition for an escape room!

**7 /10**

Clock kits that every DIY enthusiast will love to show off

BEST OF BREED

## Solder:Time Desk Clock

SPIKENZIELABS \$89.95 | [spikenzielabs.com](http://spikenzielabs.com)

**S**pikenzieLabs is known for its high-quality kits that quite often include a well-made laser-cut enclosure, and the Solder:Time Desk Clock is no exception. It's fun to put together, the integrated circuit (IC) is preprogrammed, and it includes the enclosure that gives it a truly finished look when assembled. The kit also features a real-time clock IC with battery backup, which gives it extremely accurate timing and a fail-safe for when the power goes out. You can set an alarm, use it as a stop-watch, program a scrolling message, or simply display a 'worm animation'. Yeah, try doing that with your regular old clock!



And for those of you who like to tinker with Arduino, the clock is easily reprogrammed via the standard Arduino IDE. Head on over to the website for more information about hacking this kit. You will also find all the required libraries and schematics, along with a beautiful PDF step-by-step guide.

**Left ♦**  
The finished clock looks great and is very versatile

**VERDICT**

Solder:Time  
Desk Clock

A fun kit, with a sleek case.

**10**/10

## Solder:Time Watch Kit

SPIKENZIELABS \$29.95 | [spikenzielabs.com](http://spikenzielabs.com)

**T**he lone watch in this clock roundup is the Solder:Time Watch Kit from SpikenzieLabs.

Introduced in 2011, this kit was an instant success. And why not, since it's easy to solder, unique, and really well designed? SpikenzieLabs designed a laser-cut enclosure that perfectly fits the PCB and allows for a hook-and-loop wristband. And yes, it's a bit on the large side, so you most likely won't wear it on a daily basis.



However, if you're like us, you just might use it on your desk, or attach it to your bench for a unique-looking timepiece among all your other tools and devices.

That's where this author's lives.

And if soldering isn't your thing, you can also purchase it with the PCB already assembled. But come on... that takes all the fun out of it! Head over to the SpikenzieLabs website to learn more about this unique watch, download the source code if you'd like to see how it works, and check out the beautiful step-by-step instructions.

**Left ♦**  
Wearing this watch on your wrist is sure to attract attention

**VERDICT**

Solder:Time  
Watch Kit

Admittedly a bit large for most wrists, but it's certainly unique.

**8**/10

# Angular Clock Kit

**WICKED DEVICE** \$59.95 | [wickeddevice.com](http://wickeddevice.com)

**W**e've always liked the aesthetic of an analogue clock, especially the kind that use repurposed analogue voltage meters or similar components to indicate the hours, minutes,

**and seconds.** And although this is a fairly popular project in the DIY community, especially for those who tinker with Arduino, this author has never actually built one for himself. Well, that just might change after finding this kit from Wicked Device.

The Angular Clock Kit, originally designed and prototyped by Daniel and Ken Rother, is based on the Arduino platform and is completely open source.



Right ♦  
One of the  
few analogue  
clocks on test

It should only take about 45 minutes to assemble, and no soldering is required. The kit is open source, which allows you access to all the design files, including the code. So why not make it your own? Maybe it can indicate the phases of the moon, Instagram likes, or your YouTube subscriber count? The kit comes with everything you need to get up and running, including a nicely designed laser-cut enclosure.

## VERDICT

Angular Clock Kit

A great combination of digital and analogue.

9/10

# VFD Modular Clock IV-4

**VFD** \$90 | [tindie.com](http://tindie.com)

**W**e love vacuum fluorescent display (VFD) tubes, and we really love VFD clocks. It's another really popular build for the DIY enthusiast, but it's not an easy DIY build like a simple seven-segment clock, since these devices often require some tricky electronic circuitry to get them working properly. The VFD Modular Clock IV-4 from the AkaFugu Corporation in Japan takes care of all the complexities and packages it up into a simple and elegant DIY kit.

The clock is controlled by an ATmega328P microcontroller. And the controller board also contains all the high-voltage VFD driver circuitry used to illuminate the individual digits of the display. Best of all, since it's based on an ATmega328P, you can



Below ♦  
Retro VFD tubes have a sci-fi glow

program it just like a standard Arduino. This enables you to modify it to display any combination of four numbers or alphanumeric digits that you'd like. You can purchase the kit with or without the VFDs, but we highly recommend picking it up with them included, making for a much easier build.

## VERDICT

VFD Modular Clock IV-4

An interesting, partially assembled kit.

9/10

Clock kits that every DIY enthusiast will love to show off

## BEST OF BREED

# ChronoDot

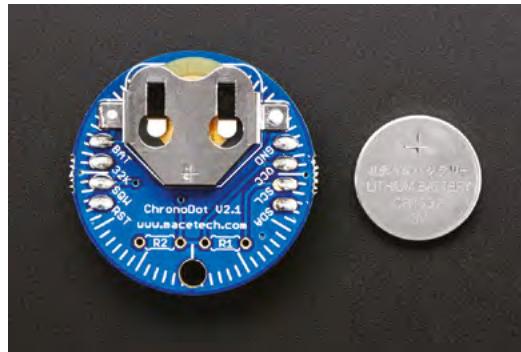
ADAFRUIT ◇ \$17.50 | [adafruit.com](http://adafruit.com)

**L**ooking to roll your own clock? One thing to consider is the accuracy. Yes, you can just use the internal clock of the microcontroller of your choosing, but you'll quickly learn that they aren't all that accurate, especially over longer periods of time. What you'll want to include in your build is a real-time clock (RTC) module. And that's where the ChronoDot comes into play.

Not only will you get accurate timing, but you also get a battery backup. This allows you to lose power, but not the time. It's a great little device that this

“  
If you need accurate and reliable time for your next project, you need a ChronoDot  
”

author has personally used on several projects. The drift is less than one minute per year, and it's really easy to implement, thanks to the header pins that easily plug into a breadboard or PCB. It communicates to your microcontroller via an I<sup>2</sup>C interface, making it easy to implement with your Arduino, and most other microcontrollers with ease. If you need accurate and reliable time for your next project, you need a ChronoDot. □



**Left ◇**  
Small enough to fit in almost any project

## VERDICT

ChronoDot  
Adding accurate time to your project is easy with the ChronoDot.

**10/10**

## BULBDIAL CLOCK - MONOCHROME KIT

EVILMADSCIENTIST ◇ \$65 | [shop.evilmadscientist.com](http://shop.evilmadscientist.com)

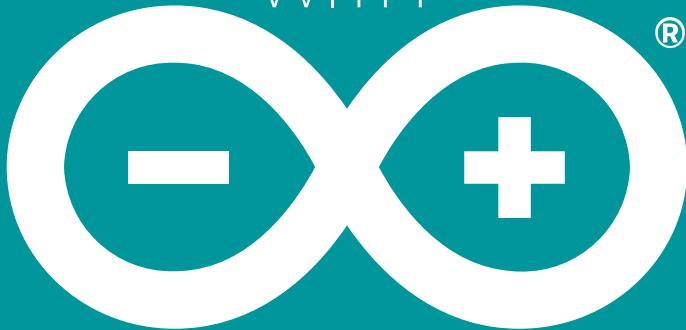
You might be asking yourself, what about the Bulbdial Clock Kit by Evil Mad Scientist? Don't worry, we reviewed it in HackSpace issue 12 a while back as part of the Best of

Breed 'Lots of LEDs' kit reviews. Let's just say it's a great kit, and one not to be missed if you are into building your own unique clock. Check out more at [hsmag.cc/issue12](http://hsmag.cc/issue12).



# GET STARTED

WITH



# ARDUINO

Robots, musical instruments,  
smart displays and more



## Inside:

- Build a four-legged walking robot
- Create a Tetris-inspired clock
- Grow veg with hydroponics
- And much more!



AVAILABLE  
NOW [hsmag.cc/store](http://hsmag.cc/store)

plus all good newsagents and:

WHSmith

BARNES&NOBLE



FROM THE MAKERS OF HackSpace MAGAZINE

# Can I Hack It? An O2 Joggler?

Can we reuse an old O2 Joggler?



**Les Pounder**

@biglesp

Les Pounder loves taking things to pieces and seeing how they work. He teaches others how to be makers and tinkerers at events across the UK. He blogs at [biglesp.com](http://biglesp.com).

- YOU'LL NEED**  
 O2 Joggler / OpenFrame
- COST**  
 £ Various
- WHERE**  
 eBay/thrift stores

**Right** ♦  
The O2 Joggler may look like a picture frame, but inside we have a full Linux-powered computer, ready and waiting to be hacked!

In 2009, O2 (a cell/mobile phone carrier in the UK) brought out the Joggler. While bearing an unusual name, the goal of the Joggler was to replace our refrigerator door – the place where a calendar, notes, and photos are stored.

The Joggler was there to make our lives easier. Sadly, the concept wasn't popular, and after a year at £150, the units were reduced to £100, and then £50. So, in 2010, we bought two units and hacked around with them. Alas, they went back to the loft, and were not used for eight years. So, when clearing the loft during the lockdown, we found them and thought 'We need to hack them!' Joggliers are still available on popular internet auction sites for around half what we paid, if you want to follow in our footsteps.

## GENERAL CONSTRUCTION

The Joggler is dominated by a 7-inch screen, and there's a metal stand at the rear. Made from plastic, the case can be easily worked with hand tools. The metal stand is secured inside the case. To get inside the Joggler, we need to remove four cross-head screws which are hidden under a large sticker on the rear. Then, at the bottom of the Joggler, there are a series of friction-fit plastic lugs that need to be carefully pulled open; then, move around the case with a plastic tool and pop the remaining plastic lugs to split the front and back apart.

## ELECTRONICS

The Joggler is a low-powered PC. Using an Intel Atom Z520 CPU running at a maximum 1.33GHz and coming with 512MB

of RAM, the Joggler is a product of 2009, where these specs were enough to get the job done. We have on-board WiFi via a Ralink RT2770 USB add-on, connected to an internal USB socket; there is also another USB 2.0 port on the right-hand side of the Joggler. On the rear of the unit are three ports: a DC barrel jack for the 5V 4A power supply, Gigabit Ethernet, and a 3.5mm external speaker connection – but honestly, the internal speakers are great! Internal storage is via 1GB flash storage which stores a custom version of Ubuntu 18.04 running the BusyBox window manager, which is the default operating system from OpenPeak.

Accessing the electronics is easy, largely thanks to its using standard machine screws and fixings. The CPU and GPU are accessible, if care is taken, as they are covered by a large thermal pad which is used to draw heat away from these components. If taking the unit apart, ensure that you take photos of where wires are connected, and ensure the thermal pad is replaced. Talking of thermals, the Intel Atom



**Above ↗**

Splitting the Jogger apart, we see the screen and touch interface connected to the mainboard via many wires. Make sure they stay connected

Z520 is a rather warm CPU, and when it gets too warm, it will thermal throttle. So, either add extra cooling, or run the CPU a little slower.

**HACKABILITY**

Can it be hacked? Well, yes, it can. The first easy hack is to use a newer operating system. Luckily, the **birdslikewires.net** website has all the resources and tools that we need to add Ubuntu 18.04 to the Jogger via an external USB drive. We performed two installs. First, we installed Ubuntu 18.04 and the Xubuntu window manager, and used the Jogger to display web content. It worked OK, but it was a little slow. So we tried another project: a custom control unit for Pimoroni's Mote light kit. We installed Ubuntu 18.04, a simple X server, then wrote a few lines of Python to create a GUI interface (using guizero) and control our lights using the Jogger's touchscreen. It works, and it's super-quick too.

For a clean look, we removed the internal WiFi card, but save this, as it is a USB device which can be reused in another project. We replaced the WiFi card with a 4GB USB 2.0 drive containing Ubuntu 18.04, and then sealed the unit back up.

But what if you want to take your hack further? As mentioned before, the CPU will throttle if it gets too warm, so adding a heatsink and fan would be a great start. A 5V fan can be connected to the power supply with relative ease. A more advanced hack

**WISE WORDS**

As this is my final *Can I Hack It?*, I wanted to part with a few words of wisdom that I have learnt.

If you can't do it safely, then ask for help! Learn a new skill with a friend who has the knowledge to teach you safely. Especially if you are learning how to work with new tools and high-voltage power. Your health and safety are vital!

Learn how to FAIL. Failure is only complete if you give up. Failure is not the end; rather, it is a time for reflection. Why did it go wrong? How can I prevent it from happening again? FAIL is an acronym: it's our First (or Further) Attempt In Learning. We learn resilience from failure and overcoming obstacles.

Share your skills. Write a blog, record a YouTube video, start a podcast, and tell others about what you have learnt. The world is a richer place when we share ourselves to help others. Think of your favourite teacher, mentor, or friend and how they helped you learn something new.

Take photographs and be methodical when tearing equipment apart. Document the process, as you will need to refer to it when putting it all back together. Store screws/part in trays/plastic cups.

Electronics can contain harmful components. Wear appropriate gloves, eye protection, and wash your hands after handling older electronics.

Be safe, never stop learning, and happy hacking!



would be to remove the unit from the chassis and integrate it into a wall unit or desk. Touchscreen control for your desk lights is possible with this!

**CONCLUSION**

The O2 Jogger may be an old piece of kit, but thanks to hackers like Birds Like Wires, we can reuse this old piece of tech to do wondrous things. So, go on, get a cheap unit online, and make it into something wonderful. □

**Above ↗**

At the rear of the unit are the power, Ethernet, and headphone outputs. Under the sticker are four cross-head screws, which are to be removed to open the case

**Right ↗**  
Don't throw away the USB WiFi card – use it in other projects!



# E3D Nozzle Fun Pack

From ultra-detail to ultra-fast in one packet

**E3D** ♦ £30 | [e3d-online.com](http://e3d-online.com)

By Ben Everard

 @ben\_everard

**T**here is no perfect nozzle for printing with hot plastic because it's a trade-off between speed and quality. Smaller nozzles can print with incredible detail, but take a long time to do it. Larger nozzles are quicker, but can leave large layer lines. Most 3D printers ship with a 0.4mm nozzle, and this is a pretty good compromise. They print small things quite well, and aren't too slow

if you want to do something big. However, if you're doing something a little outside of 'average', then you can get a big advantage by changing the nozzle.

The E3D Nozzle Fun Pack comes with a range of six different nozzles, from 0.25mm to 0.8mm. These let you pick the perfect size for your print, at least among those that are broadly compatible with most 3D printers. Speaking of compatibility, these nozzles are compatible with E3D hot-ends, including those used on official Prusa and LulzBot printers. The other popular standard is MK8, which does look quite similar, so double-check what your printer needs before ordering.

Nozzles do come a little smaller and larger than this range, but they're a bit more difficult to use. If you want to go smaller than 0.25mm, you'll probably find that you need to use specialist high-flow filaments. If you want to go larger, you'll find that on most

3D printers you won't get a speed-up as the hot-end won't be able to heat up plastic fast enough to use a larger nozzle size (there are specialist hot-ends that allow this, but they don't usually ship as standard on 3D printers). As well as heating, cooling is a problem with larger nozzles. More plastic needs more airflow to get it to solidify, so you may struggle with overhangs if you're pushing the limits with a large nozzle.

## SETTING UP

Replacing a nozzle is fairly simple, as they just unscrew and screw back in (a small spanner is included to help with this). You do need to be a little careful not to damage the threads





on the hot-end, so it's worth checking the instructions for your particular printer to see if there are any special steps to take. The biggest risk is working with hot metal, as you have to heat up your hot-end to unscrew the nozzle – make sure it doesn't drop on you or something it may damage!

As well as setting up the hardware, you need to configure the software. Your printer manufacturer may provide settings for some of these nozzles (Prusa, for example, provides settings for 0.25mm and 0.6mm nozzles in PrusaSlicer). You may be able to find some community-created ones online. However, in order to properly get the full advantage of your nozzles, you may find that you have to create some custom slicer settings. This isn't as daunting as it sounds – you can often just take an existing profile and tweak a few bits to get it working.

The advantage of having a range of nozzles is that, if one goes a little outside the range of your printer (for example, if 0.8mm is just a bit too much for your hot-end), or proves tricky to work with a particular filament type, then you can still get an advantage by dialling it back a notch and going for the next nozzle.

We've been using this nozzle set for a few months now, and found them really useful. The smaller nozzles have been great for miniatures such as game pieces. With these,

the speed isn't as critical because they're small anyway and if they're going to be used a lot, it's worth spending the time to make sure they're printed as beautifully as possible. The 0.6mm nozzle came in handy last month when we were working with wood-fill. The really large nozzles are useful when you need a lot of something quickly, such as when you're prototyping larger objects and want to see how they fit together, or when you're in the middle of a pandemic and 3D-printed face shields are useful, but a little slow to produce.

Perhaps the one downside to this is that the nozzles are all brass, so they will gradually wear out (particularly if you're using abrasive filaments, such as those with metal particles in them). That said, this is a good way of testing out how different nozzle sizes work with your printer, and if you're finding that you're using one particular nozzle size with abrasive filament, it might be worth replacing that with something like a hardened steel nozzle.

It's hard to think of a more cost-effective or easy way to upgrade your 3D printer than a nozzle set. For a fairly small outlay, you get both a big increase in resolution and a big increase in speed (though not at the same time, unfortunately). The E3D Nozzle Fun Pack has a good range of nozzles that should be straightforward to get working with a compatible printer. □

**“**  
The smaller nozzles have  
been great for miniatures  
such as game pieces

**Left** Both of these prints took about 20 minutes; the larger with a 0.8mm nozzle and the smaller with a 0.25mm nozzle. We did get some stringing with the smaller nozzle, but this is removable with the usual processes.

**Below** The kit comes with a spanner and a metal storage case



**VERDICT**  
Extend the range of your 3D printer.

**L 9/10**

# Ringo phone

Build your own phone

CIRCUITMESS ♦ £129.99 | [circuitmess.com](http://circuitmess.com)

By Ben Everard

 @ben\_everard

**Right** ♦  
The interface is easy to use, if a little retro in look and feel



The Ringo (formerly known as MAKERphone) is an open-source, DIY phone. It has an ESP32-based microcontroller board at its heart that can run Arduino or MicroPython code. This solders onto a mainboard that has

buttons, a screen, and a connector for either a 2G or 4G mobile phone module. These electronics bolt into a case of four bits of laser-cut acrylic to create a mobile phone. A slightly bulky phone, and one based on buttons rather than a touchscreen (as many modern phones are), but a phone nonetheless.

When you open the box, you're confronted with a set of circuit boards (with the surface-mount components already soldered on), a bag of components, and the bits for the case. It's up to you to put this together. You can also optionally get a tools kit that contains a USB-powered soldering iron, pliers, desoldering pump, and a few other bits.

The step-by-step instructions (available at [circuitmess.com/build](http://circuitmess.com/build)) take you through the whole process very easily. Since all the surface-mount components are in place, there's only through-hole soldering left to do, and it's all quite accessible. While a complete beginner might feel a little nervous starting out with an expensive kit like this, there aren't any bits that should cause problems, and everything is forgiving. The first step is to solder the headers onto the controller board, and this gives an easy introduction to soldering. You can also remove these without too much difficulty should you make a mistake (we can speak from experience here, as we failed to follow the instructions – a fault of our hubris, not the clear instructions – and soldered them on the wrong side of the board).

The exact model of speaker has changed slightly since the instructions were written, and is a little tight to get behind the screen, but if you loosen off the

bolts on the screen, it slots into place without much difficulty. One of the speaker wires broke on our model (CircuitMess tells us that it had a slight problem with a supplier), but it took just a quick drop of solder to fix.

## SOFTWARE

The phone software is reminiscent of early 2000s phones. The screen is colour, but the resolution is much lower than most modern phones (160×128 pixels). The joystick feels good under the thumb (it's analogue if you want to make use of this in your own software). The phone comes with a few basic apps for phone, SMS messaging, contacts, torch (or flashlight for our American friends), a music player (WAV files only), an image viewer, and there are three (quite addictive) games: Space Invaders, Pong, and Snake.

The real advantage of this phone is not the software it comes with, though: it's that you can make your own quite easily. There's support for MicroPython and Arduino, as well as CircuitBlocks which gives you a drag-and-drop Scratch-like interface to the phone's Arduino library. The firmware on the phone allows you to use these languages to create apps that can be loaded interactively and will be interrupted by phone calls and messages. As yet, there's not much of an ecosystem of apps available for this, but it's a new platform, and this may change in the coming months.

While the ESP32 does have WiFi and Bluetooth, the only default application that makes use of this is the firmware updater. Should you wish, you may be able to make use of these in your own applications.



## IN YOUR POCKET

While this is a fun project to make, the real test of any project is how it works once it's made. On the simplest level, this makes a perfectly good phone. You can make calls and send SMS messages easily (though you have to be old-school enough to remember how to type on a digit keypad, as the letters aren't printed on). We found that the phone battery lasted a couple of days with light use.

However, few of us use phones as phones these days. The Ringo phone doesn't compare to a modern iPhone or Android device. There's no web browser, no camera, no WhatsApp, etc. For some people, this will be an advantage, and for them, Ringo is an interesting option. It's distraction-free, so you won't be sucked into social media black holes, or feel you have to photograph every meal. However, at the same time, if you do find you need extra little features (and if you've got the programming skills), you can implement them yourself.

Phones are now the visible face of high technology and are often seen as black-box devices that hold enormous power over our everyday lives. The Ringo is a phone that we can control in almost any way we want, as a tool for helping to understand technology and our relationship with it. For some, it'll be a fun project and a curio; for others, it'll be the basis

of their personal communications that they retain control of. Either way, it's an entertaining kit to build and play with. □

**Above** The phone is built from these three circuit boards

**Below** In the basic build, there's a gap around the side of the phone that anything conductive could get into and cause a short, but a user on the forum ([circuitmess.com/community](http://circuitmess.com/community)) has designed a 3D-printable surround



## VERDICT

We had great fun putting this phone together. It's one of the better maker kits around, and could be a useful, customisable gadget.

**L** **10/10**

# Arduino Portenta

A powerful new pro-level microcontroller

**ARDUINO** ♦ €89.90 | [arduino.cc](http://arduino.cc)

By Ben Everard

 @ben\_everard

**F**or over a decade, Arduino has been synonymous with beginner-focused microcontrollers. Its easy-to-use hardware and software have made it easier for a generation of hobbyists and up-and-coming electrical engineers to get started with programmable hardware. However, in the last few years, we've seen products more aligned with the professional market. First, Arduino's MKR line of boards offered features targeted at industry, then the Pro version of its development environment launched with features for more advanced users. Now the Portenta is the latest in the line of offerings for users with heavier requirements.

At the heart of this new board is a beast of a microcontroller – an STM32H747 with two cores: a Cortex-M7 running at 480MHz, and a Cortex-M4 running at 240MHz. From a programming perspective, these two cores are treated separately, and each can run a different sketch. The slower of the two cores, on its own, would be more powerful than most hobbyist microcontrollers, but the M7 is really

quick (the Teensy 4 is the only faster microcontroller we've tested).

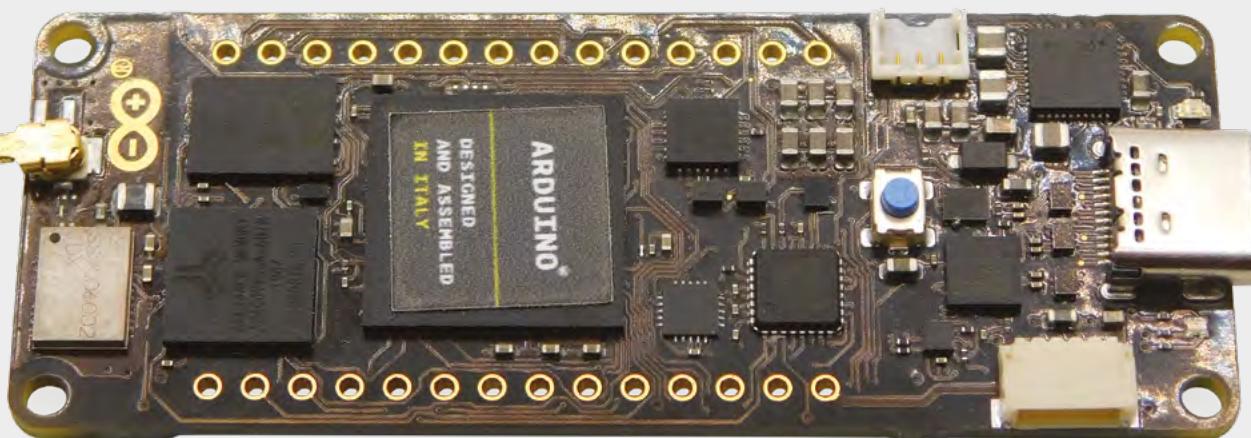
A little unusually for a microcontroller, this SoC also features a graphics engine and can output either over DisplayLink (on USB-C) or via Display Serial Interface (DSI). As well as the MKR-style pinouts, the board has two 80-pin, high-density connectors to give access

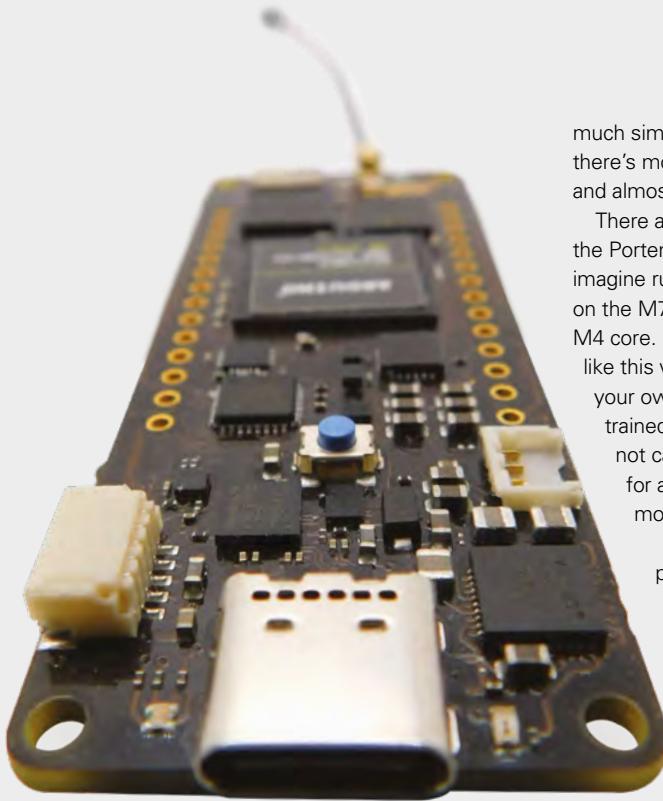
“  
There are definitely situations where something like the Portenta could work well”

**Below** You can order the Portenta with different memory and connectivity configurations, though prices and minimum order quantities are yet to be confirmed

to more of the chip's functions (including the DSI and a camera connector). Accessing these will require a carrier board – Arduino has released details for one, but as yet, no price has been announced.

There are a lot of features on this board. As well as those mentioned, there's connectivity with Ethernet,





WiFi and Bluetooth, 8MB RAM, 16MB flash, and an NXP SE050C2 crypto chip. All this comes with a price tag, and that's 89.90 euros, plus tax.

As you would expect, all this can be accessed from the Arduino IDE. Arduino has also promised Python and JavaScript support, but these weren't available at the time we tested out the unit.

The feature set and price point perhaps make this more comparable to Linux-based single-board computers than most other microcontrollers. Things like driving displays and running neural networks are more associated with more fully featured computers; however, there are some advantages to running them on microcontrollers – keeping the software stack

much simpler means there's less room for errors, there's more potential for fine-grained timing control, and almost instantaneous booting.

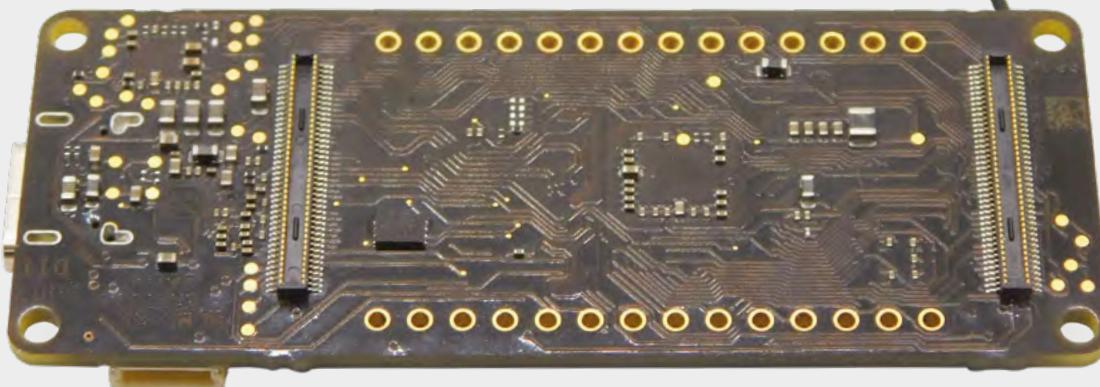
There are definitely situations where something like the Portenta could work well. For example, we could imagine running a neural network doing image analysis on the M7 core, while controlling hardware with the M4 core. However, the software side of something like this would be complex. Unless you want to train your own neural network, you're reliant on those trained by others. For an industrial user, this might not cause a problem, but it's quite an undertaking for a hobbyist to train and optimise their own model for any sort of complex AI.

If the Portenta (or similar boards) become popular, and there's a good source of neural networks trained and optimised for this hardware, then it could be an attractive option. Similarly with the hardware – Arduino describes the two 80-pin connectors as 'a new standard'. If they catch on, and there's a choice of carrier boards and different price points offering different features, they also could be very useful. If we're left with just the offering from Arduino – which has a lot of features, but again we're expecting a price tag to match – then you'll be looking at a significant cost, even if you just want to attach a single peripheral such as a camera. Again, for industrial customers, it might make sense to design your own carrier board, but it's a bit excessive to do this if you just want to access a camera on one project.

We'd love to be able to use the Portenta to its full potential, but until the ecosystem catches up to the board, it will be a significant undertaking to build something that does. □

**Left** The USB-C port supports display-link monitors for video output

**Below** The high-density connectors each give an additional 80 connections to the board



## VERDICT

A powerful board, but lack of ecosystem limits its ability to fulfil potential.

**L** 8 /10

The background of the entire page is a dense arrangement of various tropical leaves, including palm fronds and monstera leaves, set against a solid black background.

issue

#33

ON SALE  
23 JULY

# nature

Find out how makers are helping wildlife

---

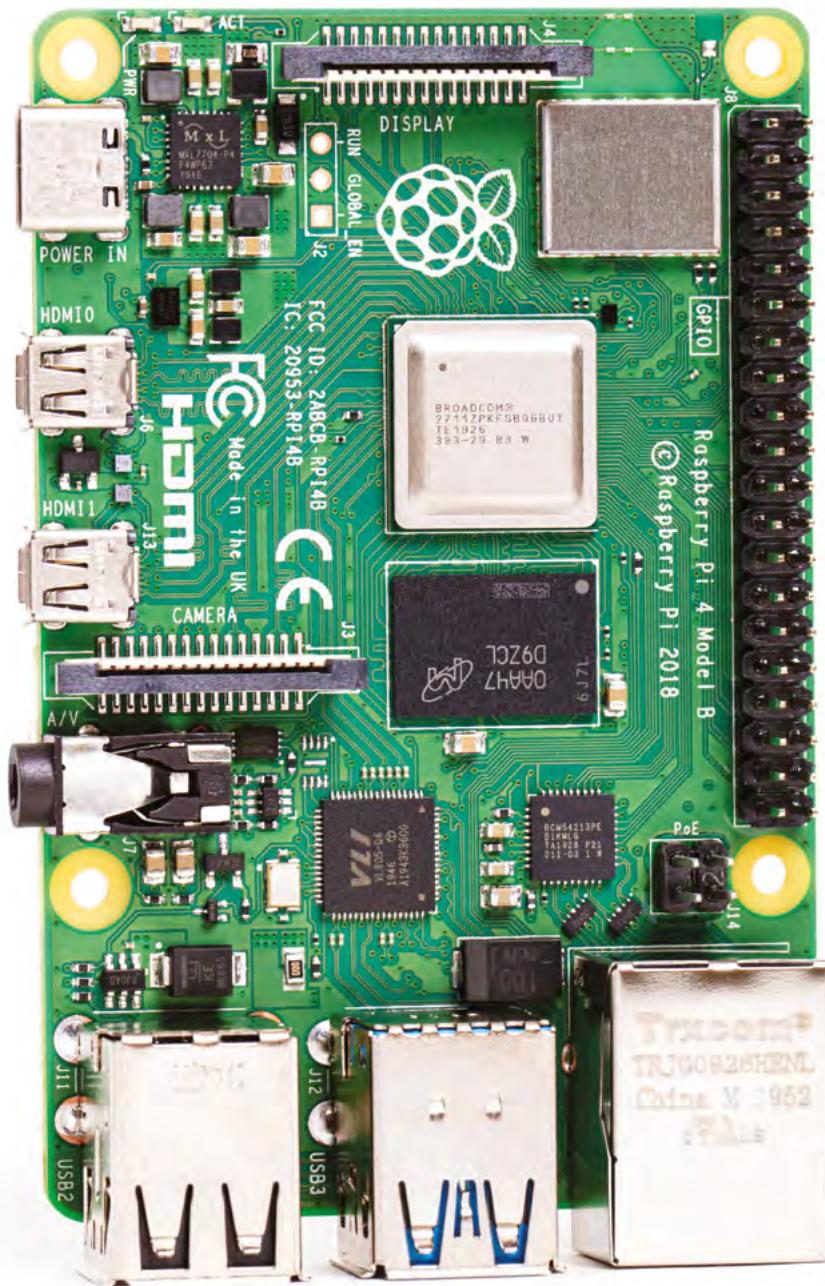
## ALSO

- Debugging microcontrollers
  - 3D design
  - Knitting
  - Music
  - And much more
- 

DON'T MISS OUT

[hsmag.cc/subscribe](http://hsmag.cc/subscribe)

# 8GB RASPBERRY PI 4



— hsmag.cc —

