PROJECTS FOR MAKERS & HACKERS

BUILD A FLAT-PACK ROCKET

MAKE ELECTRONIC MUSIC WITH A RASPBERRY PI PICO

BOOK OF MAKING

2025

MAKE A CONNECTED PLANT MONITOR

BUILD SMART HOME LIGHTING

& LOTS MORE

FROM THE MAKERS OF HackSpace MAGAZINE

Welcome to the Book of Making

Making is for everyone. From the cave paintings of the earliest humans to the satellites, cars, and computer networks of today, using tools to create has always been part of humanity. You might want to add heating to a chair, or make your own robotic cocktail machine, or build a machine that tells you when your house plants need watering. It doesn't matter what you want to build: the crucial thing is that you can do it.

We've gathered together our favourite projects previously published in HackSpace magazine, which we think show the breadth of what an ordinary person can do. Arm yourself with a Raspberry Pi or Raspberry Pi Pico, and get making!

ANDREW GREGORY
Editor andrew.gregory@raspberrypi.com

EDITORIAL
Editor
Andrew Gregory
andrew.gregory@raspberrypi.com

Sub Editors
David Higgs, Nicola King

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

DESIGN
Critical Media
criticalmedia.co.uk
Head of Design
Lee Allen
Designers
Sam Ribbits, Olivia Mitchell
Raspberry Pi Designers
Sara Parodi, Natalie Turner, Jack Willis
Illustrator
Sam Alder
Photographer
Brian O Halloran

CONTRIBUTORS
Ben Everard, Jo Hinchliffe, Rob Miles, Nicola King, Andrew Lewis, Phil King

PUBLISHING
Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com
Director of Communications
Helen Lynn
CEO
Eben Upton

DISTRIBUTION
Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS
Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
hsmag.cc/subscribe
hackspace@subscriptionhelpline.co.uk

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001. Book of Making 2025 is published by Raspberry Pi Ltd, 194 Cambridge Science Park, Milton Road, Cambridge, England, CB4 0AB.

ISBN: 9781916868366
UPC: 725274044351 45

Contents

Dragon egg

30

Felt + LEDs

12

Hands-free lighting

58

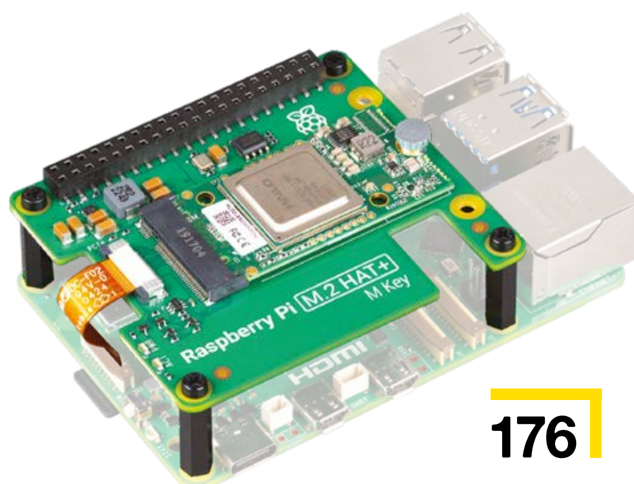
170

94 INSPIRATION

164 REVIEWS

Arduino cocktail machine

114



176

Some of the tools and techniques shown in this book 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. Book of Making is intended for an adult audience and some projects may be dangerous for children. Raspberry Pi Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in this book. Laws and regulations covering many of the topics in this book 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 this book may go beyond. It is your responsibility to understand the manufacturer's limits.

HackSpace
TECHNOLOGY IN YOUR HANDS

AFTERNOON PROJECTS

HACK | MAKE | BUILD | CREATE

When you have a couple of hours to yourself, these projects are quick to make – have fun!

PG 8

NFC TAGS

Make interactive greetings cards with short range wireless technology

8

PG 12

NEEDLE FELTING

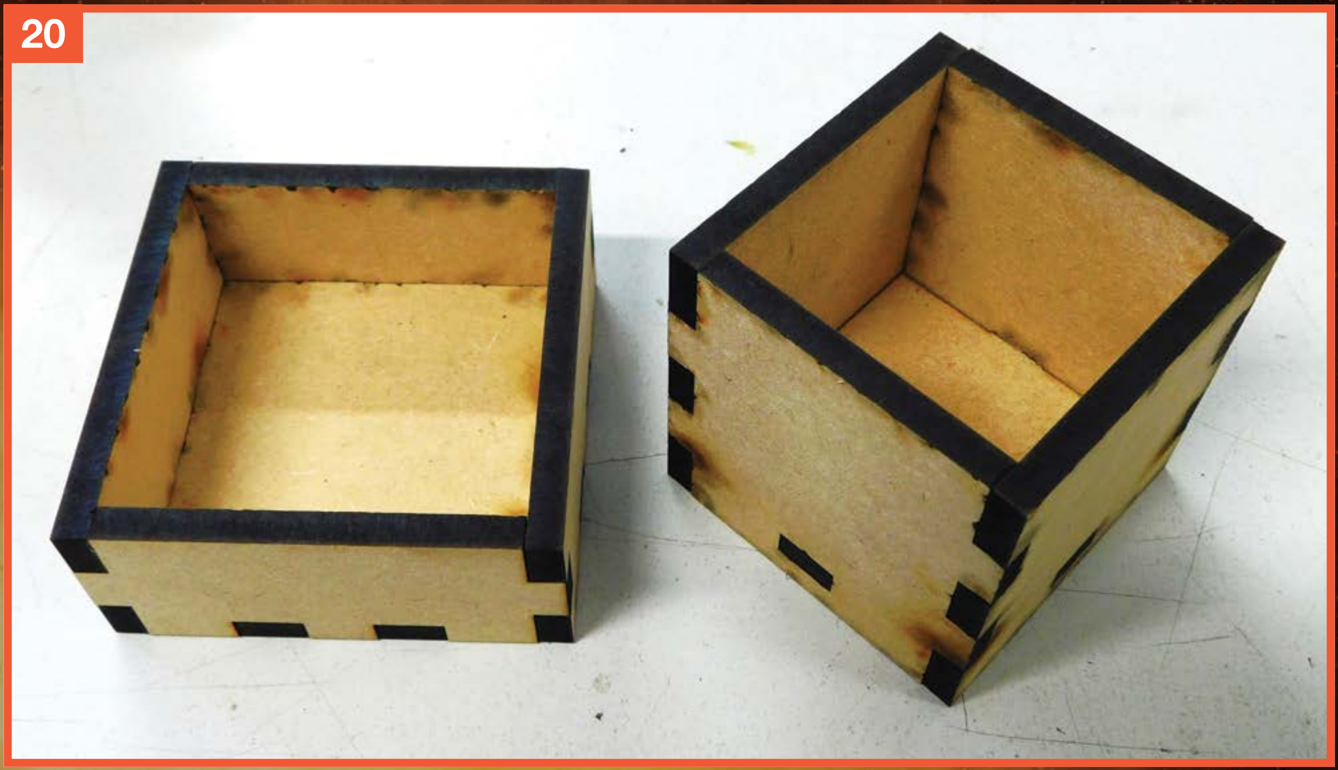
Sculpt 3D shapes in wool, then add LEDs. It's like 3D printing textiles!

PG 16

CUSTOM ENCLOSURES

Mix fabric with 3D printing to create bespoke cases and covers for your projects

16



PG 20

LASER CUT BOXES

We love the smell of laser cut plywood in the morning – and it makes pretty useful boxes too

PG 24

MACRAME

A beginner's guide to the 1970s art of tying knots to build 3D shapes

Make interactive greetings cards with NFC tags

Pep up that boring greetings card with a fun NFC tag. **Nicola King** explores the possibilities

Nicola King

@holtonhandmade

Nicola King is a freelance writer and sub-editor. She vows to spend less time searching for makery paraphernalia, and more time creating useful items from her (substantial) stash.

Above A selection of different NFC tags – when choosing which tag to use for a specific project, you'll need to consider memory requirements, read/write cycles, where it will be positioned, whether it needs to be waterproof, how it will be attached, and so on

YOU'LL NEED

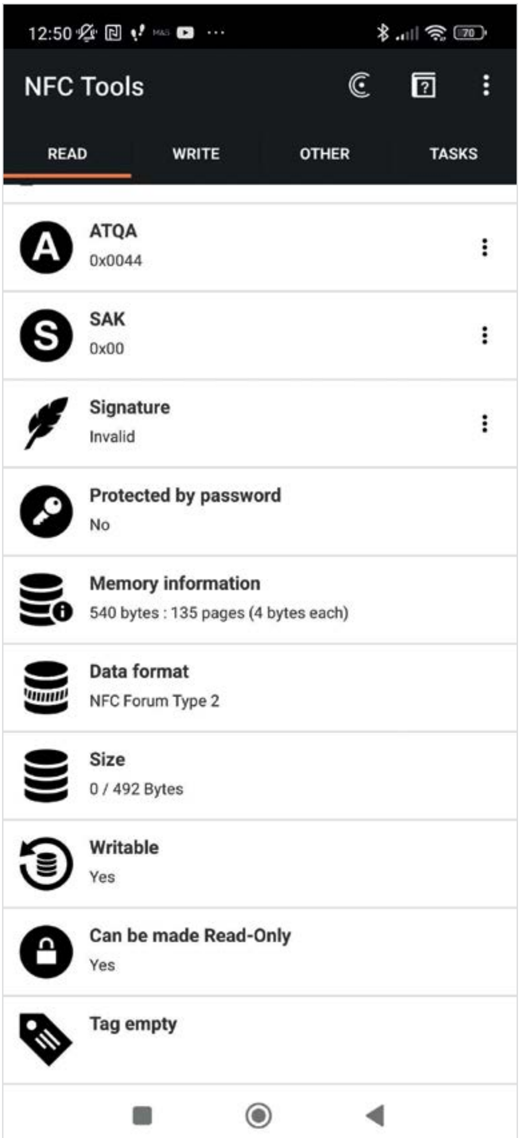
- **A ready-made greetings card, or card-making supplies**
- **An NFC tag,** e.g. hsmag.cc/TimeskeyNFC
- **An NFC-enabled smartphone**
- **An NFC app downloaded to your phone** (such as NFC Tools)
- **A pen**

There are lots of fun ways of using near-field communication (NFC) in our day-to-day lives. In this tutorial, we will be taking a humble greetings card and hiding an NFC tag inside it in order to gift an unconventional digital surprise. The best part is that you need very little in terms of 'equipment' to get going, it takes next to no time to set up, and you really don't need to be a tech whizz either, so let's jump right in.

STEP 1 ASSEMBLE YOUR SUPPLIES

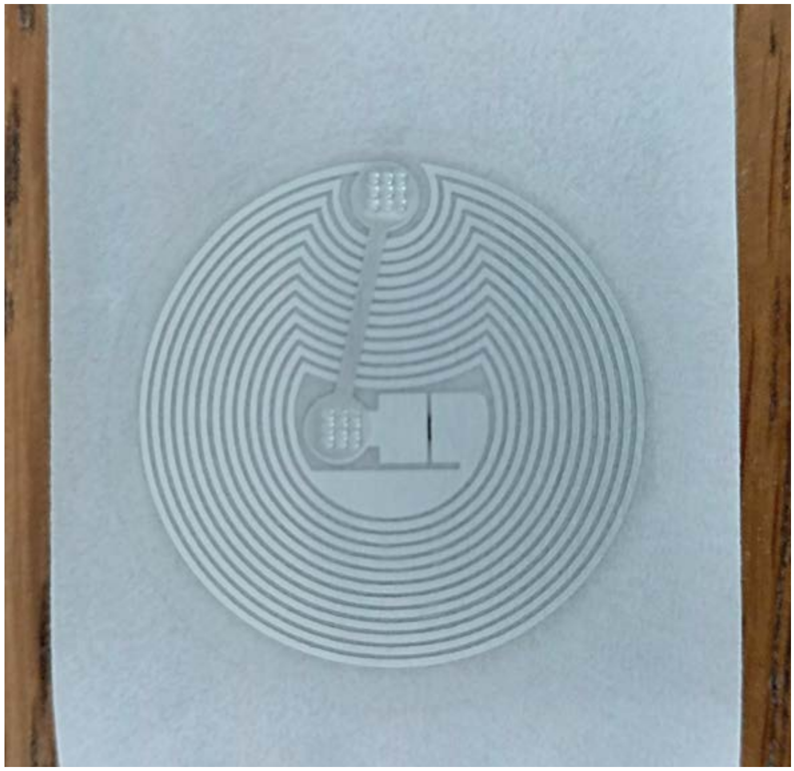
Let's firstly gather together what we need for this project. In the interests of speed, we've used a ready-made greetings card for this endeavour but, if you have the time, you could really push the boat out and make your own card for that extra handmade touch, whether that's using a digital cutting machine such as a Cricut which can cut out your chosen design, or just purchasing some blank cardstock and card-topping ephemera from a craft store for you to piece together in an artful manner – the choice really is yours.

Next, let's talk about NFC tags, and the many different options available. These unpowered tags come in various shapes and sizes, but are generally on the small side, fitting in the palm of your hand, and they are usually super-cheap to buy. We picked up a handy selection pack from Amazon which



Left The Read tab of the NFC Tools app gives you the details of the scanned tag. As you can see, this one is empty of data

contained some black and clear-coloured coin-shaped sticker tags around 2.5 cm in diameter, some similar coin-shaped NFC cards, some mini NFC cards, some credit card-sized transparent NFC cards, and five cool-looking epoxy NFC keychains. In all, the £15 pack contained 30 pieces, which offers a lot of flexibility for different NFC-related projects. Given that we are working with a greetings card, we chose a thin, light sticker tag for this venture as it will fit very neatly inside the card. The beauty of the sticker NFC tags is their simplicity and the fact that you can stick them virtually anywhere, so they can be used in places that other technologies can't.



Above A close-up of a sticker tag, showing the coiled circuit which will take its power from the nearby smartphone or other device

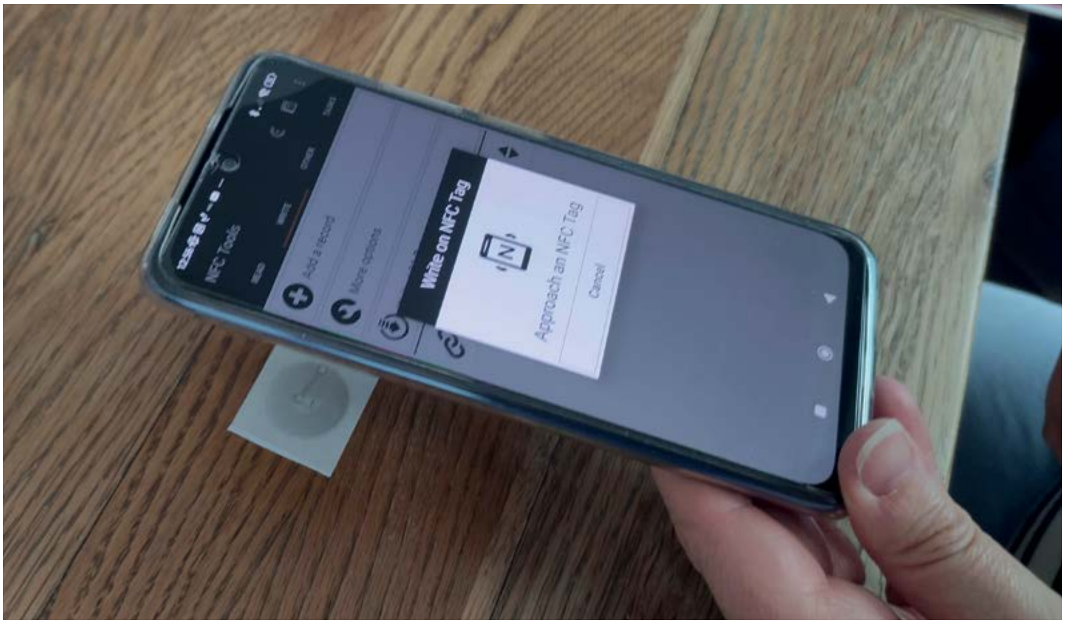
NFC tags vary in terms of their memory capacity and read speeds. The tags that we bought were all described as NTAG215, which means they have a medium-sized memory capacity (504 bytes). You can purchase tags with a bigger memory, such as those labelled NTAG216, which hold 888 bytes – it really depends on your specific needs. It's worth bearing in mind that tags with less memory will generally be

> "You could really push the boat out and make your **own card for that extra handmade touch**"

slightly less expensive. It's also worth pointing out that you can rewrite many NFC tags, which is clearly a carbon footprint-reducing route to take. So, check this out before you purchase.

Now, let's turn to our smartphone. The first thing we need to do is ensure that it is NFC-enabled. On our Android phone, we went into Settings, then into a sub-section called 'Connection and sharing', under which we found 'NFC (allow data exchange when this device touches another one)' switched on. Every phone is different of course, but this setting should be easy to find – on iPhone, it should be in the Control Centre.

Next, we need to download an appropriate NFC tag writer app that will enable us to program our →



Right After selecting the Write option, hold the phone near to the NFC tag to write the selected records to it

Below We stuck an NFC tag to our greetings card with a helpful message for the recipient

tag. There are a number of suitable apps available, including NFC Tools, NFC Tag Reader, NFC TagWriter, and others. We chose to download NFC Tools, a popular and easy-to-use app which allows you to read, write, and program tasks on NFC tags and other compatible chips.

QUICK TIP

Try not to bend NFC tags too much, especially the very thin ones – if the antenna breaks, the information stored on it is unlikely to be recoverable.

STEP 2 PROGRAMMING YOUR TAG

Once you have the app installed and launched, go to the 'Read' tab. Put your tag on the table and hold your phone over it. All of the technical data regarding the tag should appear on your phone and, as ours is a new, blank tag, it is empty of data as we've obviously not programmed anything yet.

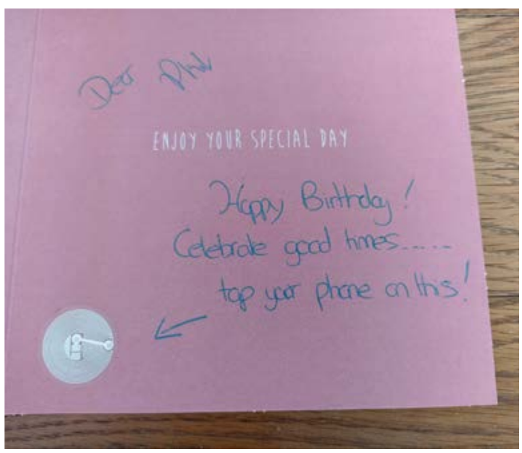
WHAT IS NEAR-FIELD COMMUNICATION?

NFC is a technology that has gained momentum in recent years, and is a set of communication protocols that enables wireless communication (sharing data, media, and more) between two electronic devices that are in close proximity to each other. Its history is rooted in radio frequency identification technology (RFID), and NFC is really an evolution from RFID tech, with both NFC and RFID having distinct use cases. NFC technology is used in things like credit cards, ID cards, transport passes, car keys, access badges, hotel access keys… to name but a few.

NFC tags are passive devices with no power of their own, and consist of a thin copper coil (antenna) and a microchip, which is used to store very small amounts of data, such as a URL or a password, for example. These tags operate at 13.56MHz. The coil allows the tag to wirelessly receive power from the NFC reader (in this case, a smartphone) through electromagnetic induction. NFC tags are activated directly by the magnetic field of the NFC sensor of the smartphone/other device that reads them. So, when the smartphone comes close to the tag, the tag is instantly energised and transmits the stored data in its microchip to the smartphone.

Officially, the two electronic devices that are communicating with each other should be about 1.5 inches (3.8 cm) apart, but they can be up to 4 inches (10 cm) from each other in practice. In our simple tutorial, we have paired an NFC-enabled smartphone with an NFC sticker (a very popular and practical form of NFC tag) which gets its power from the phone.

We are sending a birthday card and, to our tag, we want to add a link to a music video (Kool and the Gang singing *Celebration*) from YouTube to wish the recipient a happy birthday. You could, alternatively, send them a link to a digital voucher, if you are feeling generous. The choice is yours, but the point is that we are going to add a URL to the tag in this next step. Go to the 'Write' tab and select 'Add a record', and then select 'URL – Add a URL record'. Click in the URL box, and paste in the link. Press OK. Then select 'Write/XX bytes' (the XX relates to the number of bytes and will vary). The app will then invite you to approach your NFC tag with your phone to write that URL to the tag. We found that we had to put the top of our phone very close to the tag, but it worked and we received a 'Write Complete' message. Click on OK, and you have successfully written a URL to the tag. If you want to test it, just hold your phone close to the tag and the page should open.



STEP 3 TAP TO FINISH

As already mentioned, we used a sticker NFC tag, peeled it off its backing and placed it inside the card, with a note asking the recipient to point their phone at it. Note that when the recipient taps their phone on the tag, the link should open, but they may be first asked which browser they want to use if no default browser is already set on their phone. It's also worth noting that you can add multiple records to a tag, so we also added a text message.

There are many other options too – for example, you could add details of your Wi-Fi router to a tag, so that any visitors to your home/office can just tap it to gain access. You can add a link to videos, social networks, phone numbers. Have some fun playing with NFC tags – for example, we wanted to know the best route to a local garden centre. Selecting 'Add a record' in the Write tab of NFC Tools, we chose 'Open a destination address'. Next, we input the name of the garden centre, selected 'Write/ XX bytes', wrote it to the tag as before and, when we read the tag, were presented with the best route in Google Maps. You can also set passwords for the tag, lock it, erase data, and so on.

> "Have some fun playing with NFC tags – we wanted **to know the best route to a local garden centre**"

Additional options are available in the Tasks tab when you download the NFC Tasks companion app and grant it permissions. These include the ability to alter your own phone's controls, e.g. muting the volume, as well as setting timers and alarms.

It's even possible to use NFC tags to turn on lights, unlock doors, and perform other actions in your smart home setup. There are various ways of setting this up, such as using iPhone Shortcuts or sending custom URLs to an app like Home Assistant to trigger an automation.

So, what are you waiting for? Get yourself some NFC tags, download an NFC app to your smartphone, and see where it takes you. ◻

Left The NFC tag in our birthday card links to a YouTube video of Kool and the Gang singing *Celebration*

Below Some of the many record options available in the NFC Tools app

OTHER IDEAS FOR HOW TO USE NFC TAGS

We can literally only skim the surface with regards to what you can use NFC tags for. The possibilities are numerous, and we'd recommend you read around the subject a little to get some inspiration, but here are a few of our favourite suggestions:

- Make your own virtual business card – cardstock business cards are so old hat! Just program your contact information, social media details, website etc. onto your tag and away you go. The credit-card-sized NFC tags would be perfect for this. This handy explanatory video is well worth a watch: **hsmag.cc/NFCBusinessCard**.

- If you want smart home automation and some handy life hacks, NFC tags are just the ticket. You can use them to help with tasks around your home such as turning lights on and off, setting a kitchen timer, setting an alarm etc. You could even automate your grocery list – stick a tag to the side of the bread bin, for example, and scan it when you are out of bread so that 'bread' gets added to your shopping list. Take these ideas and just run with them! Try this instructional link out for more information: **hsmag.cc/NFCSmartHome**.

- Send secret messages – children might like this one. Just program your tag with your message and hide it for someone to find. Amusing more than practical…

- Lessen the chance of lost PE kits – if you have offspring who are prone to losing their PE kit (voice of experience speaking here!), then why not program their name, the name of the house they are in at school, and any other pertinent details into a waterproof and durable key fob NFC tag and attach it to the PE kit? Unlike a Bluetooth tag, it lacks the range to help you find it yourself, but it might improve the chances of it getting returned if someone scans the tag with their phone.

- Having a party? Add an NFC tag to the invitation so all the info is there at the tap of a smartphone, including directions. You get the idea… you're really only constrained by your imagination.

Needle felting: Learn to sculpt 3D shapes with wool (adding a little LED bling)

Piercing wool fibres with needles to form aesthetically appealing shapes is an easy, inexpensive, and highly therapeutic craft

Nicola King

@holtonhandmade

Nicola King is a freelance writer and sub-editor. Crafting and making stuff generally helps to keep her grounded and semi-sane…just ask her long-suffering family.

The increasingly popular craft of needle felting, also known as dry felting, is the art of transforming wool fibres into, usually, 2D or 3D shapes, with the aid of some barbed needles, and this author is happy to announce that needle felting is her new obsession. She'd read that it was an addictive hobby, but little did she realise just how captivated she'd become. At the end of her first project, she was already looking for potential needle felting accessories that her family could gift her for Christmas… we're just pre-warning you that if you try this craft once, you'll likely be back for more. It's a hobby that only requires a few materials to get started, and these are easily purchased online or from a craft store. It's also inexpensive, and very uncomplicated to get to grips with in terms of skill level. As well as that, it's extremely therapeutic… there's just something about stabbing wool endlessly with needles that is so very calming!

So, in this tutorial, we are going to needle-felt a 3D shape. And to give it that HackSpace twist that we know you all crave, we are also going to add a simple circuit so that we have some LED lights adorning our newly felted make. Light-emitting diodes undoubtedly look good on anything.

Figure 1 First attempt, and we're pretty pleased. A Christmas decoration that we hot-glued some trimmings to. It's surprisingly easy to felt the trunk to the tree, just keep stabbing!

FELTING FUNDAMENTALS

The process of needle felting involves the repetitive stabbing, or piercing, of wool fibres – this means that the needles, and specifically the barbs or notches on the needles, grab the top layer of fibre and then tangle or knit this layer with inner layers of fibres. Usually the needle notches are facing downwards, so they don't pull the fibres out as the needle exits the wool. Therefore, the wool, which progresses to a solid felt, becomes stronger after the fibres are compressed and tangled, and forms more solid shapes. The more you stab it, the more the fibres slowly matt together.

Figure 1 shows this author's very first stab (!) at needle-felting a 3D shape, a Christmas decoration, and it turned out pretty well. After some time piercing away, the fibres eventually started to condense, and the piece began to take its conical shape, with the texture and density firming up as felting progressed.

There are many types of wool and needles that can be used for needle felting (see 'Types of felting

TYPES OF FELTING WOOL/NEEDLES

WOOL FIBRES

There is a daunting array of wools available to felt with, some more fine than others, some more coarse, and newbies may be slightly bamboozled by the choice, so here is a very quick guide to a few of the options:

- **Batts** – this is wool that has been washed, teased out, and carded into batts, which is like a sheet of wool. Great for needle felting
- **Raw/unwashed wool** – straight off the sheep but, as it's not been washed, you'd need to wash your hands regularly or wash it yourself before use
- **Cleaned locks** – wool that's been washed, but not combed, so it's great for adding effects to a finished piece
- **Tops/roving** – washed and combed wool, and the fibres are all going in one direction. Perfect for felting and easy to find. Natural wool roving is what we've used in our tutorial project, and it comes in many different colours, so there is lots of choice

Different fibres and wools will create different looks – just experiment!

NEEDLES

Again, a huge range to choose from. One thing to note is that felting needles are very delicate and can break easily if you don't hold them and stab them at a 90-degree angle, which minimises the chances of breakage. Needles come in different gauges, with the gauge referring to the diameter of the needle – the higher the number, the finer the gauge. Usually you would start a project with a lower gauge to do the bulk work and the shaping of your piece, and then move onto a finer needle for the surface and detail element.

Felting needles tend to fall within the 32–42 gauge bracket and, from what we've learnt so far, a 38-gauge is a great all-rounder. A few examples of varieties of needle include:

- **Triangle-shaped** – three-sided blade with downward barbs on all three sides
- **Twisted** – spiral blade with notches that point downwards and twist around the needle
- **Reverse** – barbs on these point the other way, upwards, so fluffy finishes can be created

You can also buy needles in a pen form, which makes them easier to hold (**hsmag.cc/PenFeltingTool**).

wool/needles' box for more information). You can buy needle felting kits (from an array of craft kit purveyors) which come with everything you need to get you started; we purchased a couple of these to practise with. One came with a plaster, which we found hilarious and alarming in equal measure – needle felting obviously involves working with sharp needles but, as long as you are careful, a plaster shouldn't be required (at least, we've not needed one yet).

Do be aware that some kits come with polystyrene shapes to mould your felted shape around. Adding polystyrene to the mix undoubtedly speeds things up enormously and saves on wool fibre usage. However, be conscious of the fact that polystyrene is slow to degrade and not particularly environmentally friendly, so we'd recommend that you avoid kits that include it.

Finally, before we get started, it's worth noting that your piece of needle felting is unlikely to actually resemble what you are aiming to make until you are around three quarters of the way into your project. Be patient, as it will come together in the end. After quite a lot of stabbing and piercing, this author was not at all confident that she possessed any skill whatsoever in the needle felting department but, after some persistent needle action, her faith was restored and she could see it finally taking shape. The moral of this tale being: keep at it!

Above
Our completed and illuminated toadstool in its natural habitat. (Seasonal elf styling is entirely optional)

FUNGI FELTING

So, we have chosen to create a needle-felted toadstool because, frankly, who doesn't want one of those in their life? You can, of course, create absolutely any shape you like with any colours you wish to use and, if you want to add some luminosity as we're doing, how about creating a felted car, tree, or house? With a little practice, you'll soon see how intuitive this craft is, and you'll be felting all sorts of shapes in no time. →

TAKE CARE

Needle felting is definitely *not* a craft for young children due to the sharpness of the tools involved, (but fine for anyone aged ten or above), and you should keep the needles away from your pets too. Mind those fingers, and remember to store your needles safely out of harm's way when not in use.

YOU'LL NEED

- **Wool fibres** (e.g. carded wool, wool batts, merino wool, tops, or roving)
- **Felting needles**
- **Needle holder** (optional)
- **Thick foam pad** (hacker's economical alternative: a car washing sponge)
- **Finger guards** (optional)
- **Cookie cutters** (optional)
- **Hot glue gun and sticks** (optional)
- **Embellishments** (optional)
- **Sewing needle** (for creating the circuit with thread)
- **Conductive thread** (hsmag.cc/ConductiveThread)
- **CR2032 battery holder with switch** (hsmag.cc/SewBatteryHolder)
- **CR2032 battery**
- **2 × (or more) easy-sew white-coloured LEDs** (hsmag.cc/LEDlights)
- **Scissors** (for tidying up stray wisps of fibre)
- **Plasters** (in unlikely case of mishap – optional)

Figure 2 The stem in its raw, unfelted form, just rolled into the vague shape that we are looking for – it's about to get the needle!

STEP 1 GIVE IT SOME NEEDLE

You need to be sitting down, ideally with a flat, clear work surface in front of you – the kitchen table would be perfect. Make sure you have your foam pad in place, in order to protect your surface, and arrange all of your materials and tools so that you can easily and safely access them.

We began by creating the base or stem of our toadstool and we used some cream/off-white wool roving, pulling off a small amount and gently shaping it with our fingers into a rough stem shape (**Figure 2**). To give you an idea of size, our stem measures around 6 cm in height, and is about 4 cm in diameter. We wanted to have the option of possibly fitting the battery holder to the base of the toadstool – in the end, we fitted it under the cap closer to where we wanted our LEDs, but the choice is yours. Just make sure that you make the base wide enough if you want to attach the holder there.

QUICK TIP

Remember that the shape that you end up making will be smaller than the shape you started with, and this is normal – it will be around 30 to 50% smaller. As you felt, you are condensing the fibres.

Once you are happy with the rough size and shape, you can begin felting. Use your needle at a 90-degree angle to minimise the chance of needle breakage (as they are very delicate tools) and gently stab down into the fibre, using only the lower section of the needle. Note that you should not be pushing the needle all the way down into the foam pad. Keep your wrist straight when stabbing, and gently move the piece round, so that you are not stabbing in only one place. Continue this light piercing motion and you will find that the more you stab, the tighter and thicker the wool gets. As you agitate the fibres, they bond together. You want to get to a point where the stem of the toadstool feels quite firm. If you need to add more wool to shape it, just pull some off and start felting it into the shape. This process took us around half an hour, but time flew!

> "**Keep your wrist straight when stabbing,** and gently move the piece round"

STEP 2 CAP IN HAND

Next, we created the cap of the toadstool. Using our red-coloured fibre, we created a vaguely round shape, placed it on the foam pad, and started to work the fibre with our needle. Take your time: this is a walk, not a sprint, so don't panic that you're spending too long on it, just give it as much time as it needs.

As we wanted to fix the battery holder under the cap, and to achieve that authentic toadstool cap shape, we created a bowl-type form, so that the underside had plenty of room to hide the battery holder (see **Figure 3**). Again, keep working on the shape until it feels firm and cap-shaped. Add more fibre around the sides to form that brim if you need to – sculpt it with your needles until you are satisfied.

When you are content with the form of the cap of the toadstool, you can then add those final touches to make it look more convincing – some little white spots. Using the cream fibre, we rolled some tiny balls, (you don't need very much at all!), and gently felted them into the top of the cap. Use a finer needle, if you have one, as this is more delicate work, and remember to keep fingers safe as this surface work can be very fiddly.

STEP 3 CIRCUIT-MAKER

So far, so good – if nothing else, you've learnt how to needle-felt a 3D shape. Next, we added a

COOKIE CUTTER CREATIONS

If you are new to this craft, using a cookie cutter as a shape guide is a great idea. Whatever you have in the kitchen drawer, cookie cutter-wise, can be utilised to get you going on your felting journey.

Choose your cutter, take some wool roving, or whatever you are using to felt with, and fill the cutter with your fibre. Using the instructions we've already covered, begin felting. When you remove the shape from the cutter, you'll need to finesse the edges and do some shaping work to refine the shape. Use your needles to create definition around corners for example, and use scissors to trim away stray fibres.

If you have small star or heart shapes, you could string some up to make bunting or, for individual shapes, maybe add a loop for hanging with some thread, per our Christmas tree decoration. You can also attach your needle-felted shapes to brooch backs, hair accessories or, if you use less roving and create a flatter shape, you can make appliqués that can be attached to other items, such as bags.

little radiance to our creation. The first step was to attach the battery holder, and we decided to sew it underneath the cap. As the holder will only be a short distance from the sewable LEDs, we only needed a short length of conductive thread, so we first cut a 15 cm length, threaded our needle and knotted the end. We then attached the thread with a stitch into the base of the cap. Next, we sewed into one of the positive holes on the holder a couple of times (**Figure 4**), then pushed the needle up to one of the white spots on top of the toadstool. Then, we took one of the LEDs and sewed through the positive sewable side of the LED, again, a couple of times. We then took the needle down to the underside of the cap and unthreaded the needle, letting the short length of conductive thread drop down.

Next, we took another 15 cm length and did the same again on the negative side of the holder, sewing through the negative hole, taking the thread up to the same LED, but this time sewing through the hole on the negative side of the LED. It's then just a question of doing the same thing for the other LED, as we are using just two of them in parallel, using the two negative and two positive holes on the battery holder. (Note that you could use more LEDs if you wished and set them up in a series, linking one to the next with thread, negative to positive.)

Finally, we inserted the battery into the battery holder. Our holder has an on/off switch, so we switched it to 'on' and the two LEDs lit up beautifully. A simple circuit that just adds an extra touch. If your LEDs don't light up, check for an easily rectifiable short-circuit – for example, the ends of the conductive threads could be touching/crossing each other, or they could be touching the battery.

Figure 3
Stabbing the cap into shape. Use whatever colours you want or have to hand. You can even mix colours by laying them on top of each other while felting, or by manually mixing the fibres before you start the felting process

Figure 4
Our concealed battery holder sewn into place. Use a reasonably fine sewing needle to do this, as it needs to fit through the holes on either side of the holder when the holder is being sewn in

COMPULSIVE CRAFTING

Needle felting is a very easy-to-learn textile craft. It's portable, rewarding, and a great way of making personalised gifts for others. Compared to something like knitting or other fibre arts, you can actually produce a finished item very quickly. The sound of the needle crunching into the wool is one of the very pleasing aspects of the craft, and it's extremely tactile too. As with every skill, the more you practise, the more you will learn and, as you progress, you can incorporate things like wire into your work to help shape it and create some amazing pieces with real character. If you are looking for a way to escape the mundane, needle felting is totally absorbing. □

QUICK TIP

Start with less wool than you think you will need – it is much easier to add more, and meld it into your creation, than it is to take some away if you've used too much to start with.

I WANT TO KNOW MORE!

Here are a few information/product sources to get you going:

- There are numerous instructive books on the subject of needle felting, which you can either purchase or borrow from your local library. *Needle Felting for Beginners* (2020) by Dace and Balchin, or *Making Simple Needle Felts* (2018) by Stern are both educational options.
- If you want to add some LED bling to your needle-felted project, Light Stitches (**lightstitches.co.uk**) specialises in supplying electronic textile kits and components, and sells some needle felting e-textile kits too. It also has conductive thread, battery holders, LED lights, and more, which are useful for all sorts of HackSpace-influenced projects. Also, The Felt Box (**thefeltbox.uk**) offers a plethora of needle-felting goodness, including monthly surprise boxes if you want a treat!
- There is a huge range of inexpensive kits available to buy, and they make great gifts for someone who you think will enjoy some repetitive fibre stabbing, maybe as a stress-buster! Try this one for a start-off: **hsmag.cc/UltimateKit**.
- Of course, YouTube is brimming with helpful 'this is how you needle felt' videos, and they are well worth a watch for tips, guidance, and general inspirational ideas. Here's a useful link to help steer your felting techniques in the right direction: **hsmag.cc/AvoidMistakes**. These sheep baubles (**hsmag.cc/SheepBaubles)** are adorable, too – ewe have to give them a go!

TUTORIAL

Cases and covers with a 3D printer

Make cases and covers by 'non-sew sewing'

Rob Miles

@robmiles

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

It is a truth universally acknowledged that someone in possession of a bit of tech will want some cases and covers for it. If only to hide what you've bought from the rest of the household. But some things don't come with cases or covers, especially if they are things you have made from scratch. In this article, we'll explore how we can use 3D-printed components as stiffening and connecting elements for cases and covers.

NON-SEW SEWING

Sewing is great. It is one of the many things that the author can't do very well. Although he is the only person in the household who can operate the buttonhole maker on the sewing machine. But sewing thicker materials is hard work. And it involves sharp pointy needles. But the author is quite at home with nuts and bolts and 3D printing, so he prefers to use those skills instead of spending ages trying to thread a needle.

CUSTOM COVERS

Nothing keeps sunlight and dust off your precious devices like a cover. But they can be expensive to buy – even assuming you can find one that fits. So, why not make your own? You can buy leatherette material in a variety of colours at pleasingly low prices. And it can be bought in a width of 1.5 metres (just under 5 feet), so you can make covers for large as well as small devices. First, you cut to size, then cut out the corners, and with four seams down each corner, you

Figure 1
You can use dome bolts for a rounder finish

YOU'LL NEED

- **Leatherette (PU leather)** or cloth for the outer covers
- **Felt or thin cloth for the inner cover**
- **Spray adhesive**
- **A hole punch**
- **A sharp knife**
- **M3 bolts 4 mm long** – dome or countersunk
- **3D printer and filament**

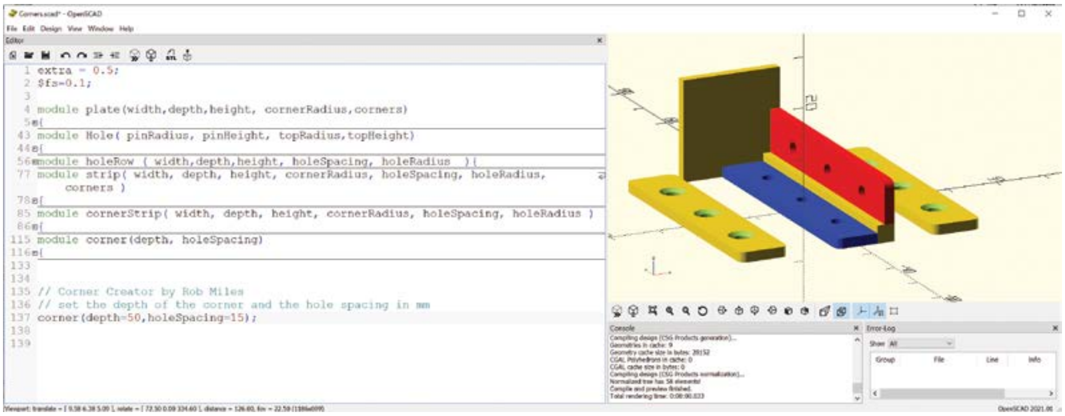


Figure 2 The red and blue elements were given colours to make it easier to see which is which when creating the design

Below You can also use felt as a liner for the inside of the case, but the author rather liked the bulb material

can make a snug-fitting cover. Stitching leatherette is not too difficult, and you can use a hole punch to make the holes if needed. But it can be hard to make your stitches consistent, and the corners that you make have no rigidity. So why not print some corners of your own and then clamp the material into them with bolts?

Figure 1 shows what we are making. The corner support is underneath the leatherette and the bolts screw straight into it, through the strip. The author has found that M3 bolts will screw into 3D-printed elements and grip tightly. A hole radius of about 1.45 mm seems to work well on an Ender-3 printer.

Figure 2 shows the designs for the clips for the edges of the covers. They were produced using OpenSCAD. You can find the designs on the GitHub site for this article: **hsmag.cc/covers-and-cases**. You can download the OpenSCAD application from: **openscad.org**. You can create corners of different lengths and hole spacings by changing the call to the corner function in the program:

```
corner(depth=50,holeSpacing=15);
```

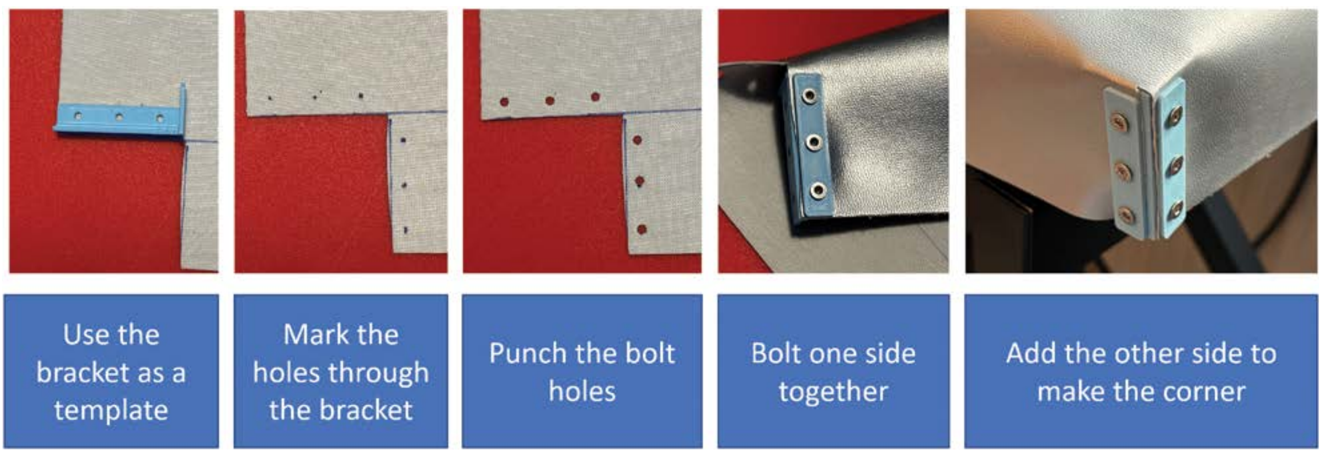
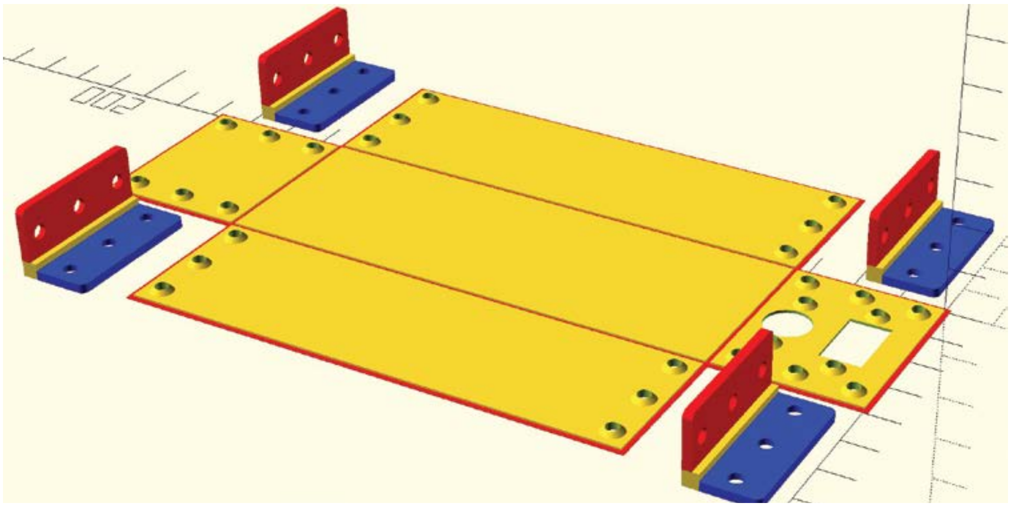


Figure 3 The holes in the plates have a larger 'countersink' hole at one side. Make sure you screw the bolt into this larger hole so that the head is slightly recessed



The OpenSCAD program will make the designs for your corner and the brackets. These can be exported as an STL file to be sliced and printed.

Figure 3 shows how you add a corner to a cover. Bolt one side followed by the other. The bolts go through the cover plate, then the fabric, and finally into the bracket. If you want to make something super-strong, you can use longer bolts and put a nut on the end. Cover the corner fitting with insulating tape on the inside to stop the corner support from damaging the edges of the thing it is covering.

A CASE FOR TREATMENT

The party camera was featured in HackSpace issue 57. It takes pictures and sends them to a thermal printer so you can make a printed record of all your party fun. The top surface of the camera exposes a PIR sensor, a camera, an OLED display, and a trigger button. The bottom of the camera contains holes for a USB cable to power the device, and a tripod socket to support it. The author thought it might benefit from a case that just enclosed the bottom of the camera. The case would be lined with cloth and covered in leatherette. It was decided to use the same 'bolt-based' corners as were used for the cover just described, but this time, the corners would be fitted to the outside of the case and be bolted into 3D-printed stiffening panels that made up the case body. This makes for a nice steampunk aesthetic, although the author has probably used more bolts than he needed to. The corners would fit perfectly well with just two bolts.

Figure 4 shows the 3D-printed components for the party camera case. The stiffening material uses one-layer thick 'connecting' elements shown in red, and three-layer thick yellow panels to provide the stiffening. The bolt holes in the panels are surrounded by extra layers to allow them to grip the bolts from the corners. When the case is constructed, the sides are folded up and the corners bolted around the outside to hold them together.

The printable components are created by an OpenSCAD program that accepts the width, depth, and height values and then produces the design for the stiffening materials and the corner pieces. The program will also generate a lid for the case, but this is not needed for the party camera. Extra code was added to cut holes in the end panel for the USB connector and the tripod supports. The code for this is available at the GitHub site for this article: **hsmag.cc/covers-and-cases**.

PAPER TEMPLATES

The best way to get the shape of your cover (particularly if you are covering something which has an angled top) is to make a paper template first and then transfer the shape onto the material you are going to use for the finished cover. Paper is useful because you can always stick an extra sheet on if it turns out you've just cut the wrong part off.

Figure 4
This version of the case design has bolt supports for covers on the end panel for the USB and tripod holes. The cover wasn't added to the finished case because it didn't seem to need them

Figure 5
You can use a 1/8-inch drill bit to cut holes in the outer for the corner fixing bolts. Alternatively, you can cut the outer cover material to shape first and punch holes in it before sticking it onto the stiffener

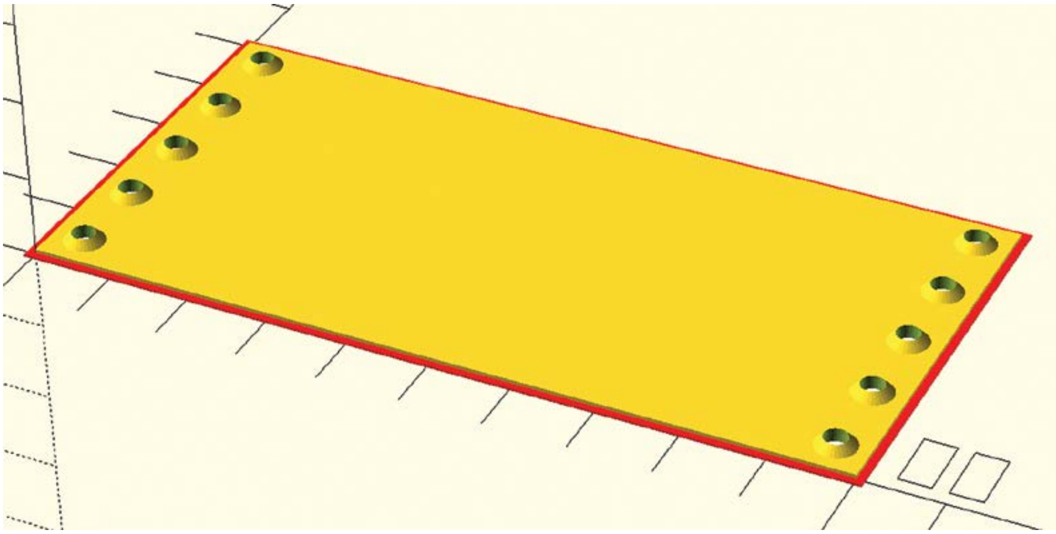


Figure 6 The 'holeSpacing' variable at the top of the program lets you set the spacing of the bolt fittings. The default value is 10 mm. The default margin around each panel is 1 mm. The 'margin' variable at the top of the program lets you change this

CASE ASSEMBLY

Figure 5 shows the sequence for making a case. The lining for the case was made from a scrap of spare fabric. You can also use felt as the lining material. The cloth and the cover were stuck onto the stiffener using extra strong spray-on flooring adhesive, which worked well. The outer cover was wrapped around the edges to finish them. The case was folded so that the raised parts of the bolt-hole reinforcements were on the inside. The outer cover and the liner material were stuck to the stiffener, and the excess material cut away once the glue had dried. The rectangular hole for the USB connector was cut with a sharp knife, and the inner lining cloth was folded back to cover the hole edges. The hole for the camera tripod was made with a hole punch and then opened up with a pen. The original plan was to print and fit covers over the two holes in the bottom of the case, but it turned out that with the corners fitted, it was not possible to add the covers. However, the case material leaves a clean edge when cut, and so the finished result was deemed tidy enough.

The case works well. It took a few attempts to get the sizes of all the components right. The outer cover and the inner lining added to the thickness of the edges so that the correct size of the corners could only be determined by printing a new iteration of the corner and testing it. The template took around an hour to print, with each corner taking around 20 minutes.

> It would take a very long time to create panels **of similar strength just by printing them**

PANELS FROM OPENSCAD

The OpenSCAD software for this project contains a function that will make panels of any size and add bolt holes and fittings on specified edges.

The panel shown in **Figure 6** was produced by the following statement:

```
makePanel(100,50,[1,1,0,0]);
```

The first two parameters specify the width and depth of the panel. The four-element array provided as the third parameter to the call of `makePanel` allows you to specify which sides of the panel should have bolt fittings in them. The first two elements in the array control the sides along the Y axis.

TAKING IT FURTHER

The author was very impressed by the strength and flexibility of the material that was produced by gluing thin 3D-printed layers between an inner lining and an outer covering. The use of single-layer thickness 'margins' to provide folds when assembling a structure also worked very well. It would take a very long time to create panels of similar strength just by printing them, and they would not have the flexibility of the ones that were produced here. It would be interesting to alternate strips of thin and thick stiffening elements to create panels that could be bent to form part of a curved case. By creating joining strips to link panels together, you could also create really large cases.

Laser-cut boxes

Create customised storage with an online generator

Ben Everard

@ben_everard

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

3D modelling is a tricky beast to learn. It's a useful maker skill, but it's one that takes time and effort to get right. Wouldn't it be much easier if someone created a tool to quickly and easily customise existing designs? Well, we have good news. The Boxes.py project does exactly this. You can choose from a huge range of existing designs, enter your own parameters, and get files that you can laser-cut.

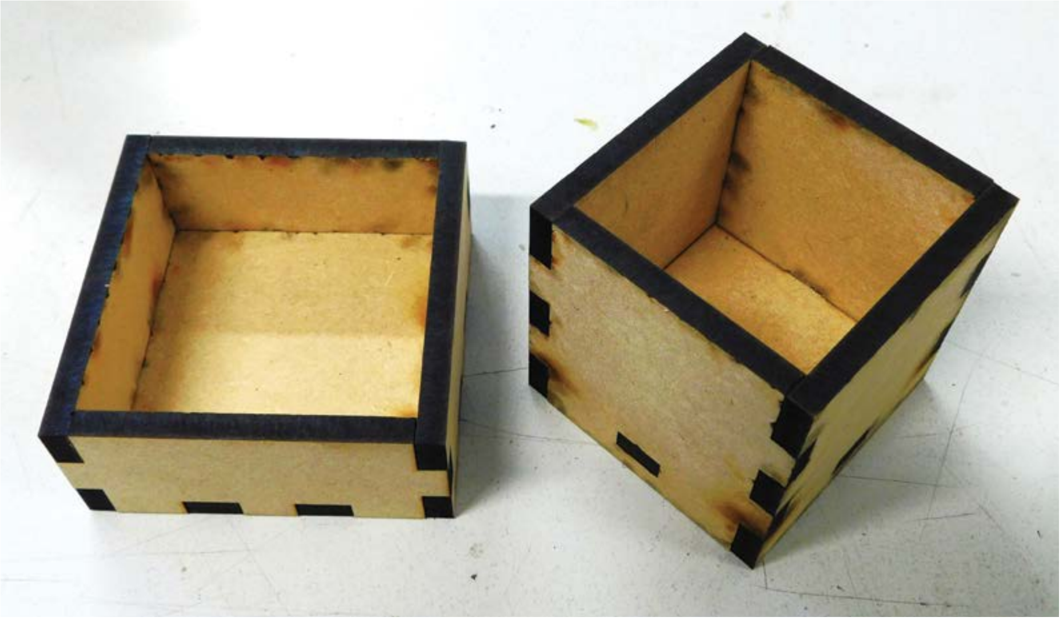
Before we start looking at the designs, we first need to get some calibration information for our cutter, specifically the kerf. This is the width of the cut that the laser makes. It varies from cutter to cutter, and even material to material.

In Boxes.py, the kerf is handled by the burn parameter, which is the distance from the edge that the laser will cut – in other words, the burn is half the width of the cut. Putting a larger value for burn will result in tighter fitting joints, and smaller values will give looser joints. This might sound counter-intuitive, but remember, the burn parameter doesn't control the kerf, it corrects for it. For some materials (such as wood), you might want very tight joints that hold together with just friction. For others, you might want slightly looser joints that leave some space for glue.

There's an included test to help you work out the burn parameter you need, which you can download from **hsmag.cc/burntest**.

You can leave all the settings at their defaults in most cases, though you might want to change the format depending on your laser cutter software. We usually use SVG or DXF, but use what works for you.

Cut out this test as you would any other laser-cut design, and try putting it together with the same



Right A lid is just a wider, shorter box turned upside down numbered edges against each other until you find the fit that you like. If none of them feel quite right, you can change the step and burn parameters to try a different range of sizes until you find settings that work for you.

Now we know the optimum value for the burn parameter, we can try our first box.

There are a lot of different box options, an almost bewildering range of box options when you first start. However, let's start in a simple place, the universal box. We're going to create a box that has a lid.

Head over to the universal box generator (**hsmag.cc/universalbox**) and enter the following, leaving the others as their defaults:

- **top_edge:** e Straight Edge
- **bottom_edge:** h Edge
- **x, y, h:** your sizes
- **thickness:** your material thickness
- **format:** the format you need
- **burn:** your calculated burn
- **lid:** flat

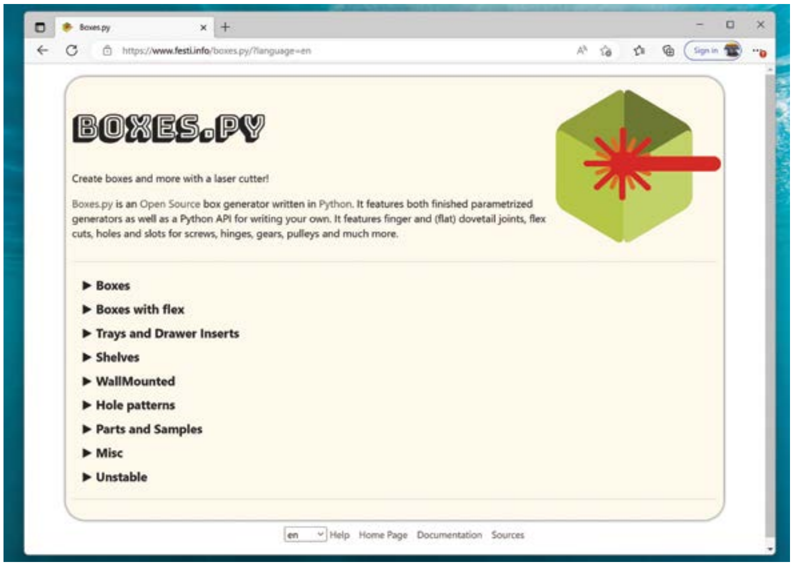
Whatever format you download it in, you should end up with a file with lines in three colours. The interior cuts are in blue, and these should be cut first. The exterior cuts are in grey, and these should be cut second, and the labels (if these are selected) are in red, and you don't need to cut or engrave these. We find it useful to include the labels in the download and design so that we have a reference as to which part is which, and simply turn off cutting for this colour; however, if you'd rather, you can deselect the checkbox to not include the labels in the file. Import the file into your laser cutting software, and cut it out.

Everything should fit together, though depending on how you set the burn parameter, you might need a bit of force. We printed ours to hold together without glue, and it needed a bit of tapping with a hammer to get everything in place.

The lid is made up of two squares, one internal and one external, and these need gluing together. You'll need to spread out some wood glue on the smaller square, then clamp it in place while it dries.

At the end of this, you should have a box with a lid.

> "If you cut thin strips into it, you can create **what's known as a 'living hinge' or flexure**"



Above Boxes.py is the name of the software, not the web address

WIBBLE WOBBLE

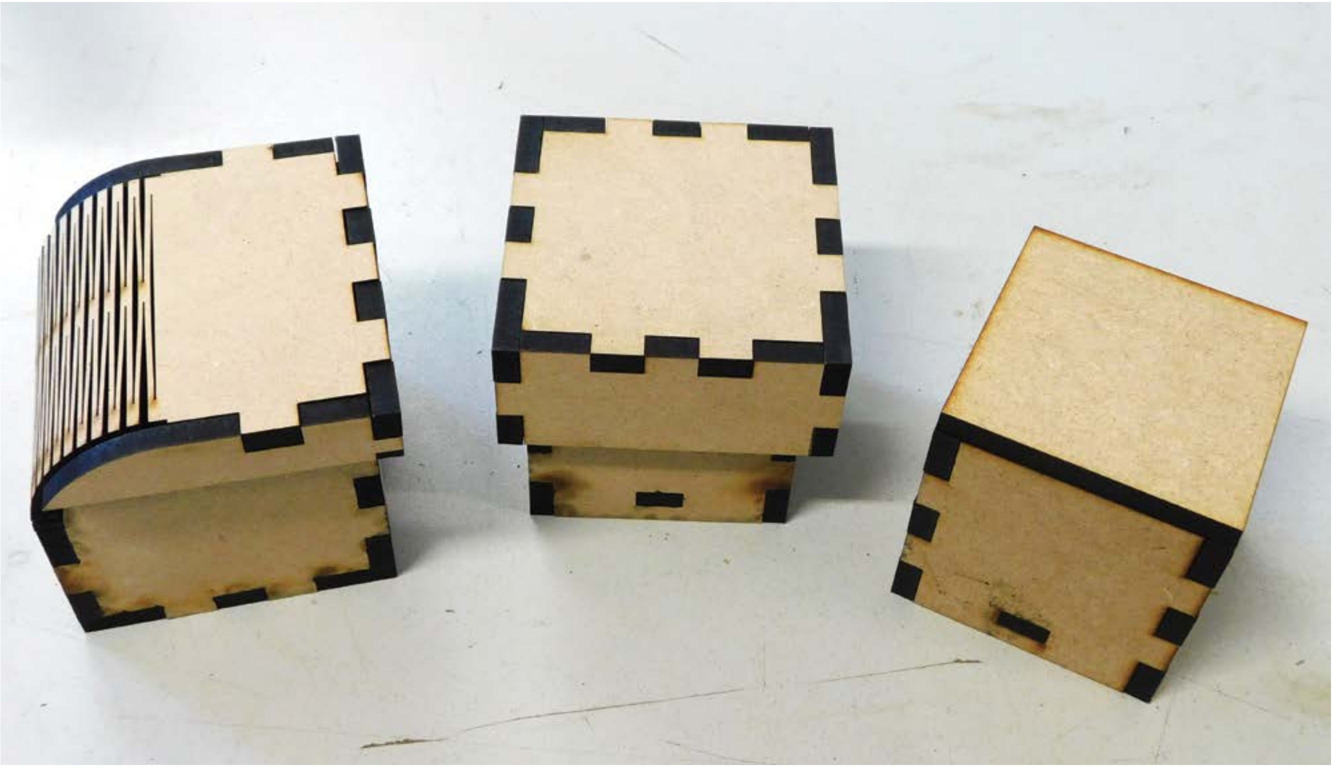
Let's take a look at another way of making a box with a lid, a flexbox. While sheets of engineered wood generally have a little flexibility (depending on their thickness), it's not usually enough to be useful. However, if you cut thin strips into it, you can create what's known as a 'living hinge' or flexure. These can be used in different ways, and there are a few different designs in Boxes.py that use them. The first thing, though, is to test out the material you have with a sample.

How well living hinges work depends entirely on the material you're cutting. Most engineered wood works well. Non-engineered wood is a bit prone to splitting, but can work as long as you don't need to bend it too many times. Acrylic doesn't work particularly well, but some other plastics might. In general, thinner materials work better than thicker ones. We did this test with 6.4 mm MDF and it worked fine.

There are a load of settings that you can tweak to alter the pattern that's cut to make the material flexible.

You can download the flex test from **hsmag.cc/flextest**. Unlike the burn test, this doesn't have multiple options in one file.

While the test pattern doesn't have any finger joints that are dependent on the thickness of the material, the flex parameters are given in proportions of the material thickness, so do make sure you enter this correctly. →



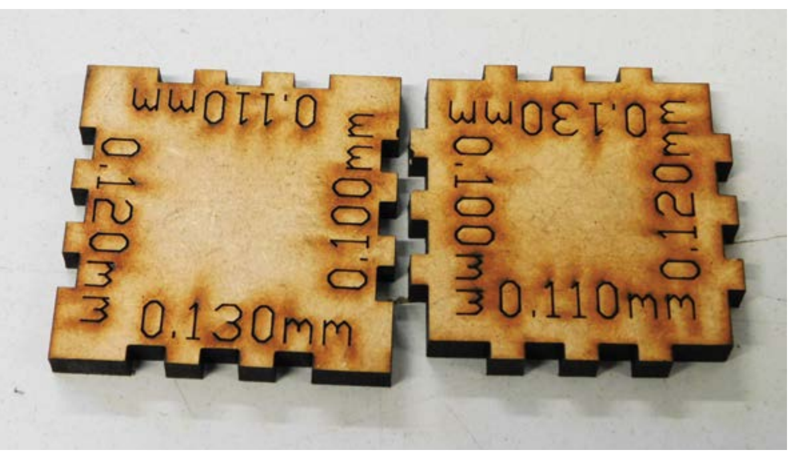
Above
Three different styles of box

It might well be that the defaults for this are fine for your use. In our experience, the defaults work quite well for thinner material, but might need adjusting for thicker material. With 6.4 mm MDF, we adjusted the Settings For Flex > Distance parameter to 0.25.

Now we've experimented with the flex material, let's make a box. While there are lots of boxes that use living hinges, we're quite fond of the unglamorously named FlexBox3, which uses a living hinge to pivot the lid.

As well as the flex parameters you experimented with before, you can adjust the radius of the bend that the material will take. The default is 10 mm, which is a little tight. If your box is big enough to spare the space, we'd recommend increasing this. On our 100×100×100 mm box, we went with 30 mm, which is gentle and aesthetically pleasing. Again, you can download the files in any format you like and cut them as before.

Below
Get the perfect fit with test cut



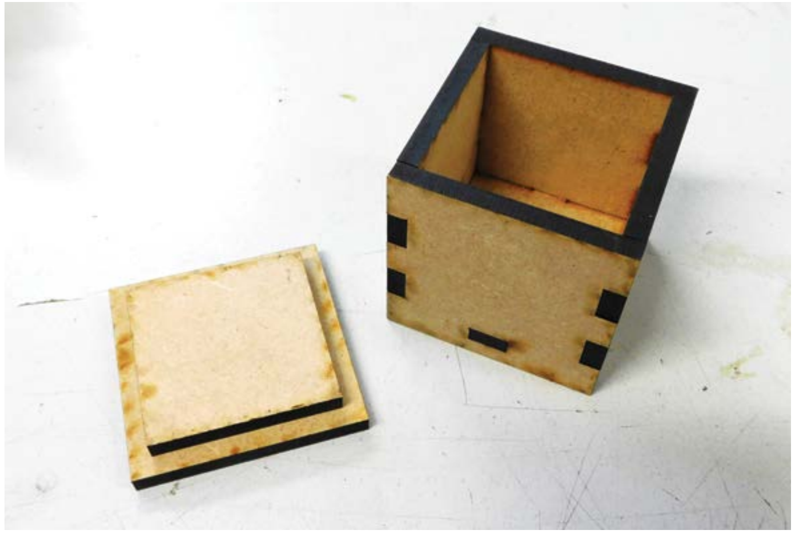
DOUBLE TROUBLE

While Boxes.py does have a huge range of different boxes, as far as we can see, there's no simple box with what we would call a 'normal' lid. The sort of lid you'd find on a shoebox – that is a lid that's basically another box but slightly larger and a lot shorter that goes on top upside down. Fortunately, since this lid is essentially another box, you can create it using the universal box generator.

The first part of the box, you can make in the same way as our first box – in fact, you can use that if you like. To make the lid, you need to know the outside dimensions of the box. You could get this from the Boxes.py generator, but given that it needs to be quite accurate and it will depend a little on just how firmly you've pushed the joints together, we'd recommend actually measuring the box.

In the universal box generator, you can now create your lid-box with the following parameters:

- **top_edge:** e Straight Edge
- **bottom_edge:** f Edge
- **x, y:** the measured sizes of the main box plus 1 or 2 mm



- **thickness:** your material thickness
- **format:** the format you need
- **burn:** your calculated burn
- **lid:** default (none)

An important difference between this and the first box we made is the bottom edge. We used h-edges previously, which are stronger and better suited to the base. This is where the finger joint from the base pokes through a hole in the side. For this box, we've used an f-edge, which is where the bottom is flush with the bottom of the side. This isn't quite as strong, but the top of our box doesn't need to be very strong, so this should be fine.

You should now be able to download and cut this box, and it should fit snugly over the other box.

"You should now be able to download and cut this box, **and it should fit snugly over the other box**"

We've looked at three examples of boxes you can make with Boxes.py, but there are loads more options available. Most of them have images to give you an idea of what's possible. Spending a little time clicking through the available options is time well spent, as you'll be able to pick the perfect enclosure for your next project.

Above A simple lid of two bits of MDF glued together

Left The universal box has a mind-boggling range of options, but you don't need to use them all

Left We're always amazed by just how bendy MDF becomes

Macramé: A beginner's guide

Get creatively knotted with a very traditional craft that's well and truly back in vogue

Nicola King

@holtonhandmade

Nicola King is a freelance writer and sub-editor. Her favourite craft changes week by week, with her sewing machine currently in quilting overdrive.

Macramé was a very popular pastime back in the 1970s, which was perhaps a golden age of handicrafts of all kinds, when this form of creative knotting was used to fabricate lampshades, plant hangers, wall hangings, and many a bohemian beaded waistcoat with intricate and delicate patterning. This author fondly remembers her mum's extensive collection of 'Golden Hands' periodicals, an informative series of magazines that sought to share crafting goodness with the multitude. Macramé, in some form, always seemed to be in every single issue.

However, like many fibre-related crafts, macramé's history predates that particular era by some time, in fact by thousands of years. For example, some decorative carvings of the Babylonians and Assyrians

Above
There are many books available on the subject, and a pattern book of some kind is always useful as an inspiration source

YOU'LL NEED

- **3mm macramé cord** (e.g. **hsmag.cc/MacrameKit**)
- **Sharp scissors**
- **A tape measure**
- **A keyring** (with a D ring-shaped base – **hsmag.cc/DRingKRing**)
- **A comb for fringing** (optional)

Left A few of the additional bits and bobs you can use within your work. The wooden beads and rings and dowels really complement the natural fibres being used

> The beauty of the craft is that there are no **hooks or needles required**

feature macramé-style knots. More recently than that, 19th century British and American sailors also made hammocks and belts from macramé, although back then it was called 'square knotting'.

Well, you can't keep a good craft down and, after falling out of fashion in the 1980s following its previous resurgence, macramé is popular once again, with millennials picking up on its relaxing benefits. You can find macramé kits in abundance in craft stores, and this author has even seen plant hanger kits for sale in supermarkets, and she's happy at the craft's revival. When she moved into her present family home many years ago, a macramé lampshade adorned the ceiling of one of the bathrooms – it went swiftly in the bin but now, as a fibre and textile crafts fan, she slightly regrets casting aside someone's vintage hard work.

Macramé is another calming and mindful hobby, and you can use it to create a huge variety of useful household items, as well as beautiful pieces of art and jewellery.

The beauty of the craft is that there are no hooks or needles required, as we are not weaving or knitting here – we are just knotting with our hands. Also, the emphasis is very much on using natural fibres such as cotton, jute, or hemp, for example – ideally these fibres are sustainable and plastic-free, so it can be argued that macramé is quite a 'clean' →

Left A selection of macramé cords – obviously, as with any fibre craft, the thicker the yarn, cord, or rope that you use, the larger the finished product will be

QUICK TIP

Work at a clear, roomy table with good lighting so that you can clearly see what you are doing.

Figure 1 Attaching cord with the useful lark's head knot. This knot is also often used in jewellery making, and is also called a cow-hitch knot

WHAT CAN I MACRAMÉ WITH?

There are several options in terms of the fibres that can be used for this craft. Below are the main natural ones to consider, although you can also macramé with synthetic fibres should you so wish. We've opted to concentrate on natural fibres as they look great, are perfect for indoor projects, and they are biodegradable. Be aware that, if you are making projects that will sit outdoors with natural fibres, that biodegradability feature will mean that your makes will eventually degrade over time. For example, cotton will yellow in the sun, and jute and hemp will be affected by rain and sun – just something to bear in mind.

Clearly, the thickness of your material is a key choice, depending on what you are making. You'll need to consider the ply, i.e. how many strands have been twisted together to make a piece of cord or rope, and what effect does that give? Thicker fibres can be more challenging to work with, but certainly make a statement piece.

- **Cotton** – the world's most used natural fibre, it is soft and fairly strong too, and popular in the macramé world. It also gives a smooth finish and is the most versatile macramé fibre. Flexible and widely available, cotton is easy on the hands and won't stretch over time. It also comes in various thicknesses, from string to cord to rope, from $^2/_3$ mm up to 40 mm.
- **Jute and hemp** – these are both fibres with low elasticity and a similar tensile strength. Jute is extracted from the stem of the jute plant, while hemp comes from the stalk of the hemp plant. Both of these can be more difficult to work than cotton and can be very rough on the hands, but they arguably have a very rustic and natural look to them, so it really depends on the aesthetic you are aiming for.
- **Linen** (in cord or yarn form) – perhaps not as common as other macramé materials, this fibre comes from the flax plant. More difficult to come across, but very strong, and often used for jewellery making.
- **Silk and leather cord** – both extremely strong macramé fibres, but perhaps less common. Leather cord is often used in jewellery making.
- **Natural yarns** – including merino wool, alpaca wool etc. Macramé knots will be much smaller if you use a natural yarn, however, as the fibre compresses so much.

hobby to pursue. Plus, if you decide to give it a try, you need very little in terms of equipment, so your craft space won't get unnecessarily cluttered and, perhaps most importantly, macramé is not hard to learn – once you have mastered just a handful of knots, you can make a plethora of items. So, in this tutorial, we are going to give you a basic macramé taster session.

LET'S GET TIED UP IN KNOTS

We're going to make a very simple keyring, and will be looking at just three basic knots: the lark's head knot, which simply attaches our cord to the keyring; the square-knot, which will create the main body of the piece; and the gathering knot, which will finish off our work.

STEP 1 CUTTING AND ATTACHING OUR CORD

A top tip is never to underestimate how much cord you are going to need. In fact, estimating how long to cut your cord length is arguably one of the most difficult stages. There's a tendency to cut cords too short, and you don't want to be joining cords halfway through your make. We'll cut two lengths of two metres each of our cord.

We then use the simple lark's head knot to attach the cord to the keyring. Take one length of cord – we've used 3 mm cord – and fold it in half. Take

Figure 2
A square-knot up close. We had to practise this a few times, but once we'd learnt it, we were away!

the looped end and thread it through the back of the D ring part of the keyring. Then, pull the cord ends through the middle of that loop in the cord (**Figure 1**). Repeat with the other length of cord. You should now have four lengths dangling from your keyring. We will be working with the two outer cords throughout. The two central cords are called the knot-bearing cords, and you should try and keep these two cords as straight and taut as possible.

> **There's a tendency to cut cords too short, and you** don't want to be joining cords halfway through your make

STEP 2 THE SQUARE-KNOT

Before we begin the main body, a word on tension. Everyone's tension is different, and we are aiming for a uniform, natural tension throughout. Don't pull the cords too tightly, but don't leave them too loose or you'll have huge gaps in your work – just pull them using a tension that feels natural for you and gives you the effect you want.

Take the left-hand knotting cord and take it over the two knot-bearing cords, and under the right knotting cord. Bring the right knotting cord to the left, under the two knot-bearing cords and over the left knotting cord (through the loop you've created). Pull the knotting cords, ensuring the knot-bearing cords are straight and taut. Now for the other side – bring the right knotting cord to the left, over the knot-bearing cords and under the left knotting cord. Finally, bring the left knotting cord to the right under the knot-bearing cords, and over the right knotting cord (through the loop). Pull to secure your square-knot (**Figure 2**). Repeat this until you have achieved the length you require.

RESOURCES

If you're interested in giving this craft a go, it's useful to do a little research before you start, and there are many books/websites that can answer any questions you may have and give you a dash of inspiration. Many modern macramé patterns have a contemporary twist so, don't worry, no 1970s throwbacks, unless of course that's the look you are aiming for:

- ***Macramé Pattern Book*** – Marchen Art – a handy guide that contains over 70 patterns that you can turn to for a multitude of projects
- ***Modern Macramé*** – Emily Katz – contains a whole heap of projects to get stuck into
- ***Macramé: Techniques and projects for the complete beginner*** – Sian Hamilton and Tansy Wilson – a great place to start for a newbie
- **YouTube – Majestic Macramé (youtube.com/@majestic.macrame)** – lots of very clear videos to help you on your macramé way
- **YouTube – Macramé School (youtube.com/@macrameschool/videos)** – make a macramé flower, make a macramé fish… it's up to you
- **Website: macrameuk.com** – includes a 'Knot bible' page for easy reference

STEP 3 FINISHING OFF

When you are happy with the length of your work, it's time to tidy up your ends. First, cut another length of cord, around 20 to 30 cm in length. Place the end of that against the loose cords, leaving a short end on the top, and form a loop, then wrap this new cord tightly around the bottom of your work several times. Then take the end of that cord and pull it through the loop you made (**Figure 3** uses a contrast colour to illustrate). Now, pull on the short end of the cord until the lower end of the cord is pulled up into the wrapped knot. Trim either end of that cord and tuck any loose ends into the wrapped knot. You can then either trim the remaining loose dangling ends to a length you like, or you can use a comb to turn them into a fringe, as we have done.

Once you've learnt a few knots, you can obviously gravitate towards more complicated macramé patterns, and maybe experiment with adding glass beads, shells etc. to create more elaborate and ornate pieces. We've also seen amazing work where artists have used a cotton hand dye and paintbrushes to colour their unique creations with a stunning ombré effect. So, get knotting!

QUICK TIP

When using natural fibres, ensure you store them well before you use them, wrapping them in something so they don't get too dry.

Figure 3
A gathering knot to pull the ends of your work together

HackSpace
TECHNOLOGY IN YOUR HANDS

WEEKEND PROJECTS

HACK | MAKE | BUILD | CREATE

When you've got a bit more time to yourself and can spend longer in the shed/garage/workshop, these brilliant builds will keep you occupied and expand your making skills

PG 30

DRAGON EGG

Build a magical Halloween prop with electronics from your favourite online auction website

PG 36

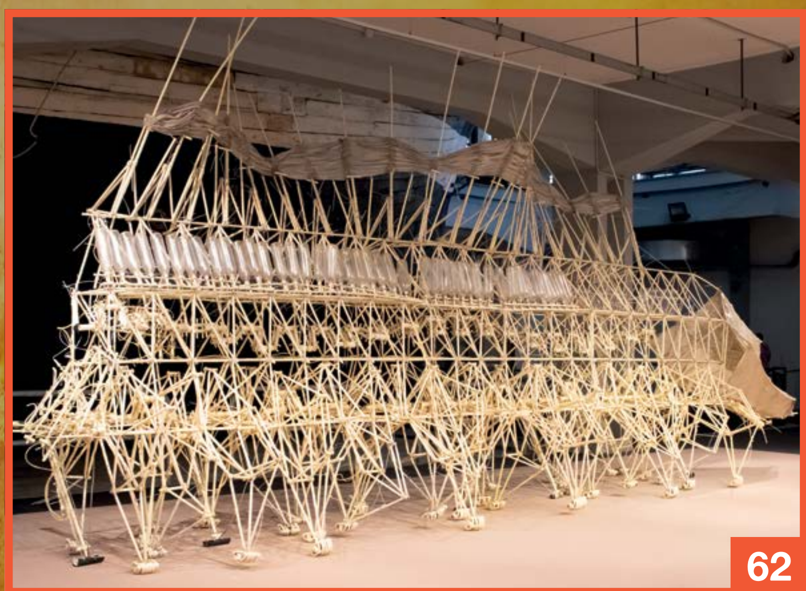
PICO PLANT MONITOR

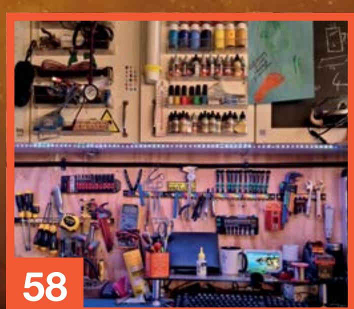
Screens are boring – read data about your houseplants with servos and tin cans instead

PG 42

PEPPER'S GHOST

Recreate a classic Victorian stage trick with a tiny screen and a Raspberry Pi Pico


62


58


72

Making a dragon's nest

Make a misty forest dragon's nest, complete with a flaming egg

Dr Andrew Lewis

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

Above
By modifying a flame-effect light bulb and an ostrich egg, this project would make a great background for reading children stories that have dragons in them, or a Halloween decoration

There's a staggering amount of mythology surrounding dragons. From giant creatures in ancient tales to the vacuum cleaner-sized yapping flamethrowers of the Discworld, they're a staple of the fantasy genre. In this article, you'll get the opportunity to create your own dragon's nest prop with firelight effects and a mysterious haze. You'll also see how to use a motion sensor module and a Raspberry Pi Pico to turn on the electronic effects when someone approaches.

This dragon's egg prop has two practical effects built into it to give the impression of fire and smoke. On first glance, it's fairly obvious that flickering LED lights are used to give the effect of dancing flames, but the source of the 'smoke' is perhaps a little more difficult to discover. It's become quite popular to hack electronic cigarettes (vapes) to create smoke effects in models – for some projects, it's a good choice. A vape works by heating a mixture of propylene glycol and glycerin with a coil, the vapours of which are then inhaled. To create a smoke effect from a vape, you need to find a way to keep the heating element active while you use an air pump to push the vapour out of the way of the coil. The dragon's egg doesn't use this approach because there's a risk that a vape coil could overheat, boil the liquid, catch fire, or burn out in continuous use. This is combined with the fact that a vape needs a special liquid to work. From an aesthetic standpoint, the vapour from an e-cigarette needs to be kept in motion to keep the heater coil clear, and the vapour tends to rise, rather like a plume of smoke from a chimney.

For this project, a low-hanging mist is more visually effective, evoking memories of the alien nest in the movie *Alien*. To achieve this look, a more tried-and-tested technology comes into play. An ultrasonic pond mister or humidifier creates an eerie, rolling mist when it's activated, and it only needs a few inches of water to work. While

DRAGON SAFETY

It does not do to leave an electrically live dragon out of your equations, if you live near him. This project mixes water and electricity, and although the voltages involved here are low and should be safe, you should always make sure that your projects are properly fused and protected with a residual current device where necessary. Make sure water cannot travel anywhere in your project where it might cause a danger. In particular, ensure water cannot escape from your project and pool anywhere near household electrical outlets or extension cables.

pond misters are effective, they do have a few drawbacks that make them unsuitable in some situations. They're quite large when compared to an electronic cigarette, being generally about the size of a cupcake. They also tend to need an AC voltage input, because they work by oscillation of a piezoelectric element, and it's generally cheaper and easier to generate that oscillation with an AC voltage source. That means that it's less easy to get an ultrasonic mister to work from a battery, and, at the same time, it's more difficult to control the mister with a solid-state device like a transistor. Luckily, none of these drawbacks are a problem for the dragon egg project. We've used a 24 V DC mister that comes with its own power supply. These are widely available online.

Begin your project by building a nest. You can use a variety of different methods for this, including weaving branches together into a bird's nest, using a ready-made basket or bowl, using chicken wire and plaster bandages, or gluing together clay beads over a form and then using resin to strengthen the structure.

After building your nest, you'll need to put in a reservoir to hold your mister. A large spray can lid is useful for this, but be careful that you choose one that's deep enough to submerge the mister without creating a large amount of splashing, or you'll lose most of your water in just a few minutes. You can disguise the containers used for the reservoir with moss and stones.

LOW LIGHTING HIDES MANY CRIMES

Route the power cable for the pond mister out of the back of the nest. It's time to feather the nest →

"After building your nest, you'll need to put in a reservoir to hold your mister"

FLAME EFFECT

One of the main features of this project is the flame effect generated inside the egg. While the effect might look daunting to create, the circuit is actually salvaged from a cheap flame-effect light bulb. It's possible that you might have considered using these flame-effect light bulbs in the past but were discouraged by the practical or safety concerns arising from including a mains-powered bulb in your project. The good news is that the while the bulb uses 240 V, the circuit that controls the light effects inside these bulbs actually runs somewhere between 3.3 V and 5 V.

Above You can really let your imagination run riot and include whatever you want when you're designing your nest. This nest uses an antler and some old keys in addition to crystals, pine cones, and lichen

Below There are lots of different brands of LED flame-effect light bulbs, but they all look very similar inside. This particular version has the PCB held in place with a few dabs of hot glue, an elastic band, and double-sided tape

Right For the ultimate in flexibility when making, you can glue beads together on a plastic form (a plastic mixing bowl, for example), then remove the form when the glue is dry. The set PVA will be flexible rather than rigid, so you can manually tweak the shape of your nest and then use superglue to make the shape permanent. Finally, you can paint the whole nest with a casting resin to make it rigid and waterproof

USER SENSING

If you want to save power and water by only triggering your dragon's nest when someone is nearby, then you can do this with a Raspberry Pi Pico, a motion sensor, and some relay modules. Connect the motion sensor power to the 3.3 V output of the Pico (VSYS), and the sensor output to GP0. GND on the sensor connects to GND on the Pico. Connect the relay modules' power to 5 V (VBUS, assuming that you are powering your Pico from USB), and the modules' triggers to GP1 and GP2, with the modules' GND connected to GND on the Pico. A simple program can be used to trigger the relays on GP1 and GP2 when the Pico receives a high signal on GP0, then turn them back off if no motion has been detected for some time.

to disguise your cable and make it look lived in by a dragon. Add in a layer of lichen, some stones, pine cones, leaves, and any other detritus that you think a dragon might stuff into a nest. Once the nest looks nice and comfy, you can move on to the egg. If you can't find a genuine dragon's egg, you can substitute an ostrich egg or a papier-mâché egg instead. Ostrich eggs are generally available in

> "Ostrich eggs are generally available in the immediate **vicinity of an ostrich, or from online auction sites**"

QUICK TIP

If you'd rather have more control over the flame effect, you can use WS2812 LEDs (aka NeoPixels). These come in many forms, but a chain should fit inside the egg and then you can control the lighting of your egg from a microcontroller. You could even use different colours as notifications.

BULB SURGERY

Extracting the flame circuit is pretty straightforward, but the process will vary depending on the exact bulb you are working with. The general process involves removing the outer cover from the bulb with a plastic spudger (or a similar tool such as a screwdriver or sledgehammer) to reveal the electronics inside. The inner part of the bulb contains a flexible circuit board with a microcontroller and LED lights, powered by a small transformer circuit in the base of the bulb. This flexible circuit is usually glued or taped around a cylindrical support in the centre so that the flame effect is visible for 360 degrees. Carefully peel the flexible circuit board away from the central support, and cut the wires that lead to the transformer. You can now connect the board to a DC power supply and test it at 3.3V. If it doesn't light up or flicker properly, increase the voltage until it becomes stable. Choose a suitable DC power supply for your board, and connect to the flexible circuit with an in-line plug and socket.

Left
Some flame-effect light bulbs use 3.3V internally, while others use 5V. If you find the bulb stops working after a few seconds, try reducing the voltage. You'll probably find something is overheating. The circuits can be quite fragile when it comes to electrical spikes, so be careful when connecting and disconnecting wires

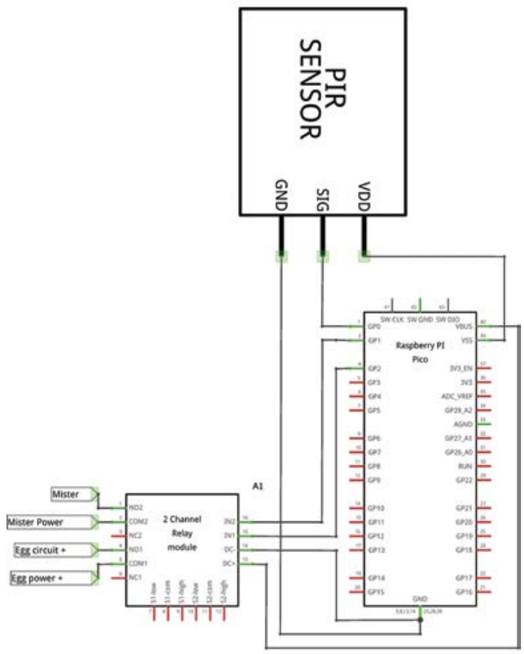
QUICK TIP

If you're interested in hacking vapes as a source of practical smoke, take a look at Dave Bodnar's notes at **hsmag.cc/AnimationSmoke**

the immediate vicinity of an ostrich, or from online auction sites. It's easier to get a blown egg (which is just an empty shell) than a whole egg, unless you need to make a really big cake and are willing to carefully blow the egg yourself without cracking it.

A blown egg will have a hole in the base, and you can use this hole to insert the flexible circuit from a flame-effect light bulb. Attach long wires to the flexible circuit, and carefully roll it up with the LEDs on the outside of the roll. Push the rolled-up circuit into the egg, and then unroll it so that it sits with the LEDs facing upwards. This step is quite fiddly, and you may find some pea-pod forceps or tweezers useful. Fix the circuit in place with a dab of hot glue, and then close over the end of the egg with glue and tissue-paper to disguise the hole.

Place the egg into the nest and route the wires under the lining so that they are hidden from sight when you're looking in. Fill the mister reservoir with water and plug everything in. After a few seconds, you should be rewarded with the sight of a fiery dragon's egg in a sea of mist.



Above
Connecting PIR sensors and relay controllers to your Pico is a simple way to control the practical effects in your props

QUICK TIP

If your ultrasonic mister isn't deep enough in the water and fires water out of the nest, use a plastic lid at an angle to catch any splashes and divert them back into your reservoir.


CODE
THE
CLASSICS
VOLUME I
SECOND EDITION

Tin can plant monitor

Build a plant monitor with servo-driven indicators instead of LCDs, LEDs, or TFTs

Dr Andrew Lewis

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

It's easy to fall into the maker trap, where your projects are driven by the availability of tools and parts rather than the desire to create something that serves your particular aesthetic. It might be more complicated, but choosing the path less travelled is a great way to build your skills and think outside the project box. In this article, you'll see how you can provide feedback about soil moisture and temperature with servos rather than the now ubiquitous LCD or TFT displays. You'll also get to play with recycled junk and build a model to house your project.

When anyone mentions a servo motor, makers tend to think of remote-control aeroplanes, walking robots, or animatronic toys. Hardly anyone thinks of car dashboards or clocks, but not every servo is a big, powerful motor, and chances are that the gauges in your car are actually servo-controlled. Light duty 9g servos (so called because their weight is 9 grams) are great for making dials or indicators because they're small and light, but they can still react quickly while moving a reasonable amount of weight. You'll use two 9g motors in this project to create different types of gauge, and you'll also learn a few basic model-making tips to keep things interesting. You will build a simple plant monitor to report on the moisture level of the soil, and get a general idea of the ambient temperature by reading the thermistor on Raspberry Pi Pico. The code for this project is available at hsmag.cc/issue60.

PAINT HIDES MANY SINS

Before you start creating dials and gauges, you're going to need somewhere to put them. You could just use a simple plank of wood, or you could get

creative and start sticking some bits and pieces together to make a strange contraption. In this article, the base object is a steampunk-esque model engine created with a sardine tin, a small tomato puree tin, a grinding bit box, and some air fittings from a compressor and a fish tank pump. Glue the pieces together to create a shape you like, and then apply a coat of undercoat to bring the pieces together visually. This sounds like a strange step, but it's very difficult to get an eye on the real shape of a model until all of the pieces share a base colour. Once that's done, it's easy to see which bits need more detail and which bits need to be removed or changed. You probably aren't working on a deadline for this project, so you have time to play around. Remember that you'll be adding some dials and electronics to your base, so don't paint everything yet, because you'll probably want to drill, glue, or otherwise modify things as you add the mechanisms.

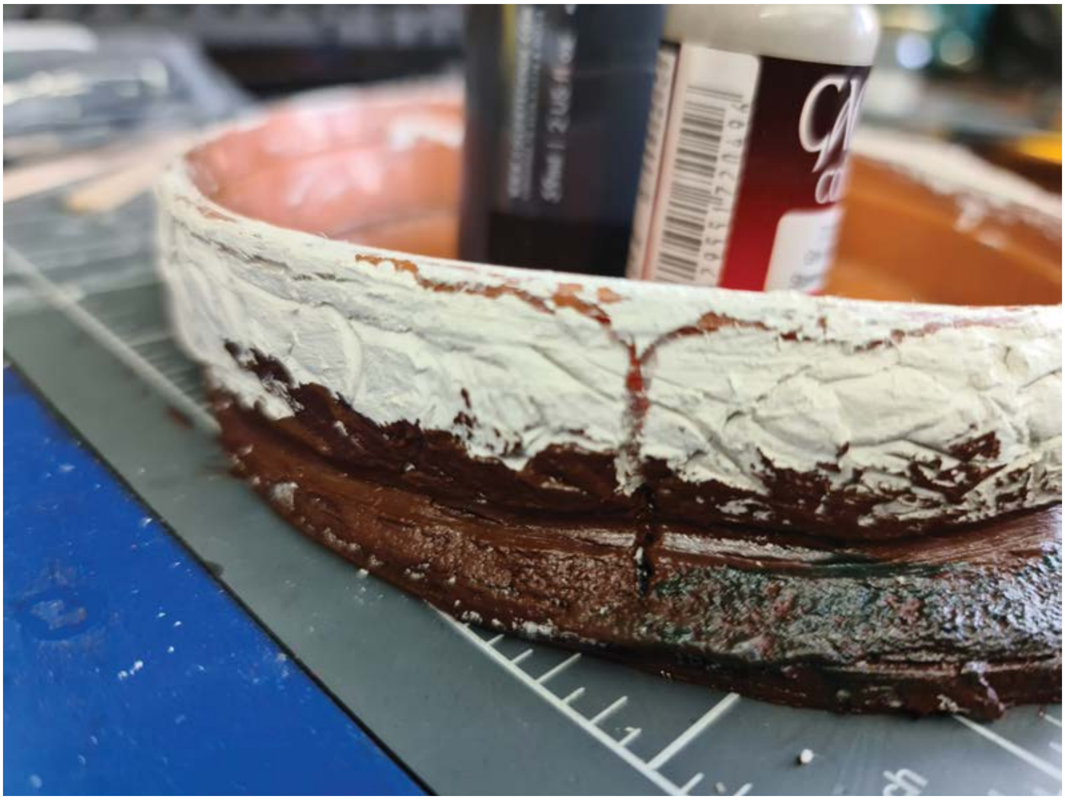
If you're feeling particularly crafty, you can make a stand to hold your plant monitor and plant pot. Glue a plant saucer to a board, and mark out an area large enough for your plant monitor to stand. You can use air-drying clay to build up the base and create walls, stairs, or flagstones. Once the clay is dry, you can paint it and apply lichen and other dressings to make the landscape look more realistic.

The first type of dial mechanism you will create is a linear indicator. A linear indicator takes the rotary motion of the servo and converts it into linear motion by using a simple rack and pinion gear system. If you're not familiar with the term, a rack and pinion gear uses conventional gear and an 'unrolled' gear to create linear motion. As the conventional gear turns around, the linear rack gets pulled along in a straight line. You can 3D-print, laser-cut, or handmake your rack and pinion gears, or just steal some from your LEGO Technic™ kit. You'll be using the linear motion created by the rack and pinion to indicate the ambient temperature. Since a 9g servo typically rotates through 180 degrees, you will want to use a rack with slightly more than half of the number of teeth that the driving pinion has. You'll need to enclose the rack into a channel so that it doesn't →

MICROPYTHON

This project uses a Raspberry Pi Pico running MicroPython to control servos and read data from sensors. If you haven't installed MicroPython before, you can find instructions here: **hsmag.cc/WhatIsMicroPython.**

Above
Adding a layer of undercoat to a project makes it much easier to get a feel for shapes and scale. For supposedly metallic items, a base coat of black followed by a chrome or copper coat makes a fair base to start adding more detailed block colours, then washes of thin colours (washed on then partially wiped back off again with a rag) to accentuate certain features. An almost dry brush loaded with a metallic colour can be used on edges to add the appearance of long-term wear, while semi-transparent gloss coats can be used to give the impression of oil or grease patches



Above You can create a small model feature to hide the moisture sensor in the plant pot. Slices of a tree branch make a nice flat surface for a rustic model table loaded with small screws and other objects, or you could create something entirely different instead

Right Air-drying clay shrinks as it hardens. That means cracking is a common problem. You can simply fill any cracks with more clay, or leave them alone and make them a feature of your landscape instead

Left
Agricultural grit mixed with PVA glue makes an excellent gravel effect. If you don't like the slightly slick and glossy look of the dry PVA, sprinkle a layer of unglued grit on top of the wet grit/PVA mixture. The top layer will still stick in place, but there won't be any glue on the top of the stones

STAY STUCK

This is the sort of project where glue plays a major role, so it's worth taking a quick look at the different types of glue that you might encounter. For model work, there are two types of glue that you'll use, and these are high viscosity and general-purpose superglues. High viscosity superglue is thick, and less likely to run if applied to a vertical surface. High viscosity superglue usually starts to bond in about 15 seconds. It's great for filling gaps and making big, bold joints along edges. General-purpose superglue is its thinner sibling, which sets more quickly (in as little as five seconds, depending on the temperature and humidity of the room). Thinner glue is great when you can't quite reach the target that you need to apply glue to, and need to let it drip into place. It's also a good way to hold strings or knots into place. Very low viscosity superglue is available for use in special cases, but it is very difficult to apply accurately, and in inexperienced hands, it will cause absolute chaos and leave you looking like you're trying to do a live-action role-play of Katamari Damacy. Waterproof PVA glue is very useful in model-making projects to keep things like fine grit or gravel in place. Simply stir the PVA and the stones together, and then lay them into place on your model. When the glue sets, the stones should be an immovable feature of the landscape. You can add acrylic tints to PVA to create the effect of muddy ground or shallow water.

come out of alignment with the pinion. You can easily create a channel by covering your rack with a couple of layers of masking tape to increase its thickness slightly, then gluing a channel together out of lollipop sticks or balsa wood. Remember, if you're painting the channel, the paint will add thickness and make the inside of the channel smaller, so it's best to be generous with the masking tape and use thin paints.

> "You'll need to move the servo to figure out the maximum and minimum limits **that it can move to within the constraints of the model you've built**"

When you're first setting up a servo with a gear system, you should make sure that you know the position that the servo is in by powering it up and moving it to a sensible known position. Once the gears are attached, you'll need to move the servo to figure out the maximum and minimum limits that it can move to within the constraints of the model you've built. If possible, keep gears loosely fitted to the servo so that they're free to slip if excessive force is applied. It's better to slip gears on a shaft →

QUICK TIP

Bicarbonate of soda makes superglue set pretty much instantly, and it's also great when used with superglue for filling larger gaps in a model.

Above Copper is a soft metal, so a copper linkage is likely to bend rather than transmit enough force from the servo to damage something important. It's also useful that you can add a little ball of solder to the end of a copper rod to keep it from slipping out of its mounting hole

PULSE WIDTH

Most servos expect to receive a PWM waveform on their signal wire (which is usually yellow, white, or orange), with the remaining two wires being used to power the servo. The Raspberry Pi Pico can generate PWM signals on most of its outputs, so connecting a servo is often as simple as picking a digital pin and plugging it in. The PWM signal used to control the servo is a typical square wave with a variable duty cycle and (for 9g servo motors) a frequency of 50Hz. The frequency is set in MicroPython using the easy-to-understand function 'machine.pwm.freq(50)'. The duty cycle is slightly more complicated to explain. In MicroPython, the duty cycle is set with the 'machine.duty_u16()' function. MicroPython uses a 16-bit resolution for PWM signals, so the range between 0% and 100% duty cycle is 0–65535 (note that in reality, the Pico hardware uses 12-bit resolution which is upscaled to 16-bit by MicroPython, so true 16-bit accuracy is not possible). To operate a 9g servo through its full 180-degree range using this scale, the duty cycle would range between 1000 and 9000, with 1000 representing 0 degrees and 9000 representing 180 degrees.

than break apart your nice new model by typing the wrong number into your code.

The second dial mechanism used in the plant monitor is a rotary gauge, and it is used to indicate the moisture level of the soil. However, since the 9g servo is a rotary servo anyway, it feels a little bit too easy to just stick an arrow onto the servo shaft and call it done. Instead, you can use metal linkages to control the bar of the moisture level scales remotely. If you're accurate enough, you can cut the linkage from a single piece of thick gauge wire. If you're not confident that you'll be able to get the measurements close enough, use two pieces of wire and join them with solder or a wire connector block once they're in place. Using linkage wires means that it's possible to transfer the motion from the servo to a remote location, effectively hiding the servo from view. In the plant monitor, the linkage wires pass from the servo on the inside of the sardine tin on the bottom of the model, through the puree tin, and through the air fixture at the top of the model. The scales pivot on a piece of wire glued to the side of the air fixture, and the linkage wire connects a few millimetres to the side of the pivot point. As with the first servo, the range of motion must be calibrated to make sure that the servo doesn't bend the linkages or pull something apart.

QUICK TIP

A stiff toothbrush or wire brush can be used to create a grass-like texture in clay. Simply stipple the brush into the surface to get the desired effect.

Left Choosing a meaningful icon to represent a particular concept is a difficult task, but words can be less effective when viewing something at a glance. The idea of dry twigs and green leaves hanging on the scale could be interpreted in two different ways, confusing a user who doesn't understand that the scales should be balanced

ACCURACY IS RELATIVE

The moisture sensing part of the plant monitor doesn't display an absolute value; it simply reports the sensor readings as one of five possible states: very dry, slightly dry, ideal, slightly wet, and very wet. The scales will tip or balance based on where in that list the reading falls.

> "The real point of this project **is to have fun and make something unexpected**"

When thinking about the technology used in this project, there isn't really much to see beyond a standard analogue moisture sensor, a Raspberry Pi Pico, and two PWM servos. The real point of this project is to have fun and make something unexpected out of the contents of your recycling bin. It's amazing how much fun you can have by sculpting, painting, and finishing a project like this. □

GETTING WET

The important thing about moisture sensors is that they generally don't directly measure moisture. The two most common types of moisture sensor actually measure either the resistance or capacitative response of contaminants in water. The value returned by these sensors is relative rather than absolute, and it varies depending on the composition of the soil, the root density of plants in the soil, the presence of stones or wood, and many other factors. This means that getting an accurate value from a sensor requires calibration for the particular soil and water combination that it's been placed in. To short-cut the full calibration process, the code for the plant monitor assumes that the moisture sensor value returned when the monitor is first powered on is the ideal moisture level. The scales on the device will indicate whether the soil is drier or wetter than when the sensor was first queried.

If you look at the code for the plant monitor, you might notice that power to the sensor isn't provided constantly. The sensor is powered via a GPIO pin that switches the sensor on before a reading is taken, and then turns it back off again when the reading has been taken. This project uses a capacitative sensor, and having the sensor powered isn't a real problem because capacitative sensors don't directly interact with the soil in the same way that a resistive sensor will. Capacitative sensors don't require a direct connection, so they are less prone to corrosion. Resistive sensors work by passing an electric current through the soil and measuring the resistive drop between the sensor probes. As you can imagine, passing a constant current (however small) through the probes will lead to very rapid corrosion. To mitigate the problem, it's generally considered good practice to power the sensor only when needed, and also to alternate the current when possible.

FEATURE

IN THE WORKSHOP: The Fairy Lantern

By Andrew Lewis

Recreate a classic stage illusion in small-scale using a Pimoroni Pico LiPo and an SPI screen

You'll need

Pimoroni Pico LiPo
hsmag.cc/PicoLiPo

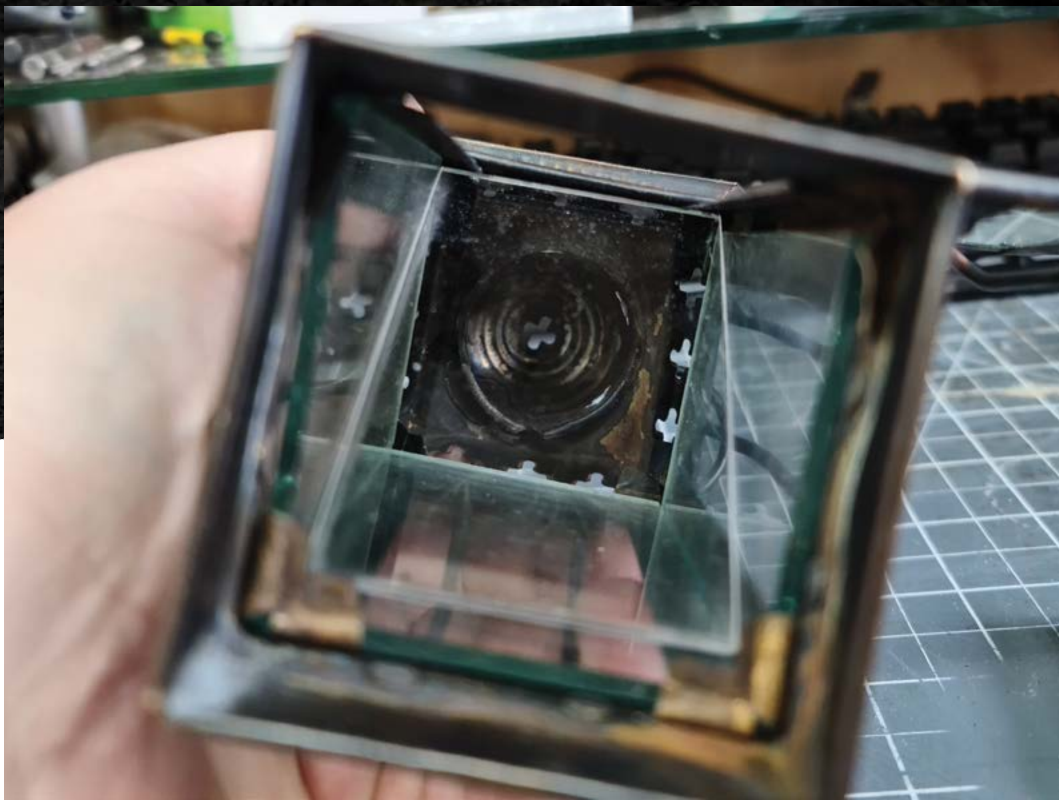
500 mA LiPo Battery
hsmag.cc/BattPack

1.3″ Square SPI LCD screen
hsmag.cc/ColourSquareLCD

Lantern or box with glass or acrylic sides

Sheet of acrylic the correct size to fit inside the lantern

Right You don't need a huge list of parts to make a big visual impact. With these few parts, you can recreate one of the most famous stage effects of the 19th century



Left The clear acrylic you use for the screen reflector should be as close-fitting to the lantern as possible, and free from scratches. You don't need to be at exactly 45 degrees for the illusion to work, so you have some leeway when it comes to finding the best position to reflect the screen

The Pepper's ghost illusion wowed the theatre-going public in the 1860s, spawning a slew of ghost-themed plays. With this project, you can create your own version of this popular illusion to capture a fairy inside a lantern, using the power of an RP2040 board, LCD screen, a suitable GIF image, and transparent acrylic.

The Pepper's ghost illusion is actually much older than you might expect, and was first described in the 1500s. Although it's not a true hologram, the eerie effect that it creates often gets called as such, and is occasionally seen as an advertising gimmick at trade shows and conferences. The illusion uses an artfully positioned pane of glass or plastic to reflect a concealed object or screen in such a way that the image appears to be floating, semi-transparent, in mid-air. The technique is also the basis of how an autocue works: since the illusion is only visible from a certain position, it's possible for a public speaker to look directly at the text of an announcement without the text being visible to those people behind the transparent screen.

IT'LL WORK IF YOU BELIEVE IN IT

To make your own version of the Pepper's ghost illusion, you'll mount an acrylic sheet at 45 degrees to a concealed screen inside a lantern. You'll want the lantern to have glass sides, so that the semi-transparent effect of the illusion is apparent to the viewer. If you try to project the illusion against a plain background, the power of the effect is lost and you might as well just put the screen up there instead.

Before you fit the hardware into the lantern, you'll need to solder the pieces together, choose a GIF image you like, size it to match the size of your screen, and then convert it into a format that the Pico LiPo can process fast enough to display as an animation. After choosing a GIF, the easiest way to resize and manipulate it is with an online service like **ezgif.com**.

Crop or resize your GIF to 240×240, and choose an appropriate compression level. Save the GIF to your computer. You can also apply different playback and effects for your animation, like reverse or ping-pong.

ALL AT C++

Converting the GIF into a useful format for the Pico LiPo is easy, thanks to the work of Larry Bank. Larry has produced some extremely useful code that lets us shortcut a lot of the issues around playing back a GIF and connecting to the screen. So, download the image_to_c application from **hsmag.cc/ImageToC**, and the code from **hsmag.cc/SPILCDCode**. The image_to_c application lets you automatically reformat the GIF file in a way that's more easily handled by the bb_spi_lcd code, which takes the reformatted GIF and displays it on an SPI LCD screen. Assuming that you're using Windows, there's a →

Quick Tip

It's easy to run out of memory quickly when you're dealing with animated graphics. The Pico LiPo has a generous allowance, but keep this in mind when choosing a GIF.

Right It's worth creating a non-reflective black lining for the base of your lantern from felt or foam. The black base with a hole cut through to expose the screen will cut down any unwanted light or reflections from the concealed electronics

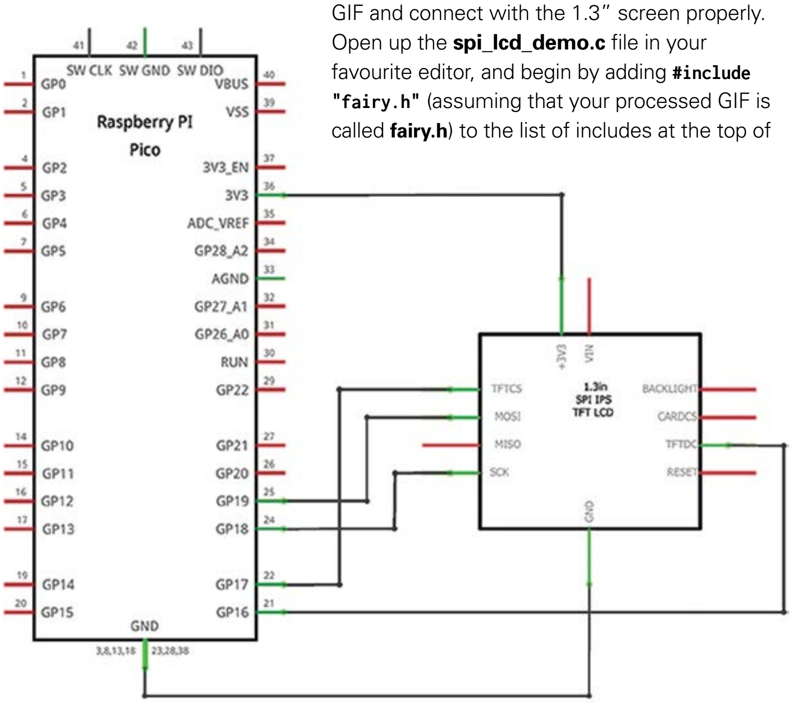
Below Solder the screen to the Pico LiPo, as shown in this diagram. You don't need to connect the BL pin. It's used to turn the backlight off and on, but it isn't used in this project

C++

Capturing a fairy in a lantern is a difficult task, and so is displaying a GIF image on an LCD. You'll be using C++ on your Pico LiPo to process the animation fast enough. Don't panic, while C++ is more complicated to set up than CircuitPython or MicroPython, it's easier than you think to get started. There are some excellent guides about how to get started with Pico and C++ at **hsmag.cc/PicoC++** if you're using Windows, or **hsmag.cc/SDKC++** if you're using Linux. The rest of this article will assume that you have installed C++, the Pico SDK, and the example files.

precompiled **image_to_c32.exe** in the **dist** folder of the image_to_c repository. If you read through the notes for the image_to_c app, you'll see that the application outputs the modified GIF to stdout. That means that if you have a file named **fairy.gif** on Windows and you want to create a properly formatted header file for C++, you'll use the command `image_to_c32.exe fairy.gif >> fairy.h`.

You'll need to make a few changes to the example provided with bb_spi_lcd so that it will load the correct GIF and connect with the 1.3″ screen properly. Open up the **spi_lcd_demo.c** file in your favourite editor, and begin by adding `#include "fairy.h"` (assuming that your processed GIF is called **fairy.h**) to the list of includes at the top of the code. Next, change the value of `#define DISPLAY_WIDTH` and `#define DISPLAY_HEIGHT 240`, to match the resolution of the screen.



Jump down near the bottom of the code and find the `setup()` function. You need to change the call to the `spilcdInit` function so that it uses the correct LCD driver settings. The correct driver is LCD_ST7789_240, so the full line should read:

```
spilcdInit(&lcd, LCD_ST7789_240, FLAGS_NONE,
CLOCK_SPEED, LCD_CS, LCD_DC, LCD_RESET, LCD_
BACKLIGHT, LCD_MISO, LCD_MOSI, LCD_SCK);
```

Finally, find the `main()` and look for the `if` statement that encapsulates a `GIF_openRAM()` call. Change the call to read `GIF_openRAM(&gif, (uint8_t *)fairy, sizeof(fairy), GIFDraw)`. The name 'fairy' here refers to the name of the const defined in the **fairy.h** file you generated from the GIF file, and is normally the same as the original GIF file name.

Save the changes, and you're almost ready to compile and upload the file to the Pico. You'll need to copy the **pico_sdk_import.cmake** file from your **pico-sdk/external** folder into the folder where you've downloaded the bb_spi_lcd repository. This file helps the compiler to locate the Pico SDK on your system. If it isn't the **bb_spi_lcd** folder, then you'll probably get an error when you try to compile. As is traditional, create a folder called **build** and navigate into it. Do `cmake ..` to create your build files, and then use `make` to build the project. After a few screens of hopefully green text have passed by, you should have a file named **spi_lcd_demo.uf2** in your **build** folder. Attach your Pico LiPo to your computer via USB, and put it into bootloader mode by turning it on with the BOOT button pushed down. Copy the UF2 file to the Pico LiPo (which should appear on your system as a drive

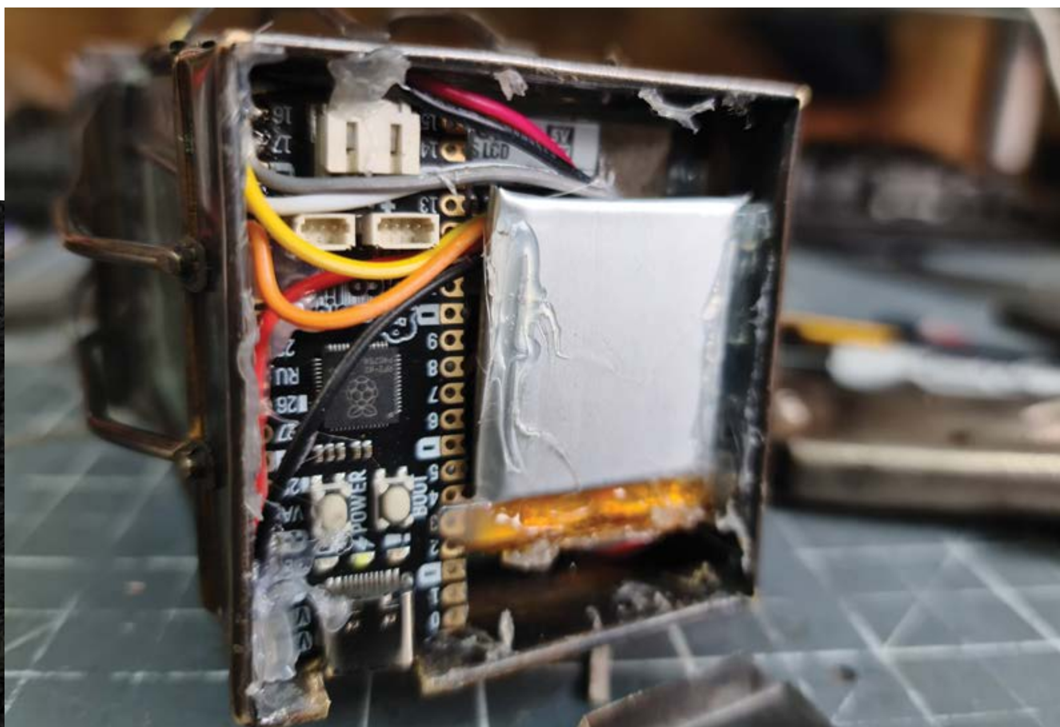
named RPI-RP2). Restart the Pico LiPo and you should see your GIF playing on the screen.

Now that the hardware is done, you can set about fitting it all inside your lantern. The exact instructions for this will vary depending on the size and shape of the lantern that you're using, but there are a few tips that might make the process a little bit easier. Firstly, pay attention to the location of the USB socket and the power button on the Pico LiPo. You'll need access to the USB port to charge the batteries, and the power button to turn the board on and off. You should also be aware of the lights on the board and make sure that they don't interfere with the illusion.

If the base of the lantern is too inaccessible, you could try fitting everything into the top of the lantern instead. As long as the screen is concealed from sight, the illusion should work fine. Keeping your cables short and soldering wires directly to the boards rather than using DuPont connectors can help to keep the footprint of the electronics small.

Chances are high that this project will get handled quite a lot, so more hot glue is probably better than less. If you're not interested in capturing a fairy, you could use the same effect to play a holo-message from the Rebel Alliance, revive a Hogwarts ghost, or visualise the great and glorious Oz in portable format. □

More hot glue is probably better than less



Quick Tip

If you got lost in all of the C++ speak, don't panic. Just copy the UF2 file provided with this project onto your Pico LiPo and it will start displaying a flapping fairy.

Above ⬆ Test-fitting the screen is important to find the best position for the illusion. Expect to assemble and disassemble the parts a few times to get things right

Left ⬅ Thanks to the small size of the Pico LiPo and screen, you can squeeze this project into even the most modest of tealight lanterns

FEATURE

HOW I MADE

By Jo Hinchliffe

UP, UP, AND AWAY!

Take to the skies with an unpowered flying machine

In Roald Dahl's *Danny, the Champion of the World*, there is a small, almost insignificant, side plot, where Danny and his father make a small hot air balloon. It's been on our list of things to make pretty much ever since. Times change, and it's now not a good idea to launch a flaming ball untethered into the skies, as it's at the very least littering and, at worst, is going to cause a fire. Sky lanterns have now been banned by numerous local authorities, but there is still heaps of fun to be had. With that in mind, we jumped right in!

We'd seen some ideas where aerostats, or balloon canopies, were flown in smaller 'hops', heating the internal air using a device on the ground enabling the balloon to perform a shorter flight. For outdoor hops, people are using camping stoves with a tin can chimney to keep the flames off the tissue or, for indoor flying, people were simply using an electric heat gun. This

What I used

- Glue sticks
- Scissors
- Sheets of tissue paper
- Paper to make a pattern
- Cotton thread
- Thin light wire, (MIG welding wire or similar)
- Heat gun
- Long ruler/straight edge
- Large set square

Figure 1 ↑
The CDAS-2 balloon filled with hot air using an electric heat gun, ready to take flight

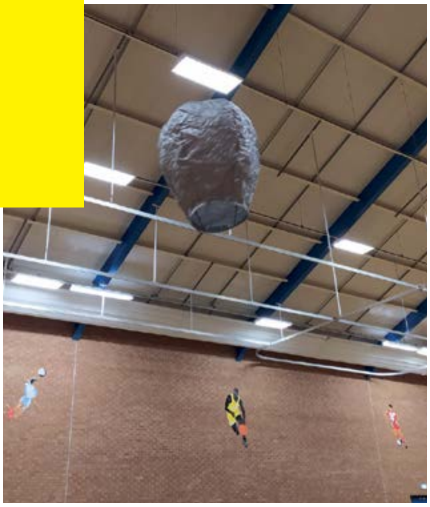
seemed like a safe and easy-to-achieve goal, so we set about making a small balloon example, which quickly escalated to larger balloon designs. We had the idea that we'd like to be able to fly an 'FPV' aerostat, where there was enough lift to haul a small camera, video transmitter, and battery up under the canopy.

As a starting point, we made something around the size of a sky lantern (**Figure 2**). As tissue paper is relatively affordable, it's a pretty good idea to do something small first, as the techniques and gluing sequences are the same on a smaller balloon as they are on the later, bigger designs. In fact, the whole process is similar. You'll need a nice clear table area to work on, and it's worth running your hand over it all to check there are no little bumps or sharp bits that might snag or rip your tissue paper. A bench built out of an old interior door is a great work surface for these projects!

For all the balloon designs, we created a pattern on some brown packing paper, and we've collated all the plans for all three designs as PDFs that you can download at **hsmag.cc/issue61**. All three of our balloon designs are made following the same process, so sometimes we've used pictures from the different builds to illustrate different points. Start by drawing a centreline along a piece of paper to the overall length and then, using a ruler, mark this line at the regular intervals indicated on the plans. For the small balloon, this is every 12.5 cm. Next, use your set square or roofing square to draw a line out from each mark, matching the distances with those on the relevant plan. To keep the patterns simple to draw, we just drew straight lines between each section to create the outer perimeter of the pattern. Having drawn the pattern outline, cut out the pattern.

> ***"YOU'LL NEED A NICE CLEAR TABLE AREA"***

Figure 2 ↓
Our first small test design was capable of short hops for a few seconds



A common tissue paper sheet size is 76 cm × 50 cm, and so, this smaller balloon is made by placing the pattern at an angle across the tissue sheets. The larger designs are made from six panels, where each panel is a pair of sheets glued together, so uses twelve sheets in all. Neatly stack six tissue sheets together, and then place the pattern on the top (**Figure 3**). Next, you pin all the →

Figure 3 For all the balloon designs, we cut the gore panels by pinning the pattern to a stack of six sheets of tissue paper

QUICK TIP

It's very handy to have a desktop fan around as you can use this to blow into the canopies to temporarily inflate them.

way through the pattern and all the sheets in numerous places to keep the pattern and tissue stack all together. Pin close to the edge of the pattern so that pin holes will be folded into your seams. With the pattern and the sheets pinned together, cut around the pattern, cutting all six sheets at once. Don't get too hung up on being incredibly precise through this process – a few millimetres of wavy cut won't make a significant difference to a large balloon construction. As you make your second or third design, you will naturally be better at the process as you refine your techniques.

Before you unpin the panels, just make a mental note of which end is which and, regardless of which design you are building, keep track of which end is where throughout the process. Place the stack away from your workspace, and clean down your bench, ready to begin the assembly process of the gores. This is a complex sequence to describe, so we've also

Figure 5 Having created three pairs of panels, we move on to joining the panels together

created a short online video tutorial that also explains the process, and you can find it here: **hsmag.cc/GoreAssembly**.

Take a pair of gore panels and place them on the bench with the base end of the gores to the left-hand side. Move the uppermost panel away from you to expose a 10–12 mm edge of the lower panel – this will be our seam (**Figure 4**). Add glue to the closest edge of the upper panel, and then carefully fold over the exposed 10–12 mm seam onto the upper panel and press it down. Once this is done, repeat this with your other cut panels, assembling three pairs.

Stack the three pairs so that they are again base to the left. Pick up the entire stack and rotate them over horizontally so that all the base ends are now on your right and the apex is now on the left. The glued seams should still be at the closest edge of the bench nearest to you. Reach over to the far side of the uppermost panel pair, and move the first single sheet back, folding it over so that you can access the lower sheet of the upper pair (**Figure 5**). Pull the entire upper pair back towards you, around 10–12 mm, exposing the uppermost sheet of the middle pair. Glue the edge of the lower panel of the upper pair and fold towards you the exposed edge that we just created from the upper layer of the middle pair to create a seam (**Figure 6**).

Next, fold the entire seam you just created towards you so that we see the lower layer of the middle pair. Again, pull the upper pair and middle pair towards you by 10–12 mm, which will reveal the upper layer of the bottom pair. Add glue to the edge of the lower layer of the middle pair, and then fold over the upper layer of the bottom pair to create another seam.

Finally, move the seam you just created back towards you and take the very top

Figure 4 Creating a seam by offsetting one panel slightly, gluing, and then folding the lower exposed panel over the top

Figure 6 ↑
Creating the second set of seams working on the far side of the panels

layer of the uppermost pair and move it over all the other layers, to be placed on top of the lowest layer which is on our work surface. Move the upper layer back towards yourself, exposing 10–12 mm of the very bottom layer. Apply glue to the top layer and fold the lowest layer over the top towards you to create the final closing seam.

If you have done this correctly, then all the seams you have created should now match when the balloon is inflated.

You can now open out your canopy and check the seams. Using a desk fan can make this easier – on a low setting, hold the base opening of the canopy open near the fan for it to inflate (**Figure 7**).

Don't worry so much that the top of the small balloon is a bit messy and possibly not fully sealed. For all the balloon designs, we created some kind of closing panel and applied it. For this small one, it's really straightforward. Cut a small disk around 10 cm in diameter of tissue paper, and glue it over the top of the balloon. You can use your hand inside the canopy to press it all together. For the larger CDAS-2 design, seen in **Figure 1**, we created a separate collection of panels to seal the apex of the canopy, which we describe later in this article.

To finish the small example, we need to create a canopy ring. This holds the base aperture open at all times and makes it easier to inflate and fly. For the small balloon, our canopy ring was roughly 22 cm in diameter, and used a single piece of 0.8 mm MIG welding wire, though any thin, stiff, and light wire would do. For the larger CDAS designs, the canopy rings are around 40 cm in diameter. For all the balloons, it's worth making the circle temporarily with some excess available, and test fit it into the canopy, adjusting to each balloon. When you are happy with the size, use a small amount of tape and superglue to create the joint. Insert the ring into the canopy and move it back towards the opening at the base. Position the ring around 25–30 mm inside the canopy. This 25–30 mm seam then gets cut into small tabs that we can glue over the ring (**Figure 8**). Once you have a few tabs holding the ring in place, you can move around the complete seam gluing tabs over the ring.

Once everything is dry, you can test-fly your creation! With a balloon similar in size to the small design, you don't really need to tether it as, in our experience, it will only fly a pretty short hop. Heating the canopy with an electric heat gun, we've found ours will hop the height of a tall room, and pause only for a second or two, before floating back down. It's a perfect project for a lighter-than-air demonstration at a school/youth group.

> **"YOU DON'T REALLY NEED TO TETHER IT"**

There was no way our small balloon was going to lift anything, so we knew we had to go bigger! We went on to design two more larger balloons that we called CDAS-1 and CDAS-2 (ConcreteDog Aerostat -1 and 2), and both perform much better. We suggest that you might consider jumping to CDAS-2, as it had numerous improvements over the CDAS-1 design, looks great, flies very well, and, in some ways, is easier to make.

For both the CDAS designs, you need to again cut six gore panels from a pattern. →

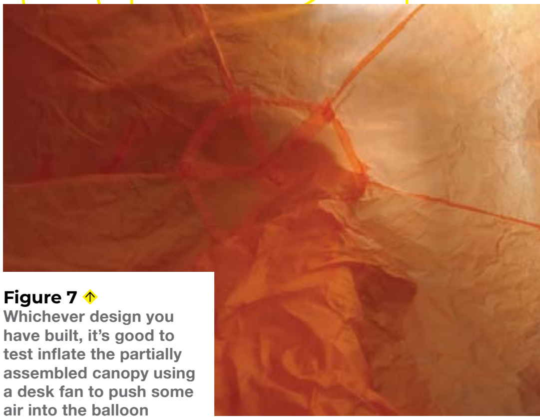


Figure 7 ↑
Whichever design you have built, it's good to test inflate the partially assembled canopy using a desk fan to push some air into the balloon

Figure 8 → We used the same technique to create and install a canopy ring for each of our designs

Figure 10 ↑ Adding a thread loop, stuck on with a patch of glued tissue paper, is a handy way to store and hold your balloon

These panels are a lot larger though and, as such, they are each cut from two sheets of the 76 × 50 cm tissue paper stuck together lengthways. The first task, therefore, is to stick together six pairs of sheets. We used a glue stick for this, and overlapped the sheets by around 15 mm. Once your six larger sheets are dry, stack them up, and pin the pattern to the top of the pile. A couple of notes here are to make sure that the sheet seams that you just glued are all aligned, so that you get a nice straight seam around the canopy, and, if you are making CDAS-2, make sure your sheets are stacked accurately, as the gore panel design fills to nearly the edge of each sheet.

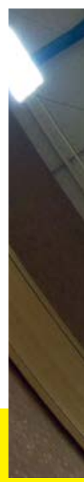
You now need to follow the same process as you did for the small balloon to cut the panels in a stack. Glue the first set of seams, forming three pairs of panels, and then assembling those panels together into the canopy. You'll notice that both the CDAS designs don't come to a point at the apex end of the pattern. For CDAS-1, we experimented with adding a larger disk to cap the top of the canopy. We ended up cutting a disk around 20 cm in diameter and stuck it to the apex of the canopy. Our approach was to stack some small boxes on the floor and balance a large plastic mixing bowl upside down on top of the stack, and then place the canopy over it, the idea being we could then use the bowl as a surface to press onto and form the curved canopy top. Whilst we managed to achieve this, it's not pretty or particularly easy to achieve. With CDAS-1 together, we added a canopy ring in exactly the same way as we did for the small balloon. We again performed a test-hop in a small room, and you could instantly tell this volume of balloon had a lot more lift. However, we didn't really like the shape of the CDAS-1 canopy – it's a bit too pointy in the top half, and we knew we could do better. One thing we did add to CDAS-1, which is now something we'd add to all our balloons, is another small patch of tissue to the top of the canopy with a double loop of cotton thread sewn and knotted through it. This has two functions; one is you can hang the canopy up to store it, but you can also have an assistant hold the canopy upright, which helps a lot when beginning to fill them (**Figure 10**).

For CDAS-2, we again glued pairs of sheets together but, this time, we created a pattern that tried to use most of the width of the sheet, so we got more air volume in the final canopy. We also went for a different approach to seal the apex which is slightly heavier perhaps, but actually adds more volume to the canopy as well. The apex ends of the gore panels are pretty wide and, instead of a disc to seal it, we created and cut 6 × 25 cm-sided equilateral

QUICK TIP

If you snag your canopy and cause a tear, or notice a slight gap in a seam, it's easily fixed with a glue stick and perhaps a small tissue patch.

Figure 9 → For the larger CDAS-2 design, we added six smaller triangle panels to create the canopy apex, using an inflated bin-liner inside to act as a form whilst gluing



triangle panels. With the canopy assembled, we glued one edge onto each of the panels at the apex edges, again using around a 15 mm overlap, and making sure that the triangle panels equally overlapped at the edges of each of the main canopy panels. We then, after some head-scratching, inserted a bin-liner into the canopy and blew this up like a balloon and sealed it with some tape. This bin-liner balloon gave us a similar shape and enough resistance to then glue the overlap seams of each triangle to each other, moving around each of them in turn (**Figure 9**). This worked well enough that we only needed to add a small, 8 cm diameter circle to fully close the apex of the canopy, and we used this circle to add our thread loop accessory. We finished the CDAS-2 balloon with a canopy ring, and we then used a vinyl cutter to make some small lightweight decals for both CDAS-1 and 2!

There's an excellent model indoor flying group locally, and they agreed that we could fly the balloons in the sports hall that they meet in once a month. This gave us the opportunity to test the balloons up to around 18 metres in height. We needed to tether the CDAS-1 and 2 designs, particularly as there was a lot of metalwork and gantries in the roof of the hall, and we wanted some control! To simply add a tether, and also a structure to attach a tiny FPV camera rig, we sewed a double thread across the canopy ring, knotting it on each side. There isn't masses of force or jerking under acceleration with these balloons, so this proved strong enough. We used a lightweight kite line and a tiny fishing swivel which allows the tether to be clipped on and removed. Both the CDAS-1 and CDAS-2 designs flew very well (**Figures 11 and 12**), and could easily reach the roof of the building and hang there for many seconds before slowly floating down. They are both capable of lifting the tether line and our tiny, low-resolution FPV camera/transmitter. The FPV rig is an Eachine TX06 camera and VTX (video transmitter), and we found a 100 mAh LiPo cell. In all, this weighed around 7 grams, (**Figure 13**). It certainly proved the FPV aerostat concept, but we're planning an upgrade in quality as a future goal. Finally, it's worthy of note to say this is great fun! It's cheap, affordable, and creates a large object that flies well. It's certainly cheap enough that we are pondering going even bigger with our next design! □

Figure 12 →
We were really pleased with the shape that the CDAS-2 design takes when filled and in flight

> **"IT'S WORTHY OF NOTE TO SAY THIS IS GREAT FUN"**

QUICK TIP

Filling these balloons with a hot air gun is an excellent approach, but you must be very careful to not touch the tissue, as it will easily scorch and burn.

Figure 11 ←
The CDAS-1 design around ten metres up, flying well

Figure 13 →
CDAS-2 at launch with its tiny FPV camera, video transmitter, and battery payload

Flat-pack rockets

An aviation device fit for a Swedish furniture superstore

Jo Hinchliffe

@concreted0g

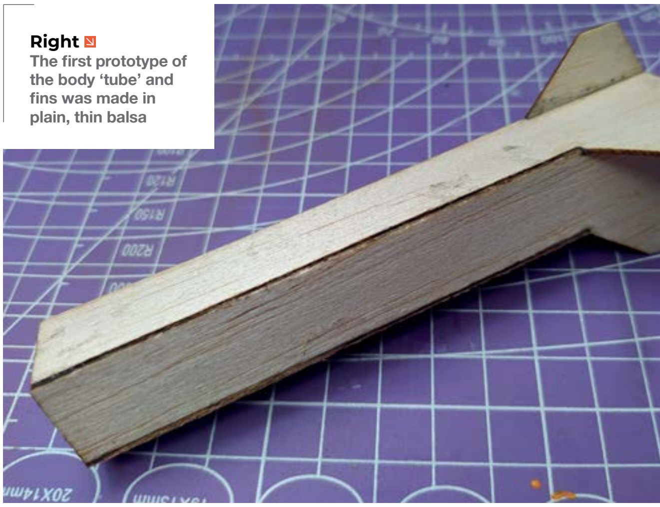
Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY space. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open-source. He also has a shed full of lathes and milling machines and CNC kit!

Above Take-off of the EXO-B, the larger of the two flat-pack rocket prototypes

Working on the EXO-S swing-wing rocket glider project, which was featured in HackSpace issue 56, saw me working with balsa wood, and I started thinking more about building rockets with sheet materials rather than tubes. Mainly as a challenge for fun and learning, but it struck me that there are advantages to using sheet material, as well as a fair number of problems to solve!

Lightweight tubes are delicate in one axis – they stand up well in compression along the length of them, so resist the general flight forces well. But, if you take a model rocket and give it a squeeze, it's incredibly easy to crush them, buckling the body tube quickly beyond repair. The buckling issue actually comes up a lot of the time when mail-ordering kit. After their journey through a postal system, they often end up a little damaged. It struck me early on in this project that a 'flat-pack' rocket kit from sheet materials could be easy and affordable to mail to people, with a good chance of survival.

I set about making sketches. My first thought was to make a long, thin four-sided pyramid, which I still think is an elegant idea, but it is a challenge to make parts of a pyramid separate, like a nose cone. I quickly came to the idea of a square-sided body, and it struck me that it seemed silly to cut and place fins on each face of the body when I could simply extend the fin out of each side of the body. This creates offset fins, but this should still create a balanced airframe in flight. The other area I considered was the nose cone. I went through numerous sketched ideas looking for the simplest solution. I considered a tapered,



Right The first prototype of the body 'tube' and fins was made in plain, thin balsa

Left Both the large and small rockets use internal bulkheads to hold the motor in position

flat-sided cone first but, again for simplicity, I moved towards the idea of a nose cone that was curved in one axis and flat-sided. As I was building the prototype, I was thinking I would make the entire nose cone hollow and bend a thin wall to form the curved side. However, I ended up going with an even simpler idea. Other early decisions were to make a really small prototype, essentially to see how the panels would align and generally work, and then to make a larger rocket, about the average size of a beginner's model rocket. I'm a sucker for a tiny rocket though, and decided that the small prototype would have to be a flying model!

> **"I'd decided to use balsa sheet in various** thicknesses for the majority of the airframe"

The other challenge with creating rockets away from standard tube components is how to create a successful design that will fly well. If you are designing a more standard rocket, you can use excellent open-source tools like OpenRocket, but there aren't tools available for more unconventional designs such as these. I've scratch-built a fair few rockets over the years, so a lot of my decisions were based on rules of thumb and previous experience. I did use a few older trusted techniques from the rocketry community to make some estimations and help inform the design. I used Inkscape to create the design files, and I used either my budget CNC diode laser rig (reviewed in HackSpace issue 47) and a CO_2 laser at a local makerspace to cut the panels. However, don't let a lack of laser put you off. You could print out your designs, cut out card templates, and then use these to mark out and cut your panels.

I'd decided to use balsa sheet in various thicknesses for the majority of the airframe as it's widely available, affordable, and lightweight.

Internally, I needed to have some bulkheads that would act as a motor mount. Again, in more standard designs, tubes are sold in a variety of materials which match the outer diameter of standard motors, but I wasn't allowing myself to stray from the flat-pack mantra! I went with a tubeless design that used a series of bulkheads that allowed a motor to be slid into place. These were cut from 1.5mm plywood as they needed to be strong enough to resist the thrust forces from the burning motor. The bulkhead at the base of the rocket has small bolts glued into position so a further external bulkhead can be bolted on to hold the motor in. Inside the rocket, the upper bulkhead stops the →

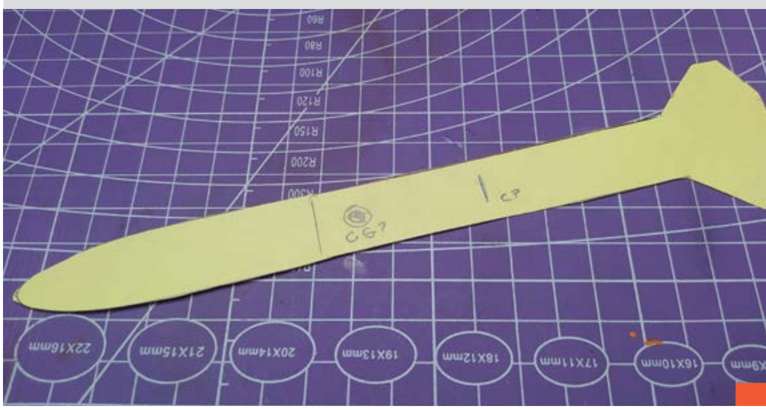
QUICK TIP

The very first piece that I contributed to HackSpace magazine was a walkthrough tutorial on how to design and simulate a model rocket using OpenRocket, back in issue 12.

Right Adding laminating film reinforces the lightweight balsa and adds a glossy low-drag finish. Sublimation printing adds fabulous decorative finishes

PRESSURE PUSHING DOWN ON ME

If you are designing a rocket using software design and simulation tools such as OpenRocket, you will probably know about the 'centre of pressure'. The centre of pressure is a point on the rocket about which all the forces on the rocket centralise. Therefore, the centre of pressure is most affected by the dimensions and sizes of the rocket airframe components, the diameter of the tube, the length of the nose cone, the size of the fins, and more. With our odd-shaped square rocket, that is trickier to simulate. We can use an older technique that gives us an estimation of where the centre of pressure is. The technique is to cut out a cross-section representation of the rocket design, and cut it out of cardstock. For this project, that's pretty easy to achieve from the drawn designs. With our cardboard cut out, we then find the point at which the cut-out balances – this point is the estimation of our centre of pressure. On the real rocket, the centre of pressure should be ideally somewhere between one and two calibres behind the centre of gravity. A calibre, in this instance, is the diameter of the body tube. Of course, our tube is actually a square, so we need to make an estimation there as well!



motor being pushed into the rocket during flight, but allows the ejection charge that pushes off the nose cone to pass through.

With my first few parts cut for the lower section, I set about assembling the first prototype. Gluing the first two sections with fins at 90 degrees to each other is probably the hardest part, and I used some square-sided blocks to help me keep things true. Once a couple of the internal bulkheads are positioned and checked for squareness, it all gets a little easier when adding the third and fourth panels. For this smaller design destined to fly using a tiny 13 mm diameter Estes motor, I used balsa sheeting that was 1.3 mm in thickness. I find sheet balsa a little variable in terms of stated thickness, so I went through my pile and graded sheets by taking numerous measurements with a set of callipers. Once assembled, I was pleased with the prototype in terms of lightness and I was also pleased with the rigidity of the square section around the bulkheads. But, I was very concerned with the strength of the fins which were very flimsy; also, away from the bulkheads, it was getting a little too flexible.

A LAYERED APPROACH

A popular approach for increasing the strength of balsa is to create a composite structure by adding a layer of another material. At this tiny scale, that might be gluing tissue paper, or thin card, or even super-fine fibreglass. However, I had an idea I wanted to try! I'd seen in RC plane-building communities that people use laminating film, which is essentially the same as office laminating pouches, to cover foam models as a reinforcement layer. This also collided with me researching craft communities using laminate films as a recipient for sublimation printing – meaning you can heat-press really nice artwork onto laminate-covered objects.

Some research ensued as I wanted to find a brand of laminating film that would be laser-safe for laser cutting. Some laminating pouches may contain vinyls which release chlorine gas when burnt and, as such, are very hazardous for human health when laser-cut, but also can cause corrosive damage to machines and extraction systems. However, there are also many laminating films that exist that are made from HDPE and PET layers, which are both considered laser-safe. Finding laminating film products that have their material contents listed, or in a datasheet, is pretty uncommon though, so it might take a while!

I pressed some laminating film onto a piece of the thin balsa using a modelling iron set to around 140 °C, and pressed each section for around ten seconds.

Figure 1 For both the smaller and the larger rocket, I went with a stack of balsa profiles glued together to create the nose

Left The EXO-B micro prototype on the launch pad, ready for its first flight

Initial results were great, in terms of increasing the resilience of the material without adding much weight. There are a few challenges, however: as the laminate film cools, it can warp the balsa material. For the small prototype, I laminated both sides of the very thin balsa, but later, for the larger rocket, I laminated only a single side. Either approach could create a little warping. After some experimentation, I found that a few approaches helped. First, heating/ironing the balsa on both sides before adding the laminating film helped to remove any moisture, and then adding the laminating film seemed to make it a little more stable. If warping did occur still, I found covering the part in greaseproof paper, heating it for another ten seconds, and then placing a heavy, flat weight on it and allowing it all to cool slowly for five minutes helped get a stable flat part. I was already extremely pleased with the structural qualities of the laminate/balsa composite, and felt the fins would definitely survive the violence of a launch, so I set about experimenting with adding graphics by thermal-transferring sublimation print designs to them. Massive thanks to the Ffiws Maker Space in Porthmadog, North Wales, for allowing me to use the sublimation printer for these experiments. I printed out some colourful patterns using an online digital camouflage generator to create the images, and then had a few attempts at transferring the print using a range of heat and pressure settings. As the laminating film is not particularly designed for this, I was using guesses at first and then adjusting. I found that a pretty high amount of pressure, and a temperature of 130°C for around 20 seconds, was pretty good in terms of transferring the design. You would probably get more depth of colour if you pressed for longer, but you run the risk of creating heat bubbles in the balsa/laminating film material.

> "**I printed out some colourful patterns using an online** digital camouflage generator to create the images"

Alongside working on the laminating and sublimation parts of the project, I had preliminarily designed a nose cone idea. I guessed that the overall design would need some weight at the nose cone end to create a stable airframe, so decided to make the top half of the nose cone out of a stack of balsa sheets glued together (**Figure 1**). I also wanted to include a payload section, an accessible area into which I could add an altimeter to tell me how high it flew. Below the stack of nose cone parts, the rest of the nose cone is closed with a sheet of thin balsa glued on one side and then also has a hinged side made with a piece of tape →

Above Cut sections of the larger EXO-B model in the laser cutter

Below right An altimeter and logging device packed into the larger EXO-B payload bay

and closed with a tiny M2 bolt and a nut epoxy glued into a small balsa block inside the payload bay. The lower section of the nose cone, or 'shoulder' that fits inside the lower section, has some small plywood bulkheads added, to which the recovery system shock cord is attached. This keeps the rocket parts and the parachute or streamer all connected, so you hopefully get it all back. For the prototype, I simply cut the nose cone parts from non-laminated balsa to work out how it would all go together and wasn't worried about appearances. But it gave me a complete rocket that I could test and then go on to launch!

With the nose weight added and the tiny PerfectFlite FireFly altimeter in the payload bay, it was time to set out and test-fly the EXO-B micro prototype. The motor of choice was a tiny 13 mm Estes A3-4T motor. On a nice calm day, we set up the launch equipment and pressed the button. Tiny rockets like this disappear very quickly off the pad and this one didn't disappoint. We managed to follow it up visually and it deployed nicely, the two halves of the rocket separating well. The small Mylar streamer we had included, which creates drag but is more packable than a parachute, unfurled and helped us track the rocket back down to the ground. Recovering the EXO-B micro, it was completely undamaged and, on examining the altimeter which logs the peak altitude and the maximum velocity, it showed very promising results! The peak altitude was 498 ft (151.79 m), and the maximum speed was 141 mph (227 kph).

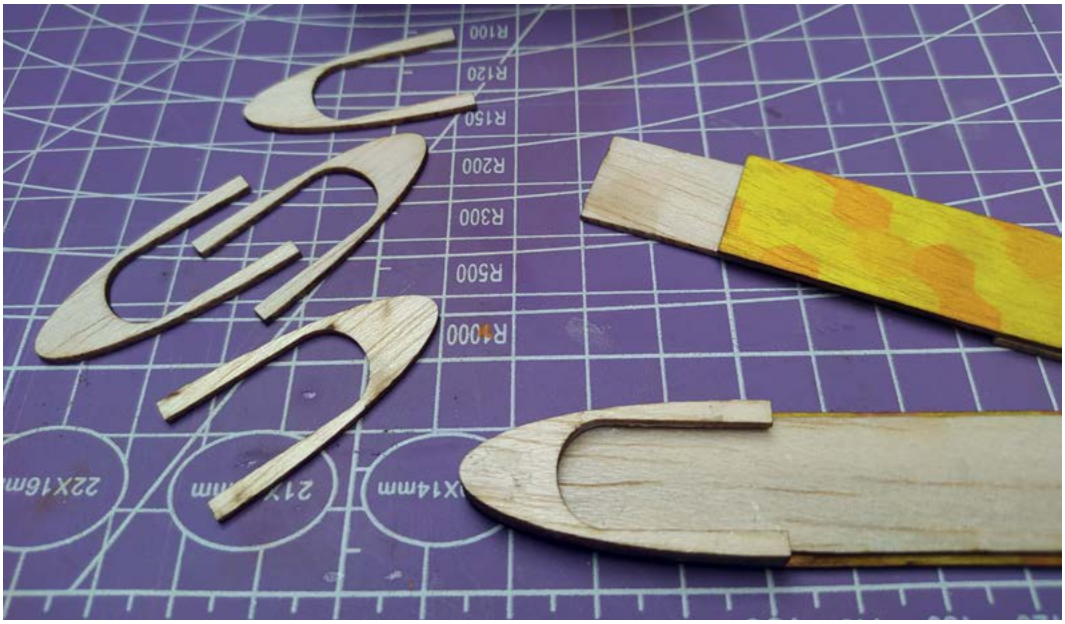
Moving to the larger EXO-B version, which flies using 18 mm diameter motors, it was largely a case of scaling up the design. I sublimated the digital camo pattern onto the balsa as I thought it looked cool, but I really should have gone with a brighter, more visible colour scheme! There's only a couple of differences in the construction of the larger EXO-B. A small obvious part is that I used only a single side of laminate onto the thicker and sturdier 3 mm balsa sheets. I also laminated and sublimation-printed the larger sheets of

YE OLD SIMULATOR

One of the difficulties with odd-shaped rockets is that you can't always use existing software simulation packages to check out the stability of the design. However, thousands of stable model rockets were designed, made, and flown way before free simulation software existed. One technique which is well worth learning and practising is the swing test. To perform a swing test, you need to attach a piece of string to the rocket at its centre of gravity. It's important that you do this with the rocket fully loaded with a motor and recovery system and any payload, so it's as it will be at a real launch. Find the point at which the rocket balances level and attach the string here. I tend to tie a loose loop around the rocket and then hang it and adjust until the rocket is balanced, and then add some tape to keep the loop on the centre of gravity. Gently begin to swing the rocket around in a big circle. I start off with about a metre of string and slowly let some string out so it ends up flying in a circle around 1.5 to 2 metres away from me. If the rocket is stable, the nose of the rocket should point in the direction of flight and it should fly level. Even if you start with the rocket the wrong way around or at an angle, it should correct into nose first, level flight. If it's unstable, the rocket may sit at an odd position or may oscillate and act chaotically. If unstable, adding nose weight usually helps. For swing tests, I often have some plasticine handy so I can temporarily add a little weight to experiment with.

QUICK TIP

I hold the current UK altitude record flown on the same motor, which was 703 ft (214 m). With the EXO-B micro getting two-thirds of this performance in a very under-optimised design, that's an excellent result.



balsa and then cut all the parts out of the sheet, rather than sublimation-printing directly onto the cut parts. The payload bay was made in a similar fashion to the EXO-B micro, but I used a bolt at each end to secure the cover. I installed some larger altimeters, as this rocket required the extra weight and I had a larger area to use, but I ran the battery connector cables out into the area below the payload bay to allow me to connect the power on the launch pad and not have to screw and unscrew the cover. Again, this larger airframe needed more weight in the nose cone, so I drilled a 10 mm hole into the centre of the tip and superglued a similar-sized ball bearing into it. As I had a little spare of the laminated printed material, I cut a matching 10 mm disc to patch up the hole on the outside.

BLOCKED-UP NOSE

One revision for both sizes is to change the stacked layer of balsa pieces that form the nose cone. I want to expand the payload area forwards into the nose cone, meaning weight can be pushed forward. This should help to create the stability the rockets need, but without adding as much ballast, meaning that the weight placed for stability is useful weight, like an altimeter.

The larger EXO-B version's first flight was on a slightly breezy day and it turned into the wind a little, which tells me it's a little over-stable. On an Estes B6-4 motor, which is a go-to motor for first flights, it achieved 366 ft (112 m). It's large enough to contain a parachute rather than a streamer, and I used a bright red one to try and help find the camouflaged rocket after landing! It's on the edge of being a little underpowered, and I'll probably test it again with a slightly more powerful motor. I also think that I could build another this size, but set it up for 24 mm motors which would increase the power availability a lot!

So, I've learnt loads and definitely proven that we can make decent rockets from flat materials. Although they should definitely be considered works in progress, I've published the part designs on this repository, should anyone like to build their own – **hsmag.cc/FlatSheetRockets**. For me, I'm already working on an improved version of the 13 mm micro one, which has the hollow nose cone idea implemented. I'm also considering if I could build a 13 mm motor one that is so lightweight it might get close to my current UK altitude record. Watch the skies for updates!

Above
For the next build of the EXO-B micro, I've adapted the nose cone stack to create more room to position nose weight and payload

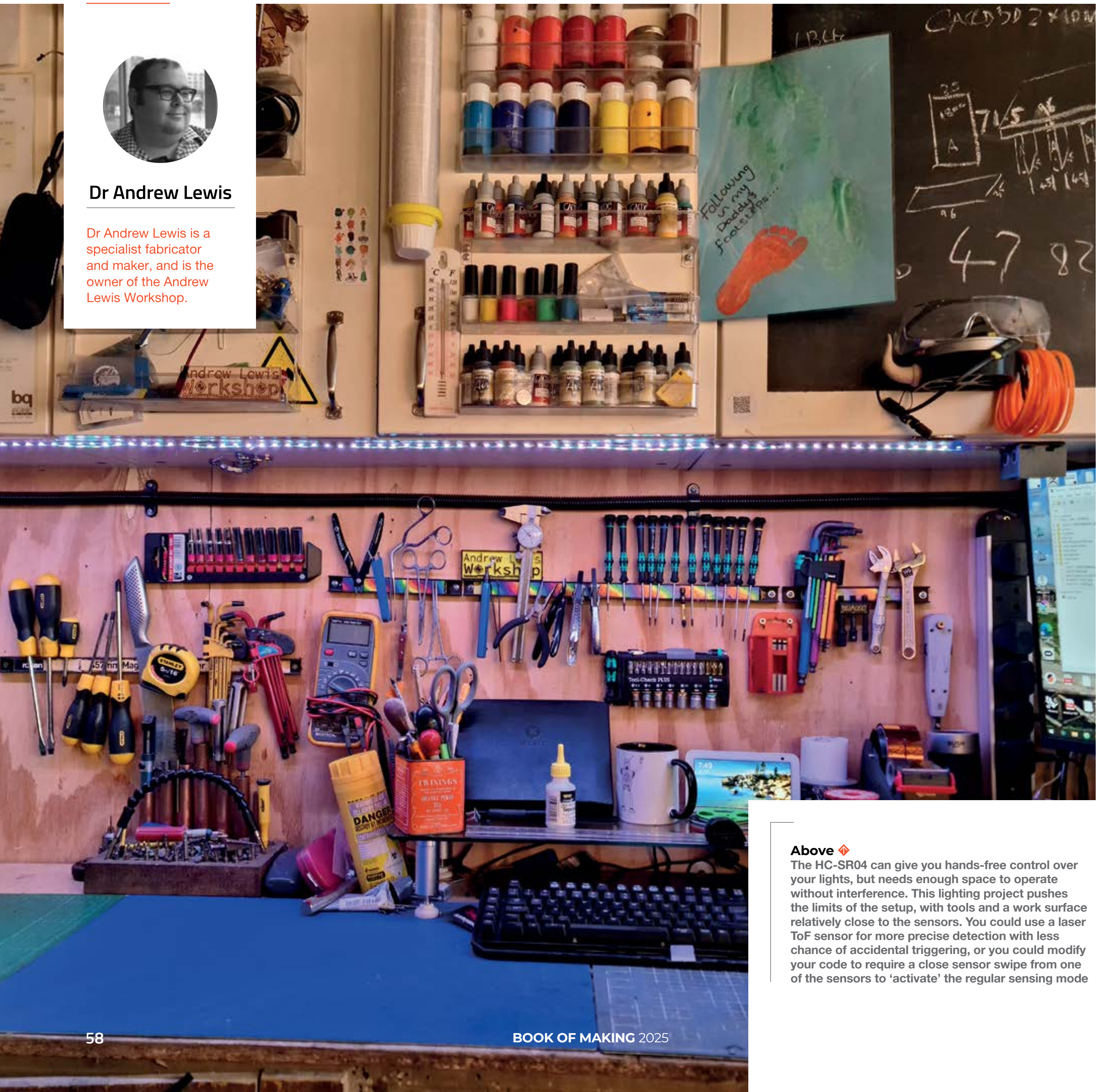
Below
Building another EXO-B micro in a pleasing hexagon-patterned livery

Hands-free lighting

Shine a light on your projects without ever having to flick a switch

Dr Andrew Lewis

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



Above The HC-SR04 can give you hands-free control over your lights, but needs enough space to operate without interference. This lighting project pushes the limits of the setup, with tools and a work surface relatively close to the sensors. You could use a laser ToF sensor for more precise detection with less chance of accidental triggering, or you could modify your code to require a close sensor swipe from one of the sensors to 'activate' the regular sensing mode

QUICK TIP

The cone angle of the HC-SR04 ultrasonic sensor means that building a light over a couple of metres will be impractical. For longer lighting runs, try using a VL53L1 laser time of flight (ToF) sensor instead.

Left If you're feeling ambitious, a fourth sensor could be used to control the colour temperature of the lights, to provide a warm or cool working environment

It isn't always easy to turn a light on and off in the workshop. Your hands are covered in oil, the switch is too far away, and speech control only works if you can remember the name of the light you want to control. With a Raspberry Pi Pico, a strip of NeoPixel lights, and some ultrasonic sensors, you can create a dimmable strip light that only shines where you want it to. In this project, you'll learn how to control NeoPixel LEDs from a Raspberry Pi Pico running MicroPython, and how to use multiple ultrasonic sensors to detect the position of your hands.

To paraphrase Adam Savage, we all need more light in the workshop. This is particularly true as we age. Our lenses become less clear, the parts of the eye that detect contrast changes begin to die off, and the muscles that control our iris become less able to react to changes in light. These changes can affect our daily lives in a number of ways, but most notable is the need for more light when working on fine projects.

The problem with just adding more lights is that you don't always want the light to be everywhere. Badly positioned lighting will shine in your eyes and blind you, rather than help you to see. Constantly turning lights on and off disrupts your flow while you're working, and it isn't always easy to adjust the position of a light if your hands are covered in oil or wet paint.

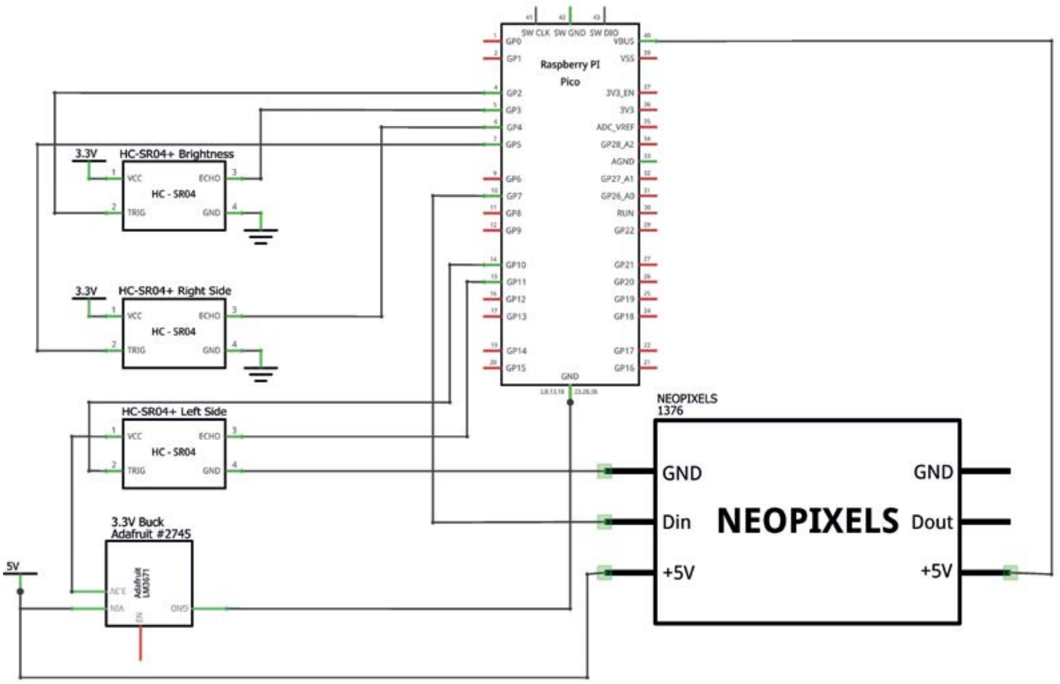
> "The problem with just adding more lights is that you don't always want the light to be everywhere"

SWIPE RIGHT TO LIGHT

This project addresses these problems by using three ultrasonic sensors to control a strip of NeoPixel lights. Two of the sensors are mounted at each end of the strip of lights, and measure the distance to your hands. The remaining sensor is mounted perpendicular to the lights and allows you to adjust the brightness of the light strip. By positioning your hands in the air between the sensors, you can control which area of the light strip will be illuminated, so that you only turn on the areas that you need to use. →

YOU'LL NEED

- **1 × Raspberry Pi Pico**
- **1 × NeoPixel/ WS2812 LED strip**
- **3 × HC-SR04+ ultrasonic sensor**
- **MP1584EN DC buck converter, set to output 3.3V**
- **2.5mm panel mount DC power socket**
- **Suitable 5V DC power supply** (Your requirement will vary depending on NeoPixel length)



Right Connecting the circuit together is not complicated, and pin choices are more to do with convenient positioning of cables than any special features of the Pico

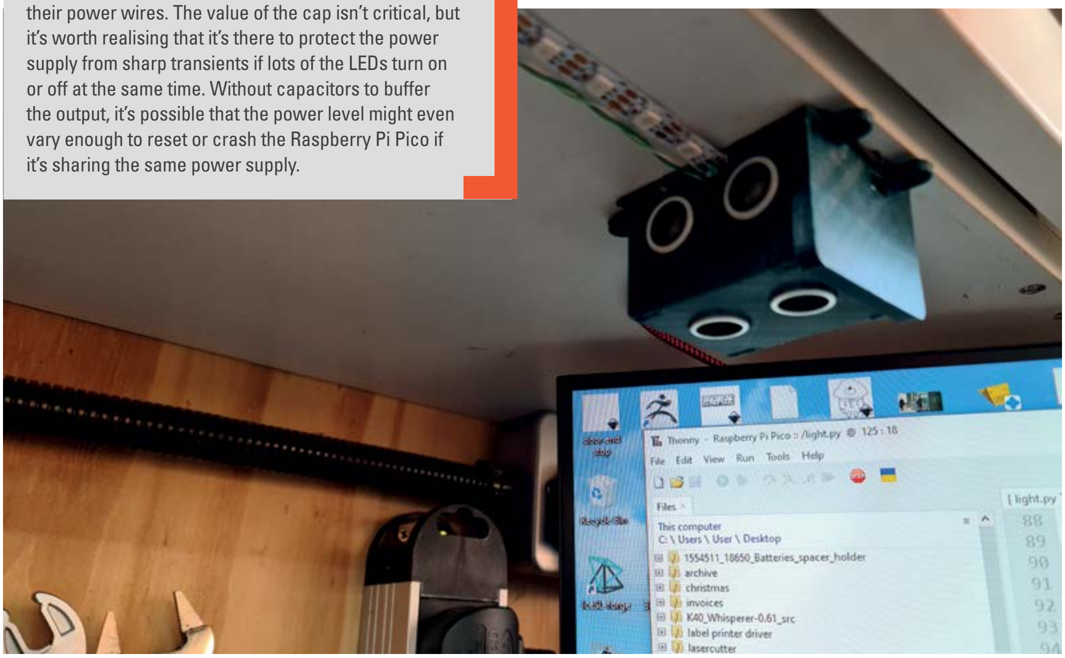
Below You can mount the light using double-sided foam tape. However, the adhesive used on some LED strips can lose its effectiveness in rooms where the temperature gets very warm or very cold. In these cases, some clips and screws can be a better choice

POWER PROBLEMS

NeoPixel strips run from 5V, and they are power-hungry beasts. The more NeoPixel LEDs you have per metre, the more power you are going to have to feed to them to keep them happy. For longer lengths of NeoPixels, you'll need to inject an extra 5V power into the strip every few feet, or your LEDs will start to show the wrong colours and generally misbehave.

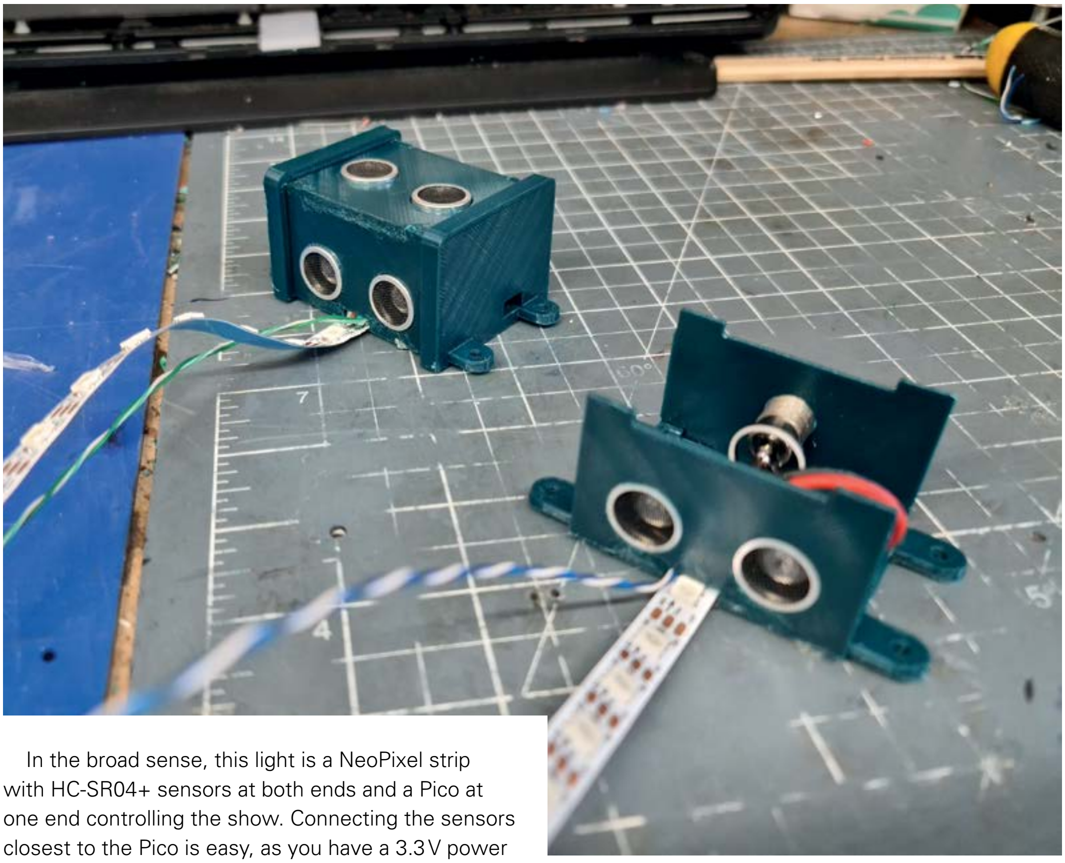
If you look at the circuit diagram for this project, you'll see that the LED strips have a capacitor across their power wires. The value of the cap isn't critical, but it's worth realising that it's there to protect the power supply from sharp transients if lots of the LEDs turn on or off at the same time. Without capacitors to buffer the output, it's possible that the power level might even vary enough to reset or crash the Raspberry Pi Pico if it's sharing the same power supply.

While the Raspberry Pi Pico doesn't require much power at all, once you add multiple ultrasonic sensors and a length of LED lights into the equation, you're going to need quite a chunky power supply to keep everything running and, if your NeoPixel strip is over two metres, you should probably think about adding multiple power supplies to keep the LEDs looking at their best. If your lights are on the shorter side, you should be able to get away with using a single power supply for everything.



QUICK TIP

A simpler version of this project could use the ultrasonic sensor input to turn a light or relay on or off using a simple hand gesture.



QUICK TIP

If you feel up for a challenge, why not use a Raspberry Pi Pico W, and give the lights some additional Wi-Fi compatibility?

Left Twisted pair wire will help reduce the possibility of interference over longer cable lengths

In the broad sense, this light is a NeoPixel strip with HC-SR04+ sensors at both ends and a Pico at one end controlling the show. Connecting the sensors closest to the Pico is easy, as you have a 3.3 V power connection that you can use built right into the Pico, and the cables between the sensors and the Pico are nice and short. The sensor at the opposite end of the light is a different matter. The sensor is sitting at the far end of the NeoPixel strip, but needs connecting to 3.3 V power and the trigger/echo pins need to be connected to the Pico's GPIO pins.

NO MORE OILY SWITCHES

Solving the power connection issue is simple. If the power input for the light is at the far end of the NeoPixel strip, the Pico can use power from the strip itself via the VBUS pin. In this way, the Pico will also function as a step-down voltage converter for the two sensors connected to the 3.3 V pins. This leaves the remaining sensor at the opposite end of the cable, which can be connected to a step-down power converter set to produce 3.3 V. For the trigger and echo pins, it's advisable to make the connections with twisted pair copper wire to reduce the possibility of interference over the longer length of the wire.

With all of the circuitry complete, you can position your light. Remember, the HC-SR04+ has a 15-degree cone, so the detection area will be quite large, and you'll need to make sure that there are no obstructions in the detection area. If you need a more precise sensor, consider the VL53L1 laser-based ToF sensor instead.

POTENTIAL PROBLEMS

The most common ultrasonic sensor used by makers is probably the HC-SR04, which has 5 V logic. This is a problem for Pico users, since the Pico uses 3.3 V logic, and isn't tolerant to 5 V signals. One option here is to use a level shifter, like the TXB0108, to act as an intermediary between the sensor and the Pico, ensuring both receive the correct voltages on their signal lines. Alternatively, the slightly more expensive HC-SR04+ is a low-voltage version of the sensor that run directly from 3.3 V without needing the level shifter.

The HC-SR04+ has a 15-degree cone. Reading that value from the data sheet might make you think that the effective size of the sensor area increases as the distance from the sensor increases. To put it another way, the further you are from the sensor, the more chance that you'll accidentally wave your hand into the area that the sensor is monitoring. It's actually a bit more complicated than this, because the cone varies between somewhere around 40 degrees and 15 degrees, depending on the distance from the sensor. The practical upshot is that this light can't be positioned somewhere that it's likely to encounter interference from objects statically positioned in or moving through the cone, although things like parallel walls aren't generally a problem because the angle of incidence between the beam and the surface will see the echo reflected away from the sensor.

The next quirk that you need to consider is that you'll be using multiple sensors, two of which will be facing each other. That means you'll need to fire each sensor in sequence and wait for a response before firing the next to avoid the sensors interfering with each other. It's also worth noting that other ultrasonic devices like distance gauges and car parking sensors could also interfere with the sensors used in the lights.

The most unexpected issue encountered with these sensors is that flexing the PCB can shift the position of the sensors, which affects the distance reading slightly and can, in some cases, cause an object not to be detected after a certain distance. Making sure the board isn't under any physical stress is important to maintain accuracy.

3D-printed linkages

Get moving in the right direction

Ben Everard

@ben_everard

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

Making a mechanism move is easy – you can add a motor or a handle to turn. However, making things move in the right way, if you don't need simple spinning, is much harder. Converting between a relatively limited set of options for input movement and whatever particular motion your project needs requires thought, planning, and more often than not, a linkage.

A linkage is a series of bars usually joined together by pivoting joints. By varying the lengths of the bars and the way they're joined, we can create incredibly complex movements. Perhaps the most famous complex movement is Theo Jansen's Strandbeest, which uses a linkage to convert the spinning of a wind turbine into a lifelike walking action. Most linkages aren't quite so on display, though. You'll find linkages in many mechanical devices. A few more common linkages are as follows.

Right Linkages are used to transfer the movement of controls on motorbikes

The Ackermann steering linkage can be used to join the front two wheels of a car so that they turn correctly. The challenge here is that the wheel on the inside of a bend needs to turn slightly further than the wheel on the outside. This is because it's slightly closer to the centre of curvature and, therefore, following a circle with a smaller radius.

Watt's linkage was born of the Industrial Revolution. James Watt needed a way of converting the reciprocating motion of his steam engine into linear motion. The solution – Watt's linkage – wasn't perfect, as the resulting motion isn't quite a straight line, but it was close enough for his purposes. It's still used today in the rear suspension system of some cars.

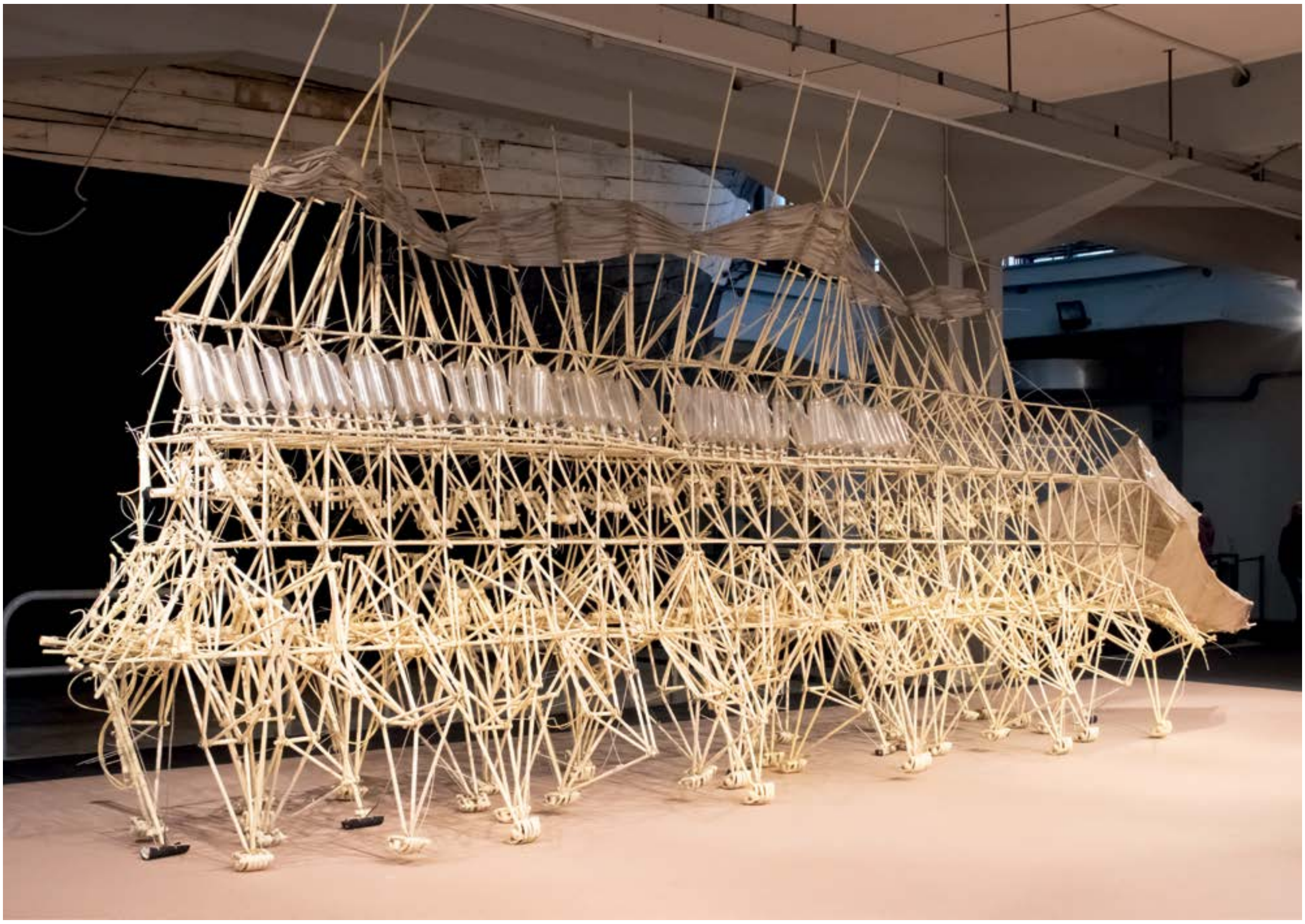
The Chebyshev lambda linkage is another approximation of a straight line. We saw this four-bar linkage in HackSpace magazine issue 66, where it was used to drive the wings of a flying eagle.

FREE AS IN FREEDOM

One of the key features of a linkage is the number of degrees of freedom. Basically, this describes the way in which it can move. A rigid structure has zero degrees of freedom.

A single point, unencumbered by any joints of fixings, has three degrees of freedom. It can move in any direction. You can reduce the number of degrees of freedom by linking it to something. This could be by making it stationary (in the real world, this would equate to it being fixed to a larger body or the ground), or by linking it to another point using a fixed bar.

For example, if you took one point and anchored it to the ground, that would have zero degrees of



freedom. If you added another point linked by a bar hinged around the fixed point, you would then have one degree of freedom.

Many useful linkages have only one degree of freedom. And this means that they move in very predictable ways.

JOINING LINKAGES

Linkages are mostly quite simple to design because they are made up of solid bars. The only real challenge is how to join them together. Usually, you want to join bars together with hinge joints that let them rotate in one axis. Generally, these want to be strong, low friction, easy to print, and small. This is quite a challenging set of requirements, and there are a few options with different pros and cons. The challenge usually comes down to the pin that joins the two parts of the hinge together. Is it part of one of the bars? A separate piece? Something additional that's not 3D-printed? Let's look at the options:

> "Linkages are mostly quite simple to design because they are made up of solid bars"

- **Attached pins:** You can simply print your linkage so that the pin that goes through the hinge is attached to one of the bars. This isn't particularly common because it's hard to secure the other bar to the pin in a satisfactory way. The pin is also the weak point, and this means that if it breaks, you have to replace the entire bar, not just the pin.

- **External pins:** In this method, each bar has a hole and you also print some pins to fit through. It's quite common to make these snap-fit, but →

Above Though not 3D printed, the Strandbeest is one giant linkage that walks down the beach Credit: Nigel Hoult, CC-BY-2.0

Above A four-bar linkage gives this box a slightly unusual hinged lid

we've found these to be a bit tricky in practice. PLA is quite brittle so, while it can work, it's often a very fine line between designing something that has enough overhang to stay in place, while not requiring too much bend that it breaks. There is also no good print orientation for these. If you print them vertically, the layer lines go around the pin, which can make the snap-fit mechanism delicate. If you print them horizontally, the layer lines go along the pin and the joint can have a lot of friction. This isn't to say that these can't work – external pins are probably the most popular way of joining 3D-printed linkages – however, there are a lot of compromises that can make it tricky to design a satisfactory part.

- **Machine screws:** If it's tricky to 3D-print the external pins, then the next best option is to make them out of a standard piece of hardware, and that often means machine screws. These can work well, but they need a locking nut or a double nut to avoid coming loose. Also, these will

DESIGNING LINKAGES

If you're already familiar with FreeCAD, the Sketches tool can help you design linkages. It's not really intended for this purpose, but it's a 2D tool with an understanding of degrees of freedom and connections.

To do this, first start a new project, then go to the Part Design workbench and create a new body, then a sketch. You can pick any plane for this. This sketch won't directly lead to the final design – it just gives us a space to play with different setups and constraints. As an example, we'll create a parallel movement four-bar linkage.

The first thing you need to do is create four lines with their ends linked by coincident constraints. They should be roughly in a square, but we'll set the exact lengths later. The coincident constraints work like hinges. You can click and drag one and it'll move the whole linkages around. Make sure that you haven't accidentally created any additional constraints – sometimes FreeCAD adds horizontal or vertical constraints when you create lines.

If you're not sure how to do any of this, take a look at our *FreeCAD for Makers* free e-book, which goes through the basics of how to work with FreeCAD.

You can now click and drag any point and it'll move the linkage. However, at this point, it'll move about in all sorts of ways – there are far too many degrees of freedom, not least because the links aren't a fixed length. The next step is to add 'Fix a Length' constraints to all the linkages. The exact lengths don't matter, other than the fact that opposite sides need to have the same lengths. We went with 50 mm for the bottom and top, and 20 mm for the sides.

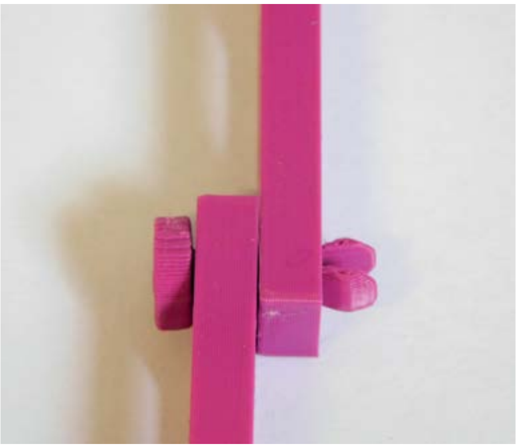
Now, if you click and drag parts, it should move like a rigid object. However, it might still move around. We need to fix one of the bottom corners in place using the Lock constraint. This fixes it in place. Locking one of the corners still allows the whole thing to rotate, though. We could lock the other bottom corner in place, but it's hard to get it lined up properly before this, so it's slightly easier to add a Horizontal constraint to the bottom bar. At this point, there should be one degree of freedom left (unlike a typical sketch, we're not looking to fully constrain it). This mechanism is called parallel movement because if you move one of the sides, the other side moves and remains parallel. You should find that if you move either of the top corners, that's exactly what happens.

This is a fairly straightforward four-bar linkage. There are plenty of others, and some can create complex movements. Once you've created a sketch with the movement you want, you can take a note of the positions, lengths, and joins, and then design the parts for your linkage.

The FreeCAD Sketches tool is a great way of getting a feel for a linkage, but it's not the only option. You could also consider:

- Cardboard and split pins. OK, this isn't the most technically advanced method, but cardboard and split pins are quick and easy to use, and are a great way to get a feel for how a mechanism will move.

- There are some linkage simulators available online, for example **hsmag.cc/desmos**. While these can be useful, you might find them limited, as they tend to be designed to create a specific type of linkage.



Above The external pin in this join snaps into place, but leaves the joint a little loose

wear quite quickly under heavy use, as the teeth dig into the plastic. They are also very heavy in comparison with 3D-printed parts and this can affect the way in which a part moves.

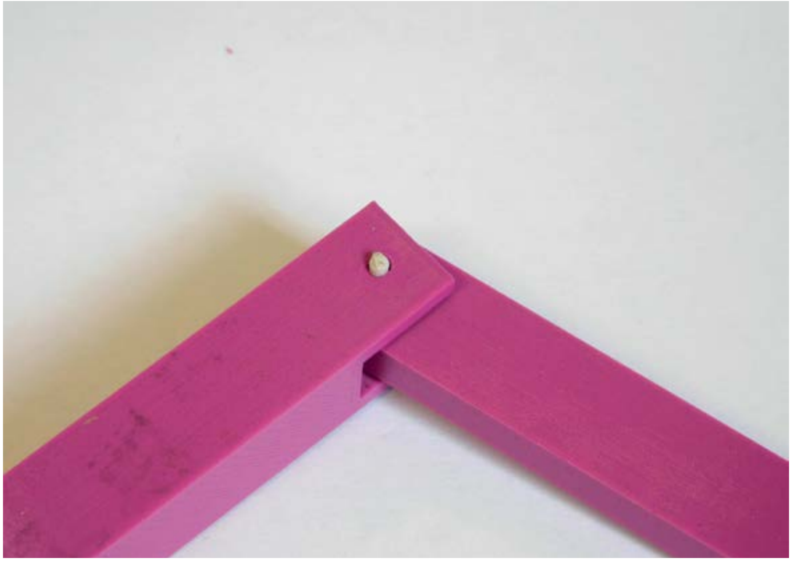
- **Filament:** There is another option that can give you the best of both worlds – use a short length of filament as the pin in the hinge. Filament has a standard and predictable width (typically 1.75 mm, though might be 2.85, depending on your printer). It's cheap, circular, and, because it doesn't have layer-lines, it's strong and has lowish friction. You'll either need to very precisely size your holes or use a drop of glue to hold it in place. When printing the holes vertically, we find 2 mm holes work well because they collapse slightly at the top. For small- to medium-sized linkages, these are our favourite joints. They're easy to design, strong, smooth, don't have much slop in them, and don't require anything other than filament and maybe a little glue.

WHAT FILAMENT?

Generally, you want linkages to be stiff, and PLA is the stiffest common 3D printer filament.

The only other significant consideration might be friction. If you need your linkage to keep running with minimal lost energy, then a material with a lower coefficient of friction, such as nylon, might be worth considering.

It's also possible to use some stock material to make the bars, and just 3D-print the hinges. Plastic, carbon fibre, and metal bar stock are all available, each with different properties, and it would be easy to 3D-print the joints between the bars to make the linkage.



Above A short amount of filament in a 2mm hole gives a free-moving strong joint

- **Bearings:** You can get bearing cartridges cheaply. The most common ones are standard sizes for skate wheels, but other options are available. They're cheap and very low-friction, and can be a great option if you need your linkage to run very freely or survive a lot of use. However, they are very heavy in comparison to 3D-printed parts. They certainly have their place, but are probably overkill for most cases.

- **Print-in-place:** Some of the most impressive 3D-printed mechanisms work straight off the print bed. While this can look cool, it's often not the most practical option for linkages because it's very hard to balance friction, sloppiness, and the tolerances you need for print-in-place. That's not to say that it can't work – we've seen some great print-in-place linkages – but it should be used with caution.

Linkages are fantastically useful mechanisms, as they convert one type of movement into another. With electronics, we often work with rotary motion, whether that's from servos or motors. Linkages can let us use this to control all sorts of different hardware. There are lots of standard linkages for common movements, so you may find that you can just borrow the work of some 19th-century industrialist for your next design.

The biggest challenge with 3D-printing linkages is making sure that they move freely without too much slop. We've had success using filament for the hinge, but other things can work well. □

Changing motion with linkages

A round of applause for ourselves

Ben Everard

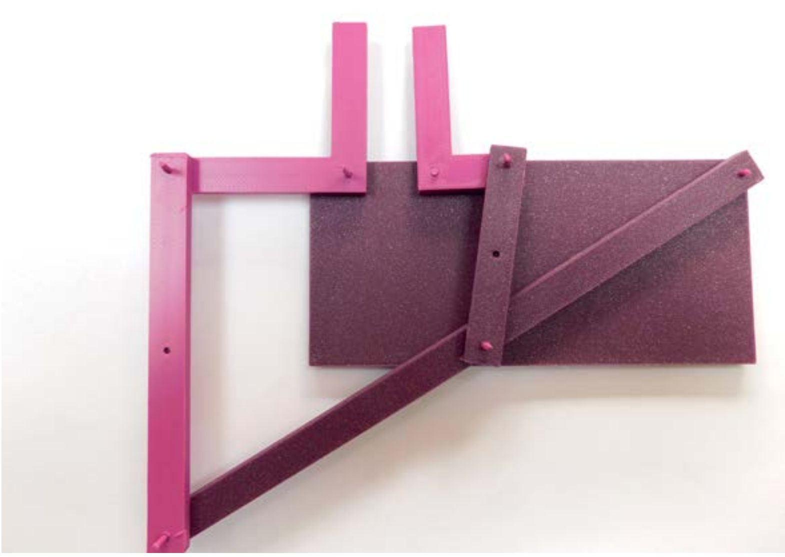
Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

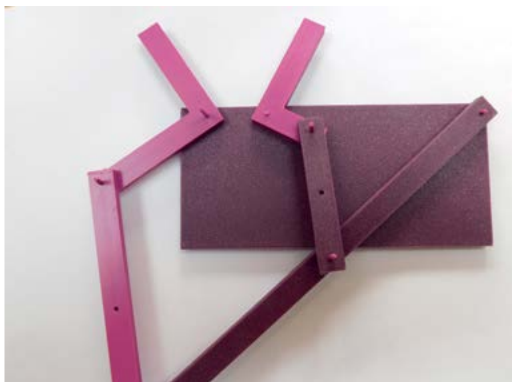
We've already looked at linkages – ways of joining together moving parts using solid bars. Now, we're going to design one. We want to create an arrangement of rods and joints that can only move in one way, and that's to make a pair of hands clap. We want both hands to move toward each other.

Alternatively, hands can be on rods that rotate around a point. However, we need one to rotate clockwise and one to rotate anticlockwise.

'Bell (crank) linkage' is a name for an L-shaped rod attached at the angle in the middle of the L. This linkage gives us a way to transfer motion between different directions, because pushing the base of the L up makes the top of the L move sideways. In this way, we can transfer (almost) vertical motion into (almost) horizontal motion. We need to do this for both hands, so we have two Bell linkages. By flipping one around, we can get anticlockwise motion on one linkage and clockwise on the other. This does present a problem because one of the linkages will be further along the arm than the other, and the further along the arm it is, the more motion it will get as the arm moves. We can simply offset this by making the arm on one of the Bell linkages longer than the other.

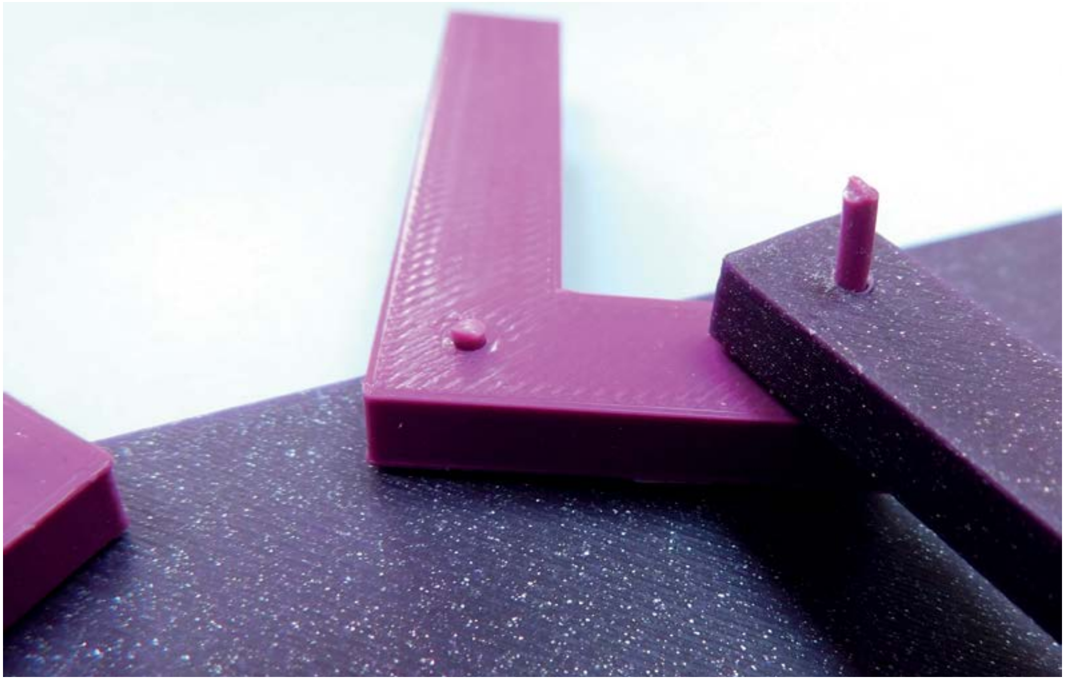
This doesn't quite solve the problem though, as we still need a way to actuate this pair of Bell linkages. We opted for an arm on a pivot. If we attach both linkages to this then, as we rotate the arm, it will move the two Bell linkages. However, one of the Bell linkages will be further out than the





Above Open …

Left … and shut



Left
Push the filament into the hole, and trim the end for a strong pivot

other, so will rotate further. The solution to this is to make one of the Bell linkages have a proportionately longer base to the L, and then the leverage effect counteracts the fact that it moves further.

> **"Spend a bit of time looking at other linkages and** familiarise yourself with some of the simple ones"

Once the plan was sketched out, calculating the lengths of everything was just a little trigonometry.

DESIGN DECISIONS

As with many linkages, once the design is finalised, everything seems very straightforward. This doesn't mean it was straightforward to design. The best advice we can give for designing linkages is to spend a bit of time looking at other linkages and familiarise yourself with some of the simple ones. Don't forget the importance of CAD – Cardboard Assisted Design. With an old box, some scissors, and some pivots (pretty much anything you have to hand can work, from split pins to screws), you can quickly test out some designs. We've tried different techniques for joining our rods together and came to the conclusion that short sections of filament performed well, so we've continued to use these. Most of the links are, therefore, bars with 2 mm holes in them at the ends. These are really easy to draw in FreeCAD. We created an object that took data from a spreadsheet and made the bars suitably sized.

The Bell linkage bars are slightly different because these need to have 90-degree bends in them. We created them using two sketches – one with the outline of the bar which we then padded, and one with the holes that we then pocketed.

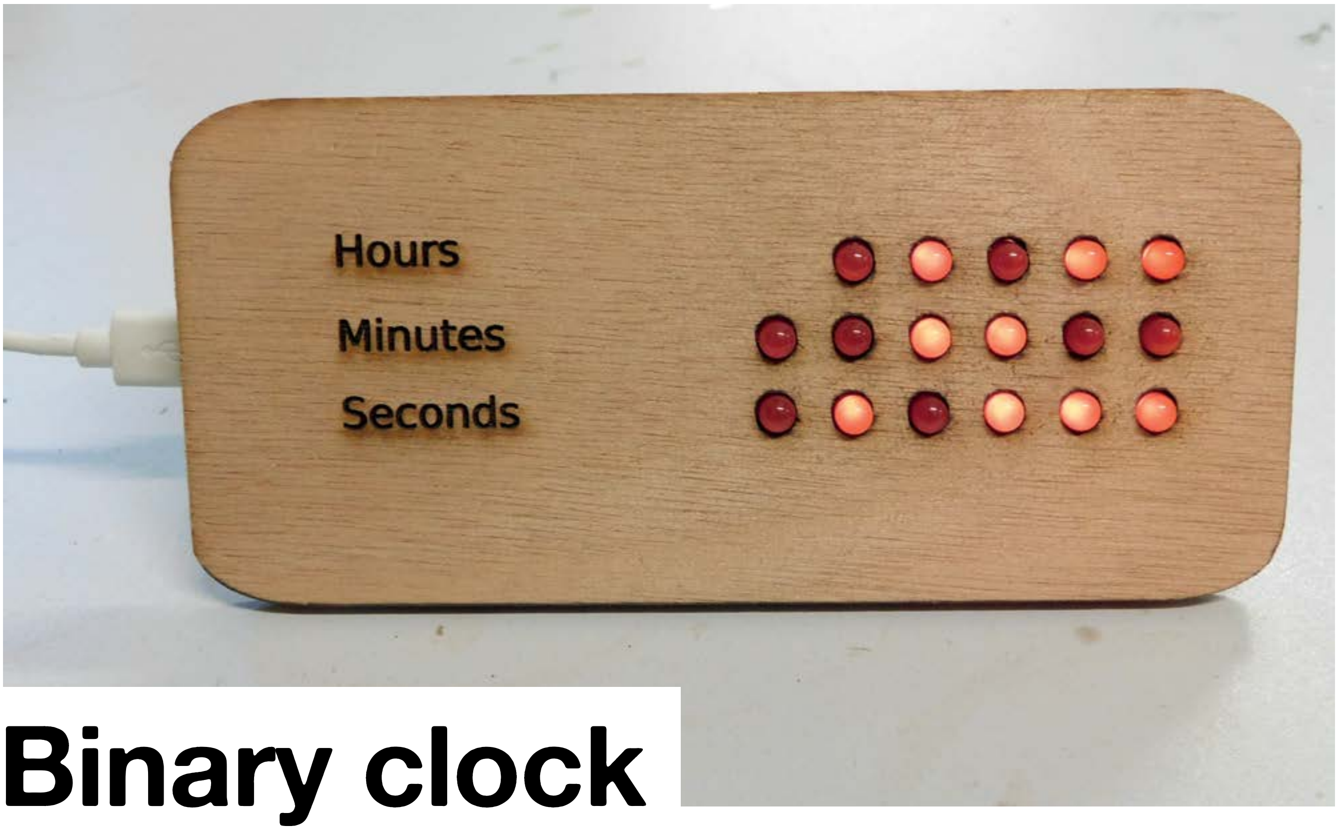
Finally, you need to add a base, whose sole purpose is to hold everything together.

Once we had the linkage set, the only thing left was to add the hands. We traced around our hands and cut them out of cardboard. A bit of sticky tape holds them onto the linkage. Of course, you could print out hands for this if you prefer. □

EXTENDING

It would be fairly easy to extend this project by linking the main bar to a wheel that you could turn, either by hand or with a motor. This would make it easy to generate a lot of applause. Equally, you could use the movement to do something else – clang some cymbals together or get a pair of hands to wave goodbye (or hello).

Above
Tape on a pair of hands and clap away



Binary clock

Keep track of time, and learn to count in 1s and 0s

Ben Everard

@ben_everard

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

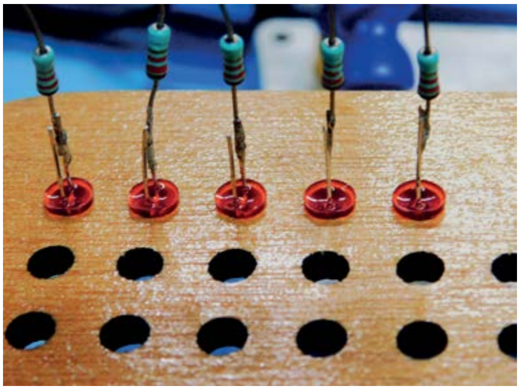
Binary is the key concept in how computers operate. If you drill down far enough on any operation, transmission, or storage, you'll find 1s and 0s, on or off, positive or negative. This ability to build up to complex behaviours from a simple distinction between, and manipulation of two states, is what computing is all about.

Yet, despite this being absolutely fundamental to the way computers work, it can be a bit alien to us humans. We're brought up in a world of ten digits, not just two, so it can be hard to get an intuitive sense of binary, especially if you don't use it very much. Our solution is to make a clock that displays time in binary.

We're going to use MicroPython for this because it's quick and easy to get started wirth, and has access to Pico's timekeeping features. While MicroPython is fairly standard across several boards, this particular project requires having an on-board real-time clock (RTC), so won't work with all boards. We're going to be using a Pico W, but it might work on others that have both wireless networking and RTC hardware. Along the way, we'll learn how to connect Pico W to the internet to get the time, and how to use Pico's Real Time Counter to keep track of the time in hours, minutes, and seconds. Let's take a look at this first.

There's a protocol for getting time on a network, and it's unimaginatively named Network Time Protocol (NTP). This actually does a few things around synchronising time over a network with various degrees of lag, but we don't need to worry about being a few milliseconds out, so we're just going to grab the current time.

We're also not going to worry about daylight savings time. We're doing all this in MicroPython, so we're going to take the simplest solution – twice a



year, you'll connect to your clock and adjust the offset manually. If you want, you could add a toggle switch that flips back and forwards to adjust for daylight savings.

We suspected that setting the time on an RTC via NTP might be a popular thing to do, so we wrapped up the example code for NTP in a simple library that does just this – you can get it from **hsmag.cc/SimpleNTP**.

You'll need to copy the **simple_ntp.py** file over to your device – you can do this by opening it in Thonny, then selecting 'Save as' and MicroPython device. Remember, it must have the same name on the device.

We'll shortly look at what's in that file, but for now, we'll use it to grab the time and set the RTC. You can do this with the following code:

```
import network
import socket
import time
import struct
from machine import Pin
import simple_ntp

led = Pin("LED", Pin.OUT)

ssid = 'your_ssid'
password = 'your_password'


wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(ssid, password)

max_wait = 10
while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    max_wait -= 1
    print('waiting for connection...')
    time.sleep(1)
```

```
if wlan.status() != 3:
    raise RuntimeError('network connection
failed')
else:
    print('connected')
    status = wlan.ifconfig()
    print( 'ip = ' + status[0] )

led.on()
simple_ntp.set_time()
print(time.localtime())
led.off()
```

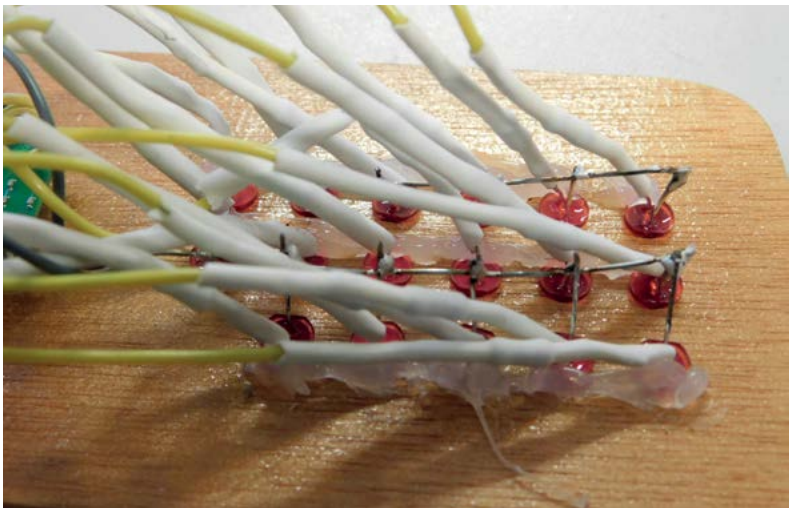
Obviously, you'll need to put in your network's SSID and password, but other than that, it should just run. At the end of it, it'll print the result of `time.localtime()`. Firstly, we should note that this is a misleading name as it won't print your local time, it'll print the time in GMT. Secondly, it won't actually print the time in a usual format, it'll print it as a list. The items are: [year, month, monthday, hour, minute, second, weekday, yearday]. All of these should be set correctly. →

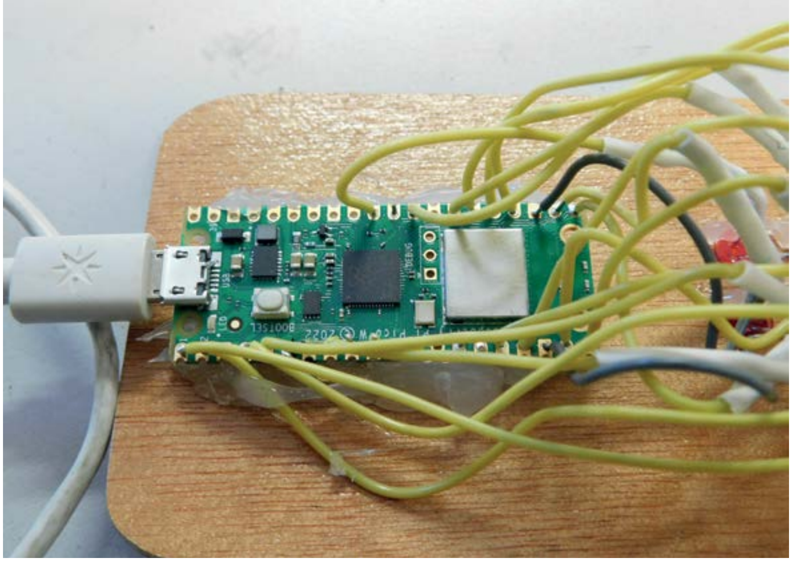
DIMMING

Our program turns the LEDs all the way on or all the way off. This is fine for us, and we can control the brightness by picking the right resistor. However, you can control the brightness of LEDs by flicking them on and off really fast. This is known as pulse width modulation (PWM). Pico W does support PWM but is limited to just 16 pins, so we can't use it in this project. We could create a PIO program that took in details of all the pins that should be on, and do a global PWM across all of them, but this is more than we need.

Left Soldering resistors in-line is an easy way to build circuits without a circuit board

Below The ground wires connect across all the LEDs





Above Pico W is a great choice for this because it's cheap, has a lot of GPIOs, has WiFi for time, and has an RTC

This uses `simple_ntp.set_time()` to set Pico W's RTC to the correct time. We won't go through the whole of this, but let's look at a few key bits. The function is defined with:

```
def set_time(offset=0, delta=2208988800,
host="pool.ntp.org"):
```

There are three optional parameters here. The first is `offset`, which is a number in seconds that you want to set your RTC's time to – this is useful if you want to set it to a particular time zone. The second is `delta`, which you can probably ignore most of the time, but it's the difference between the way time is encoded in NTP. The final parameter is the NTP server. Again, `pool.ntp.org` is probably a good choice for the vast majority of cases. This isn't a specific server but a domain that will resolve to an NTP server close to your location. By being close to you, it should (in theory at least) have a low latency and, therefore, you'll get a more accurate time. However, for our purposes, the difference between high- and low-latency servers should be irrelevant.

Now we've got our time, we will need to create the hardware to display it. Here, it's up to you what you want to do. We obviously need a lot of LEDs, but how you mount them depends on what you have and what look you want to create. You could create a bare-bones look on perfboard. You could design a PCB for it. You could get some wood and drill holes to mount the LEDs in. Or, you could 3D-print a front panel for it. We opted to laser-cut ours because we wanted a wooden front panel, but any of these options should be reasonably straightforward.

Our laser-cut panel has holes for 5 mm LEDs in a line, so you can push the LEDs in place and secure them with a drop of hot glue.

The circuit is quite simple because the output is in binary, and our GPIOs work in binary. Each GPIO connects to a current-limiting resistor, then to the positive leg of an LED (usually the longer leg), and the negative leg of the resistor connects to ground. There are more LEDs than ground connections available, so you'll need to connect them all together.

The resistors need to be at least 220 Ω, but you might want to make them higher to limit the brightness of the LEDs. This will depend a lot on the particular LEDs you have, and how bright you want them. We found that around 1 kΩ usually works well, but it's obviously worth experimenting to see what you like.

> We've used the classic 5 mm red LEDs that give a retro feel, **but 3 mm LEDs will give you a more compact design**

You can use any LEDs that work with 3.3 V. They can be any colour and any form factor. We've used the classic 5 mm red LEDs that give a retro feel, but 3 mm LEDs will give you a more compact design. You could use surface-mount LEDs if you're making it on a PCB. You could even get creative and use 3 V flex LED 'noodles' to build a more abstract-looking clock.

We need five pins to display the hours (in 24-hour format) and six each for minutes and seconds. That means 17 LEDs in total on the GPIO pins 0 to 16.

We found it easiest to first solder the LEDs and resistors together, then put the LEDs in their holes, and then solder the connectors to Pico W's GPIO pins and ground. One thing you need to be aware of is making sure that there's no risk of short circuits if anything gets bent. We did this by covering the LED positive leg and resistor in heat shrink. That way, all the ground wires are exposed, but it doesn't matter if anything shorts between them. It would perhaps be a bit more secure if the grounds were also protected – it's up to you how you set it up. If you don't have heat

NEW TO MICROPYTHON?

If you've not used MicroPython before, you have an exciting time ahead of you! It's a great way of programming Pico and Pico W with very little setup. It's powerful, and user-friendly. There's a quick guide to getting started at **hsmag.cc/DocMicroPyth**. For a more in-depth guide, take a look at our book *Get Started with MicroPython on Raspberry Pi Pico*, which you can download for free or buy in print at **hsmag.cc/mpbook.**

shrink, you could also cover it with electrical tape, but given how much there is to do, it might be easier to just get some heat shrink.

Now we've done the hardware, it's time to turn our attention back to the software. The main thing we need to do is convert the time as numbers into the LEDs that we want to turn on and off. We've done this in a method called **display_num**. This method takes two arguments: the number you want to display, and a list of pins on which to display it.

Converting a number to binary isn't too hard. We've used **enumerate(pins)**. This is a useful Python method when you want to loop through an iterable but still want a counter – as you can see, it returns both.

Each loop, we want to know if the number remaining is greater than or equal to the current digit. If it is, we light that digit up and take the value of that digit away from the number, if it's not, we turn that LED off and move to the next number.

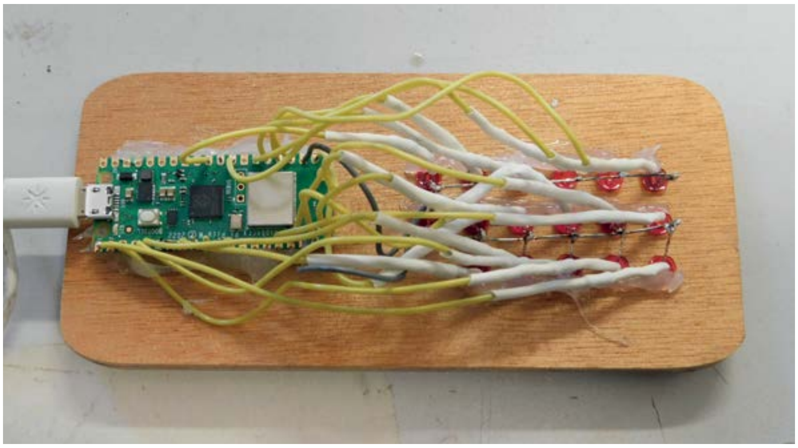
Since Pico W's RTC can keep track of time, we can just grab the time when the computer boots up and let Pico W take care of the rest.

Here's our final code for the clock:

```
import network
import socket
import time
import struct
from machine import Pin
import simple_ntp

led = Pin("LED", Pin.OUT)

ssid = 'your_ssid'
password = 'your_password'

hour_pins =  [Pin(4,Pin.OUT), Pin(3,Pin.OUT),
Pin(2,Pin.OUT), Pin(1,Pin.OUT), Pin(0,Pin.OUT)]
minute_pins = [Pin(17,Pin.OUT),Pin(16,Pin.
OUT),Pin(15,Pin.OUT),Pin(14,Pin.OUT),Pin(13,Pin.
OUT),Pin(12,Pin.OUT)]
second_pins = [Pin(26,Pin.OUT), Pin(22,Pin.
OUT),Pin(21,Pin.OUT), Pin(20,Pin.OUT),Pin(19,Pin.
OUT),Pin(18,Pin.OUT)]

wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(ssid, password)

max_wait = 10
while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    max_wait -= 1
    print('waiting for connection...')
    time.sleep(1)

if wlan.status() != 3:
    raise RuntimeError('network connection
failed')
else:
    print('connected')
    status = wlan.ifconfig()
    print( 'ip = ' + status[0] )

led.on()
simple_ntp.set_time()
print(time.localtime())
led.off()

def display_num(number, pins):
    for counter, pin in enumerate(pins):
        if number >= pow(2, len(pins)-
(counter+1)):
            number = number - pow(2, len(pins)-
(counter+1))
            pin.value(1)
        else:
            pin.value(0)

while True:
    display_num(time.localtime()[3], hour_pins)
    display_num(time.localtime()[4], minute_pins)
    display_num(time.localtime()[5], second_pins)
    time.sleep(0.1)
```

This is a very bare-bones clock, but you can extend this basic timepiece in any way you like. You could add an alarm, a date function, a time control, or anything else you'd like to keep track of. □

Below We attached Pico W with hot glue, but you could also use screws in the mounting holes



Recycling PLA

Turn filament scraps into beautiful new objects

Ben Everard

@ben_everard

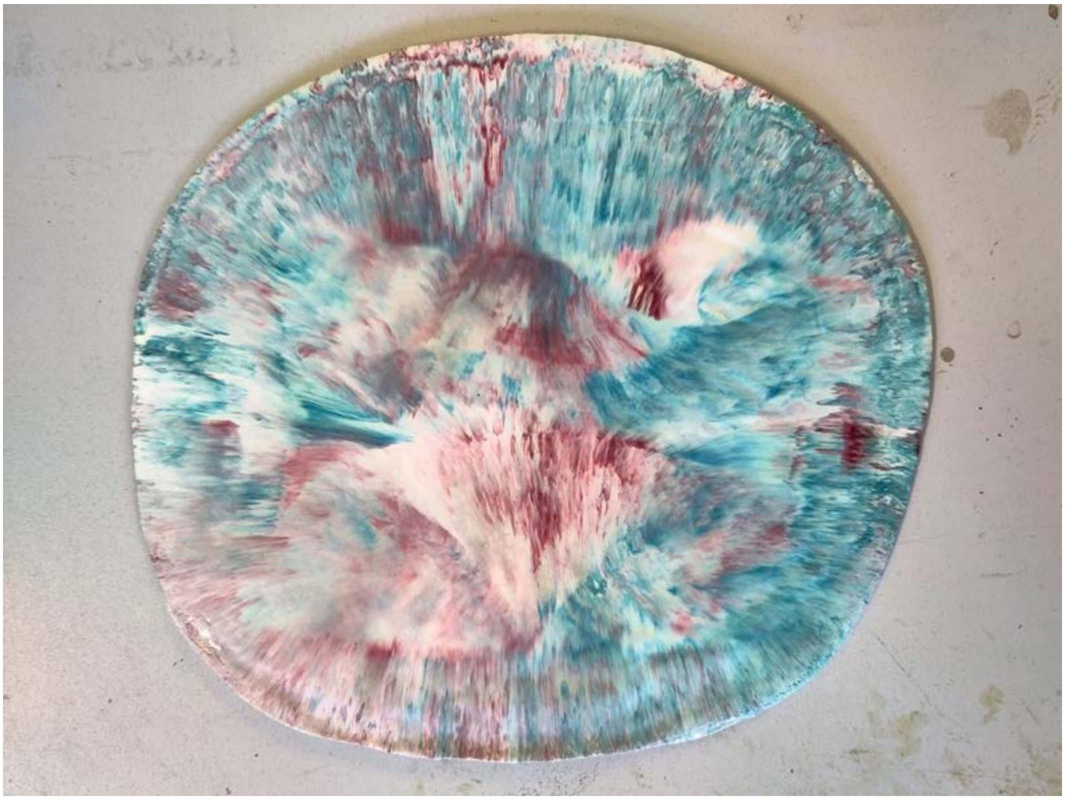
Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

Above & Right In some early tests, we ended up with very blocky colours, but we fixed this by remelting the sheet and using the fold-and-twist method

Let's be honest, if you have a 3D printer, you probably generate quite a bit of plastic waste. There are ways to minimise this – and you absolutely should – but you can't eliminate it entirely. There will always be some failed prints, and sometimes the first version of something you've designed isn't perfect (and neither is the second, third, or fourth). Can we reuse this waste plastic rather than throw it away?

This project started because of an off-hand comment in our local hackspace: "Wouldn't it be nice if you could feed failed 3D prints back into the printer." This is something that comes up fairly regularly in the 3D printing community, and there are a few projects that attempt to do this, either by reforming scrap PLA into filament, or by making an extruder that can be fed with shards of plastic rather than a spool of filament.

There are a couple of key problems you have to overcome. The first is that you need quite a bit of equipment. At the very least, a shredder to break up the plastic, and an extruder to melt this plastic and squeeze it through a nozzle (either on the printer itself or to make new filament). Both of these are reasonably costly machines. Once you've got the stuff, it's a fairly time-consuming process. And finally, if you get any particles of metal or anything that's not PLA in the mix that are larger than 0.4 mm, then you'll jam your nozzle when you try to print it.



Overall, it's not impossible, but it's not entirely trivial, and the amount of effort is quite large to create a spool of filament that you could buy for under £20.

The project almost died there until we realised that the 3D printer isn't the only machine that we used plastic in. We also used acrylic sheets in the laser cutter. What if, instead of making PLA filament for the 3D printer, we made PLA sheets for the laser cutter?

We've got a heat press (for pressing vinyl and sublimation prints onto fabric), which should be able to squeeze soft plastic into a sheet if we can work out the appropriate settings.

First, though, we needed some scrap PLA. We left a box in our local hackspace asking for donations, and they quickly turned up. Recently we've also spent a long time testing the Prusa MK4, and that meant a lot of printing. While we try to test printers with useful prints, there are only so many things we need, so a large number inevitably end up as scrap.

There are a lot of variables to tune, but the first ones were part size and temperature.

Part size affects a few things – mainly aesthetics (more on this later) and how fast it melts. We don't have a shredder, so getting small parts is mainly a case of using pliers, a vice, a bandsaw, and a hammer in various combinations to break up bits into smaller pieces. After much sweating and swearing, we've finally come to the conclusion that you don't actually need to break up the plastic in most cases. The best →

> We've finally come to the conclusion that you don't **actually need to break up the plastic in most cases**

SAFETY

Plastics give off fumes when they melt. Some give off more dangerous fumes than others, and some give off more than others.

Obviously, this includes your 3D printer. However, we're melting a lot more plastic a lot quicker than you do on a 3D printer, so it's going to give off more.

Most of the fumes given off by PLA aren't known to be particularly toxic, but there's a lot unknown about exposure to these sorts of chemicals. The risk is increased significantly if you do this regularly.

There are a few things you can do to increase your safety. Most obviously, you can increase your ventilation – open a window and ideally point a fan at it to increase airflow. You can also work with the coolest possible temperatures. PLA emits a lot more fumes once temperatures get to around 200°C.

Finally, you can wear a respirator. It needs to be both correctly fitting and rated for filtering volatile organic compounds (VOCs) to be effective.

The exact combination of measures you need will depend entirely on your setup, so it's up to you to understand the risks if you choose to follow this method.

solution is actually to blast it with a hot-air gun until it's soft enough for the initial squish in the heat press.

We found that we got reasonable results between 180°C and 200°C, with the optimal being around 190°C.

To protect the press, we used baking sheets on both sides. Our press only heats from the top, so we had to periodically flip it to ensure it melts from both sides. Once it's been squished to the desired

> "Blast it with a hot-air gun until it's soft enough **for the initial squish in the heat press**"

thickness, take it out and place it under something heavy to keep it flat while it's cooling (we found that a few hefty books worked well).

The result is a fairly flat sheet that laser-cuts reasonably well.

As a general method, this works, but it has a couple of problems:

Above & Right
We love the combination of black and orange, but it wasn't really working until we tried the swirl pattern

1) depending on the parts you started with, you can get a lot of bubbles trapped in the sheet

2) the colours can be very blobby, and it's not the most aesthetically pleasing effect.

The solution to both of these problems is to mix the plastic a bit more. For this, you'll need some heat-proof gloves. We've tried leather work gloves and silicone oven gloves and both work fine. The only ones we've had trouble with are work gloves with some sort of flexible rubber grips on them which melted into the plastic.

To get rid of the bubbles, simply folding and twisting the plastic a few times seems to work well. This might feel like it'd introduce air, but it really doesn't.

If you're just here to create functional parts, then you can finish here. You should have some reasonably consistent, smooth sheets that you can laser-cut. They're also easy to work with using hand tools as well.

Beauty is, as they say, in the eye of the beholder, and what we find interesting might not be what you find interesting. That said, there are some basic techniques that you can use to create an interesting look.

The first and most obvious thing is to think about your colours. If the colours don't work together, then the pattern doesn't really matter, it's not going to look good. You can either hold onto the PLA until you have some more colours, or you can use them to create functional parts or test pieces that don't have to look good.

We found two ways of creating interesting-looking prints.

The first is to have a roughly even mix of two colours to start with. You can then melt these down to a sheet with the colours separated – one colour on one side, the other colour on the other side. You can still mix this up a bit to get the air bubbles out by folding into the middle or perpendicular to the split in the colours. Once everything's fully soft, it's time →

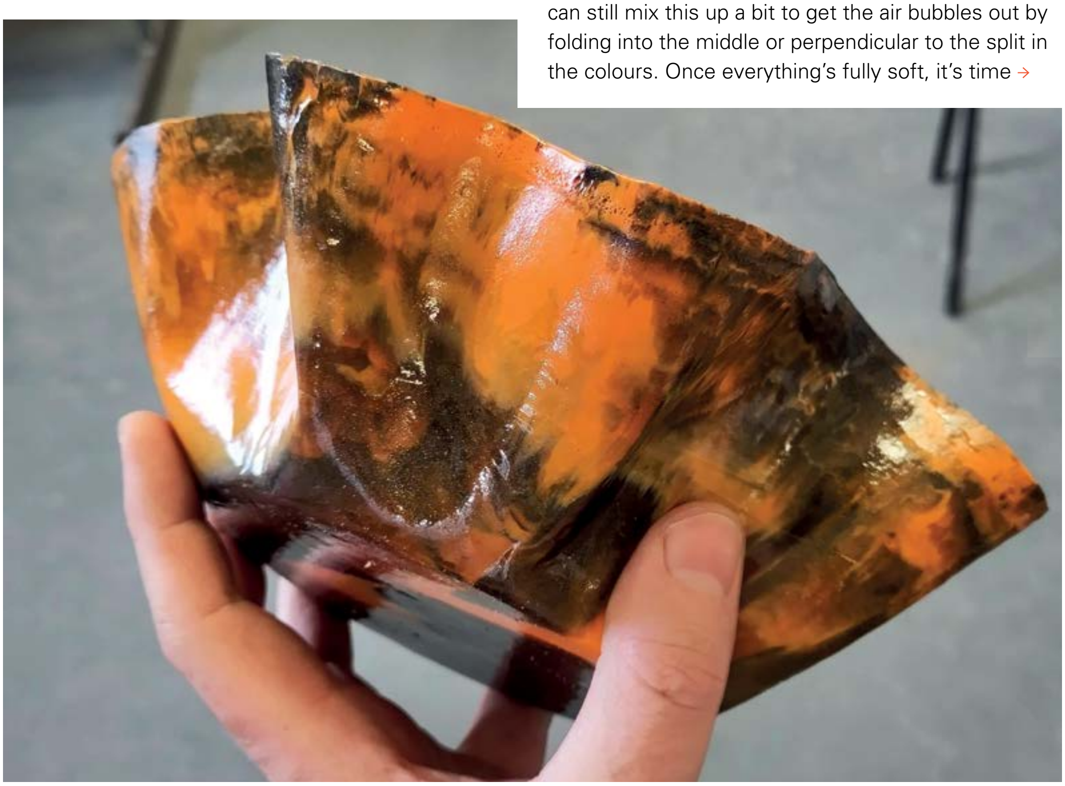
OTHER **PLASTICS?**

We've focussed our efforts on PLA because it's the most popular plastic for 3D printing, but any plastic that can be printed with an FDM printer is a thermoplastic that can be heated and reformed. Should you give this a go with other plastics?

Firstly, you need to think about safety. Different plastics give off different levels of fumes, and these have different levels of toxicity. We would recommend staying away from ABS and nylon unless you are confident in your setup.

PETG might be worth experimenting with if you have enough to try. You will, however, need to work at hotter temperatures.

Obviously 3D prints are not the only source of plastic that can be recycled. We've seen other makers have success with similar techniques using HDPE (high-density polyethylene) – you can recognise this as it has a number 2 inside the recycling sign. However, we've not experimented with it.



Left
A quick blast with a hot-air gun can really make your bowl shine

Above
Before the laser cutter broke, we tested out some jewellery designs

to mix. Roll it perpendicular to the colours, then fold it over and twist to swirl the colours together. How much twisting takes a bit of practice to get right. Too little and it's still very blocky. Too much and you just mix the colours together and end up with something fairly brown. Get it right and you can generate some really interesting swirly effects. The amount of mixing you want is often dictated by the size of the thing you're making. If your final objects are small, you'll want a tighter, smaller swirl. If they're bigger, you might want a loose swirl. There's no real substitute for experimentation here.

> "Get it right and you can generate some **really interesting swirly effects**"

COMPOSTING

In theory, PLA is compostable. However, the brutal truth is that it's almost impossible to compost it. If you try, you may well be making the situation worse. We would strongly recommend that you don't attempt to compost PLA unless you are very familiar with the specifics of both PLA and the system you'll be using.

PLA will break down in very specific conditions (basically, it needs to be held at over 60°C for an extended period of time). Doing this on a home compost heap isn't impossible, but it is hard. And you need to ensure that all of the PLA is held at that temperature. If you fail to fully compost the PLA, your compost will be full of half-decomposed microplastics that you'll be contaminating the environment with.

There are some industrial composters that do meet the requirements of composting PLA, but it's by no means all of them. You can't just assume that you can add PLA to a food waste collection and it'll be dealt with. In fact, you might make the situation much worse – if the composters aren't expecting PLA in the waste, they may reject an entire batch of food waste and send it to landfill to stop their compost from being contaminated with plastic. If you do want to compost your PLA, it's imperative that you speak with the waste company to ensure that they will process it, otherwise, you could find that you not only don't get your PLA composted, but that you stop a lot of other compostable waste from being composted.

What should you do with PLA? Well, it's complicated. Take a look at the safe disposal box for more information.

The second technique requires smaller pieces with a very mixed range of colours. Mix them as evenly as you can before you melt them and only mix a little to break up the bubbles.

Those are the two effects we're particularly pleased with, but that's just our personal style.

At this point, we had a technique for generating some interesting-looking PLA sheet, and had done some initial tests laser-cutting them. However, the laser cutter chiller at Bristol Hackspace had a 'bang and spark' error that took out the power to the building and hasn't worked since. A new one's on order, but we want to do some more testing before talking more about laser-cutting this material.

Fortunately, though, there's more than one way to work with sheet PLA, and we've been experimenting with slump-moulded bowls. The idea is really simple. Take an upturned bowl, or similarly shaped object. Heat up a sheet of PLA until it's soft. Plonk the PLA on the upturned bowl and it'll sag down to make a bowl shape. This is very similar to a technique that's often used with old vinyl records.

The amount you heat the PLA sheet affects the final result. If it's very soft, it'll instantly drape over the bowl, and you may find some of the pleats fold together. If it's only a little soft, you can get a gentler, more open bowl. There's no right or wrong here, it's just a matter of what you prefer.

This technique works best if you start off with a circular sheet of PLA. You can either cut this out of a different-shaped sheet, or you can make a circular sheet by depositing a roughly circular blob of PLA in your press after twisting and mixing it.

You can use a heat gun to make minor changes to the shape of your bowl as it cools.

The final thing to give your bowl a glossy appearance is 'flame polishing'. This is where you melt the outer layer of PLA and let it smooth together to create a shiny finish. While this is called flame polishing, it's best to do this with a heat gun. Experiment a little to find the temperatures and flow rates that work best for you, as this will vary a bit between heat guns, and you can also get different effects, from a satin finish to a high gloss.

From a fairly innocuous discussion, we've been down a rabbit hole of plastic recycling and ended up with a technique for generating PLA sheet that's easy to work with and can look great.

Below
A pile of PLA scraps waiting to be pressed into a sheet

Below
You can see in the texture of the bottom-left bowl that we didn't get the bubbles out properly

SAFE DISPOSAL

If you can't compost or reuse your PLA, what should you do? The answer really depends on what your local waste system looks like, and it's impossible for us to give hard-and-fast rules. For example, this author has a different waste stream at home and in the office, and the waste is processed slightly differently.

At home, he has domestic plastic waste. PLA isn't often recycled commercially because there isn't a lot of it in the waste stream. Even if it was, it's pretty hard to get 3D prints to be recognised correctly. You'll need to label them, and ideally, test this label with a plastic sorter.

However, the particular plastic recycling plant that is used by this author's waste collection team has two parts to it. Firstly, it recycles any recyclable plastic. Secondly, it incinerates non-recyclable plastic and uses this to generate electricity that both powers the recycling plant and feeds back into the grid.

In this situation, the PLA (which is plant-derived, not oil-derived like most plastics) essentially becomes a biofuel that happened to be a 3D print along the way. That doesn't necessarily make it great for the environment, but at the same time, it doesn't seem too bad.

In the workshop, the general waste goes directly to a different waste-to-power plant, so PLA should go directly into this, and again, it will be used as a biofuel.

This, however, is the author's specific setup. Your waste streams will be different, and you'll need to familiarise yourself with where different bits of waste end up to know where the best place to put your PLA is.

Laser-cutting PLA

Turn old 3D prints and support material into boxes, jewellery, and other items

Ben Everard

@ben_everard

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

We've already looked at how to recycle PLA from old 3D prints and support materials into sheet material. The resulting sheet is about 1 to 3 mm thick. In this article, we'll start with those same recycled sheets and look at how to laser-cut it.

PLA has some slightly unusual properties that make it great for 3D printing; the main one being that it softens at a very low temperature. When laser-cutting, this means that the bit that the laser hits obviously vaporises (as happens with other laser-cuttable plastics).

However, the heat from this spreads out and softens much more plastic than you would usually expect. It's quite common to find that the part is still soft when you've finished cutting. The make-up of the laser bed is really important. If there's not enough support, you may find that your parts sag under their own weight once they soften from the heat. You might also find that some of the softened and liquidised PLA sticks to the print bed and can make it tricky to get small parts off. A bit of a bash usually gets things unstuck, but the more delicate the part, the more of a problem this is likely to be. Sharp corners tend to get a bit rounded off as it just melts away – the same goes for thin branches. If you're particularly good with your laser control software, it might be possible to plan the job so that cuts on either side of a hole or branch are done at different times, to give the parts more time to cool down, but this isn't something that we've tested.

We've also had problems where small parts were cut out and then blown slightly by the air-assist, and rewelded themselves to the main piece. Most of the time, we've had success prying them out afterwards but, again, the smaller and more delicate the part, the harder it is. On the whole, we've had very few problems with parts (or holes) over about 2 mm × 2 mm, and things get a bit tricky once you go smaller than this. That said, with a bit of post-processing, we've had some much smaller parts come out fine.

> It might be possible to plan the job so that cuts on either **side of a hole or branch are done at different times**

The final part that makes things a bit tricky is that the sheets we've made aren't perfectly uniform, and can vary in width by about 50%. With care, you can minimise this but, unless you upgrade to a purpose-built sheet press, you're probably not going to be able to eliminate this. Obviously, this means that you need your power and speed settings calibrated for the thickest part of the material. This means you're overpowering some of the cuts.

Below It can be challenging to get to pieces with balanced colours

FLAME-POLISHING

Flame-polishing is the process of heating up the part quickly so the outer surface melts, then cooling it down. This process is very effective on PLA because of its semi-crystalline structure. You can end up with a very glossy surface that really brings out the colours.

We've had good results using an electronics head gun with a fine nozzle. Set the temperature to about 260°C and low to moderate airflow. The aim is to just melt the outer surface, so you don't want to be in any one place for long. Keep moving and keep checking on the shininess. More melt will give you more shininess, but also means more bubbles and risk of deformation (see below). However, it does have some downsides:

- The chances are that you've got some bubbles trapped under the surface of your part. If you heat it up, these bubbles expand and can pop in the liquid PLA. If you manage the temperature well, this often isn't a big problem. On parts with lots of detail, the bubbles often don't really show up. On parts that have larger smooth surfaces, the bubbles can be really noticeable.
- Thin branches can go soft and get blown around. You can poke them back a bit if they don't get stuck. This is really noticeable on geometric designs, and less obvious on more organic designs.

This means that you need to think about the appropriate finishing method at the design stage. If you're hoping to flame-polish your work, make sure these two issues won't be problems for you.

Above Just add a jump-ring and an off-the-shelf earring hook to turn your pieces into jewellery

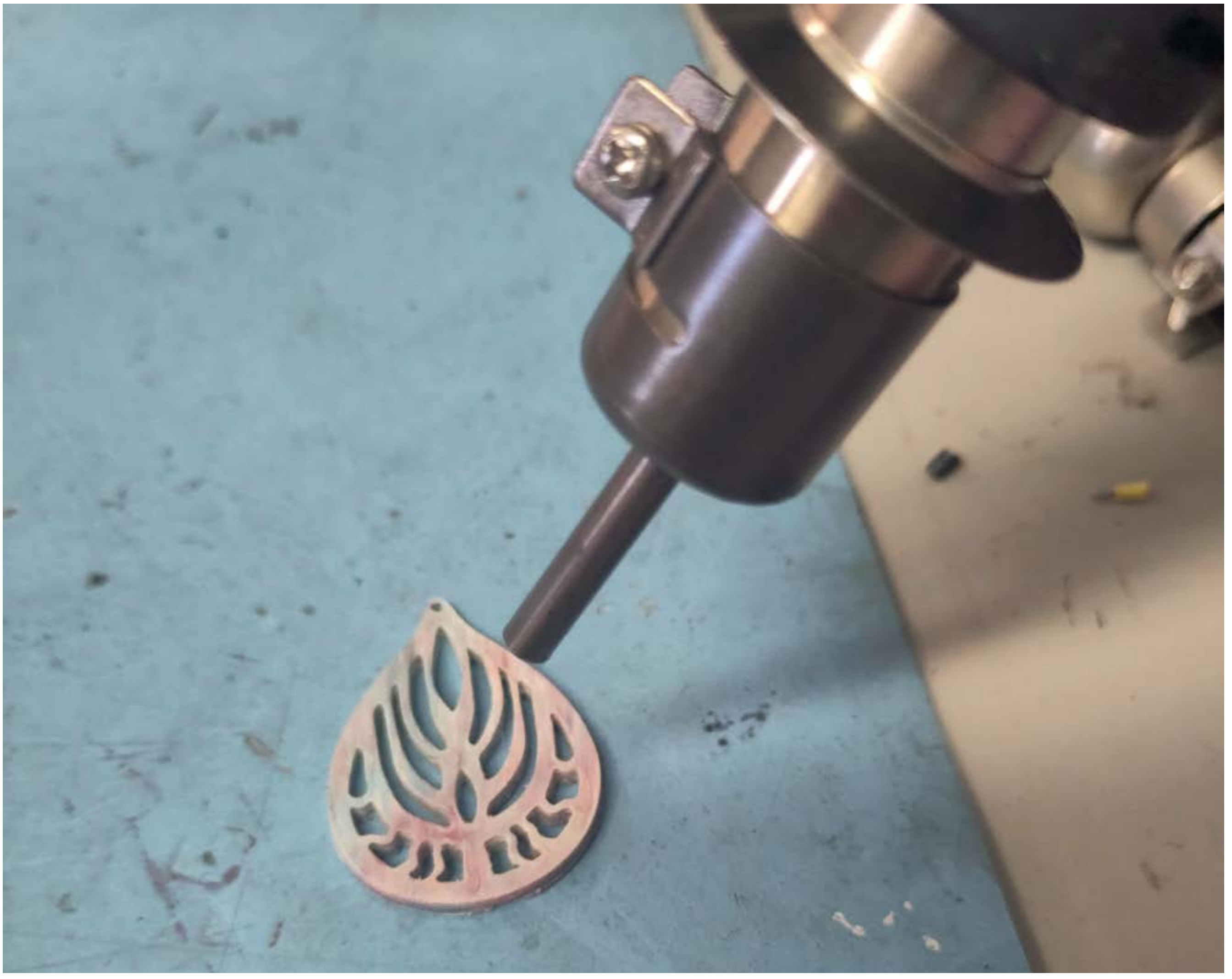
There are some potential benefits to how PLA reacts to the heat as well. When cutting finger joints, we found that if we took the parts straight off the print bed and fitted them together, there was enough softness from the heat to both help tight-fitting parts get together, and then weld themselves slightly together, ensuring a really solid part.

JEWELLERY MAKING

We really like the look of the swirls of colour you get in the recycled PLA sheets, so we tested the process out by making jewellery.

The first problem is that with this method of pressing sheets, you end up with an irregularly shaped sheet. We'd recommend the method mentioned last month – of scooping up the molten plastic, twisting it into a blob, and then re-pressing it. Not only does this mean the colours swirl together much more pleasingly, but it also means that you end up with a circular sheet. Most CAD programs are designed to work with rectangular sheets, but at least a circle is sufficiently regular to be easy to work with. Use a tape measure to find the smallest distance across, then draw a circle of this diameter in Inkscape (or your CAD program of choice), and whatever fits in this circle, should fit on the sheet. You can delete the circle before cutting.

As we've mentioned, there are a few aspects that make small parts tricky to cut. We hedged our bets on this by cutting more than we needed. When making earrings, for example, we cut three of each shape and picked our favourite two. This → also helps because the pattern is quite complex, so it can be hard to place designs to get two earrings



Above A quick blast with hot air can give a shiny finish

DESIGNS

Getting the right designs is key to getting good results. We've found two approaches that work well. The first is simple designs with areas to show off the pattern in the plastic. These can look great with a matt surface and minimal post-processing.

The second is organic designs with narrow, but not too narrow (around 1.5 to 3mm wide), branching features. These can look great flame-polished.

Obviously, it depends on your personal taste, but these are the general areas that we've had the most success with.

Unfortunately, there doesn't seem to be a good repository of open-source designs for laser cutting in the same way that there is with 3D printing. There are many sellers on Etsy selling laser-cuttable templates for jewellery of varying quality (and we don't know of any way of actually verifying if a seller actually has a licence to sell the designs that they're selling). Alternatively, you can design your own.

with equally balanced colours. By cutting three, you have a better chance of finding two that work well together. This might sound wasteful, but don't forget that any you don't use can just be put back in the recycling mix for the next sheet.

Once you've got your parts off the cutter, the next challenge is the finishing. The laser leaves quite a lot of residue. Some of this is smoke and some of this is molten plastic that's been blown away from the cut line. This is exacerbated by the fact that PLA can have two very different finishes, depending on how it cools. Sometimes you'll find it ends up matt, sometimes really shiny. We won't go into the technical details, but you often end up with a shiny outline next to the cut and a matt finish elsewhere. This can leave it looking a bit untidy.

How big a problem this is depends on exactly how the part has come off and what it'll be used for.

Let's examine a couple of examples of how to get parts looking good. The first star pendant design is

Above
Different patterns can give very different effects

quite large and has space to show off a lot of the pattern from the plastic. Some negative space adds an opportunity to give it a bit more detail.

The relatively simple design of this means that there aren't many lasered edges, so not much needs removing. A quick sanding down with 180 grit sandpaper removes almost all the imperfections. There are still some slight blemishes on the final piece if you look closely enough, but it's pretty hard to make these out.

The second piece is more organic-looking. A lot of details have been cut with the laser, which means

> **By cutting three, you have a better chance of** finding two that work well together

that it needs more work. We start with a heavy sand with 60 grit paper. This removes most of the waste material, but leaves it rough and with a lot of burrs in the details that are hard to remove. We go through the grits to 180 grit paper. This removes the worst of it, but leaves it looking very muted. A flame-polish (see box) brings out the colours with minimal bubbling, but some parts warp a little.

Above
Larger, flat surfaces show off more of the pattern, but also more defects

SAFETY

As far as we can tell, PLA is a relatively safe plastic to laser-cut. The big caveat here is fumes. While PLA isn't particularly bad for fumes, generally you don't want to breathe in any fumes from laser cutting, so an enclosed laser cutter with an extraction system is very strongly recommended. The second caveat is that, while we believe there aren't any particular risks associated with laser-cutting PLA, it's not something that we've been able to get a lot of information on.

For us, working with PLA and an enclosed laser cutter with fume extraction fits within our risk tolerance, but it is experimental and, before working on this, you need to decide for yourself if it's acceptable for you.

Laser-cutting recycled PLA is a great technique that can lead to really interesting-looking designs that are also kind to the environment. However, it's not a universal technique that can be used to make anything look great. You need to think about your design and take the time to post-process if you really want to end up with something that looks great.

Making music on Pico

Use CircuitPython to create complex sounds

Ben Everard

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

Most microcontroller programming environments have a way of making beeps, and even if there's not an in-built method, it's a pretty simple thing to do – just flick a pin on and off quickly (but not too quickly) and you have an audio signal that you can send to a speaker, headphones, or other audio devices.

CircuitPython, however, has just gained a complex audio synthesis module called synthio. This lets you not just create beeps and boops, but control these sounds in a variety of ways. Let's take a look.

You'll need a board that supports synthio. RP2040 boards, such as Pico and Pico W, are a great option for testing things out and having a play using headphones; however, a board with dedicated audio hardware (such as the Adafruit RP2040 Prop-Maker) will make it easier to connect a speaker.

Above You can use a jack socket if you want, but crocodile clips are great for prototyping

Before we dive into the code, let's take a look at the hardware. If you're using a board without dedicated audio hardware (such as a Pico), you just need to connect a pair of headphones between one of the GPIO pins and ground. The easiest way to do this is to clip crocodile clips onto the headphone jack.

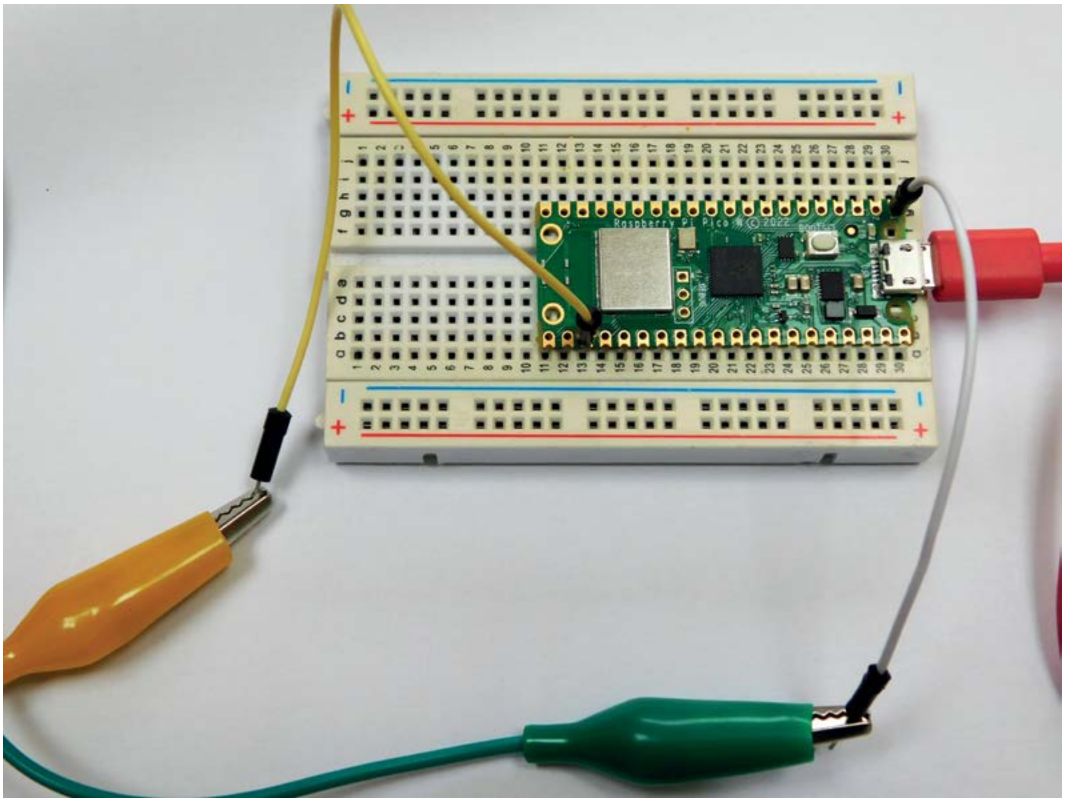
You can probably get away without adding a resistor in there as well, but it's a bit safer to put a 220 Ω resistor between the GPIO pin and the headphones. This stops the current from getting too high (especially if you are using crocodile clips that can accidentally touch).

Ideally, you should have a capacitor between the GPIO pin and ground. This is because the signal coming from the GPIO pin isn't an audio signal, but a high-frequency PWM signal that, when averaged out, becomes the audio signal. The capacitor creates a low-pass filter that does this averaging out. We find that this isn't generally necessary because the rest of the circuit tends to act as a low-pass filter, so it works fine without it.

We used GPIO0, but you can adjust this to any you want.

Let's take a look at a really simple example:

```
import time
import board
import digitalio
import synthio
import audiopwmio
import audiomixer

audio = audiopwmio.PWMAudioOut(board.GP0)
mixer = audiomixer.Mixer(channel_count=1, sample_
rate=22050, buffer_size=2048)
synth = synthio.Synthesizer(sample_rate=22050)

audio.play(mixer)
```



Left You can get away without a resistor, but it's a bit safer to add one in line with the GPIO pin

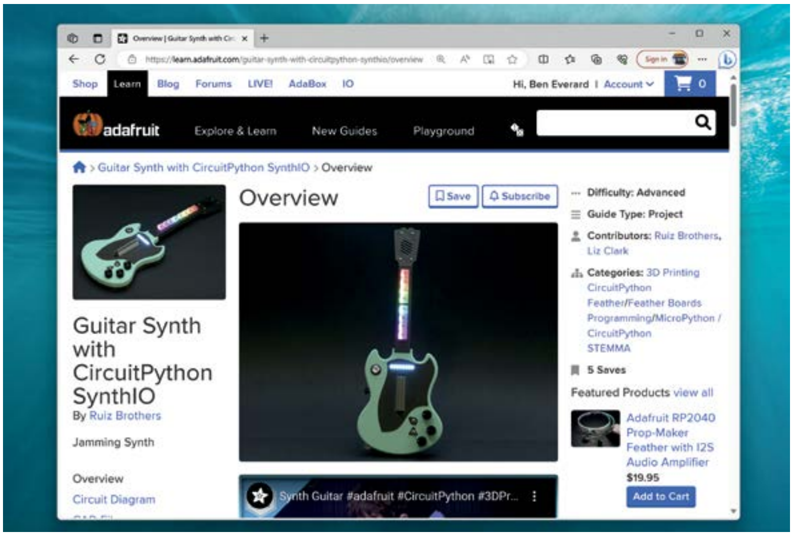
```
mixer.voice[0].play(synth)
mixer.voice[0].level = 0.1

while True:
    synth.press(65)
    time.sleep(0.5)
    synth.release(65)
    time.sleep(1)
```

Here, we use three objects. There's a **`PWMAudioOut`** which handles the output. There are other output options if you have additional hardware, for example, if you want to use an I2S DAC. There's a **`mixer`**. Technically, we could get away without this, in which case, the audio would play at full volume, but this can be a bit loud on headphones, so we prefer to always use a mixer to enable us to keep control over how loud we go. In future examples, you'll also see how a mixer can be used to combine sounds. Finally, there's a **`synth`** which is used to generate the sound.

In this case, we just loop through, pressing and releasing note 65. The notes are given numbers in the MIDI system, and 65 is F#.

When we press it, it goes on full; when we release it, it goes off instantly. This is OK, but it sounds a bit harsh. Wouldn't it be better if we could fade the sound in and out? Synthio has a solution to this – it's called envelopes:

```
import time
import board
import digitalio
import synthio
import audiopwmio
import audiomixer

audio = audiopwmio.PWMAudioOut(board.GP0)
mixer = audiomixer.Mixer(channel_count=1, sample_
rate=22050, buffer_size=2048)
synth = synthio.Synthesizer(sample_rate=22050)
audio.play(mixer)
mixer.voice[0].play(synth)
mixer.voice[0].level = 0.1

amp_env = synthio.Envelope(
                                attack_time=0.05,
                                sustain_
level=0.2,
                                release_time=0.5
)

while True:
    synth.envelope = amp_env
    synth.press(46)
    time.sleep(1.25)
    synth.release(46)
    time.sleep(1.25)
```



Above There's an example guitar created by the Ruiz Brothers at hsmag.cc/synthioguitar

Envelopes are a way of controlling the volume over time using the parameters: **`attack_time`**, **`attack_level`**, **`sustain_time`**, **`sustain_level`**, **`release_time`**, **`decay_time`**.

At first, the volume will ramp up to **`attack_level`** over the period of **`attack_time`**. Then it will drop down to **`sustain_level`** volume over **`decay_time`**. It will then stay at the same volume for **`sustain_time`**, and finally drop down to zero volume over **`decay_time`**. All times are in seconds, and volumes are between 0 and 1.

Now we can play notes that sound a bit more natural, let's take a look at combining them. The following code uses two **`synth`** objects to play notes at different times:

```
import time
import board
import digitalio
import synthio
import audiopwmio
import audiomixer

audio = audiopwmio.PWMAudioOut(board.GP0)
mixer = audiomixer.Mixer(channel_count=1, sample_
rate=22050, buffer_size=2048)
synth1 = synthio.Synthesizer(sample_rate=22050)
synth2 = synthio.Synthesizer(sample_rate=22050)

audio.play(mixer)
mixer.voice[0].play(synth1)
mixer.voice[0].level = 0.1
mixer.voice[1].play(synth2)
mixer.voice[1].level = 0.05

amp_env = synthio.Envelope(
                  attack_time=0.05,
                  sustain_level=0.3,
                  release_time=0.3
)

amp_env2 = synthio.Envelope(
                  attack_time=0.05,
                  sustain_level=0.5,
                  release_time=0.5
)

beat = 0
synth1.envelope = amp_env
while True:
    if beat % 2 == 0:
        synth1.press(60)
    if beat % 2 == 1:
        synth1.release(60)
    if beat % 4 == 0:
        synth2.press(46)
    if beat % 4 == 2:
        synth2.release(46)
    time.sleep(1)
    beat += 1
```

We can't use a simple repeating loop here because that wouldn't let us play notes at different rates, so we've created a beat loop, then we can detect which beats we want to start and stop different notes at.

ON ANOTHER NOTE

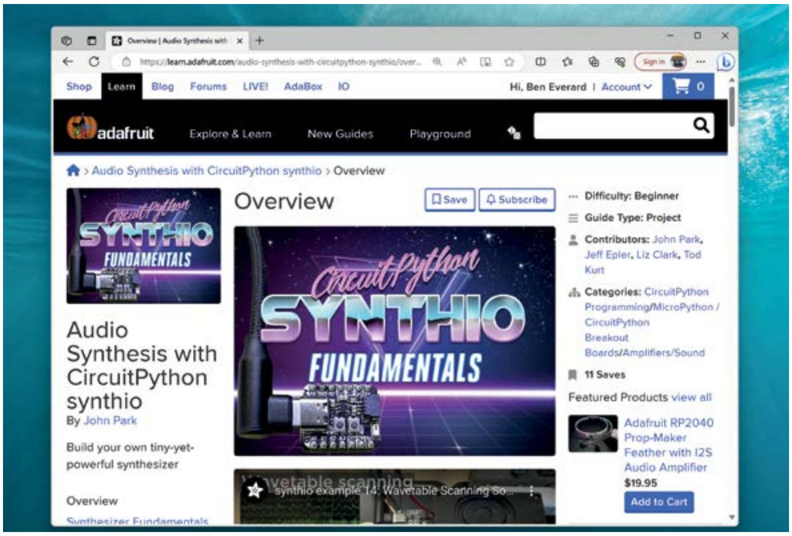
So far, we've set the volume structure of the note, but we've still been playing the same sound – a square wave beep. We can create far more complex

> **Now we can play notes that sound a bit more** natural, let's take a look at combining them

sounds than this. Synthio uses a wavetable function to create the audio signal. This basically means that you can create an array of volumes that define the shape of the audio signal for a note. This could go abruptly from 0 to 1 as in a square way; it could go up linearly like a triangular wave; it could have a curve like a sine wave, or just about anything else. Each wave shape produces a very different sound.

```
import time
```

```
import board
import digitalio
import synthio
import audiopwmio
import audiomixer
import ulab.numpy as np

audio = audiopwmio.PWMAudioOut(board.GP0)
mixer = audiomixer.Mixer(channel_count=1, sample_
rate=22050, buffer_size=2048)
synth1 = synthio.Synthesizer(sample_rate=22050)
synth2 = synthio.Synthesizer(sample_rate=22050)

audio.play(mixer)
mixer.voice[0].play(synth1)
mixer.voice[0].level = 0.2
mixer.voice[1].play(synth2)
mixer.voice[1].level = 0.05

amp_env = synthio.Envelope(
                            attack_time=0.05,
                            attack_level = 1,
                            decay_time = 0.1,
                            sustain_level=0.3,
                            release_time=0.3
)

amp_env2 = synthio.Envelope(
                             attack_time=0.05,
                             attack_level = 1,
                             decay_time = 0.1,
                             sustain_level=0.1,
                             release_time=0.1
)

SAMPLE_SIZE = 512
SAMPLE_VOLUME = 32000  # 0-32767
half_period = SAMPLE_SIZE // 2
wave_sine = np.array(np.sin(np.linspace(0, 2*np.
pi, SAMPLE_SIZE, endpoint=False)) * SAMPLE_VOLUME,
                     dtype=np.int16)

note1 = synthio.Note(synthio.midi_to_hz(60),
waveform=wave_sine, amplitude=1)

beat = 0
synth1.envelope = amp_env
while True:
    if beat % 2 == 0:
        synth1.press(note1)
    if beat % 2 == 1:
        synth1.release(note1)
    if beat % 4 == 0:
        synth2.press(46)
    if beat % 4 == 2:
        synth2.release(46)
    time.sleep(1)
    beat += 1
```



Above How far do you want to disappear down the synth rabbit hole?

We've only really scratched the surface of what you can do with synthio here. It's a hugely powerful system for creating music using microcontrollers. There's a more in-depth introduction online at **hsmag.cc/synthiofundamentals**. ◻

SYNTH TERMINOLOGY

Synthio uses similar terminology to many other synthesizers. You'll come across envelopes and mixers similar to those we've used in this article in many other bits of software and hardware. Here are a few other things that synthio provides that you will also find elsewhere:

- Low-frequency oscillator (usually known as an LFO) is a special type of oscillator that unsurprisingly runs slowly, or at least slower than typical audio frequencies. The purpose of this isn't to generate audio sounds directly but to influence other audio. For example, you can use it to bend a note in a vibrato-like effect.
- Filters block some audio signals while allowing others to pass through. The type of filter defines what frequencies are allowed to pass through. Synthio Synthesizer objects have the following methods to create filters: high_pass_filter, low_pass_filter, and band_pass_filter. Once they are created, they can be attached to notes or synthesizer objects.

Rad your ride with Pico

Turn a manual scooter into a motorised monster

Dr Andrew Lewis

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

Scooters are fun, and can be a good alternative to cars and bikes when considering the daily commute. They can even be useful while at work if your job involves moving between campus buildings several times a day, or travelling back and forth in large warehouses or hangars. Electric scooters are even more fun, but they are also expensive. In this project, you'll see how to take an ordinary manual scooter and make it electric without resorting to expensive hub motors or conversion kits. This is a two-part project. In the first part, we'll take a look at the components we'll use and how we'll need to modify our scooter to make it all work. In the second part of the project, you'll see how to power your motor from custom batteries.

The anatomy and function of an electric scooter isn't mysterious or difficult to understand. The user stands on a platform, activates a throttle, a motor hums into life, and the wheels of the scooter begin to turn. When the user wants to stop, they release the throttle and apply a brake or use their foot to slow down.

A manual scooter works in a similar way, but with more leg action and less throttle – surely we just need to add a motor, some batteries, and a speed controller, and that's everything, right? Well, it isn't quite that simple. Firstly, manual scooters come with very primitive brakes (if they have brakes at all). The wheels on the scooter are typically not designed to be driven, and just have a simple bearing and possibly a fork with a suspension spring. Making a manual scooter move under motor power means making some very permanent changes to the chassis of the scooter, mounting the motor near to the rear wheel, enlarging the rear fork to accommodate better brakes and a drive sprocket on the rear wheel, and modifying the wheel itself so that it can handle the extra stress coming from the motor.

Most of the issues with the wheel can be dealt with by using some 3D-printed parts and a healthy dose of epoxy resin to reinforce the wheel. The drive sprocket for the motor and the disc to upgrade the brakes can be mounted on opposite sides of the rear wheel using 3D-printed parts, and using

> **"Once you have a solid wheel, you can build your** rear fork to match its width"

plenty of resin to fill the space between the two sides will make sure that the bolts have plenty of solid wheel material to make contact with when the forces from the motor and brakes are transferred through them.

Once you have a solid wheel, you can build your rear fork to match its width. If you have access to a welder, you can do this very quickly by cutting the original rear fork into pieces and re-welding it with →

OFF-ROAD

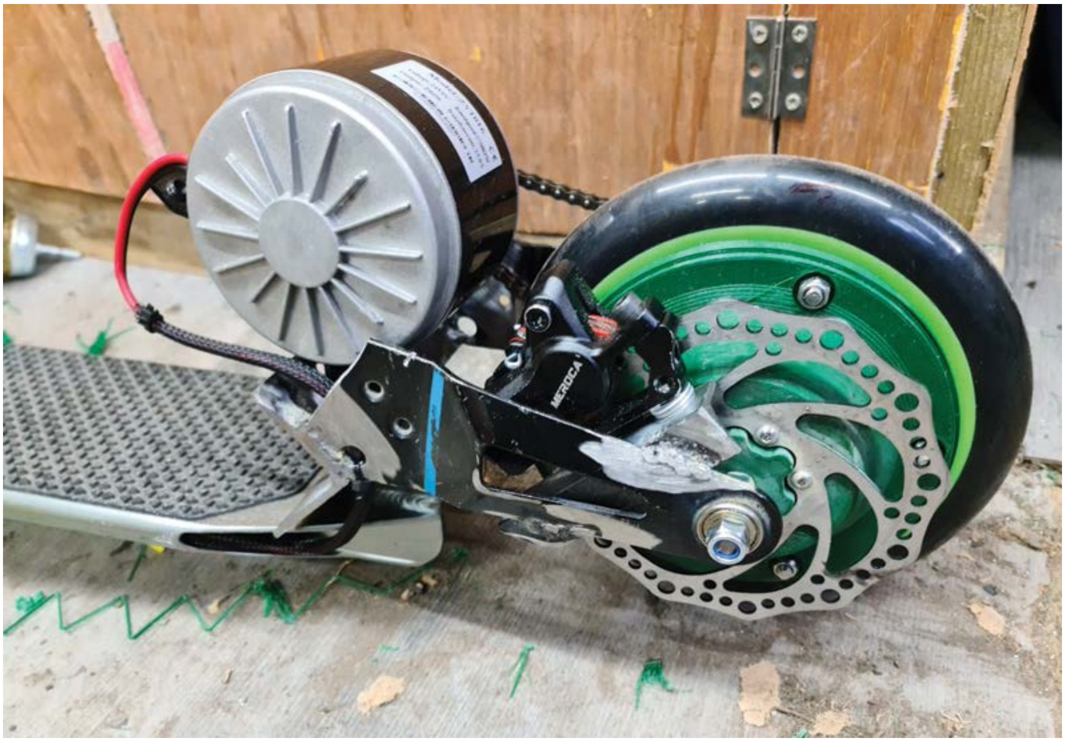
If you're thinking of using an electric scooter in the UK, you're going to be disappointed to learn that you're not going to be able to ride it on the open road, or on a footpath, or even in a bike lane. Electric scooters are in a class of vehicle known as 'powered transporters'. It's the same place you'll find one-wheels, hovercrafts, and electric skateboards. In UK law, a powered transporter is the same as a motor vehicle, and as such you would be expected to license and maintain it in exactly the same way as a car or motorbike. Unfortunately, scooters don't have some of the important features needed to pass an MOT (Ministry of Transport) test (like headlights, indicators, registration plates, etc.) – and so they will not be legal to ride on the road, and you can't get insurance to drive them. However, they are still technically classed as motor vehicles, so you aren't allowed to ride them on the pavement either. If you *do* ride on the pavement or on the road, you could be fined several hundred pounds and risk losing your driving licence. You can ride them on private land – such as corporate campuses – with the landowner's permission. At the time this article was written, the government is developing new standards and adjustments that will eventually make electronic scooters road-legal in the UK. How long that process will take depends on how quickly the government decides to move.

TAKE CARE

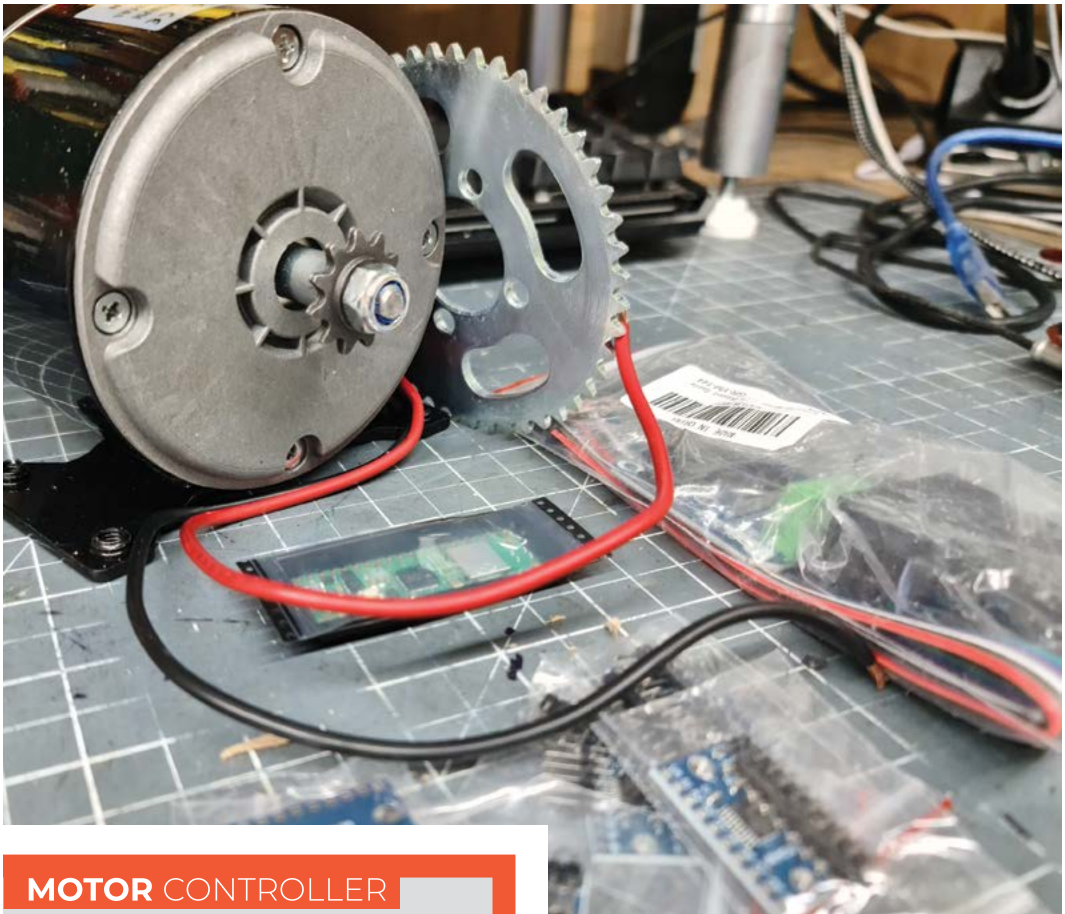
Scooters – whether powered or not – carry some risks. It's your responsibility to make sure the scooter doesn't go faster than you personally feel comfortable with. You are also responsible for ensuring that any chassis modifications you make are sufficiently strong. We strongly recommend you err on the side of caution as your decisions on both these risks could impact not just you, but the people around you.

QUICK TIP

Cycle chains can be joined with a split link, so you don't have to buy a chain exactly the right size. Simply remove excess links, join the chain with the split link, then tension the chain by moving the motor.



Left Most scooters don't really bother with an effective brake, because they don't go very fast and you can stop them with your foot. If you're on a motorised scooter, you'll need a more powerful rear brake to arrest your more considerable forward momentum. A disc brake from a bicycle will do the job. Just remember that the brake will only stop the scooter, not you



Right Using a motor with a sprocket and chain is much cheaper than using a hub motor and allows you to increase the torque from the motor mechanically

MOTOR CONTROLLER

Most BTS7960 motor controllers are H-bridge controllers that have a similar pinout. The boards have two power inputs, one of which (often labelled as pins B+ and B-) supplies power for the motor. Next to this input are the connections (M+ and M-) that directly attach to the motor.

The other power input is used to supply the logic side of the board, and is labelled as Vcc and GND. This logic level power input runs at 3.3 V or 5 V and is isolated from the motor power connection for safety. With a Raspberry Pi Pico, you should be powering this with 3.3 V. The remaining connections on the board are used to control the speed and direction of the motor, and optionally monitor the current used by the motor. We won't be using the current-sensing features, so the pins labelled R_IS and L_IS can be ignored. The L_EN and R_EN pins enable the motor to turn clockwise or anticlockwise when they are high, or disable the motor when low. Obviously, only one of these should be enabled at a time. The LPWM and RPWM pins accept a PWM signal from the Pico, or a solid logic high to power the motor at 100% power. Since a scooter generally doesn't need a reverse gear, you only need to connect either the left or right enable and PWM pins to the Pico, depending on which way around your motor is connected. Assuming that you have a right-hand motor setup, you would make the motor move by setting the R_EN pin HIGH and providing a PWM signal to RPWM.

> "You will probably find that you won't want to run your motor at 100% speed under any circumstances"

a larger gap. If you don't have a welder, you can achieve similar results using threaded bar and locking nuts with a spacer between each side of the fork.

For something like a light scooter, a 20 amp, 24 volt motor in combination with the gear reduction from the sprocket drive should have plenty of power to move an adult at terrifying speeds. By controlling the power to the motor using PWM signals from a Raspberry Pi Pico, you can provide very smooth speed control without losing as much torque as you would if you simply reduced the overall supply voltage. You will probably find that you won't want to run your motor at 100% speed under any circumstances, and will throttle the maximum PWM value to something less than bowel-voidingly high speeds. □

QUICK TIP

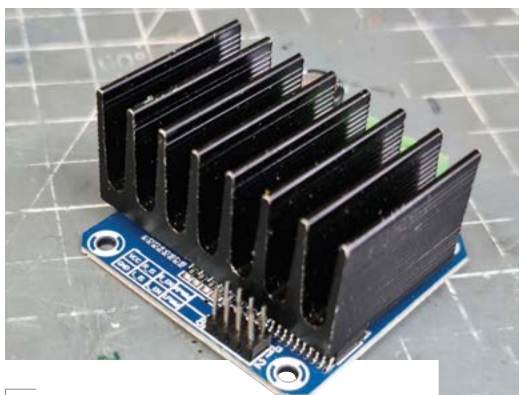
In MicroPython, the Pi Pico PWM output ranges between 0 and 65025, so setting a PWM pin with `pwm.duty_u16(32512)` would give you about 50% power.

ANALOGUE IN

Setting the motor speed is only half of the process; you also need to get a value from the user by reading a throttle control. It's tempting to think about creating a simple throttle control using a potentiometer, but that would be a bad idea. If you've ever used an old radio, you'll know that the potentiometer used in the volume control gets worn out, and the volume of the radio will suddenly change or crackle at full volume for a fraction of a second. This is annoying in a radio, but if it happens on a motor controller and the motor suddenly speeds up to 100% without warning, the result could be catastrophic. To avoid this problem, most electronic throttle controls use a Hall probe and a magnet to ensure smooth speed changes. The Hall probe and magnet aren't in physical contact and there are no tracks to get worn, dirty, or broken. The probe normally has two power pins and a signal pin, which outputs an analogue value to reflect the strength of the magnetic field it detects. As the magnet moves closer, the field gets stronger. Checking the position of the throttle controller is as simple as connecting up and polling the analogue pin of a microcontroller, just as you would with a potentiometer.

Above
You will need to mount your motor so that the motor's sprocket matches up with the sprocket on the wheel. The easiest way to do this is to fix a mount for the motor on the rear of the fork with slotted holes, and use the adjustment that the slots provide to position the motor correctly. You can also use this technique to tension the chain between the motor and the wheel

Left
You can see here how much wider the wheel becomes once you allow for the sprocket and brake disc. Widening that rear fork might seem a bit daunting, but it's actually pretty easy to do. Getting rid of the in-built suspension makes things much easier if you're struggling, and it won't actually make that much difference to the feel of the scooter



Above
To control a large motor, you will need to use a powerful motor controller like the BTS7960, which is often found on motor control modules that can handle around 43 amps, when appropriately cooled

Rad your ride with Pico

PART 02

Turn a manual scooter into a motorised monster

Dr Andrew Lewis

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

In the first part of this series, you discovered how motor controllers and throttle controllers can be used by a Raspberry Pi Pico to control an electric motor on a scooter. In this article, you will find out how to make custom battery packs for your scooter and add some features that will keep you safe while you ride.

Electric motors have high torque, but without adequate power to supply the motor you will have a very disappointing ride. Twenty years ago, a lead-acid or nickel-cadmium battery would have been the most appropriate choice for powering an electric bike. Now, lithium polymer (LiPo) batteries are a much better alternative. As you probably know, LiPo batteries are not as friendly as other types of battery, and will sometimes indicate their opposition to mistreatment by swelling up with explosive gas or turning themselves into a raging inferno. Discharging a LiPo battery too deeply can permanently damage it, and so will overheating it. All of these constraints mean that creating your battery pack requires some careful planning.

To begin with, you'll have to decide the basic voltage that you need your scooter battery to provide. With a charge level that hovers between 3.7 V to 4.2 V per cell, a 24 V motor should work with six cells in series. You will also need to consider the continuous discharge current of the batteries that you use. Some 18650 batteries only discharge at 1C (C being the rate at which the battery can be charged or discharged, such that a fully charged 2500 mAh 1C battery would provide a maximum of 2500 mA for one hour). To power a scooter motor, you will need a battery with a 10 A or even 20 A continuous discharge rate, depending on the motor that you are using.

Now that you know how many batteries you need, and have the right type, you'll need some method of using them safely. You will need a battery protection board. Battery protection boards allow you to easily build your own battery packs from 18650 cells, and look after all of the difficult jobs

Above
Here, you can see 18650 cells in a 3D-printed holder, and a battery protection board. When they are welded together into a complete battery, they will be encased in plastic to protect them. An additional rubber shim will be placed behind the protection board to make sure that no short circuits can occur

BE AWARE

As stated in part one of this tutorial, you're not going to be able to ride a scooter like this on the open road, on a footpath, or even in a bike lane in the UK. Electric scooters are in a class of vehicle known as 'powered transporters', which makes them the same as a car or a motorcycle in the eyes of the law. Scooters don't have some of the features needed to pass an MOT test, so they are not legal to ride on the road or the pavement. You risk fines of several hundred pounds and the loss of your driving licence if you do ride them in public. The government are developing new standards and adjustments that will eventually make electronic scooters legal in the UK, but they are not legal yet.

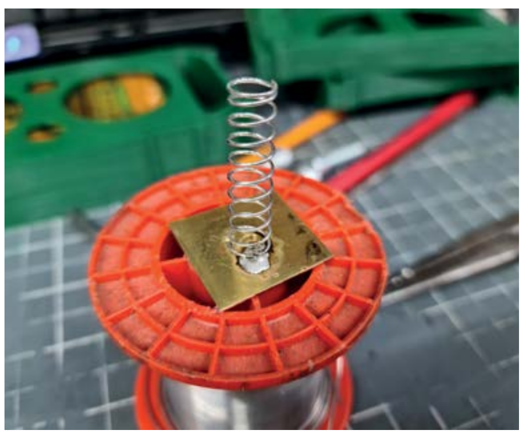
Left
The battery and motor controller for this scooter are mounted at the front. The battery is removable, and can be swapped out in a couple of seconds. Batteries are charged using an identical battery holder to the one mounted at the front of the bike, except the charger is connected to a bench power supply

TAKE CARE

Scooters – whether powered or not – carry some risks. It's your responsibility to make sure the scooter doesn't go faster than you personally feel comfortable with. You are also responsible for ensuring that any chassis modifications you make are sufficiently strong. We strongly recommend you err on the side of caution as your decisions on both these risks could impact not just you, but the people around you.

> **Battery protection boards allow you to easily build** your own battery packs from 18650 cells

like preventing excessive discharge, monitoring the battery voltages, and recharging the batteries safely. Protection boards are usually rated by current and have an S value associated with them. The S value tells you how many batteries a board is designed to protect. For example, a 24 V 15 A 6S board would have six 18650 cells connected to it, giving you a 24 V battery with a maximum discharge rate of 15 A. The battery protection board connects to each cell individually so that it can keep a close eye on what's happening. That means a 6S battery protection board will have the cells connected →



Above
If you're going to use a spring to power a device, make sure that it's thick enough to handle the power it will be transferring. If it isn't, either add multiple springs, or copper tabs with thicker braided wire at the top and bottom of the spring. Overheating power connectors can cause fires

QUICK TIP

Modern LiPo batteries have a minimum safe voltage of around 2.5 V. If you allow the battery to drop below 2.5 V, it may never recharge or work properly ever again.

QUICK TIP

Battery protection boards don't usually include temperature protection. To protect your batteries against overheating, you should consider placing a low-temperature (possibly +85°C 40A, 28V, as an example) thermal fuse in the battery pack. The thermal fuse will break the circuit if the battery heats up too much.

Right Keep your cells neat using plastic cell supports, and when you're done with your spot welding, give everything a nice generous coating of Kapton tape before the cells go into their final home (the 3D-printed removable pack, in the case of this scooter conversion)

> "You do not want any opportunity for the system to short out and cause a fire"

together in series with an extra connection between each cell for a total of seven connections, and two other connections that serve as the input/output of the battery pack.

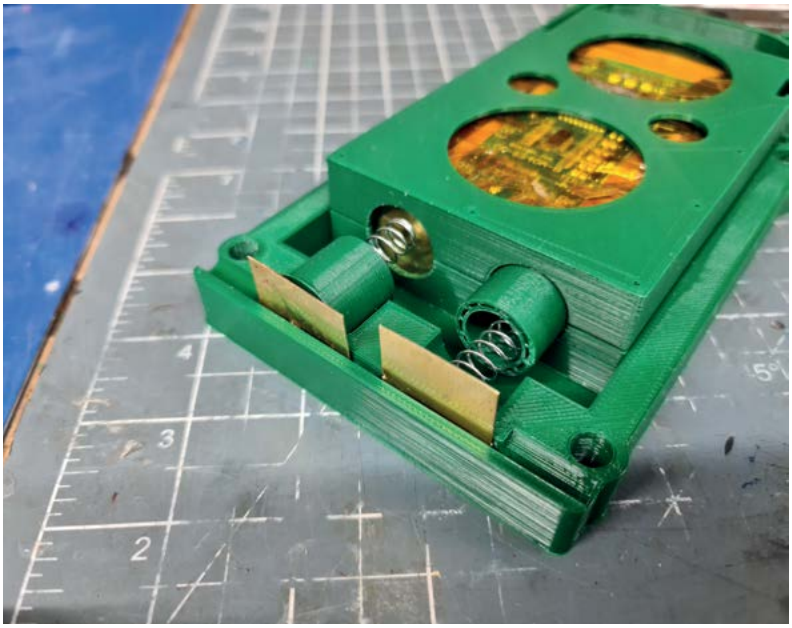
What's good for your motor isn't necessarily good for your controller, and trying to power a Raspberry Pi Pico directly from 24V will release plumes of magic smoke into the air. You need to add a small buck converter that takes the 24V battery power and reduces it down to 3.3V for the Pico and the logic side of the motor controller. Most importantly, you need to add a fuse to your scooter, immediately after the battery connection. You do not want any opportunity for the system to short out and cause a fire. □

BATTERY WELDER

One thing that 18650 batteries are particularly sensitive to is heat, which can make connecting them together a bit of a problem. The sort of temperatures needed to melt solder are too high for an 18650 battery to endure, and attempting to solder cells together will end with either a completely dead battery, or one that performs far below its expected rating. To join 18650 cells together, you really need to use a battery welder. The battery welder is a miniature spot welder that allows you to join nickel strips to batteries (and PCBs) without excessively heating the battery. The power used to spot-weld the metal strips is tightly focused, and dissipates very quickly before it can cause damage by overheating the cells. Battery welders are relatively inexpensive, and are worth purchasing if you have any power tools that might need battery packs replacing in the future. They often come with a selection of nickel strips to suit different battery needs.

QUICK TIP

It's best to think about using removable battery packs if you are going to ride your scooter regularly. Having to wait while the hard-wired batteries recharge is not ideal.



Above
The box below the battery houses the motor controller, fuse, and DC-DC buck converter that reduces the 24V battery power down to 3.3V for the Pico and motor controller logic. The key in the side can be removed to disconnect the enable signal for the motor. This isn't actually that important, since if you were leaving the scooter, you'd probably remove the battery anyway

Above
If you are making your own battery packs rather than using the 3D model files provided with this article, you need to make sure it's impossible to connect your batteries in reverse. Providing reverse voltage to the buck converter would probably kill every single component in the scooter, and cause a great deal of swearing

SPI DISPLAY

Although it isn't strictly necessary, it's quite nice to have a display on your scooter to tell you how much power you're applying at the throttle, and optionally monitor your battery levels using a potential divider connected to one of the analogue inputs on the Pico. A round ST7789 SPI display can be connected to the Pico using the instructions at **hsmag.cc/SPIDisplay**, with some example code included with this project to show how it can be used as a simple power gauge.

HackSpace
TECHNOLOGY IN YOUR HANDS

INSPIRATION

HACK | MAKE | BUILD | CREATE

A collection of quite brilliant projects to dazzle and inspire

PG 96

LARGE-FORMAT CAMERA

Create super high-quality images on a camera you printed yourself

PG 102

HEATED SEAT

Winter is coming: add battery-powered heating to keep your bum nice and warm

PG 108

ROCKET LAUNCHER

Blast off with home-made rockets, ABS piping, and compressed air

HOW I MADE

PRINT YOURSELF A LARGE FORMAT CAMERA

Create super high-quality images using a camera you printed yourself

By Rob Miles

Modern digital cameras use image processing technology to produce high-quality results. But this makes everyone's pictures look perfectly identical, and it takes a lot of the satisfaction out of getting a good picture (at least for the author). In this article, you'll discover how to use retro technology and 3D printing to create a super high-quality film camera that gives your photographs a real sense of occasion.

HAVE CAMERA, WILL TRAVEL

Figure 1 shows the completed camera. The body is 3D-printed from designs at **hsmag.cc/WillTravel**. The body was printed in white PLA, which turned out not to be the most lightproof of colours. You should use a darker colour. The camera has since been sprayed with multiple coats of matt black paint – both inside and out, to make it more light-tight. The light proofing is not quite perfect on the author's camera, but this is only a problem if the camera is

Figure 1
The hand grip on the left contains a cable release which is linked to the shutter on the front of the camera. You can trigger a shot by pressing down the plunger on the handle. This makes it much easier to use. Plus, the author thinks it looks cool

What I used

- A lens and shutter assembly. Search your favourite auction site for 'large format shutter lens'
- Some 4×5 film holders. Search for '4×5 film holders'. You put your unexposed film into them, and they attach to the back of the camera to expose the film. These can be found quite cheaply
- If you are making the helicoid version of the camera, you will need a 'M65 to M65 Mount Lens Adjustable Focusing Helicoid 17–31 mm Macro Tube Adapter'
- 6 × M3 6 mm countersunk bolts and 6 × M3 3 mm (length) × 5 mm (outer diameter) brass knurled insets
- 3/8" to 1/4" tripod screws adaptors
- If you want to add the viewfinder, search for '2 Functions Mobile Phone Lens 0.45 × Wide Angle Lens and 12.5 × Macro HD Camera Lens' on your favourite shopping site
- A dark 'changing bag'. This is a light-tight bag where you transfer your film into the film holders that are loaded into the camera
- A 4×5 cm ground glass screen. The best place to buy one of these is from Intrepid Camera (**intrepidcamera.co.uk**). It also sells Fresnel lenses
- A length of elastic 'bungee' cord, as used in jackets and hoodies. This holds the back screen on the camera and stretches to allow you to insert the film holder
- Designs for your camera. You can download them from here: **film.kolve.org/3ddesign**

used in direct sunlight. He plans to add some felt inside to improve this. The camera lens at the front is mounted onto a helicoid (the black cylinder with the three knurled rings behind the lens in **Figure 1**) which is turned to move the lens in and out and adjust the camera focus. There are also designs available which use 3D-printed threads on the lens holder to screw the lens in and out.

TO INFINITY AND BEYOND

One of the first things you need to do when you have built your camera is discover the focus position for infinity. Ideally, this would be when the lens is closest to the camera. Then, as you move the lens further away from the camera, it will focus on closer objects. You identify the infinity position by pointing the camera at a distant object and then adjusting the lens until the image is sharp. You can use this for 'zone focusing', where you just set the lens for the distance the camera is from the subject, and everything in that zone will be sharp. If you look at the camera in **Figure 1**, you will notice some white dots painted on the lens mount. The two you can see identify the infinity focus →

KEEPING THE COST DOWN

Photography can be expensive. You can spend a lot on a modern digital camera. But you can have fun and make amazing results for a lot less cash. The most expensive component of the camera is the lens and shutter assembly. This focuses the image and controls the amount of light that reaches the film. These lenses can cost over £100, but the good news is that, unlike the latest digital camera, an old lens will hold its price very well. It might even go up in value. You can swap a single lens between multiple cameras, as they all use the same fitting. Film and developing your photos are getting expensive, but there are some very good low-cost films around, and you can find bargains with expired sell-by dates. Home processing is cheaper and quicker than sending your shots away and is a lot of fun. You can pick up a second hand scanner to get your negatives into your computer, or you can use a smartphone to photograph the negatives and an app to convert them to positive images. For around the cost of a few video games, you can get yourself started on a creative and fun hobby.

Figure 2 You can add a Fresnel lens on top of the ground glass to make the image brighter

position. When the two dots are aligned, the camera is focused on the far distance. There is also a dot for 2m focus distance.

IN THE FRAME

To set up your photograph with the camera, you attach a viewing screen to the back, as shown in **Figure 2**. The screen is held in place with elastic cords. It contains a sheet of ground glass which is placed in the position that the film will be in when the picture is taken. The lens projects an image onto the ground glass. Because of the way lenses work, the image is inverted and flipped left to right, which makes the viewfinder interesting to use, but you do get used to it. The image can be hard to see, especially if you are outside. You can improve your view by putting a 'dark cloth' around the camera and over your head to keep out external light. This also gives your pictures a proper Victorian sense of occasion.

If you are using zone focusing, you can set your focus using marks on the lens and use the round optical viewfinder on the top of the camera to frame your picture. The viewfinder is based on a close-up lens which is sold for use on mobile phones. When you

Right →
The bright parts of the sky have come out completely black, whereas the dark area under the table is white in the negative

are ready to take your shot, you slide a film holder in front of the viewing screen.

WELCOME TO THE DARK SLIDE

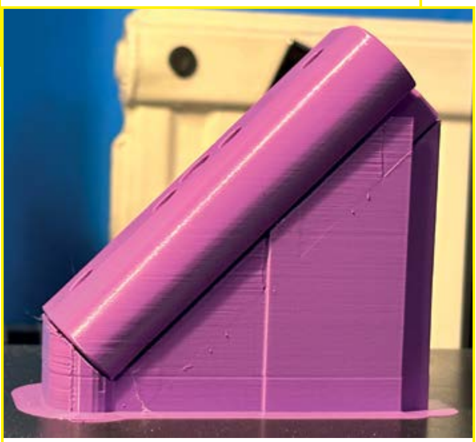
The film holder contains a 'dark slide' that protects the film from light. When the holder is on the camera, you remove the dark slide, take the picture (having remembered to set the shutter speed and aperture), and then put the dark slide back into the holder and remove the holder from the camera. A film holder contains two sheets of film, one on each side. The author owns two film holders, which means he can go out and take four pictures before he must come home again. Before any pictures are taken, the film and empty holders are placed into a special light-tight changing bag, which has light-tight cuffs sewn into it. You put your hands through the cuffs and then load the film into the holders. After the pictures have been taken, you use the changing bag to get the film out of the holders into either a light-tight box to be sent away for processing or a development tank.

ACCENTUATE THE NEGATIVE

The development process produces a negative image; the darker parts of the negative are made from tiny grains in the film which have been illuminated and then turned to silver. You make a print from a negative by projecting it onto photographic paper which, like film, turns darker when exposed to light. This reverses the light and dark of the negative, resulting in a positive image. If you don't want to do that, you can use a scanner to create an image file that can be processed by software to create the correct appearance.

HANDLE ANGLE PRINTING

You can see the camera handle on the left-hand side of the camera in **Figure 1**. It has a complex shape that made it very difficult for the author to print. If it was printed vertically, the handle came unstuck from the printer, causing the print to fail. If the handle was printed horizontally, the slot for the cable release, which runs the entire length of the handle, needed to be printed with internal support that was very hard to remove. In the end, the problem was solved by printing the handle at a 45-degree angle so that it had lots of contact with the print bed and needed no internal support. This was perhaps a bit wasteful with filament, but it did make for a very good-looking result. This is something that you might like to try if you have a tricky thing to print.



"LOAD THE FILM INTO THE HOLDERS"

THE BIG PICTURE

The camera takes pictures onto sheets of film that are 4×5 inches in size. These images contain lots of detail and, because the lens is a long way away from the film, you can get beautiful out-of-focus effects. Lenses have a property called 'depth of field', which determines how much of an image is in focus. A lens with a large depth of field would take a sharp picture of both you and a tree in the distance behind you. A lens with a shallow depth of field will be able to take a sharp picture of either you or the tree, but not both. You might think that having more sharpness in a picture is a good thing, but you can get some very pleasing effects by blurring parts of an image. The picture of the duck in **Figure 3** looks better because the →

LARGE FORMAT CAMERA

"OUR 3D-PRINTED CAMERA...LETS PHYSICS DO THE HARD WORK"

trees and fence in the background are blurred. Smartphone cameras are very small and put their lenses very close to the image sensor. Because of the physics behind the way lenses work, this means a phone camera has a large depth of field. Phones use sophisticated software that can analyse an image, decide which bits are in the distance and blur them. The results can be impressive, but our 3D-printed camera just puts the lens a very long way from the film and lets physics do the hard work.

Figure 3 The lens was focused on the duck, which is very sharp. The trees in the background are blurred

DIGITAL VS ANALOGUE

A digital camera breaks a picture down into a grid of picture elements or pixels. Each pixel stores red, green, and blue intensity values representing the colour of that position in the image. An image is recorded by a grid of sensors producing light intensity values at each pixel. The value is the amount of red, green, or blue light falling at that pixel. These values are used to calculate a colour for each pixel. Analogue cameras use film coated with silver halide particles called 'grains', which are chemically changed when hit by light. Chemicals in the film developer convert the changed grains into opaque grains of metallic silver. Colour film uses three grain layers which are sensitive to different light wavelengths. During development, these are bound to dyes which react with the silver in the grains to produce the complementary colour for that primary. Red areas are green, green areas are red, and blue areas are orange.

You could say that film images are more 'real' because they have not been subjected to any digital process. A difficulty with digital cameras is that, because each pixel only measures one primary colour, the final of each pixel is calculated using the colour values of the pixels around it. This is why different digital cameras produce slightly different results; they each use different algorithms to combine the sensor values. An advantage of analogue photography is that the colour resolution is spread over the whole image. An advantage of digital photography is that there is no film to be bought and developed, and you can view the images as soon as you have taken them. Both have their place, and they are both great ways to take pictures.

CONTROLLING EXPOSURE

When you take a picture with your phone, the camera works out how much light the sensor needs to get a properly exposed image. When you take pictures with the printed camera, you need to set the shutter speed (how long the film is exposed to the light) and the aperture (the size of the hole the light comes through). The easiest way to do this is to use a light meter application on your phone. You set the speed of the film in the app (how sensitive the film is to light), point your phone at the scene, and then copy the reading values onto the camera.

If your shutter speed is less than a thirtieth of a second, you will need to put the camera onto something solid. Otherwise, camera shake will blur the image.

A camera can only capture a fraction of the range of light and dark areas in a scene; if you get your exposure wrong, you might find that the faces of your subjects are too dark or too light. You deal with this by measuring the light from the parts of the scene you care about. As you take more pictures, you can use this limited range to your advantage, making a subject stand out by blowing the background out to white or turning the background dark. Next, you will start using lights to control how the picture looks, and hey presto! – you are turning into a proper photographer.

PRINTING A CAMERA

The most expensive camera in the world is still just a box with a lens at one end and something light-sensitive (film or an electronic sensor) at the other. So, if you want to make your own camera, you only really need a light-tight box. However, it's not quite that simple. The size of the box must exactly match the lens you are using. A lens has a particular 'flange focal distance' (FFD), which is the 'in focus' distance from that lens to the surface of the film. You need to make sure that the box you make is the right size for your lens. The website at **hsmag.cc/WillTravel** has a lookup table you can use to find the FFD for your lens. You can then select an appropriate design from the ones available.

The Will Travel body contains threaded holes you can use with 3/8" to 1/4" adaptors to allow you to put the camera on a tripod. There are three accessory shoes you can use for flash-guns, lights, or the viewfinder. There are also holes into which you can fit spirit levels. The handle can be used on either side of the camera body.

OCCASIONAL PHOTOGRAPHY

The 3D-printed camera brings a real sense of occasion to taking a picture. On a good day, the results will stand comparison with very expensive cameras. The lenses and shutters might be old, but the results they generate are very impressive and can be made into enormous images. Also, taking photographs 'the hard way' teaches you a huge amount about photography, and how to use the limitations of the medium to get the best results. If you are of a certain age, you might even find it fun to revisit the days of home development and seeing pictures appear before your eyes in the darkroom. The author has had a lot of fun with this and hopes you do too. □

FANCY FITTINGS

The lens mount is fitted onto the camera body using bolts that screw into brass fittings which are embedded in the camera body. The author had never used this technique before. It is a very quick and simple way of creating strong and reusable fittings.

To put the fittings into the camera body, you will need a special tip for your soldering iron. You can find one by searching your favourite e-commerce site for 'heat-set insert installation tip'. The brass fitting is heated up on the tip and then pushed into the hole in the camera body.

The author hasn't tested the fittings to destruction, but they seem to be a lot stronger.

FEATURE

HOW I MADE

By **Andrew Lewis**

WINTER IS COMING

Fighting the effects of winter in your workshop with heated chairs and insulation

Winter is coming. It's cold, it's wet, and it can be the most difficult season to find the literal and figurative energy to keep making things. The cost of heating is getting higher for everyone, and a dedicated workshop can quickly turn into a money-eating monster in the winter months. For all of the advantages of having a workshop, there are some pretty big disadvantages, too. A permanent workspace is exactly that. It continues to exist in space even when you're not using it. Some tools, consumables, and machinery can be ruined by cold temperatures, and in winter, a workshop becomes an extra place to heat. In this article, you'll find out how to heat a workshop more effectively, and how to stay warm yourself without relying on power-hungry heaters.

A maker's haven for the summer will become a burden without proper winter planning. It isn't just about feeling the cold, it's about how the cold effects your equipment. Inaction through ignorance or simple failure to prepare for the winter

can lead to expensive failures when the temperature drops below freezing. To get the best out of your workspace, you'll need to address both of these problems. Keeping your equipment in working order, and keeping yourself warm while you're working.

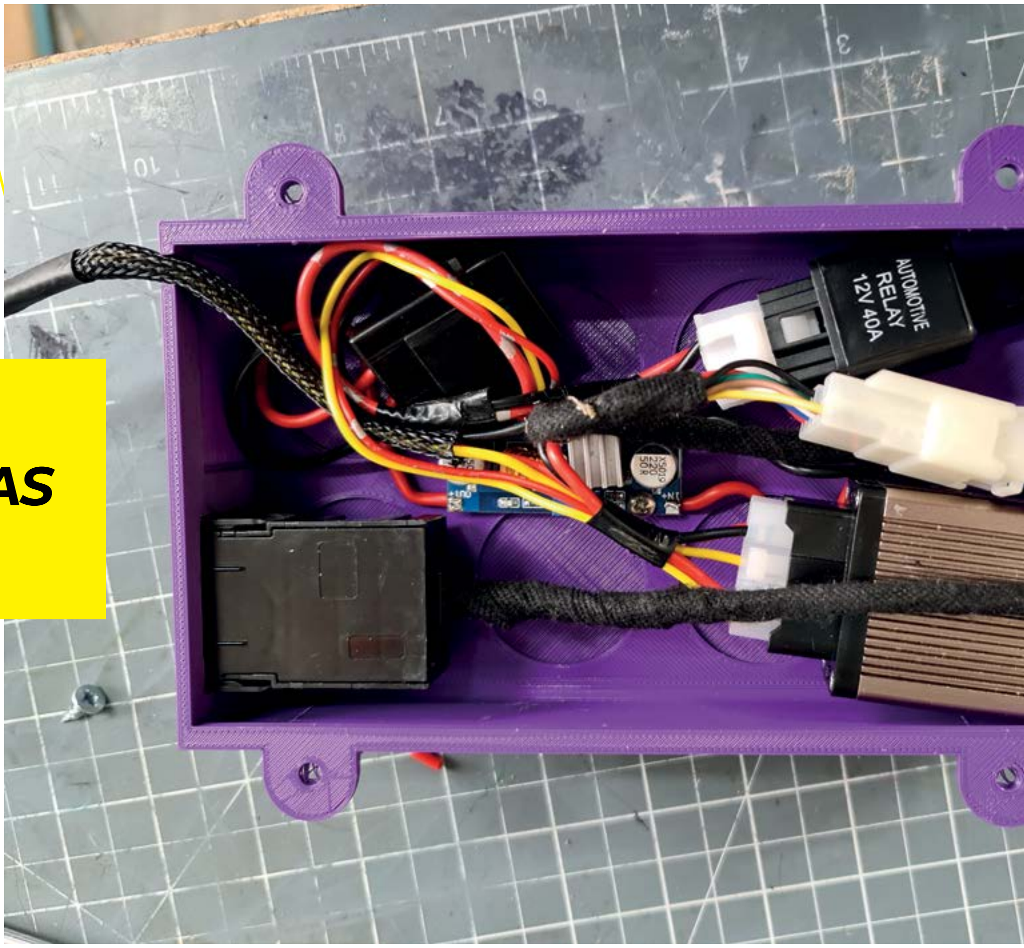
When you're working on larger projects, you're generally moving around and doing things that will help keep you warm. Making sure that you're wearing appropriately warm clothing means that you'll be happy in your work. However, if you're doing fine work, sitting still at your workbench, you'll start to feel the cold very quickly. So how do you solve this problem? You can borrow a solution from the automotive industry and create your own heated seat. Heated seats are popular in modern cars, and it's not uncommon for people to want to retrofit them into older vehicles. There are plenty of kits available online that feature heating pads and controllers designed to fit into car seats. With a little bit of fiddling, it's possible to add one of these kits to most office chairs, and you can power it with an electric tool battery that will last between two and five hours (depending on how warm you want to be). →

Above ↑
Heating the person, not the space is a common way to increase comfort levels cost-effectively. While it won't solve all equipment problems, strategically placed low-power heaters like the one installed in this chair can make a big difference

"HEATED SEATS ARE POPULAR IN MODERN CARS"

STAYING SAFE

When powering the heating elements from an 18 V power tool battery, you will need to use a DC power converter to reduce the voltage down to 12 V. A minimum of a 5 A DC converter is recommended, given the power requirements of the heating elements. The first item in your circuit after the battery should be an appropriate fuse. In this case, a 5 A fuse should be plenty. Adding a thermal fuse to each heating element is advised. A 76°C 3 A fuse like the SETFuse X0 can be used to cut power if the heating element ever malfunctions and overheats. Position the fuse in-line on the positive wire, and fix it somewhere on the heating. You can keep it in place with Kapton tape. If the temperature of the element gets above 76°C, the fuse will blow, preventing a possible accident.



"REMOVE AS MANY STAPLES AS YOU NEED"

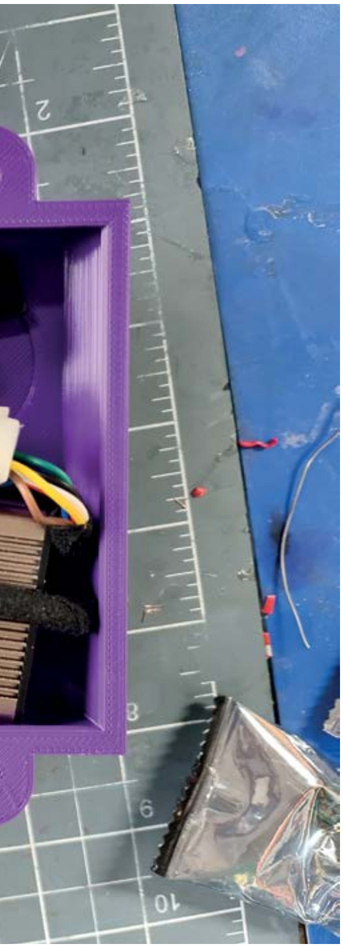
Right →
You can see all of the components that come with the seat heater here, along with a DC power converter that regulates the power from a tool battery to 12V

WATER WOES

A hidden danger of a cold workshop is condensation. If the workshop and its contents get cold enough for water to condense on them, chaos will ensue when you try to use them. Firstly, water can condense, drip, pool, and rust your machines. Secondly, water inside electronic equipment or on sensitive items like laser cutter mirrors will lead to disaster. One way to deal with this problem is to use chemical moisture traps to capture water from the air. Another way is to employ an electronic dehumidifier. The best approach is to keep the temperature of the workspace stable at a temperature that's high enough and dry enough to prevent moisture from condensing. A heater with a frost switch can help with this, turning on to keep the temperature somewhere above freezing. It's also possible to include a small heat source (such as a very low wattage incandescent bulb) to effectively heat the inside of some equipment and prevent condensation from forming. This can be a good solution for equipment with relatively large enclosures, like a laser cutter or an enclosed 3D printer, where you can use the machine's heated bed to keep the internal chamber warm. Heating an enclosure like this will use power and cost some money, but it's cheaper than having to replace components that have been damaged by condensation. If you do find that your equipment has gotten too cold, you'll have to gradually warm it and make sure that the water inside has evaporated before you start using it.

START AT THE BOTTOM

A typical seat heater kit for a car will have four heating pads, a relay, a dashboard switch, and a small circuit board or sealed box that lets you control how hot the seat will get. Often, these kits don't come with any instructions, and you might be a bit confused by the number of power wires that are sticking out of the control box. Normally, you will have two thicker red and black wires, which are the main 12V power input for the heater. You'll also have another thin wire that controls a main power relay, and possibly a thin wire that will make the switch light up on the dashboard when the headlights are on. Because you're probably not going to be driving your workspace around at night, you can safely connect these wires to the positive 12V line and forget about them. If you look at the switch for the heaters, you'll notice that there are two dials. One dial is for the passenger seat, while the other is for the driver's seat. As you'll only be using one seat, you can use these dials to



HEATING THE ROOM

A well-insulated workshop (or any other building) can remain remarkably warm without any heating. However, it's difficult to achieve such high levels of insulation unless the building has been built specifically with eco-living in mind from the beginning. For most people, some level of heating is going to be necessary. You can discount any idea of portable gas or paraffin heating, as they produce high levels of very moist air and fumes that make them practically useless or downright dangerous to use in an enclosed workspace. Conventional electrical heaters are potentially useful, although convector heaters typically use several kilowatts of electricity every hour, making them a very expensive option.

control the temperature on the back and the seat of your chair separately. Simply connect one pad to the passenger connector, and the other pad to the driver's side. You'll need to cut the wires from the control box, since it's designed to be fitted in a car a metre or more away from the seat, and you'll be fitting the box underneath your chair about 15 cm away from the heater panels.

Fitting the heating elements inside the office chair is the fiddly part of this particular project. If you're lucky, your chair's seat cushions will be removable and have zips or Velcro strips that you can undo to get inside the padded parts of the chair. Unfortunately, the vast majority of office chairs are not designed this way. The fabric for the seat will more likely be stapled onto laminated board, and you'll need to disassemble the chair to get access to the inside of the seat cover. Often, you'll need to remove the arms and legs of the chair, and then disconnect the seat from the back-rest if it is removable. You should be able to see the staples once the chair is disassembled.

Remove as many staples as you need to get inside the seat cover, and then slide the heating element in between the outer cover of the seat and the interior foam. This seems like a simple instruction, but in reality, you will probably spend 20 minutes cursing this article, the person who wrote it, and all of the ancestors of the person who designed the chair while you are trying to slide the →

Below ↓
Removing staples is much easier with a decent pair of pincers. They offer easy leverage and are less likely to accidentally grip and rip your chair's fabric than ordinary pliers

CHILLY CHALLENGES

It isn't just machines and people that suffer the effects of the cold weather. Greases and oils behave differently in cold weather, becoming more viscous and potentially becoming unusable. Some industrial paints can be ruined if they get too cold, and most paint won't brush or dry properly if it's below 10°C. Spray paints won't spray properly in cold weather, and butane gas canisters will stop vaporising gas properly at roughly the same temperature. The pressure in acetylene tanks will drop as the temperature decreases, too – so don't be too surprised if that full tank reads almost empty now that it's colder. Battery power drops off dramatically in cold weather, meaning that you'll need to change and charge batteries more often on power tools, and some items like multimeters that use disposable batteries might start to give battery warnings or read incorrectly. If you're going to need temperature-sensitive consumables, it's probably best to bring them into the warm for a few hours before you use them.

Right →
Pulling plugs and wires through fabric and foam can be tricky if you've never done any upholstery work. Push pliers through from the outside of the foam first to open up the hole, then grip the plug and pull it through to the outside. Don't use too much force, as you're likely to bend or break the plug

Below ↓
Screw the project box in place under the chair so that you can access the switch and the battery easily. The base of most office chairs has surprisingly thick wood, but take care not to use over-long screws

stubborn heating element flat against particularly grippy upholstery foam. Worse, you need to hold the heating element in place with double-sided tape, removing the backing while it's in position inside the chair, and then finally restaple the outer cover back into place while remembering to keep the wires for the heating element hanging outside the fabric.

SIMPLE BUT WARMING

All that remains is to connect your control box to the heating elements, and stash all of the electronics in a box under the seat. Make sure that you position the box so that

QUICK TIP

You can fix one of the remaining seat heating elements to the underside of a self-healing cutting mat, and use it to warm the mat up. This will help keep your hands (and tools) warm while you're working.

COOL CUTTERS

One of the most temperature-sensitive machines in the workshop is a laser cutter. The laser tube is fragile. The coolant used in lasers can freeze in the winter, cracking the laser tube, bursting coolant pipes, and damaging pump diaphragms. On top of this, bearing grease will stiffen in cold weather, increasing the chance of a stepper motor stalling, and contraction and expansion from temperature changes can throw off the alignment of lenses and rails. Running the laser when it's too cold can cause the tube to crack from thermal shock.

PRINTER PROBLEMS

3D printers are sensitive to cold, but not necessarily in the same way that other precision machines are. Issues with thickened bearing grease and frame warping do exist for 3D printers, but for machines without an enclosed chamber, the temperature of the environment can be critical for a successful print. If the room is too cold, prints can warp and fail, and bed adhesion might also be affected. For resin printers, the temperature of the resin affects the cure time, and a cold resin tank can cause unexpected failures on previously successful prints. A good solution for 3D printers is to (if possible) build an enclosed area just large enough to house the printers and their accessories, and then heat that area using the printer's bed heater or an incandescent bulb for some time before you start printing.

the dashboard switch is easily accessible while seated, and connect a 3D-printed battery connector that matches your power tool batteries to a 12V DC regulator. Connect the battery, turn on the heaters, and warm your heart by making comfortably.

It's one thing to pump heat into your workshop, but if you don't have any insulation, then you're just going to be throwing away money. Use whatever you can to improve the insulation in your workshop. Cover shed windows with bubble wrap to reduce heat loss through the glass, and use the best insulation you can get your hands on to insulate walls and doors. Even

> ***"COVER SHED WINDOWS WITH BUBBLE WRAP"***

layers of cardboard will help reduce heat loss through walls, ceilings, and floors. However, cardboard will absorb moisture and will lose it's effectiveness when wet. Foil, bubble wrap, or foam insulation is a much better option if you can afford the extra expense.

Draught excluders and neoprene foam tape can be very useful if your workshop is in an outbuilding or shed where doors and windows don't necessarily fit as closely into their frames as they do inside a regular house. Plugging up those leaky seams can go a long way towards reducing heat loss. □

HOW I MADE

By Jo Hinchliffe

AIR POWERED ROCKET LAUNCHER

There are lots of ways to launch rockets into the air. Most obviously, with a combustible engine but, here, we're going to look at a different method: compressed air.

We've kept in mind various safety considerations where we can. That said, there is some inherent risk. Make your own sensible judgements and don't put others in danger. Only attempt to recreate this project if your skill level and materials allow you to do it within your own risk tolerance.

Lots of these DIY cannon-type projects have historically used PVC waste pipe fittings, with many a recipe online referring to 'schedule 40' piping. This isn't a term that's used in the UK to describe pipes, but we were heartened to see, on a visit to our local DIY centre, that many of the waste pipe products in the UK are now manufactured from ABS rather than PVC. While this doesn't instantly make them impervious to pressure, ABS does seem safer in terms of toughness and how it fractures when compared to how PVC splinters.

What I used

- Length of 32 mm ABS solvent weld waste pipe
- Length of 21.5 mm ABS solvent weld waste pipe
- 2 × 32 mm ABS solvent weld coupler tubes
- 32 mm to 21.5 mm ABS reducer
- 32 mm ABS threaded inspection closure
- 21.5 mm ABS solvent weld 90-degree section
- 2 × 21.5 mm to ¾" BSP adaptor
- Brass ball valve male-to-male ¾" BSP
- Thread-on Schrader tyre valve
- PTFE tape
- Electrical tape and glue
- Sticky tack
- Card stock

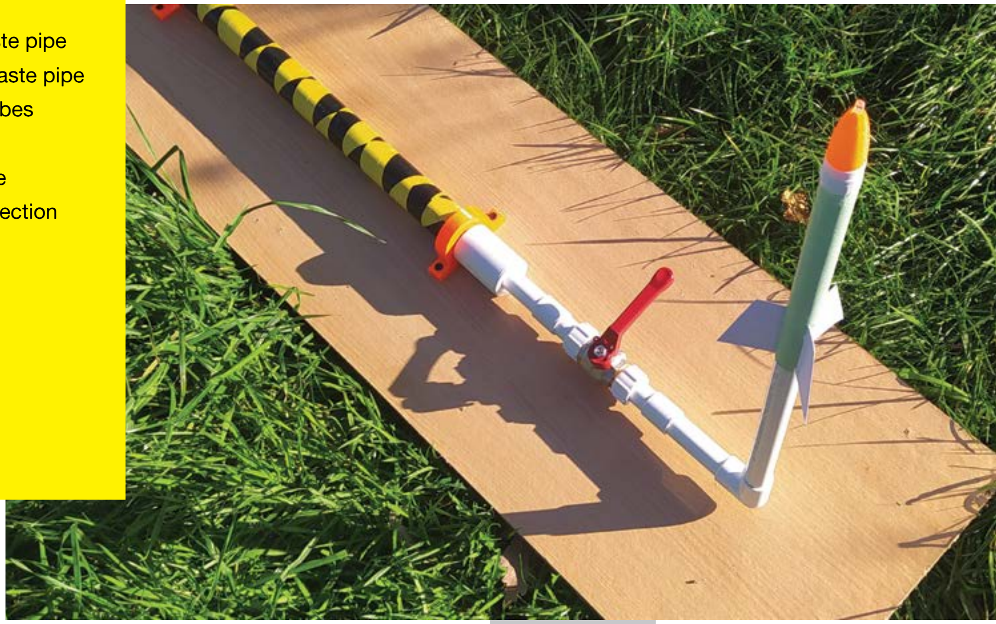


Figure 1 ↑ Using pipe cement to solvent weld a coupler and inspection cap onto the end of our 32 mm pipe

The principle of the air launcher is there is a reservoir side of the system that has air pumped into it to build the pressure. This pressure is contained in the reservoir until a valve is released, allowing the pressurised air to be pushed out through the system. The rocket is then sat on top of a pipe on the low-pressure side of the system.

To build the high-pressure reservoir side of our system, we used a length of 32 mm waste pipe and some plumbing accessories. It's important that all the accessories are 'solvent weld'-style and are designed to be cemented together rather than 'push-fit', which use a simple pressure and gasket/seal system to clip together. The later push-fit-type components wouldn't be able to hold any useful amount of pressure. We cut a length of the 32 mm waste pipe to 75 cm using a hacksaw, taking care to keep the ends of the pipe as square as possible. There are special tools available that you may have in your toolbox for cutting plastic plumbing pipes and would give more perfect pipe ends, but a reasonably close to square cut will suffice.

At one end of the cut tube we fitted an accessory coupler, which allowed us to then glue into place the inspection-style threaded end cap (**Figure 1**). This is a pipe-closing accessory that has a threaded cap, allowing the pipe to be terminated but with the option of opening it if needed. The cap is sealed with a large rubber gasket inside and has worked well with an input valve placed through it. For our valve, we have used a threaded Schrader valve, the kind of which are used in bicycle and car tyres (**Figure 2**, overleaf). You can find the threaded versions online which have a flanged base, an inside and an outside seal, and a threaded shaft and nut. Drilling an 8 mm diameter hole through the inspection cap, we can insert the valve with a seal →

STAY SAFE!

The risk with this – or any pressurised system – is that if it starts to crack, it can suddenly explode. As a maker, it's up to you to understand and mitigate this risk to a level you feel comfortable with. The first thing you can do is keep the pressure manageable, and this means using a pump with a pressure gauge. Stay under 2 bar (30 PSI) – that's about most bike tyre levels. Secondly, make sure you're wearing appropriate protective equipment. At a minimum, that should be suitable eye protection. If you're handling it while pressurised (which we don't recommend), wear gloves as well.

The further you can be from the pressurised system, the safer you are, so any onlookers should stay back (especially younger makers). The person charging and firing the system should stay as far away as possible – for example, using a pump with a long hose gives you more protection than having to stand right next to it.

Finally, an extra protective layer around the high-pressure side provides even more safety. This can be a wrapping of PVC tape or (even better) a heavy blanket. It's impossible to make this project completely risk-free. It's up to you to decide how it fits with your own risk tolerance.

Figure 2 ↑
A threaded Schrader-style valve with rubber seals

each side and tighten the nut down. This creates a great seal. We can then, in turn, thread on the inspection cap to form that end of the reservoir.

At the other end of the high-pressure reservoir, we needed to step the 'tank' down to the 21.5 mm diameter pipe size which connects to our release valve. Simple solvent weld reducers are available, which again require an accessory coupler attaching to the 32 mm tube into which they are inserted. Once the reducer was in place, we cut a short length of 21.5 mm waste pipe to connect our reducer to the 21.5 mm to ¾" BSP threaded component. The component we used for this is slightly less common and not stocked in general DIY shops, but they are widely available online. We ordered a pair of them from **hsmag.cc/SolventWeld**.

> "WE NEEDED TO STEP THE 'TANK' DOWN"

We should say that we bought these components to match the ¾" BSP brass ball valve that we had decided to use for this build. We wanted the valve throat of the release valve to be a similar diameter to the 21.5 mm tube; however, you may find another valve with a different threading which would suit, but ensure you can find a way to connect it into the system, though. With the 21.5 mm to ¾" BSP plastic adaptor cemented onto our small section of 21.5 mm pipe, we could then connect the brass ball valve (**Figure 3**). We'd explored some cheap solenoid valves considering we could make a decoupled launch system that could be operated remotely with a battery and a switch; however, we found that these valves, primarily made for water/liquid systems, would leak air and therefore weren't suitable. On assembling the plastic threaded connector onto the brass valve, we realised that, mechanically, this was the weakest point, and the plastic threaded section was slightly prone to leaking air. To rectify the leaks, we sealed the thread using some PTFE tape. After assembling the system and checking it, we opted to use some cement in the threaded cap to seal it into place, adding a little mechanical reinforcement.

We did exactly the same for the plastic threaded adaptor on the other side of the valve: sealing with tape and reinforcing with cement. These were the last parts that we cemented. We realised that the lower pressure/output side of the system didn't really need to be cemented together as the tubes are only really guiding the fast flow of released air. This provides a couple of advantages, one is that you can remove the output side, making the system smaller to transport, and also, you can experiment with different configurations and launch angles.

To make a rocket, we definitely encourage experimentation! We're sharing what we did, but feel free to try lots of different ideas. One of the most crucial factors for a successful rocket launch is that the tube that forms the rocket body is not too tight or too loose when slipped over the launcher's 'barrel'. Our rocket tubes are made by rolling card around a mandrel or form, with this being made from a length of 21.5 mm waste pipe, but with one strip of thin tape added to slightly increase the diameter of the mandrel. This seems to work well to get a good sliding-fit tube to base your rocket around. If your resulting rocket doesn't launch and rather the compressed air hisses out slowly, it's highly likely that your rocket is not a good fit – either too tight or too loose on the launcher.

For most of our rocket experiments, we've used around a 75 mm strip of card stock cut across the width of a portrait piece of A4 card. We found that this wraps around the mandrel with around 5 mm of

QUICK TIP

We've kept in mind various safety considerations where we can. That said, this is very much a 'do this at your own risk' project. Make your own sensible judgements, and don't put others at risk!

Figure 3 ↑
The reducer, 21.5mm to ¾" BSP couplers, and the brass ball valve installed

tab crossed over for gluing. Our technique is to pre-roll the card a little around the mandrel to get it to curve. Then, we wrap it tightly around the mandrel – don't worry too much if it's straight or twisted. Holding it in place with one hand, we then place around five rubber bands over the card on the mandrel to hold it in place. Once held, you can twist the card a little until the overlapping ends are even and square, and the tube isn't twisted. Once in place, we tend to shift one rubber band off the cardboard and place a smear of PVA glue under the flap and press it back together. After holding this section for a few seconds, you can then replace the rubber band over it to keep it in place. Then, in turn, move along the tube, adjusting the rubber bands a little at a time, inserting glue, and re-clamping with the rubber bands. Eventually, when you get to the other end, the seam should be neatly glued and closed all the way along. Leave the tube on the mandrel for a few more minutes, but then remove the rubber bands and make sure that the tube can move up and down on the mandrel, checking that it hasn't been glued on, then leave it to dry for an hour or so (**Figure 4**, overleaf).

For simple sets of fins, we've found that cutting six fins of the same shape out of the same thin card stock as the body tube, and then laminating them together in pairs to form three fins, seems to make a tough enough setup. As for fin design, again, we'd urge you to experiment. We've hand-cut fins that are around 6 cm tall, with 2 cm on the leading edge and 4 cm on the trailing edge – these worked well. We've also got fancy and used our vinyl cutter to cut semicircular fins from card stock (we looked at cutting card with vinyl cutters in issues 58 and 59).

In larger, more accurate rocketry, people use all manner of fin guides and jigs to accurately place fins on a rocket body tube to ensure they are straight and true. For these air rockets, however, it's OK to hand-place them and judge by eye. One good tip for gluing on the fins is to place a drop of quick-drying superglue at either end of the fin and then put PVA glue along the middle section. This means that you →

GETTING STUCK

All of the permanent joints that aren't threaded together in our build are cemented using pipe cement to solvent weld the parts together. Solvent cement is widely available and reasonably priced and is often sold alongside waste pipe products in DIY shops or plumbing suppliers. It creates a very secure joint very quickly. You simply apply a generous layer, 1–2mm thick, on both mating surfaces and let them stand for around ten seconds before pushing them together. You'll tend to squeeze out some glue as the joint is pushed together, so it's handy to have some paper towels to hand to wipe off the excess. The joints are strong enough to handle after a few minutes, but not fully cured for a few more hours. Being cautious, we left our system for a day for everything to fully cure before testing it with any pressure.

Below ↓
Making rockets to launch is simple, quick, and cheap

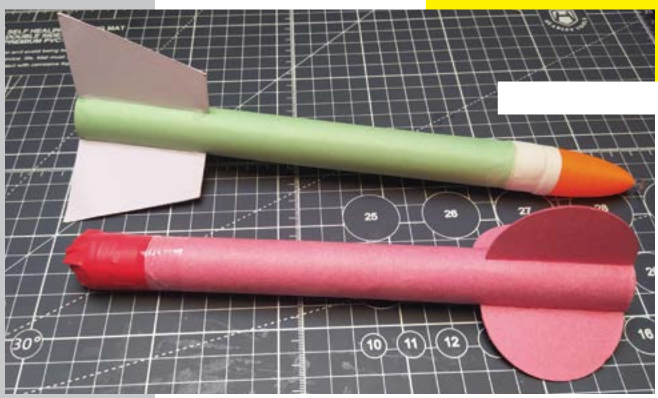


Figure 4 ↑
Using a section of 21.5mm pipe, with a strip of tape added, to create a good-fitting rocket tube

can hold the fin in place for around 30 seconds – the superglue will 'weld' the fin in place, drying quickly, and then you can leave it unclamped for an hour or two while the PVA dries. It's worth adding glue fillets to the fins where they join the body tube. After the initial glue has dried, add more glue on either side of the fin, building up a couple of layers to add strength.

You'll need to add some nose weight to enable your rocket to fly well. You will also need to seal the end of the rocket so that the compressed air doesn't leak out. A simple but effective way to do this is to begin by adding a layer of tape around the upper end of the rocket and folding it into the inside of the rocket to make a little surface. We've then added about 5–7 grams of sticky tack in a ball pressed into that little tape shelf. Next, add a little more tape around the tube and press it over onto the sticky tack. It might take a few layers, but you should get to a point where the tube doesn't leak any air when you blow into it from the fin end.

Before you run off and test-fly the rocket, think about the area you are in – is it too small, or are there any hazards? If you are doing this activity with young children, they tend to get very tempted to try and catch the rockets – you might want to discourage that as, although they are quite light, they do come in pretty fast and could cause a paper cut! Make sure you've read and understood the safety stuff in this article before having a go! We'd recommend that for the first few attempts, you put very little pressure in, just enough to hop the rocket off a couple of metres, and then increase the pressure for more spectacular results. We've found that, even sticking to our 2 bar limit, we can get flights that are really impressive to onlookers (**Figure 5**).

It's a great system, and the rockets are so quick and easy to make that it really does encourage experimentation. One experiment we wanted to try was

"YOU'LL NEED TO ADD SOME NOSE WEIGHT"

Above ↓
Our air cannon rocket launcher and LED-tipped rocket are ready to launch!

Right →
Adding tape to allow you to press in some sticky tack to add weight before covering with more tape

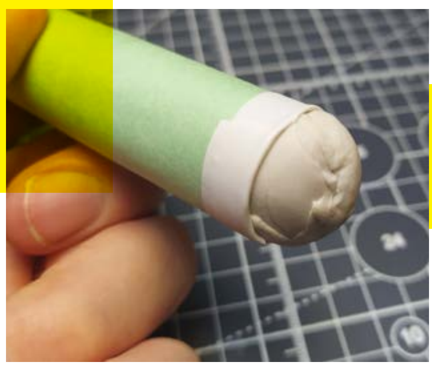


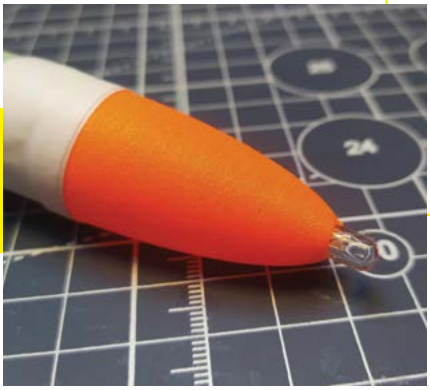
Figure 5
Great fun can be had launching rockets with only a little pressure

incorporating a flashing LED into the rocket for some night launches. Whilst you could totally tape an LED and coin cell onto a rocket, we wanted to try a slightly neater solution!

We wanted to incorporate the LED into a nose cone to make the upper end of a rocket look a little neater, and for it to be nice and aerodynamic. We quickly modelled a nose cone in FreeCAD using the excellent Rocket workbench add-on. We've compiled all the FreeCAD tutorial series into a book, *FreeCAD for Makers*, which explains how to install additional workbenches and more – download a free copy here: **hsmag.cc/freecadbook**. Having modelled a nose cone, we then used the Part workbench to create a cylinder to cut the tip off the nose cone, which would eventually form the 5 mm hole when we 3D-printed the nose cone using Vase mode.

As the nose cone model had a base and the hole was closed, we used Prusa Slicer to set up Vase mode printing and told it to ignore the base and top layers. This means that the nose cone prints as a single wall and is open at both ends. We, again, have cut some corners to assemble the LED system, and it's a pretty quick experimental build. We inserted a fast-blinking RGB LED into the nose cone hole and splodged some hot glue behind it to seal it into position. The long LED component legs stick through into the interior of the nose cone, and you can insert a coin cell between the legs to power the LED. To keep the battery in place, we used a small amount of tape folded into a U-shape and pressed over the component legs and the battery and pressed together gently with a pair of pliers. We then back-filled around the battery with some sticky tack to help bring the nose cone up to weight and also to help seal the nose cone. Finally, we attached the nose cone using PVC tape, pulled and stretched tightly over the joint to try and stop any air leaks. The observant of you will note that there is no way to turn the LED on or off, so the battery and nose cone final assembly takes place just before launching!

The LED works really well as an effect, and also to track the rocket and recover it on landing. Launching over grass tends to help keep the nose cone intact, but if you do catch a stone, you can often bend and squash the assembly back into service. It's great fun trying to capture long-exposure images of the LED rocket. In our experiments, an exposure time of between two and four seconds seemed to work pretty well. It's great to look back at the rocket trajectory – you can make estimates of flight-time based on the images and the exposure time. □



Left
A simple nose cone with a fast-blinking LED glued into the tip for night flying

Below
A two-second exposure captured the night flight up to apogee quite well, showing the LED flashes on the way!

HOW I MADE

By **Sven Kroesen**

ARDUINO COCKTAIL MACHINE

Automated, not stirred!

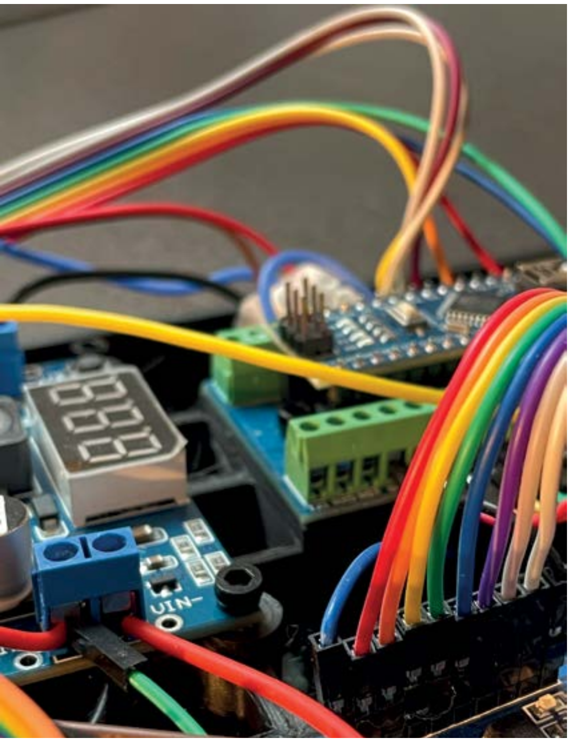
Around a year ago, I bought an Arduino Uno after fiddling around with them at college for a few weeks. I already owned a 3D printer, so this seemed like a great addition to the hobby. After completing a couple of simple projects, it was time to try something more complex. On the weekends, I enjoy making cocktails for myself and my friends. A Long Island iced tea has always been our favourite, but with six ingredients, it's quite a hassle to make.

So I thought, why spend a few minutes making a drink when you can spend a few months automating it? So I did.

THE MACHINE

In essence, the cocktail maker is simply seven peristaltic pumps controlled by an Arduino Nano, on which all the recipes are stored. These peristaltic pumps are super-handy components that allow you to accurately dose liquids without them coming in direct contact with the pumps' mechanism. When you know the amount of liquid that is pumped through per unit time, it's very easy to use delays in your code in order to pour drinks accurately.

Previous projects of mine have always exclusively used 3D-printed parts and off-the-shelf components. Because I wanted to try something new this time, and since it would



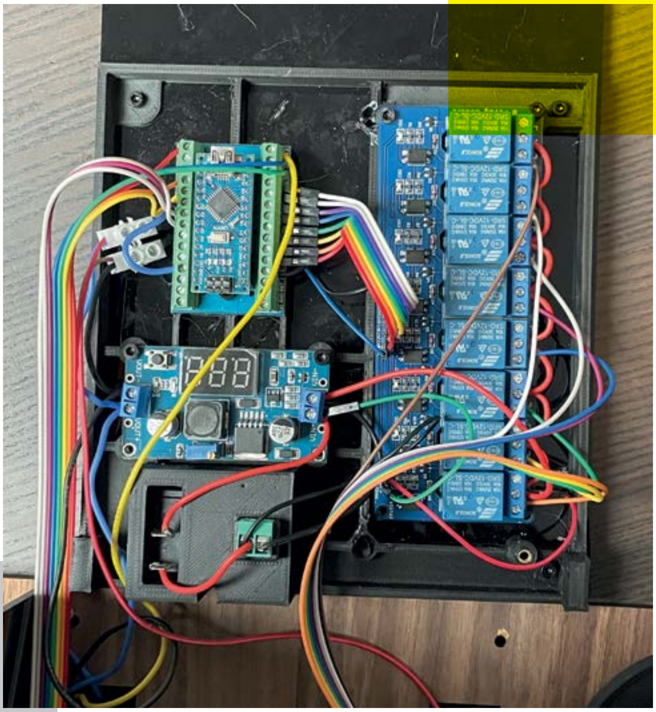
turn out to be so big, I had the biggest parts laser cut. Doing so helped give the machine a much nicer look. In addition, LED rings hidden under the bottles and glass finish the machine's look.

My goal was for anyone that comes over to be able to use the machine without assistance. So, controlling the machine had to be very simple. For this reason, the only way to interact with it is by using the rotary encoder that's placed under the TFT display. When the cocktail maker is turned on, this screen immediately displays a list of available recipes. Using the knob, the user can navigate through the different drinks, change what bottles are stored, and access various machine settings. Telling the Arduino which bottles are stored as well as their locations helps it determine if all necessary ingredients are present to make a cocktail. If that is not the case, the cocktail is greyed out in the menu. It can still be prepared, but the missing ingredient will be omitted.

To make a drink, the user simply pushes the button with their drink of choice selected. A list of ingredients is then displayed on →

Above ↑
You can add whatever drinks you like and let the Arduino know where they are

Above Left ↖
The finished machine ready to serve up a delicious drink



Right →
The Arduino controls relays that switch the pumps on and off

> ***"ALL YOU NEED IS A 2D PROJECTION OF YOUR DESIGN"***

the right-hand side of the screen and the user is asked to confirm their choice. The next step is to input the strength of the cocktail and the amount of ice desired. At 50% strength, a regular drink will be prepared. At 100%, your drink will not contain any mixers. 0% is the same thing but for the alcoholic parts. The ice selection bar works similarly, where at 0%, your drink will not contain any ice. Any higher and the amount of ice will also increase. When both choices are confirmed, the preparation of your cocktail will start. The screen automatically updates to let you know how much liquid is still left to pour. When everything is done, the machine lights up and you can take your finished cocktail!

BUILD PROCESS

During this project, I used two new techniques to put everything together, which turned out to be extremely helpful. The first is the aforementioned laser-cutting of wooden and plastic panels. I had never done this before, but I heard it was possible to order them online, so I decided to give it a try. If you want to follow along with this project, I assume you already have some knowledge of basic CAD design. For laser cutting, all you need is a 2D projection of your design which you can turn into a DXF file with any modelling program.

All I had to do after that was upload them on the website of a local laser cutting place and wait two weeks. It turned out to be surprisingly cheap, too. Numbers are relative, but all together, it was cheaper than the sum of the electronic components. To keep this price down, it helps to design the parts in such a way that they fit snugly next to, or even within, one another. This way you can cut them all out of one big piece. The cocktail maker is cut out of two such panels: one made out of a reflective black plastic, and another made from wood. These parts don't only look nice, but they also provide rigidity to the structure. On their own, they cannot be held together, though. In order to do this, I tried another new technique.

Brass inserts. These little things allow you to put metal taps into your own 3D-printed objects, which is very handy. At first, I tried to simply use tiny nuts and bolts with brackets in the corners, but the further along the build got, the harder it became to reach in certain places, making it necessary to try a different technique. Using the inserts is very easy if you have a soldering iron handy. If you're going to copy this project, you will probably need one anyway. To use them, simply put circular holes in your designs that are just snug enough. Put the insert on the top of a heated soldering iron and slowly press it into place, making sure no plastic gets on the inside. After the plastic solidifies around it, it will be as strong as the plastic it's placed in, which should be plenty for this application if you use the right materials and density for your 3D prints. I like 30% in this case. Doing this for all components makes assembly surprisingly simple, as you only have to screw them in place. Even allowing you to make a frame to put all electronic components in instead of using a PCB.

THE MECHANISMS

Effectively, the machine consists of three parts: the pumps, the ice mechanism, and an electronics rack hidden in the back. We'll talk about the pumps first.

As explained earlier, they are called peristaltic pumps. Instead of turbine-like mechanisms used in many other types, these pumps sort of squeeze the liquid through a flexible PTFE tube. This ensures the fluids never come into contact with anything but those tubes. For this reason, it is perfect for transporting liquids intended for consumption and is, for example, commonly used in medical devices. Starting out with this project, I ordered six of these pumps from China. Since it usually takes a while for the parts to arrive, I try to buy everything I will need in one go. Sometimes, unfortunately, this means a part doesn't work exactly as expected. In those cases, I still try to make the best of it. For this project, the pumps I ordered turned out to be a little slow when pumping. A large glass needed almost a minute to completely fill up using two pumps, which was too slow for my needs. As a solution, I decided to order a different type of pump that is much faster. Since most cocktails have one dominant ingredient, this faster pump can be used to speed up the whole process. Often this ingredient is Cola. The pumps are all mounted in a 3D-printed rack in the top of the machine. A gap is left open so the new seventh bottle can be stored inside the machine. →

Below Left ↙
Liquid is sucked up through these tubes by the pumps

Below ↓
Laser-cut panels made assembly more straightforward

This seventh bottle is used by the faster pump and should be for the primary ingredient in most cocktails. It is a bit of a mess, but this inner container can also store some ice to keep the bottle cool and for putting in your drink. To achieve this, a so-called Archimedes screw is used. This mechanism lets you vertically lift up small objects or, in certain designs, even liquids. We are using it to lift up ice cubes through the gap that leads to the glass. The ice cubes do have to be small enough to prevent jamming the screw.

Working with cooling elements was a little out of reach for this project, so I decided instead to just let the ice slowly melt. All cracks in the interior are covered up with a food-safe resin to prevent leaking, except for a tiny hole under the Archimedes screw. From here, the water leaks through another PTFE tube to the back of the machine. Here, you can either let the water leak away or redirect it to a basin/sink with another tube.

All of this is controlled by a single Arduino Nano. The recipes are stored in a struct data type, with each the name of the drink, references to all ingredients, their volumes, and whether it contains alcohol or not. This last characteristic is necessary to adapt the recipe to the user's alcoholic strength preference. When a drink is selected, the Arduino gets to work. Because we are using peristaltic pumps, we know exactly how much liquid flows every second when they are powered on. By simply dividing the required amount according to the recipe by this number, we know the number of seconds each pump should run for. These pumps, together with the motor for the screw, are controlled through eight relays.

To know which pumps to turn on, we first have to tell the Arduino which ingredients are present and where they are placed. Before we select a recipe, we can input this data by going to a special menu where you first select one of the seven platforms and then one of about 30 pre-installed ingredients. This data will be stored even when the machine is turned off, thanks to EEPROM.

Even when not all ingredients are present, the Arduino will find all corresponding pumps and turn them on for the number of seconds we calculated earlier. All seven tubes are individually led to the console in the front. I

Right →
Here you can see the six peristaltic pumps

"LONG ISLANDS ARE, OF COURSE, VERY TASTY!"

attempted to connect them all together in one single bigger tube but this caused a lot of back-flow and leaking. There are purpose-built connectors to achieve this, but at this point, I was nearing the end of the project and I did not intend on spending any more money on it. It is also not recommended to 3D-print such components. 3D-printed objects are inherently not food-safe because of the tiny cracks that exist on their surfaces. These spaces allow bacteria to hide away and are very hard to properly clean. Or even impossible when it's hidden inside a machine. So, in this case, I preferred my ad hoc solution.

The remainder of the electronics are very simple. The pumps and screw motor all need 12 volts. With a maximum of 2 amps, this is supplied by a simple power supply. Since the microprocessor needs no more than 5 volts, this is stepped down first. This power is also used to power the NeoPixel rings in the platforms and the display on the front. I am still very much learning about electronics, so in my projects, I like to use screw terminals. These allow you to change and adapt as you go while still giving rigid connections.

CHEERS!

Altogether, making this machine has been a great learning process. Laser cutting seems like a viable tool if your design is large enough since you usually have to pay for the whole sheet. In our case, some leftover space was used to cut out a door sign with mine, my girlfriend's and, of course, the cat's name engraved into the surface. After weeks, the cocktail maker still works beautifully. Soft drinks come out a little flat and ice unfortunately does melt, but I had a ton of fun building it, and Long Islands are, of course, very tasty!

Above
This screw mechanism lifts the ice up and into the drink

Above Left
A spinny knob is a great way to control a machine

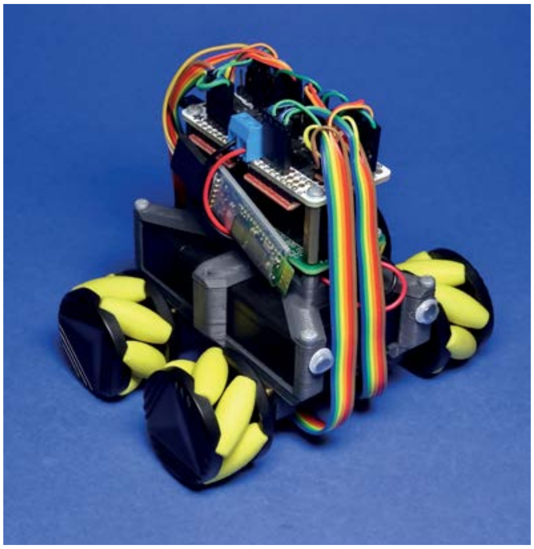
HOW I MADE

By Dr **Christopher Parrott**

L-HEX: A LIDAR-EQUIPPED OMNIDIRECTIONAL ROBOT

There is a special kind of enjoyment that comes from making robots. Creating something that can interact with the real world, using parts you have built and code you have written, feels almost magical. Every robot offers its own challenges in mechanics, electronics, and computing that, when overcome, make the end result all that more rewarding. This is the reason for my ever-growing robot collection!

L-Hex, a lidar-equipped hex (as in six) mecanum-wheeled robot, is the latest addition to my collection. It is triangular in shape, measuring 162 mm in diameter and 120 mm tall. On each side are mirrored pairs of mecanum wheels for movement, driven by DC motors with encoders. Taking up a good portion of the robot's height is a low-cost, 360-degree light detection and ranging



Above
The Mini Mecanum robot showing its stack of components

Above
L-Hex in all its RGB, Mecanum-wheeled, lidar-using glory

(lidar) scanner, which uses a spinning infrared laser to measure distances to objects. At L-Hex's heart are two Raspberry Pi RP2040 microcontrollers: one operating the lidar and the other driving the six motors.

The combination of lidar scanner and mecanum wheels is interesting as they let L-Hex sense its environment with great detail and use that information to move in any direction to follow or avoid obstacles without the need to turn to change direction.

HOW IT STARTED

At the tail end of 2019, I was exploring the use of mecanum wheels on small robots. For this, I created a car-like robot dubbed Mini Mecanum. It used four 50:1 N20 motors with magnetic encoders for feedback, two TB6612 dual motor drivers on a protoboard, a Teensy microcontroller, a Bluetooth serial receiver, and a 3 × AA battery pack for power. All of this was mounted on a bare-bones 3D-printed chassis designed in Fusion 360. 3D printing has been my go-to manufacturing method ever since I was introduced to it at university a decade ago.

To drive Mini Mecanum, I programmed the Teensy to take serial commands received via Bluetooth and translate them to motor speeds, which, along with measured speeds from the encoders, were fed into proportional-integral-derivative (PID) control loops to produce accurate motor outputs.

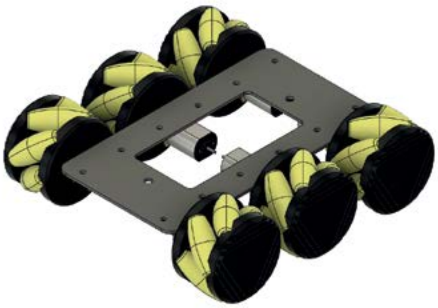
For the robot's mecanum wheels, I attempted to find ready-made ones online. It took some time, but I eventually found a Taobao listing for N20-compatible mecanum wheels – offered in a range of colours, too! Being unfamiliar with ordering from Taobao though, I took advantage of my Sheffield connection and reached out to Pimoroni to see if it could source them, which it could! This was a whole year before getting employed by the company →

Above The three colours of wheels I acquired

Below Should I go with three wheels on a side, or six?

– back when I had only met its crew a few times at Raspberry Pi events.

In early 2020, the mecanum wheels were listed on Pimoroni's website in yellow, green, and black. Surprised by having colour choices, I got a set of each and fitted the yellow ones onto Mini Mecanum. They worked better than I had hoped! Not only could the robot move in all directions with ease, but I was able to enhance my code to have it drive autonomously in a square, and even rotate as it translated along each edge!



SIX WHEELS

Having a working mecanum robot, I started thinking up ideas for other robots to create. I was particularly interested in using more than four wheels as I had seen videos of industrial robots with twelve or more used to move trains and aircraft around factories. Also, my code enhancements had theoretically removed any limits on the number of mecanum wheels that could be used together, so I wanted to test this with hardware.

Seeing as six wheels was the natural step up from four, I sketched up two ideas in Fusion 360. The first idea was a truck-like arrangement with three wheels on each side. The second idea was a triangle arrangement with two wheels on each of the three sides. I shared both concepts on Twitter, and the triangular shape seemed to gather more interest. This shape also appealed to me as I could distinguish each side with its own wheel colour!

Above ↑
The first assembly of Hex Mecanum

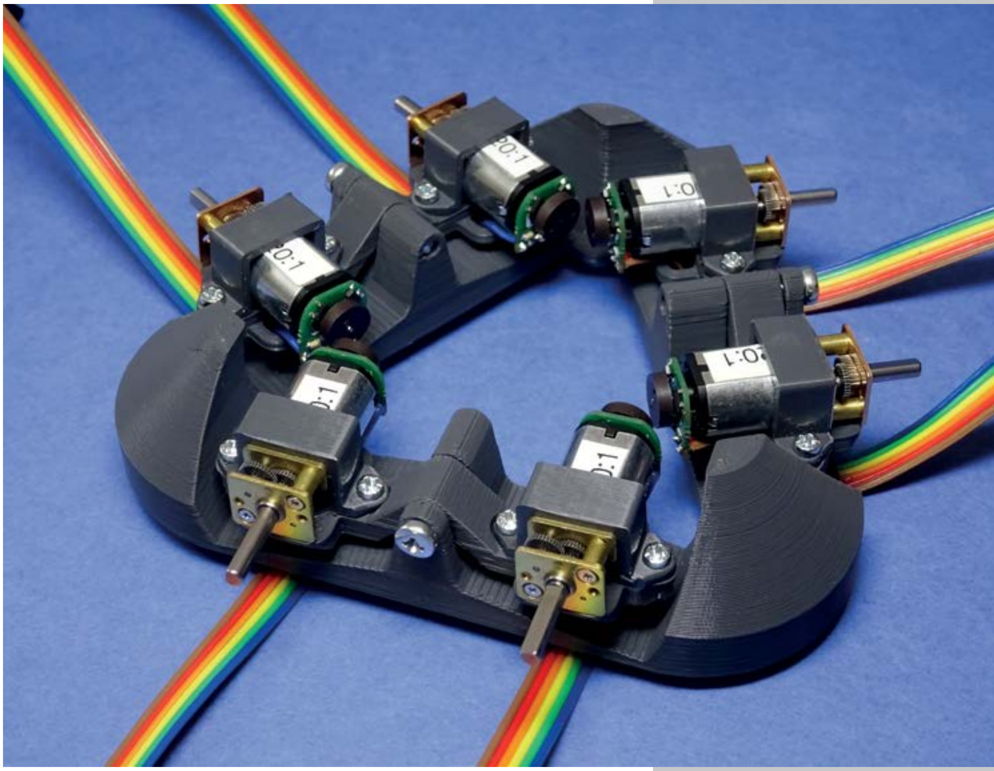
WHAT ARE MECANUM WHEELS?

Driving in any direction on a surface, known as omnidirectional motion, is a beneficial trait for robots needing to navigate complex environments or accurately position themselves to manipulate objects. One way to achieve this is to use a family of wheels known as omni wheels.

Omni wheels are specially constructed wheels with many free-spinning rollers around their perimeters. These rollers let the wheels slide in their rolling direction, creating the effect of having practically zero surface friction in that direction. By arranging multiple wheels so that the driving directions of each overlap the rolling directions of the others, omnidirectional movement can be achieved.

A typical omni wheel has its rollers angled at 90 degrees to the rotation axis of the wheel, making them slide sideways. This gives rise to two common chassis arrangements, triangle and square, both with one wheel on each side. Moving such a chassis is a matter of driving each wheel with a proportion of the motion in that direction.

Mecanum wheels differ from typical omni wheels in that they have their rollers at 45 degrees, giving them the tendency to slide diagonally. Placing four mecanum wheels, as mirrored pairs, on a car-like chassis makes their driving directions and rolling directions naturally overlap. As a result, movements of this chassis behave the same as they do for a regular fixed axle car, with the bonus of being able to strafe sideways by having each side's wheels drive towards or away from each other.



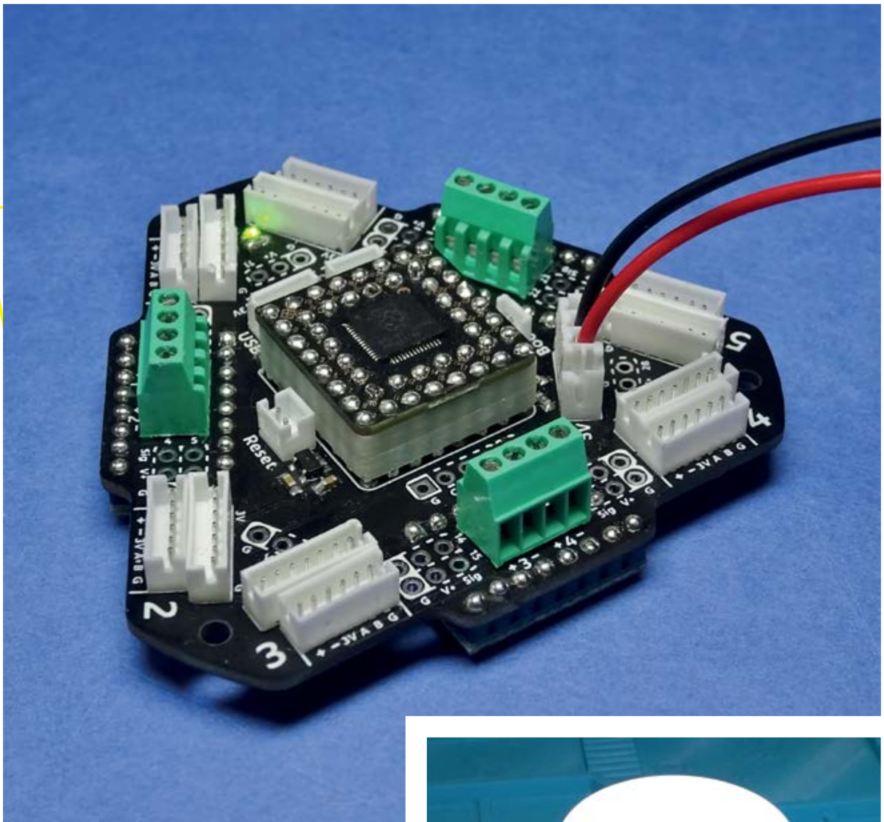
Above ↑
The underside of the frame prior to wheels, showing the suspension pivot

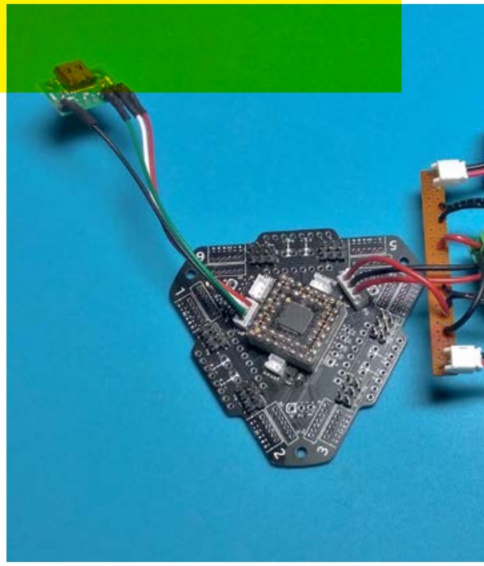
I began the new robot, dubbed Hex Mecanum, by making a frame to mount the motors to. Since 50:1 N20s worked well on Mini Mecanum, I opted to use them on this robot too. One valuable piece of feedback I received when sharing the robot idea was to include suspension on the wheels to ensure they all stayed in contact with the ground to

> **"I WAS INTERESTED IN USING MORE THAN FOUR WHEELS"**

work properly. For this, I designed a simple see-saw mechanism for each triangle side that had a motor on each end and a pivot in the middle.

For driving the motors, I initially soldered up a protoboard with three TB6612 dual-motor drivers. Unfortunately, its rectangular shape did not fit well on the triangular frame. Wire routing was also →





Above
The Hex Driver board assembled and externally powered

Right
The moment that gave the inspiration to add a lidar to the robot

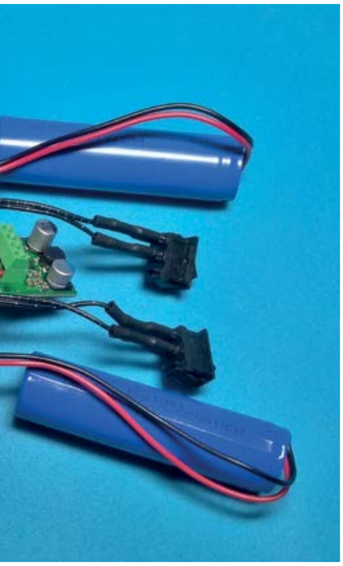
an issue, with cables from motors crossing over others to connect to the correct spots. Not seeing an immediate way to overcome these issues, I put the robot to one side for what ended up being a whole year!

THE ELECTRONICS

I picked Hex Mecanum back up in 2021. I had recently become a product engineer at Pimoroni, helping design and launch products for the newly released Raspberry Pi Pico and RP2040 microcontroller. As part of the role, I was learning PCB design with EAGLE. This reminded me of Hex Mecanum's problems and how a custom triangular PCB with conveniently positioned outputs could solve them.

Around this time Pimoroni was developing PGA2040, a breakout board for the RP2040 that included the essential components to make the chip run, and exposed all 30 GPIO pins. This was an ideal fit for the project as it was small and gave enough GPIO to cover the 24 needed for the six motors and encoders, with some spare for extras. Plus, I was already becoming quite familiar with programming the RP2040.

I started work on the PCB, called Hex Driver 2040, creating the schematic with all



Above ↑
The power electronics, showing the batteries and regulator

the components needed. To keep the size down, I swapped the motor drivers out for DRV8833 breakouts from Pololu. Then came the board design stage, where I quickly found that using a square alignment grid on a triangular board left me with no space to route all the traces I needed, even though I could see that there was space. My solution: ‘route’ the board in Fusion 360! I imported all the component positions and then drew lines for each trace manually, applying angle and distance constraints to them to get the layout I wanted. I could even apply symmetry operations to duplicate similar sections.

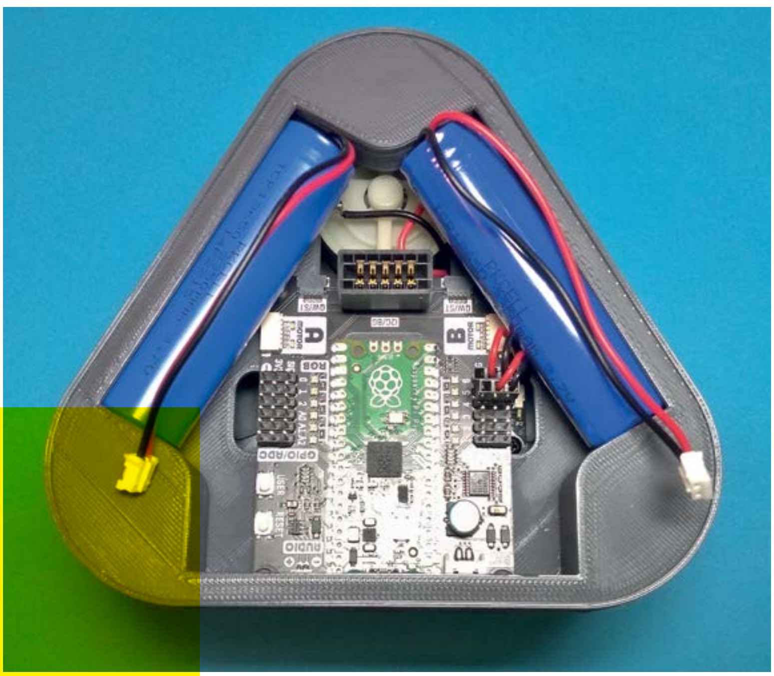
Once finished, I hoped I could just export the lines from Fusion and import them into EAGLE as traces, but I could not find a way to do it. My workaround was the laborious process of manually copying each line’s coordinates from Fusion and pasting them into EAGLE’s trace properties. This was time-consuming to say the least, but the end result was worth it. To avoid having to repeat this in the near future, I included alternate connector landings to support other projects that could benefit from a triangular board. This meant the final hex driver was capable of driving 6 × DC motors with encoders, or 3 × stepper motors, or 12 × servos, with up to 7 × I2C sensors or a single SPI sensor.

I ordered the completed hex drivers from JLCPCB, which arrived just in time for PGA2040’s launch. I soldered one up straight away and did some simple motor movements to test it worked as intended, which it did! I then started converting

“I PUSHED MY ROBOTICS AGENDA ONTO PIMORONI”

my Mini Mecanum software over to it but hit a snag when it came to motors and encoders with RP2040. Realising I would need to write my own support to get the functionality I wanted, I pushed my robotics agenda onto Pimoroni so that I could implement it on work time. This worked, resulting in the Pimoroni flavour of MicroPython receiving motor and encoder support, and a bunch of robot products getting launched too! This wasn’t a quick process though, delaying Hex Mecanum for another year.

I flashed the hex driver with pirate-brand MicroPython and used my own motor feedback example to control the motors. Then, once I finished porting Mini Mecanum’s code and assigned each wheel to its correct location, I connected the chassis up to a bench →



Left ←
The Inventor 2040 used to drive the lidar. This is actually a pre-production, non-Pico W version, but final robot has a Pico W version

Above ↑
The lower half of the robot assembled, with power system installed

Left ←
Some of the final prints that were done

power supply, and the motions 'just worked'. The chassis was able to translate in various directions, turn on the spot, and move and turn simultaneously, all as I had hoped.

Right ↗
L-Hex render from Fusion 360

THE ROAD TO CAMJAM

Not long after that success, the Cambridge Raspberry Jam was announced. Being the first event in a long while and the 'première' Jam, I immediately registered for the Show & Tell with the plan to display some robots, including Hex Mecanum. The robot was far from complete though, needing a power source, a chassis that enclosed its internals, and, ideally, a feature that would benefit from its omnidirectional motion.

For power, I went with two 18650-style rechargeable battery packs connected in series. These fit neatly along the sides of the chassis, and gave 7.4 V nominal, which was plenty to power a 5 V step-down regulator for the hex driver, and could also be used to power the motors directly.

Around this time, Twitter user @NotBlackMagic had found and documented their efforts with a cheap lidar scanner from China. Pimoroni got some in for me to test, and following @NotBlackMagic's information, I quickly got one running from a spare Inventor 2040 W. It needed a motor driver output, and a logic level converter to shift between 3.3 V and 5 V UART.

While working on the lidar one evening, I casually rested it on top of Hex Mecanum,

and it fit almost perfectly. This was the feature that turned the project from Hex Mecanum into L-Hex!

To complete L-Hex's chassis, I updated the Fusion model with the latest components, tweaking their positions until everything fit well. I then drew a solid shape that enclosed them, which I sliced into sections to work on in stages. These were slowly chipped away at, adding and subtracting material, similar to carving a piece of clay. To check the tolerances of various pieces, I would often slice and 3D-print just the relevant geometry, saving valuable time on what would have otherwise been multi-hour prints. I printed all of L-Hex's parts, going for a white and black colour scheme to match the lidar, with coloured highlights on each side to match the wheels.

With CamJam fast-approaching and software still needing to be written, I opted to simplify matters by treating L-Hex as two independent systems, without any communication between them. The drive system was programmed to perform random on-the-spot rotations, as this seemed safe for displaying on a table.

The lidar system was programmed to show audiences what the robot 'sees' on an RGB LED ring added at the last minute. With these systems working, I performed the final assembly and packed L-Hex up for transport to Cambridge.

CamJam was a great experience, with L-Hex and the rest of my table receiving a good amount of attention. The RGB LED ring proved to be a great addition to the robot, with lots of attendees having fun moving their hands around the lidar and seeing how it reacted. I even had a brief visit from Eben Upton, who said he had been following the project!

After CamJam, I had plans to complete L-Hex's software, but the push to get it ready burnt me out somewhat, so I am giving it a break. If this four-year-long story has highlighted anything though, it's that extended breaks are not unexpected, and often lead to new opportunities that would not exist had the break not happened. For now L-Hex is having a well-deserved rest on a display shelf alongside the rest of my robot collection!

Below
It took some work to get there, but L-Hex was ready in time for the Cambridge Raspberry Jam

"I WOULD OFTEN SLICE AND 3D PRINT JUST THE RELEVANT GEOMETRY"

20 Amazing RASPBERRY PI PROJECTS

Armed with a Raspberry Pi and a few electronic components, a world of adventure awaits the intrepid maker

Phil King

A long-time Raspberry Pi user and tinkerer, Phil is a freelance writer and editor with a focus on technology

While a Raspberry Pi is a capable computer, and can even be used as a desktop PC replacement, its GPIO (general-purpose input/output) header is what makes it different and truly powerful. This enables you to hook up electronic circuits and components such as LEDs, sensors, motors, and servos.

To help inspire you to get creative when connecting electronics to your Raspberry Pi, we've compiled a list of 20 of the most impressive and ingenious projects around. Dazzling light-up devices include an interactive model Stargate, LED cube, London Underground map, and spinning POV (persistence of vision) display.

Moving into robotics with motors and servos, you could make a piano-playing, metal-detecting, chess-playing, or air hockey-playing robot. Or maybe a LEGO submarine or even a ceiling planetarium. If you fancy something musical, how about a synth guitar or glockenspiel player? Or get arty with a CNC drawing machine or even a social media jacket. The only limit with a Raspberry Pi project is your own imagination.

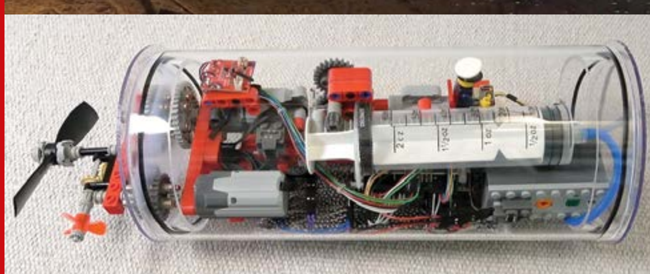
LEGO Submarine 4.0

The Finnish maker behind the Brick Experiment Channel on YouTube loves using LEGO to create working models that address complex physical and mathematical challenges. One of his most impressive creations yet is this working radio-controlled LEGO submarine (his fourth attempt).

With the processing power of a Raspberry Pi Zero 2 W on board, it uses a servo motor and LEGO rack and pinion system to operate a syringe as a piston ballast; this draws in or expels water to raise and lower the sub's buoyancy level to make it rise or fall in the water. Tungsten pellets are used for ballast. An absolute pressure sensor obtains accurate depth readings, while a micro laser sensor measures the distance to the 'sea bed'. Positioning is aided by a PID (proportional integral derivative) controller loop in the Python software.

Naturally, protecting the electronics from water damage was a key consideration in the build. To this end, a fair amount of time and effort went into creating the submarine's transparent acrylic case with tight-fitting end caps with rubber seals.

hsmag.cc/LegoSub



GIANT Battleships

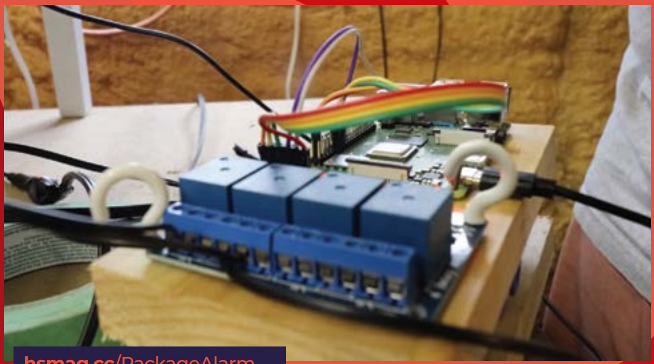
Maker Dan Aldred has taken the concept of the classic Battleships game and blown it right up. After previously recreating the board game on Raspberry Pi with a Sense HAT, he opted to go bigger by using a giant 10×10 LED matrix that he'd already created for light shows and discos. Cleverly, the NeoPixel LEDs in the grid have their light diffused by glass jars spray-painted white.

An innovative method was also employed for the user to choose the coordinates to target torpedoes at the hidden ships in this solo version of the game: a 1960s Bakelite telephone fitted with a Raspberry Pi. As you turn its dial to select digits, a circuit with a GPIO pin is interrupted by the relevant number of clicks. Each coordinate is sent in turn (using Python sockets) to the LED board's Raspberry Pi.

The project even makes use of the telephone's handset speaker to give voice instructions to the user and play sound effects such as explosions. →

hsmag.cc/Battleships

PACKAGE *Thief* DETERRENT



hsmag.cc/PackageAlarm

Like many people, Canadian maker Ryder (of the Ryder Calm Down YouTube channel) has experienced the problem of packages delivered to his home being stolen by porch pirates while he was out. Armed with a Raspberry Pi 4, a security camera, and a few deterrent devices, he set about solving the problem.

First of all, images from the camera connected to his Raspberry Pi are processed by a custom machine learning model (trained using Google Cloud AutoML) to detect if there is or isn't a package. If one has been taken unexpectedly, the Raspberry Pi sends signals to a relay switch board to activate a variety of alarms to entice the thief to drop the package. Those surprises for thieves include a sprinkler, a loud truck horn, and a flour shower.

Naturally, since he didn't want to trigger the alarm himself, he has trained the system to deactivate when it recognises him in the frame. One slight issue is that the AI sometimes thinks his cats are packages!

ROBOT ARM *Clock*

Even a stopped clock is right twice a day, but maker Hendrik Ohrens wasn't content with that when his timepiece broke. Instead, he opted to build a robotic arm to physically move the clock hands to the correct time every minute!

Having explored inverse kinematics and computer vision, Hendrik came up with a far simpler solution. Using Python code running on a Raspberry Pi 3B+, he trained the robot arm manually for each tiny movement of the minute hand required for a complete rotation.

Once trained, the Raspberry Pi relays the precise positional instructions to a connected Arduino MKR board equipped with a shield to control the arm's four servos.

Instructions, code, and files for the arm's 3D-printed parts can be found in his GitHub repo: **hsmag.cc/RobotArmGit**.

hsmag.cc/RobotArmClock

LONDON *Underground* MAP DASHBOARD

hsmag.cc/TubeDashboard

While many makers create Raspberry Pi projects for use in the home or just for fun, Richard Kirby built this one to help with his work. As a test manager for a company working on automating parts of the London Underground (aka Tube), he needed a way to monitor the real-time status of the train lines to instantly alert him to any problems.

Made from 5 mm plywood and printed network map, this large dashboard is ideal, featuring 284 individually drilled holes to house multicolour NeoPixels. A Raspberry Pi Zero 2 W gathers data from the Transport for London (TfL) Open Data site to get the statuses of the various lines, which it then converts into animations: 'Good Service' is represented by fully lit LEDs for that line; 'Minor Delays' is rapid flashing from 50% to 100% brightness; while 'Severe Delays' has slower flashing.

The map is also connected to a Bluetooth speaker for service announcements, with text-to-speech used to turn the data into spoken words.

DeMOOR *Orrery*

After seeing a 240-year-old orrery – a mechanical model of the solar system – on the ceiling at the Eise Eisinga Planetarium, Chris DeMoor was inspired to build one of his own. Rather than copying the pendulum clock-driven system of the original, however, he opted to use six Raspberry Pi Zero boards.

Each controlled by a Raspberry Pi Zero, the six inner planets of the solar system move in orbits around his ceiling in real time. Two different methods were employed for their movement. Mars, Jupiter, and Saturn are attached to front-wheel drive, 3D-printed cars which run on tracks on the upper, non-visible side of the project. As their orbits would be too small for tracks, Mercury, Venus, and Earth are mounted to dishes connected to rotating stepper motors.

To obtain its correct position, each planet's Raspberry Pi connects wirelessly to a web server which plots their orbits using the jsOrrery JavaScript library. The orrery planets are then moved accordingly. →

demoor-orrery.com

PIANO-PLAYING Robot

hsmag.cc/PianoRobot

Regretful that the piano in his living room was played all too rarely, Étienne Allaire came up with the idea of a robot that could play music. To do so, he mounted a wooden frame with 15 solenoid switches to an electric keyboard. Controlled by a GPIO pin on a Raspberry Pi 3B, connected via a 16-channel relay module, each solenoid pushes down on a key to play a note.

The system can read a MIDI file, convert it into on and off signals for each note, and play the tune, although there is a limitation as to how fast it can play due to the speed of the solenoid actions. A basic UI allows the user to choose between scales, arpeggios, or melodies.

The most impressive feature, however, is the robot's ability to read previously unseen sheet music – using a Raspberry Pi Camera Module – and play it. The open-source tool Audiveris uses optical character recognition (OCR) to read the notes on the sheet music.

STARGATE Model

As a fan of the *Stargate-SG1* series, Kristian Tysse was inspired to create his very own Raspberry Pi-based scale model of a Stargate, complete with animated spinning wormhole. To achieve the latter effect, he created an infinity mirror with the light from 122 LEDs reflected back and forth. The model itself was painstakingly created in great detail, built from 3D-printed parts.

Most importantly, it includes a DHD (Dial Home Device) to dial any Milky Way address featured in the show. Acting as a USB keyboard, its key presses are sent to a Python program running on Raspberry Pi, causing the Stargate's symbol ring to rotate accordingly and each of the seven chevrons to light up and move inward – done using stepper motors.

If a correct set of symbols is dialled, the wormhole is established and audio clips from the TV show are also played. The Stargate can even dial other Stargates over the internet, and receive incoming wormholes from them. Find instructions for making one on the website.

thestargateproject.com

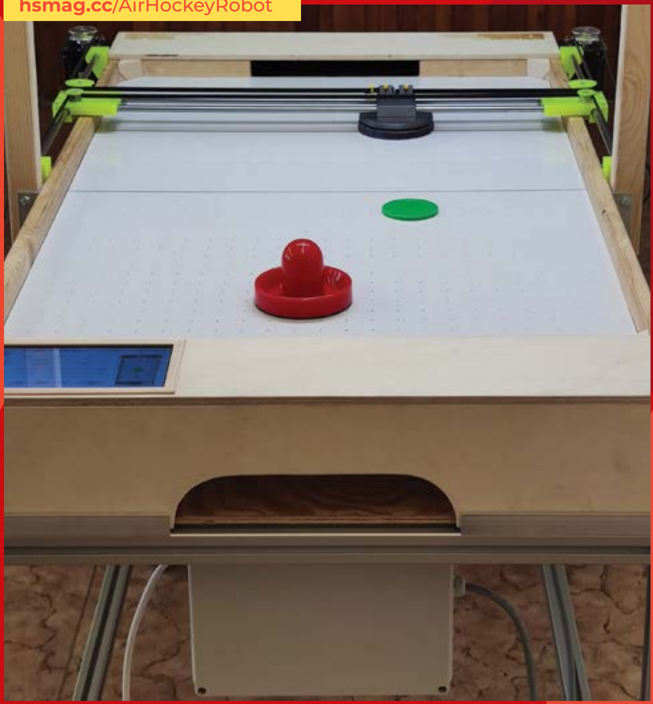
AIR *Hockey* ROBOT

When students Ondřej Sláma and Dominik Jašek needed to write their course thesis, they chose the Air Hockey Robot they had built from scratch. Taking about a year, the project involved creating robot movement control algorithms, computer vision, game strategy algorithms, and a user interface.

The game table itself was designed in Fusion 360 and built from plywood, with a mesh of 920 holes creating an air cushion. The processing power for the robot's optical puck recognition and AI strategy is provided by a Raspberry Pi 4 connected to a Camera Module mounted in the overhead part of the frame.

For the mechanical aspect, the pair opted for an 'H-bot' design to move the robot's paddle. Held in a 3D-printed housing, the paddle is moved from side to side using a pulley and belt system with two stepper motors. You can find the code used for various parts of project in their GitHub repos (linked from the YouTube video).

hsmag.cc/AirHockeyRobot



CNC *Drawing* MACHINE

Made by Yannik Dolde, this seriously impressive Raspberry Pi project repurposes old CD drives as motors for the X and Y axes of a CNC machine that can draw a picture. The pen is mounted in a 3D-printed holder that moves up and down using a servo with a spring-loaded metal rod to regulate the pressure on the paper. A Raspberry Pi 3B+ with custom Python software runs a user interface and reads a standard G-code file, line by line, to send commands to an Arduino Mega to move the motors.

hsmag.cc/CNCDrawing

LED *Cube*

An LED cube is a popular maker project, using LED matrix panels for its sides. Inspired by other cube creators, Sebastian Staacks originally envisaged crafting a stationary mood light for his living room. However, his wife's disapproval led to him turning it into something more useful: a parametric animation reflecting the status of his PC's Ryzen 5 CPU.

As it's only intended to be viewed from one angle, his 'cube' comprises three 64×64 RGB LED matrix panels. These are held by a 3D-printed frame which also contains a Raspberry Pi 2 with an Adafruit LED Matrix Bonnet. It cost Sebastian less than €150 to make.

Making use of an OpenGL shader, the cube displays impressive glowing effects in real time based on the CPU's temperature and its core loads. →

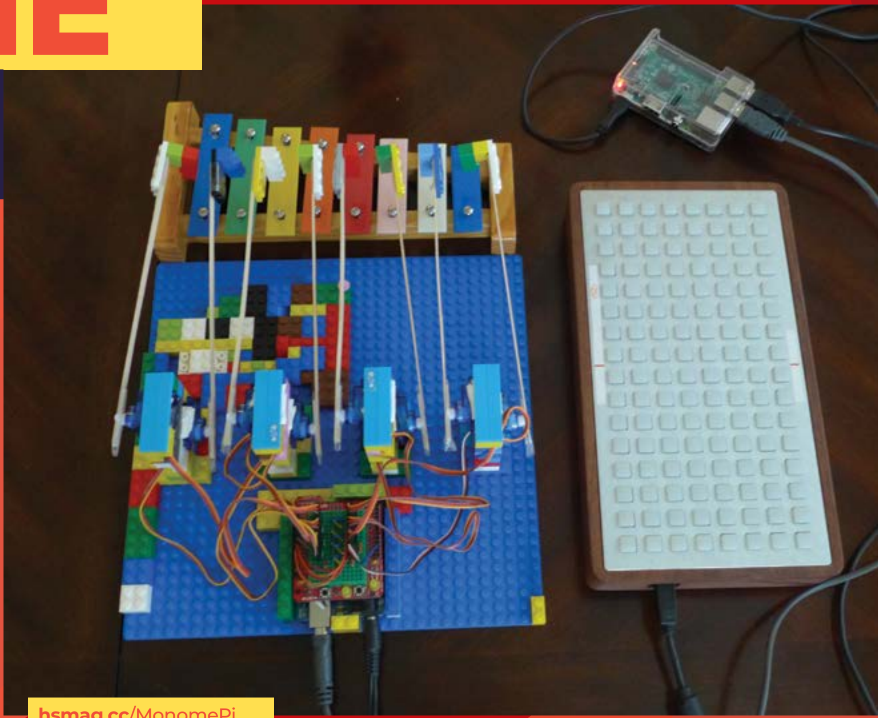
hsmag.cc/LEDCube

MONOME Pi

Joon Guillen's unusual music box features old and new technology working in perfect harmony. For input, a Monome Grid controller is connected to a Raspberry Pi 3 running a step sequencer program which registers the user's button presses on the Monome, lights them up, and sends serial commands to an Arduino Uno.

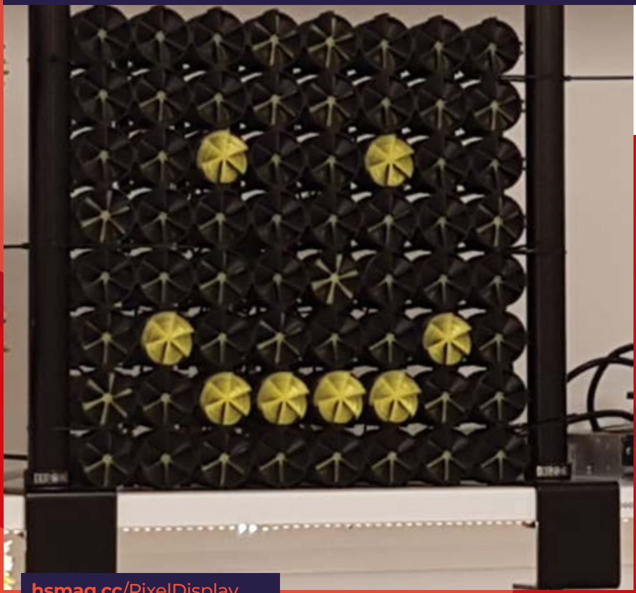
The Arduino is connected via a ProtoShield Kit to eight servo motors, which move hammers to play the correct glockenspiel notes to match the pattern shown on the Monome. The makeshift hammers are made from coffee sticks, sticky tape, and LEGO blocks borrowed from Joon's daughter – which shows that you don't need to be a master craftsperson to build an interesting project.

Joon says that you could substitute the Arduino with a Raspberry Pi servo/motor driver board, and the Monome with a touchscreen or web UI.



hsmag.cc/MonomePi

PIXEL Electromechanical DISPLAY



hsmag.cc/PixelDisplay

When considering how to make a decorative display for an office party, Gavan Fantom thought a standard LED matrix would be too obvious and simple. Instead, he designed and built Pixel, an ingenious electromechanical matrix display controlled by a Raspberry Pi and comprising no fewer than 448 3D-printed parts. Each of the display's 64 'pixels' is turned on and off by a servo rotating a 3D propeller-like shape to reveal its bright yellow blades from a black casing.

While it was inspired by traditional flip-disc electromechanical displays, its unique pixels can also be rotated to intermediate positions to achieve greyscale-style shading and ripple effects. Each pixel comprises seven 3D-printed parts, one servo motor, and two nails to transmit the latter's rotation to reveal the vanes from the casing. To drive 64 servos from a single Raspberry Pi, three Pololu Mini Maestro 24 boards are used.

The end result is an amazing display that just has to be seen in action – check the YouTube videos in the linked blog piece.

PUITAR

Finding his brother's old electric guitar difficult to play, Behruz Farshi opted to transform it into a digital instrument using a Raspberry Pi Zero. His Puitar (pronounced 'pit-tar') features a keypad matrix of 22 frets and six strings. When the player presses a string onto a fret, it forms an electrical connection that is detected by one of Raspberry Pi's GPIO pins and the appropriate note is played – which could be a piano or any MIDI sound you like. To expand the number of pins, an IO Pi Plus board is used.

To create the electrical matrix, Behruz drilled holes into the fretboard and soldered wires to the frets from below. While his prototype Puitar can't handle polyphony, and accidentally touching strings together can cause it to malfunction, it's a clever concept that could be adapted and possibly modified into a full-on synth axe.

hsmag.cc/Puitar

DIGITAL *Zoetrope*

One of the oldest animation methods, a zoetrope is a spinning drum with a set of static images that, when seen in rapid succession from a slit in the side, appear to be moving.

Fascinated by the concept since he was a child, maker Brian Corteil decided to make his own zoetrope using 15 e-ink displays – Pimoroni Badger 2040s. Brian designed the zoetrope drum using the SolidWorks CAD program, creating outlines to be laser-cut from 3 mm and 5 mm plywood, while some other parts were 3D-printed.

A Raspberry Pi 4 is used to send screen updates to the e-ink displays via USB connections, while a Pico in the base controls the drum rotation via a motor controller and monitors the emergency stop buttons. All the electronics spin around with the drum – to avoid the main central power cable twisting, there's a split ring in the base.

A second Raspberry Pi 4 connected to a flatbed scanner also enables animations created on a cell sheet to be scanned and uploaded to the zoetrope.

hsmag.cc/DigitalZoetrope

DISCOVERER *Metal-Detecting* ROBOT

Building a robot vehicle is one of the most popular Raspberry Pi projects, but this one is rather special and could be useful. Maker Ingmar Stapel came up with the idea after watching a TV show about people trying to find gold with a sophisticated metal detector.

For the chassis, he used a plastic storage box to contain all the electronics, adding PVC piping around the exterior to hold a dual-servo pan-and-tilt camera – which enables the streaming of live video to a web dashboard – along with a Gary's Pulse-AV metal detector. A step-down converter is used to change the detector's 12 V output to 3.3 V for a GPIO pin on the Raspberry Pi.

Discoverer's four motors are driven via L298N H-bridges. As well being remote-controlled with a range of 350 metres, the robot can be programmed to move autonomously from one waypoint to another thanks to the addition of a GPS receiver and a Sense HAT for the compass. →

hsmag.cc/Discoverer

SOCIAL MEDIA *without the* INTERNET

Imagine what a physical, real-world version of interacting on social media might look like. That's just what interactive artist Tuang Thongborisute (Tuang T) wanted to explore, leading her to create this very special jacket as a performance art project.

The blazer features numerous electronic elements – controlled by a Raspberry Pi, aided by an Arduino – which enable people the wearer meets to engage in six social interactions. For instance, a follow is made by tapping a pressure-conductive resistant sheet on the right shoulder, while unfriending involves a push of a button located on the left side, near the heart. Adding a new friend is achieved by shaking hands, connecting two conductive rings on the wearer's fingers.

The jacket also has a 7-inch touchscreen for further interaction and to show status information. In addition, three tiny cameras broadcast the real-time interactions on a local network for performing in a closed environment like an indoor gallery.

hsmag.cc/SocialMediaJacket

RASPBERRY *Turk*

raspberryturk.com

This amazing chess-playing robot was inspired by the 18th-century 'Mechanical Turk'. While the latter had a human player concealed inside to play its moves, the Raspberry Turk has a Raspberry Pi 3 for its brains.

Created by Joey Meyer, this open-source project is built into a small 3×3 ft (91×91 cm) table. A box on one side houses all the electronics, while the robotic arm is mounted on its top. The arm is built with Actobotics components, Dynamixel servos, and some 3D-printed parts.

The arm's motion is controlled by the rotation of two servos attached to gears at the base of each link. Another servo controls the gripper mechanism which uses an electromagnet to pick up pieces. Naturally, great precision is required to move pieces on the board, so Joey built up a dataset of the arm's movements to aid his mathematical model.

To evaluate the positions of the pieces, a high-mounted Raspberry Pi Camera Module captures a view of the board which is then perspective-transformed using OpenCV. The Stockfish chess engine is used for its AI.

Pico Projects

AUTOMATIC Dog Ball LAUNCHER

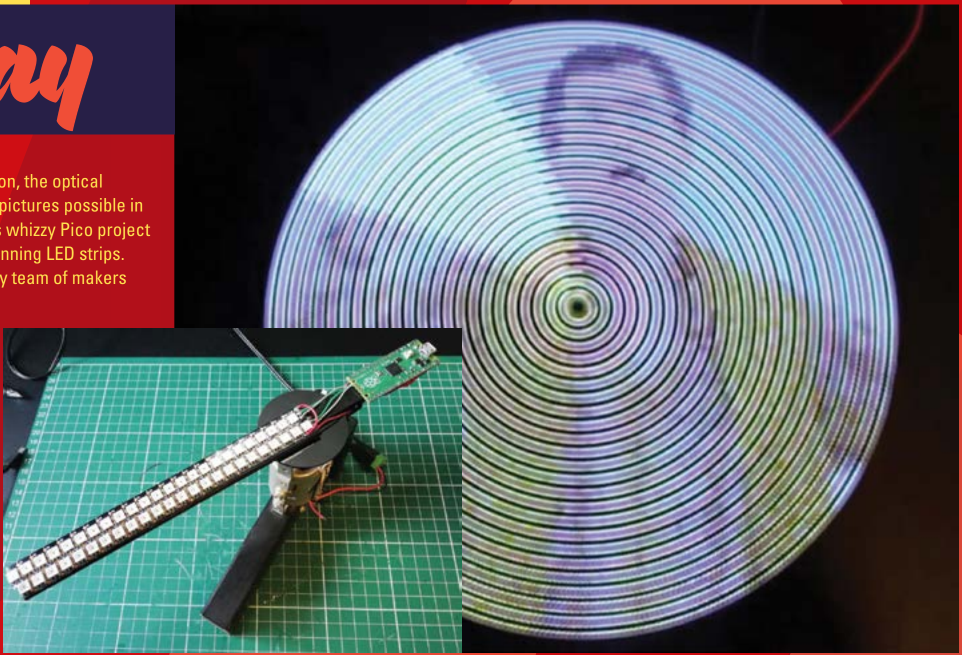
Rather than buy a commercial automatic dog ball launcher, maker Brankly opted to build one of his own, based around a Raspberry Pi Pico microcontroller. After playing around with different shapes, he found that a spherical design looked the best; it was also small enough for his 3D printer to print.

When a ball is placed into the launcher's funnel, it is prevented from falling into the launch channel by a piece of plastic that's controlled using an SG90 servo. The ball is then detected by a sensor, prompting Raspberry Pi Pico to get ready for launch.

To vary the distance of the ball launch each time, to make it more fun for the dog, a motor controller is used to randomise the speed of the motors using PWM (pulse-width modulation). Once a ball has been launched, the motors turn off and the servo blocks the entrance again so that the machine is ready for the next ball.

hsmag.cc/DogBallLauncher

POV Display

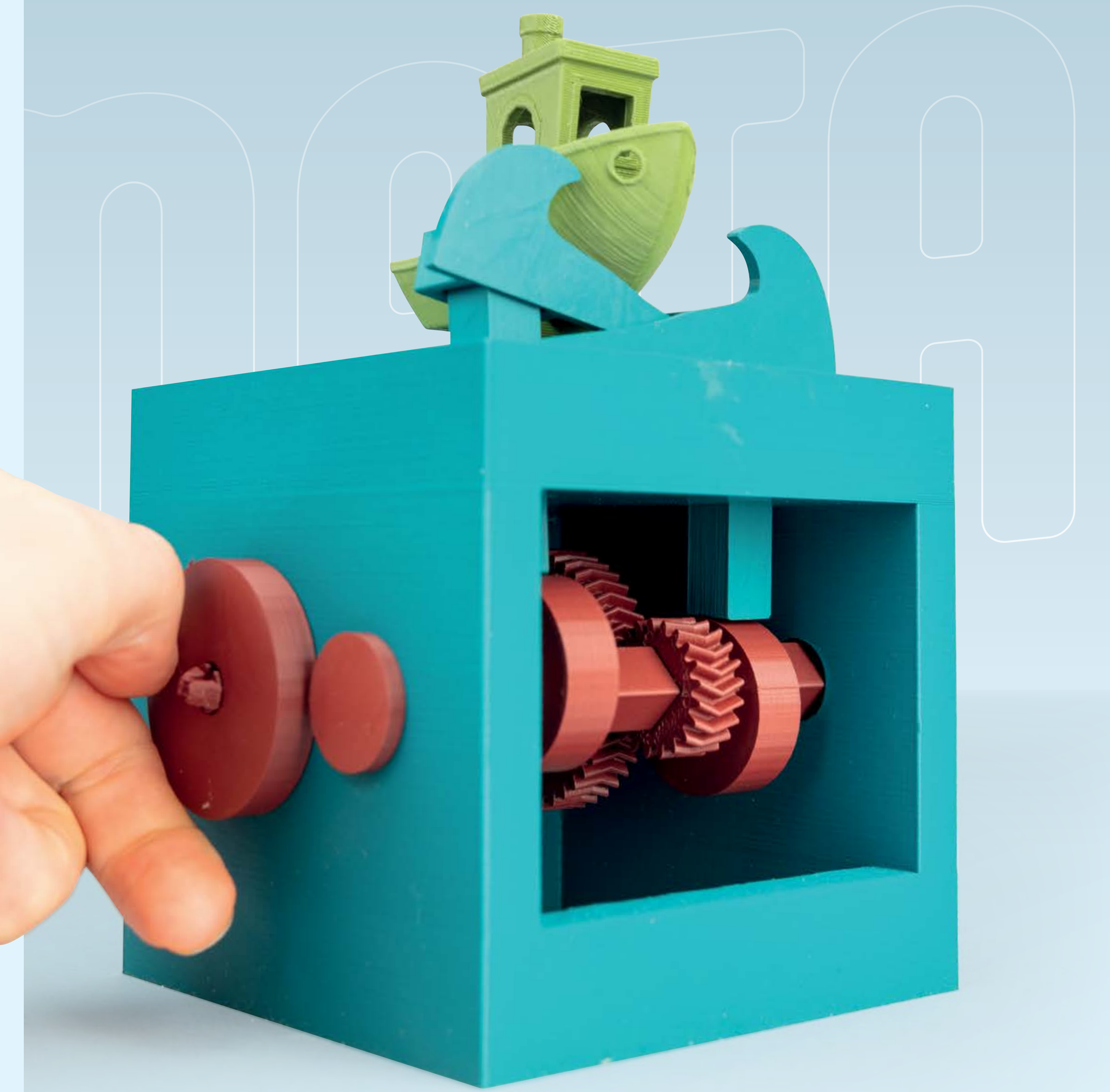


'POV' stands for persistence of vision, the optical phenomenon that makes moving pictures possible in cinema and TV, and on which this whizzy Pico project relies to depict an image from spinning LED strips.

Created by Japan-based family team of makers HomeMadeGarbage, it makes use of two of the Pico's PIO (Programmable IO) state machines to control, in parallel, a pair of super-bright 24-LED DotStar strips on its rotating arm. The arm, which also holds the Pico board, is spun at high speed by a motor while the LED strips are blinked in carefully controlled sync with the rotation speed to display a static or animated image.

A wireless charging module is employed to power the spinning arm, using a coil on the underside of the latter and another on top of the motor. □

hsmag.cc/PicoPOVDisplay



AUTOMATA

Learn mechanical engineering with a 3D printer

Generations of mechanically minded folk have played with automata. These are mechanisms that – in some way – move by themselves, and the term is usually reserved for things that aren't electronic.

They can be powered by springs or gravity, but in the case of the 3D-printed ones we're looking at, they're operated by turning a handle. The automata consist of some form of mechanism to transform this rotation into the animation for a scene. We'll see flying animals, hunting felines, and ocean waves.

These automata can show how relatively simple mechanisms can create quite lifelike movement. As well as being fun little games, they give the maker a safe space to explore different tools, techniques, and materials.

In this feature, we're looking at 3D-printed automata. It's common to think of 3D printers as simple replicators that don't require any thought or effort in their use, but this isn't the case. Like all tools, 3D printers require skill and knowledge to let the user get the most out of them, and obviously, 3D design is a whole set of skills in its own right. Automata give us a space for experimenting with these where there's not much risk if things go wrong. Getting your automata to work will require thinking about forces (and the orientation of these forces), tolerances, and the mechanical properties of the materials we're working with – as well as the mechanics of the design.

We're going to take to the skies, catch mice, sail the seas and more with just the help of some filament and a printer. Let's see what it takes to make your prints move. →

PICKING PLASTIC

We were able to print all the parts for these automata in PLA, but this isn't necessarily the best plastic for the job. Nylon's low friction or the toughness of PETG or ASA might make them a better choice for some of the parts.

One significant advantage of PLA is that it comes in a wide range of colours, so you've got plenty of stylistic choices. That said, if you don't want to clog up your cupboards with spools of different colours, you can paint your parts. This probably won't work well for moving parts such as gears, but for some other bits, it's a quick and cheap way to add more colour. The cat and bat models in this article are painted as we didn't have a suitable colour PLA to hand.

EAGLE IN FLIGHT

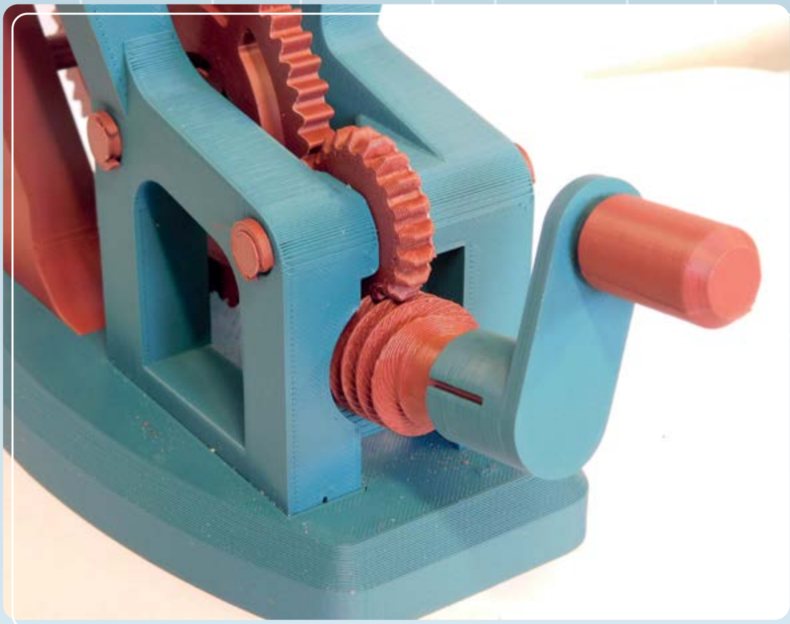
Gears, linkages, and a realistic flight

hsmag.cc/flyingeagle

This machine is frankly a ridiculously complex mechanism for a 3D print. There are detailed instructions on the printables site at **hsmag.cc/flyingeagle**, but friction is the main enemy on a build like this. You need to make sure that your parts fit together smoothly. A big part of this is the point where your print lies on the print bed – you can get a bulge on the bottom layer known as elephant's foot.

If you haven't got your bed perfectly level with the right Z offset, you can end up with the first layer spreading out a little more than the other layers. When parts have to fit together snugly, this can cause problems. There are two ways around this: tune your printer, or rectify this problem afterwards with a knife or sandpaper. You might need to do a bit of both.

Above The linkage is a bit stiff in practice, so you'll need some sandpaper and patience to get it flapping



Above The large gear reduction means you have to turn the handle a lot for the eagle to fly

Even with a perfectly tuned printer, you might find that you need a light sanding, particularly on the peg that goes through the eagle's body.

The friction on this eagle puts a huge stress on the axles. We found that we had to print these at 100% infill to make them strong enough.

DRIVE CHAIN

There are a lot of gears in this build, but that's really the most straightforward part of it. The handle turns a worm-gear that drives a gear chain that ends in two bevel-gears taking the rotation into the wing mechanism. This drive mechanism does two things: it takes the rotation from the handle and moves it to the place it's needed, and it reduces the speed from a fast handle wind to a slow wing-flap.

The movement of the wings looks like one big linkage of levers, but it's probably best thought of as two mechanisms. The bird with wings, and the levers that drive this.

Linkages of stiff rods can create a range of movement. However, they can be a little difficult to design because it can be hard to work out exactly what arrangement of rods creates what movement.

The specific movement this eagle uses is known as a Chebyshev lambda linkage. This was originally designed as a linkage to turn rotational movement into a straight line. Well, almost a straight line; it's more like a flattened D shape. In this model, this movement is where the lower lever joins the wing. This moves the bird up and down and the slight side-to-side movement is used to extend the wings for the downstroke and extend them for the upstroke.

Once you've got everything up and running, it's an absolutely delightful automaton. It's almost impossible not to turn the handle each time you pass it. →

Left The bird stretches out its wings for the downstroke

FEATURE

FLYING BAT

A simpler flying machine

hsmag.cc/flyingbat

This bat automaton used a pair of crank-shafts to both raise and articulate the wings. It's almost all 3D-printed, but you will also need some 3mm wire (which makes the pivots) and four M3 screws (about 10mm or so). We did experiment with replacing the 3mm wire with 1.75mm filament, and it just about worked but was a little janky – we suspect it wouldn't last long. If you happen to have some 2.85mm filament tucked away somewhere, it would probably do the job perfectly.

The two cranks are kept in sync by gears on the back. Each crank-shaft drives two rods. One (which has a short joint on the end) plugs into an appropriately shaped hole in the wing. The second joins an L-shaped linkage on the main pivot and drives the opposite wing-tip.

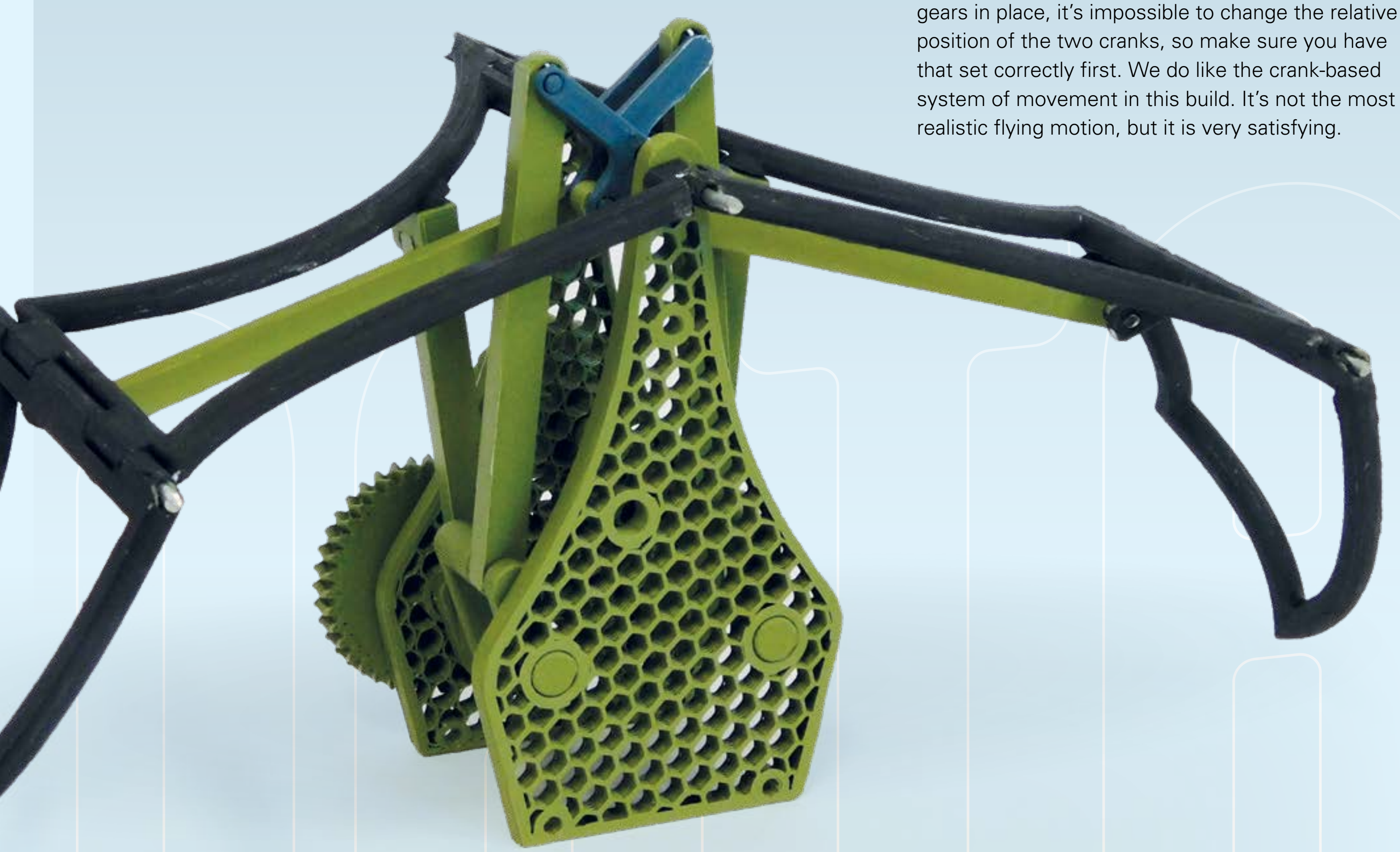
There are very few instructions on the project, and the photos don't do a great job of showing you how to make it. However, we've done the hard work of figuring it out.

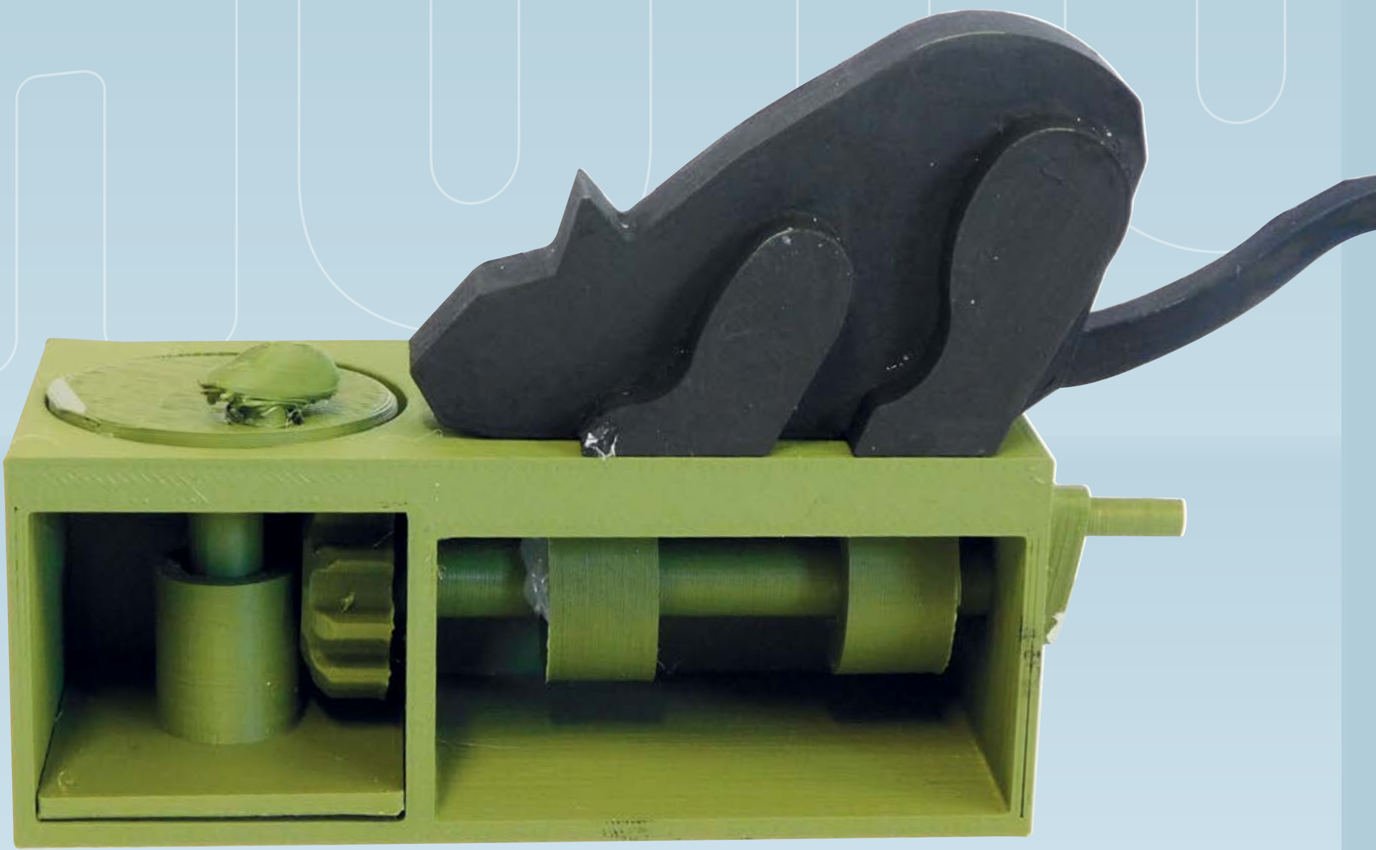
The one instruction that Printables user Kingfisher does give us is a good one. The front and back panels get their distinctive hexagon look by printing them with no top and bottom layers, and just hex infill. The pattern isn't in the model, it's exposed infill.

There's not a lot to the build once you've worked out how the mechanics go, which you should see from the images here. However, you do need to glue it together, so once you've put it together, there's no going back if you've made a mistake.

We've taken photos of the key parts which should help you put things in the right place. The only particular trap we found is that once you've glued the gears in place, it's impossible to change the relative position of the two cranks, so make sure you have that set correctly first. We do like the crank-based system of movement in this build. It's not the most realistic flying motion, but it is very satisfying.

Below
The linkage looks a bit like a real bat's shoulder muscles





CAT & MOUSE

hsmag.cc/catandmouse

Predator and prey

This model was uploaded seven years ago by Thingiverse user MakerMatic. It's designed to be push-fit together, but we found the joints very loose, so you will also need some glue. A handle turns a camshaft, which has two cams on it. Both pushing up levers – one leaver is the cat's arm, and the other is its tail. At the other end of the shaft is a bevel-gear that rotates a platform with a mouse on it.

Looking at the creator's video, it appears the gears should fit together properly, but due to a happy accident, they don't on our build. This means that rather than moving around smoothly, the mouse does little darting runs as the gears catch then skip.

This model is easy to print and make, but there is a slight problem – one that you'll find in a lot of 3D-printed mechanisms – Z axis strength.

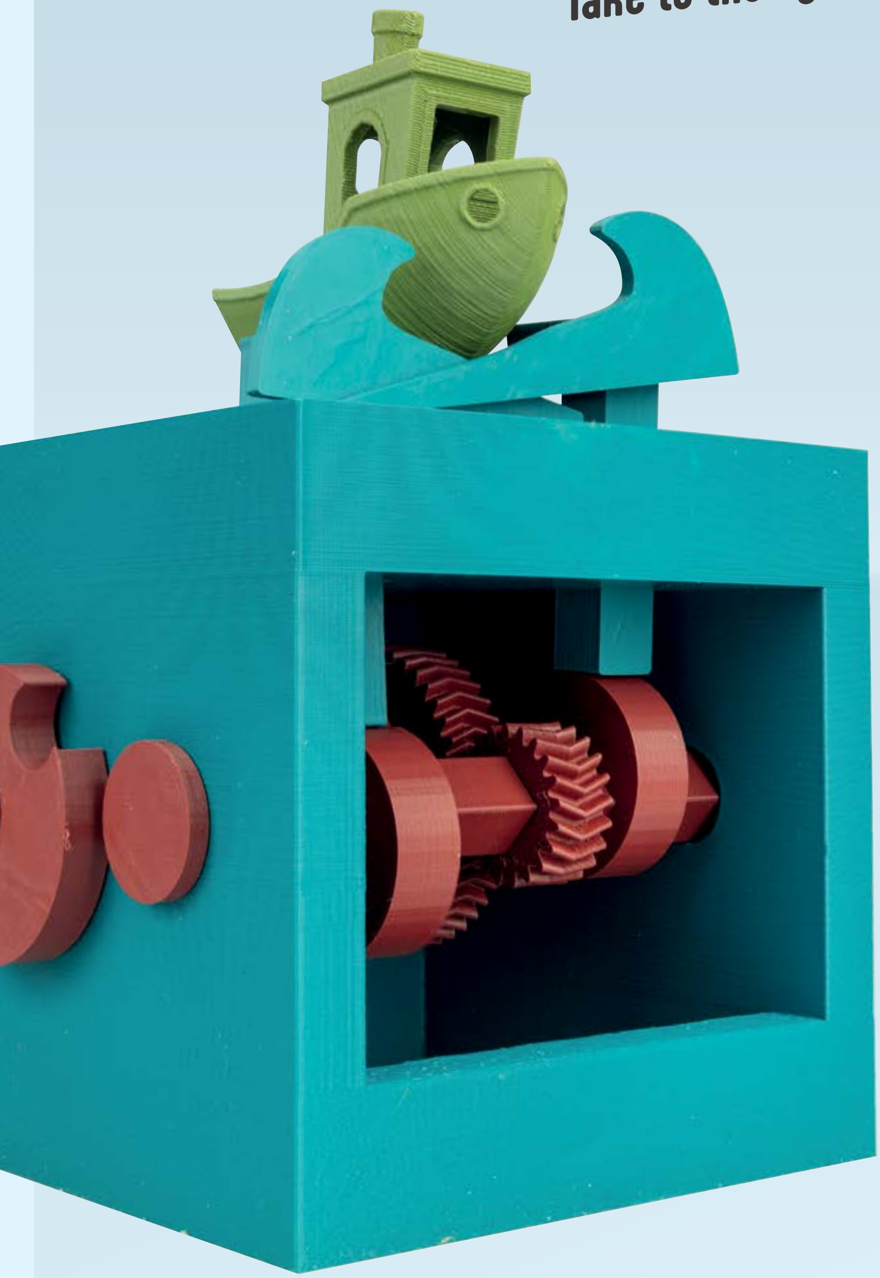
3D-printed parts can be strong in some directions, but they're weakest at the layer lines. Thin parts are particularly weak. In this build, there are a few parts that most naturally print this way – pins to hold the levers in place and the handle. There's no good way around it. Some makers use non-3D-printed parts for this (such as wire or bolts), but that can make it harder to replicate the build.

Compared to the other automata we've looked at here, the movement is much faster. Although it's not truly erratic, the flick of the cat's tail does look like a real hunting feline. →

Above Turn the handle to see the cat try and catch the mouse

BUILD YOUR OWN SAILING BENCHY

Take to the lightly-animated seas



As a magazine about making things, 3D printers pose a bit of a conundrum. After all, all you need to do is load up the filament, slice the model, and press print. However, that's not really the full story. There's also the mechanics of how it works, and you can look at how to design and tweak the model. In this feature, we're going to cover all these angles, and we'll start with printing and assembling the models.

You can download all the bits you need from Printables at **hsmag.cc/automaton**.

You'll need one of everything except:

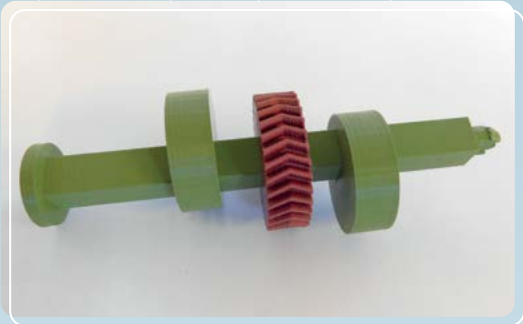
- 3 × cams
- 3 × cam followers
- 2 × axles
- 2 × axle handles

We've taken the slightly odd decision to use square axles. Squares are not generally known for their ability to rotate smoothly in a round hole. While there's a bit of clunking, they do actually rotate reasonably well. The square axle does mean that the cams and gears can slot on without the risk of slipping. If you're assembling it permanently, you might want to add a drop of superglue to hold them in place, but it runs fine without this and that allows it to be completely disassembled to change the gears of cams to change the movement (which we'll look at later).

First, push the rear axle through the hole and thread on a cam the larger gear – it doesn't matter which order they go on in.

You can now push on one of the axle handles and it should clip into place.

You can now push the second axle through the hole. On this you'll need to put on a cam, then the smaller gear, then a third cam. It's important at this point to get things the right way up, as you can get different effects by having the cams the same way up



Above The axle and handle snap together

(having both waves rise together) to having them at different angles. We like the effect of having them 90 degrees out.

You also have to be careful how you put the gear on for two reasons. Firstly, the herringbone has to be the right way around to mesh with the other gear, so you might need to flip it over. Secondly, you won't be able to mesh the gears once the axle is in place, so you'll need to mesh the gears first, then feed the axle through.

With those in place, you can clip the handle on the end of the axle and everything should be linked together. Turn one of the handles (either one) and it should all just work.

You now just need to add the bits that move. To do this, position the cams below the holes, drop the followers through the holes, and put the waves and Benchy on top of the followers.

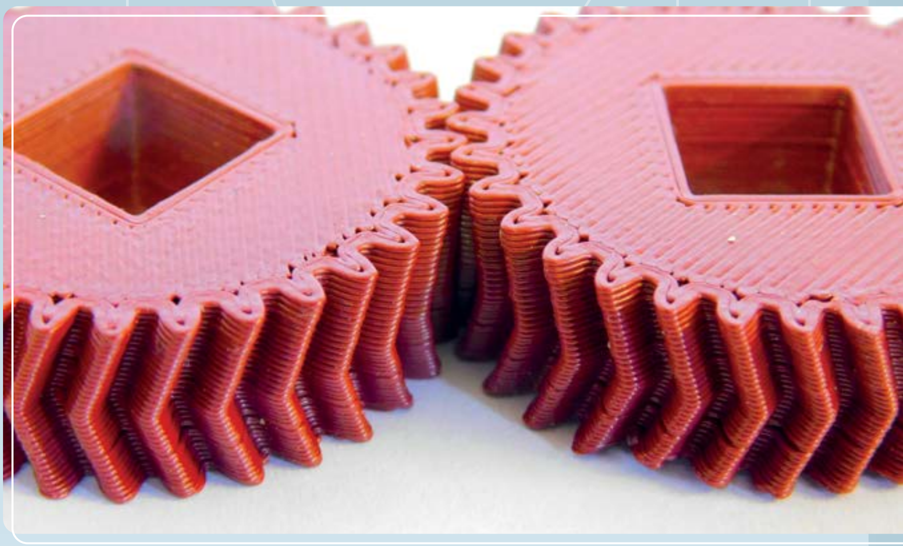
At this point, you should be able to put your finger in the hole on the wheel and give it a spin. Benchy should go up and down, and the waves should break over the bow.

That's how you make it – let's now take a look at how it works.

It's not complicated. There are basically two different types of mechanisms in the box: gears and cams. The gears connect the two axles, and the ratio of teeth on the gears determine the ratio of the speed of the axles. In our case, one gear has twice as many teeth as the other axle, so that axle spins at half the speed of the other axle.

Cams are a way of converting rotational movement into linear movement. The ones in our design are off-centre circles, but they can be many other shapes. The important bit about a cam is that as it rotates around the axis, the surface is different distances from the axle. This makes the cam follower move up and down. They can have more than one protrusion that makes them go up and down multiple times in a single rotation. The followers sit on top of the cams and transfer the linear movement to the object on top. In our case, the followers are held down by gravity, but they can also be held down by springs.

The Benchy itself is also modified slightly and rocks back and forwards on a pivot.



Above The square axles makes it easy to keep the cams and gears secure

DESIGN

We're not going to go through the whole of the design of this automaton. If you're interested in becoming more familiar with the process of 3D design, our free e-book, *FreeCAD for Makers* (**hsmag.cc/freecadbook**), has details about how to go about this.

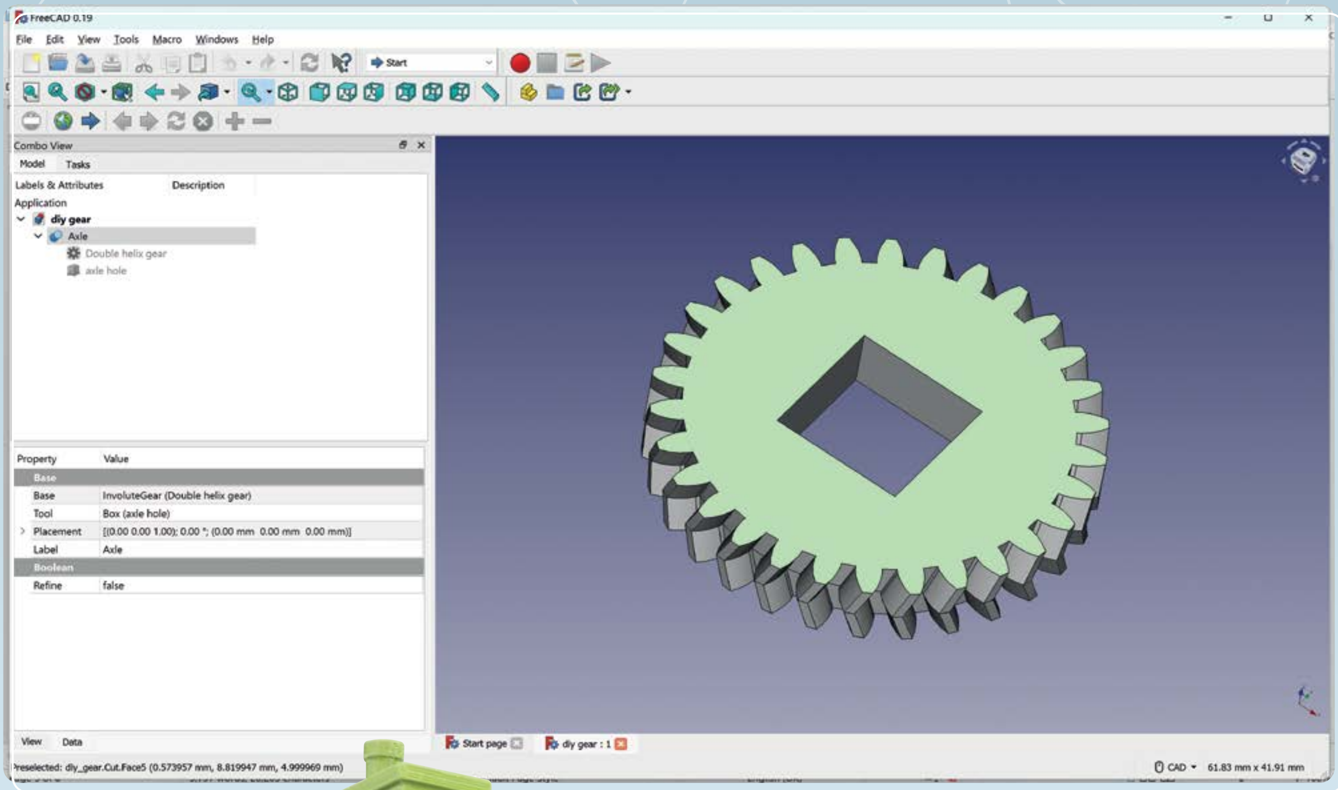
However, we will look at the two key parts of the mechanisms and how to modify them.

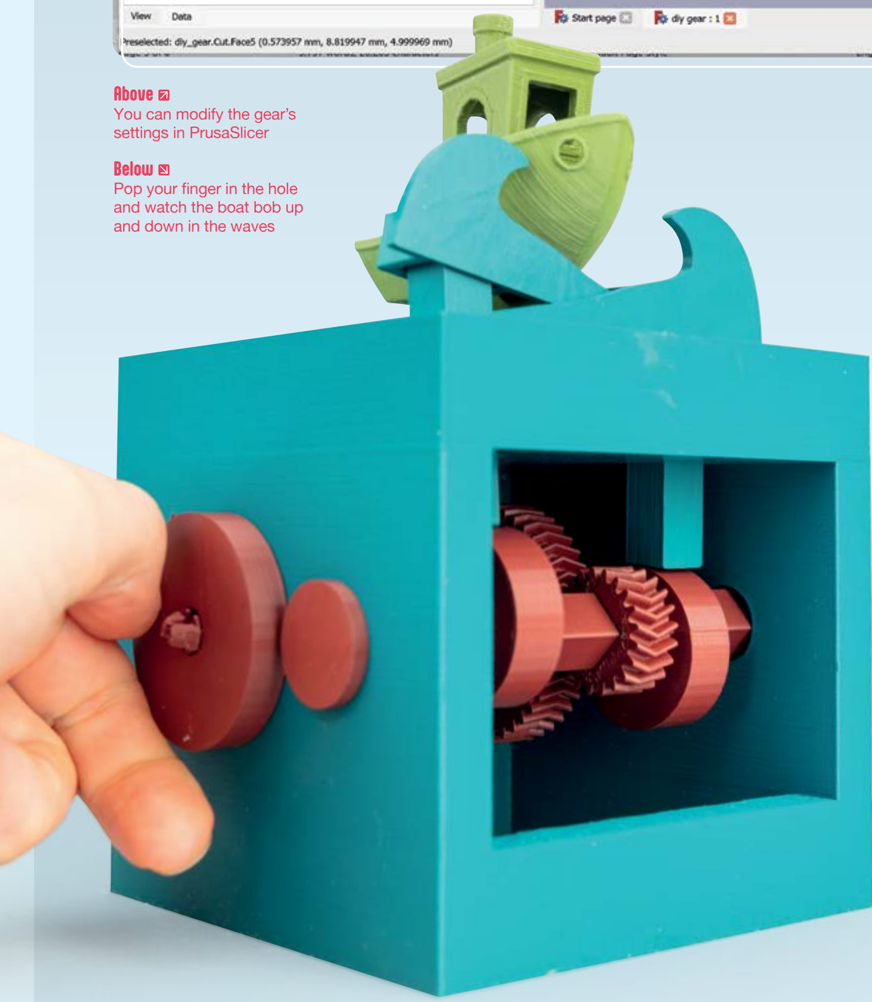
The cams are the easiest thing to adjust. The axles are 9.9 × 9.9 mm. This means that you need a square hole slightly larger than 9.9 × 9.9 mm in your cams. How much larger depends a lot on your printer. We found that a 10.2 × 10.2 mm hole gave an easy fit even if there's a bit of elephant's foot, but you might be able to get away with a little smaller. It doesn't have to be particularly tight to stay in place.

You don't even need to venture into CAD software if you don't want to. Let's take a look at how to create an off-centre square cam using PrusaSlicer. This cam will give your boat a much bumpier ride than the circular cams that the original design comes with, but all boats experience storms sometimes!

Open PrusaSlicer, and on the blank print bed, right-click then select Add Shape > Box. This will add a 1-inch cube to the print bed. In the left-hand pane, you can change the size, but before you do, you'll →

> "Benchy should go up and down"





Above You can modify the gear's settings in PrusaSlicer

Below Pop your finger in the hole and watch the boat bob up and down in the waves

need to click on the padlock otherwise it'll scale it proportionally in every dimension.

Enter a Z height of 10 mm, and X and Y as 28 mm.

We now need to add the axle hole. Right-click on the box and select Add Negative Volume > Box. This now creates a shape that will be subtracted from the other shape when it's sliced. Again, you can use the left-hand pane to resize it and then move it into position. Where should it go? That's up to you. Near the centre would give four bumps per rotation, whereas near one of the edges would give a rise and fall with four bumps along the way. What effect do you want to create?

GRINDING GEARS

The Benchy is controlled by one of the axles, and the waves by the other. We can control the relative speed of the two by changing the gear ratio. This means you can have the waves go faster, slower, or the same speed as the boat. What effect do you want?

In order to create gears at a different ratio, we need to dive into FreeCAD. We're using FreeCAD 1.9 with the Gears workbench. We'll guide you along here, but if things seem strange and unfamiliar, it might be worth reading through the first couple of articles in the *FreeCAD for Makers* book to familiarise yourself with the tool.

We first need to install the Gears workbench. You can do this by going to Tools > Addon Manager, and scrolling down to freecad.gears. Highlight this and click Install. Once installed, you can use the

Workbench drop-down (which will probably show the Start workbench) to select the Gear workbench.

Now you can create a new project (File > New) and we can start building our gears.

In FreeCAD terms, the gear we're creating is an external involute gear. You can create one of these by clicking on the left-most yellow icon in the Gear workbench.

Now we need to make sure that our two gears fit in the space we have. The distance between the centres of the two axles is 72 mm. However, you can't just create a gear with an arbitrary radius, because it might not have a whole number of teeth. Instead, there are two parameters which determine a gear's size: the module and the number of teeth.

The module is the size of the tooth, and both gears must have the same size module in order to fit together. The pitch diameter of a gear (which excludes the bit that overlaps with the gear it meshes with) can be calculated as the module multiplied by the number of teeth. The maths is obviously simpler if we use a module of 1. Since we're only interested in the radius, but the radius of both gears, the whole formula collapses and we just need to make sure that the sum of the teeth on both gears is the distance between the centre of the two gears – in this case, 72.

GOING FISHING

One slightly unusual feature of the gear is the 'herringbone' arrangement of the teeth. We'll be honest and say that the appearance was a significant factor in choosing this style of gear, however, it does have a practical use – it locks the gears together, meaning they won't slide sideways out of alignment, even if they're not fixed to the axles. Of course, a drop of glue would do the same, but as we said, we like the way this looks.

You can create a herringbone gear in FreeCAD by selecting the gear and scrolling down through the properties until you get to the Helical section. The Beta parameter is the angle of the teeth (we went with 20 degrees), and set Double Helix to True to enable the teeth to angle in both directions.

This will create a gear but with no hole for an axle, so again, we need to add a square hole in the middle.

Now use the Workbench drop-down to change from Gear to Part.

In FreeCAD, there isn't quite the same concept of negative volume that there is in PrusaSlicer (at least not in the Part workbench). Instead, we can create a regular box and use a cut to remove the box from the gear. First, you need to switch to the Part workbench, then click on the yellow cube to create a cube solid.

You can set the length and the width of the cube to be 10.2 mm. However, the cube will be positioned with one of its corners at the origin while the gear will be positioned with the centre of the gear at the origin. This isn't what we want. You can simply calculate the offset for the cube (-5.1) to put it in the right position; however, you'll have to adjust this if you want to change the tolerances. Another option is to use the expressions.

In the Base section of the cube properties, click on Placement to open up the options for X, Y, and Z. Click into X, then press the 'fx' button to open the expressions box. In here, you can type:

```
-Width/2
```

And in the Y box, you can enter:

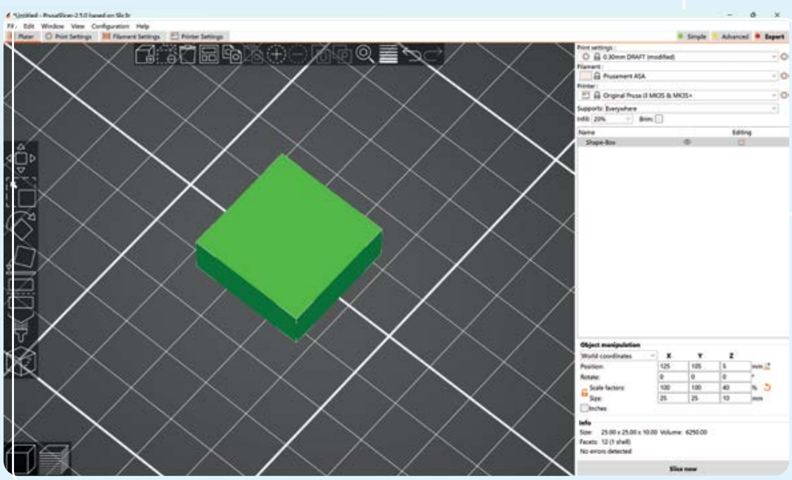
```
-Length/2
```

The cube should now be centred on the gear. If it isn't, right-click on the cube in the cube in the left-hand pane and select Recompute, as it sometimes doesn't do it instantly.

We've uploaded a file with this all setup as **diy gear.fcstd** to the Printables project. You can download this, and then just modify the number of teeth. To do this, open the file in FreeCAD, then open the DIY Gear > Axle drop-down and select Double Helix Gear. This will open the properties box in the left-hand pane, and you can update the number of teeth.

Have a play with this automaton and see the effects, then set forth and design your own. Explore your own mechanisms using gears, cams, cranks or whatever you like. You'll have some fun, and hopefully learn a bit along the way as well. □

"WE LIKE THE WAY THIS LOOKS"



Below You can design really simple parts directly in PrusaSlicer

FEATURE

LEARN TO CODE

FOR MAKERS

Learn to code and bring life to your projects by making them light up, sense their surroundings, and move

BY PHIL KING

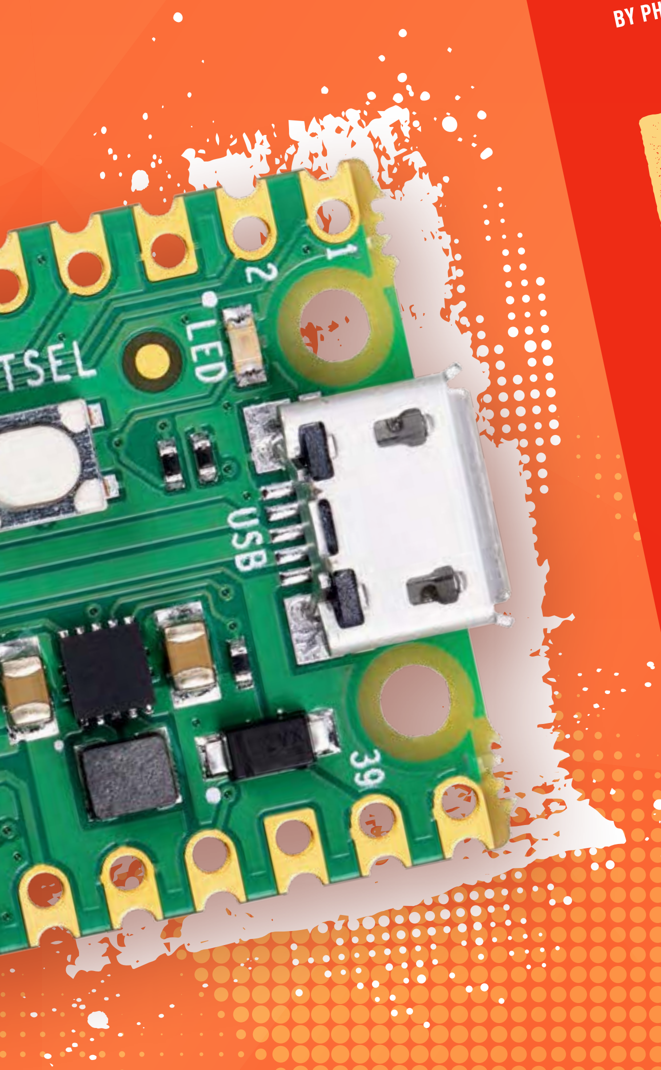
Digital making and learning to program go hand in hand. While you can borrow code from other makers to use in projects, and it's often a good starting point for adapting, it's useful and empowering to learn the basics of programming.

In this introductory guide, we'll explore some of the key coding concepts using a couple of variants of the easy-to-understand Python language. Throughout, we're using the Raspberry Pi Pico and mainly MicroPython, but the same techniques can be adapted for use with other microcontrollers or single-board computers (e.g. Raspberry Pi) that support Python or its variants.

We start out simple with the 'Hello World' of digital making: getting an LED to blink. We then add a push-button to trigger it, using a conditional statement. Controlling a multicoloured RGB LED is next on the list, with three GPIO pins used to set its shade. This is followed by a guide to lighting NeoPixel strips using CircuitPython and an LED animations library.

Next up, we move on to reading sensors, starting with a DHT11 to sense temperature and relative humidity. Using a moisture sensor placed in the soil, we then create a plant monitor that sounds an alert when it needs watering. Last but not least, we get moving by controlling motors for a wheeled robot, along with a servo for precision movement.

With a bit of coding knowledge, you can bring any project to life and get it working exactly as intended. Do let us know what you make! →



> "With a bit of coding knowledge, you can bring any project to life"

CODING WITH LEDS & BUTTONS

CONTROL AN LED AND TRIGGER IT USING A PUSH-BUTTON

YOU'LL NEED

- Raspberry Pi Pico with soldered headers and MicroPython installed (see hsmag.cc/PicoMicroPython)
- Breadboard
- 1 × LED (any colour)
- 1 × 330Ω resistor
- Push-button
- Male-male jumper wires

Light-emitting diodes (LEDs) are a mainstay of the digital making world, used in many electronics projects. Whichever flavour of Python you're using, they're easy to turn on and off – in the code or by adding a push-button.

Programming Pico

To program your Pico, connect it via USB to a computer. You'll need a Python IDE to enter and run your code. We're using Thonny (**thonny.org**), with its Python interpreter set to 'MicroPython (Raspberry Pi Pico)'.

Blink an LED

Blinking an LED is the digital making equivalent of a simple 'Hello World' program. For this example, we'll use a Raspberry Pi Pico microcontroller and code it with MicroPython, but you could fairly easily adapt it to use another type of board and/or CircuitPython.

Wire up the circuit on a breadboard as in **Figure 1**. A red LED has its longer, positive leg connected – via a 330Ω resistor – to the GP15 pin on Pico; its shorter leg is connected to a GND (ground) pin. It won't light up unless we trigger GP15 in our code. We'll use a time delay to turn it on and off every second. First, we need to import a couple of MicroPython modules. The **machine** module enables you to read and control Pico's GPIO pins. The **utime** module is used to measure time and add delays.

```
import machine
import utime
```

Next, we define an object for our LED using the **Pin** class, with the pin number as 15 (for GP15) and setting it as an output with **OUT**:

```
led = machine.Pin(15, machine.Pin.OUT)
```

To turn the LED on and off repeatedly, we'll create a **while True:** loop that never stops running (unlike a **for** loop that runs a set number of times). Indented lines of code form part of the loop and so are repeated in turn.

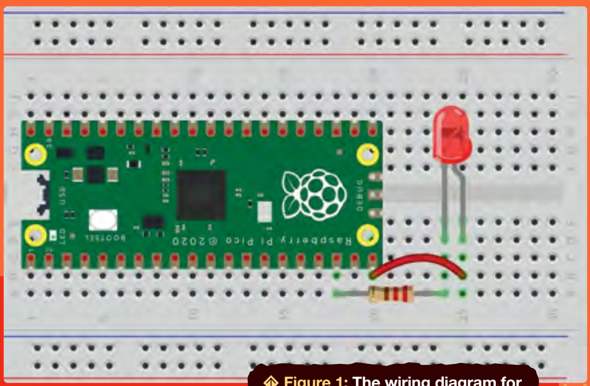


Figure 1: The wiring diagram for connecting an LED. Make sure to use a resistor to limit the current

```
while True:
    led.value(1)
    utime.sleep(1)
    led.value(0)
    utime.sleep(1)
```

Make sure you indent the lines under **while True:** by four spaces. Setting the **led.value** to 1 turns on the LED; setting it to 0 turns it off. The **utime.sleep** command adds a delay of the number of seconds in the brackets. The complete code is:

```
import machine
import utime
led = machine.Pin(15, machine.Pin.OUT)

while True:
    led.value(1)
    utime.sleep(1)
    led.value(0)
    utime.sleep(1)
```

Run the program and your LED with blink. An alternative is to use the **toggle** command to change the LED to the opposite state (on or off): **led.toggle**.

Push the button

As well as controlling output devices like LEDs, we can read the signals from input devices such as a push-button. We'll add a button to our previous circuit, with one side connected to GP16 on Pico, and the other to the 3V3 pin (see **Figure 2**) – pushing it completes the circuit.

First, let's try reading the state of the button. As before, we import the `machine` and `utime` modules. We define an object for the button, with its pin number and setting it as an input; we also set it to PULL_DOWN mode – see 'Pull up or down'. We then use a `while True:` loop to repeatedly check the button's state; if it's pressed, we print a message to the Shell.

```
import machine
import utime
button = machine.Pin(16, machine.Pin.IN,
machine.Pin.PULL_DOWN)
```

```
while True:
    if button.value() == 1:
        print("Button pressed!")
        utime.sleep(2)
```

The indented lines are only executed when the `if` condition is met. Note the use of == to check if value is equal to a number (whereas = is used to set a variable to a value).

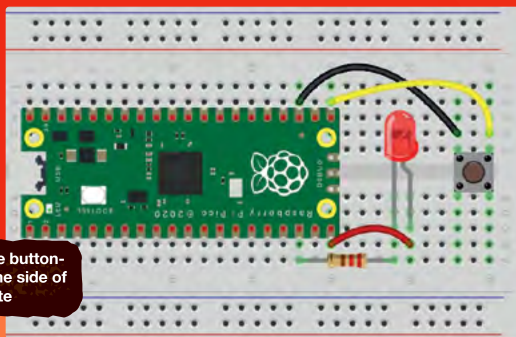


Figure 2: The wiring diagram for the button-activated LED. Use the two pins on one side of the button or those diagonally opposite

Pull up or down

Pico's GPIO pins can have their built-in resistors set to pull-down or pull-up. If the former is used, as in our example, the button should be wired to 3V3 and when not pressed, the input reading is 0. If pull-up is used, the button should be wired to GND and when not pressed, it'll read 1.

Button-activated LED

Now, let's combine the parts of our programs to light the LED whenever the button is pressed.

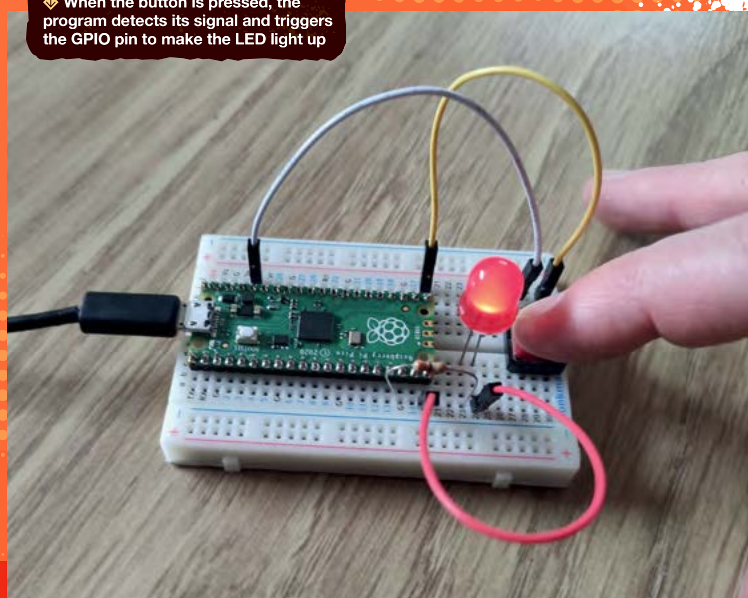
```
import machine
import utime
led = machine.Pin(15, machine.Pin.OUT)
button = machine.Pin(16, machine.Pin.IN,
machine.Pin.PULL_DOWN)

while True:
    if button.value() == 1:
        led.value(1)
    led.value(0)
```

Run the program. Whenever you press the button, the LED will light; when unpressed, the LED will turn off.

Want to toggle the button on or off whenever you press it? Replace the `while True:` loop with this:

```
while True:
    if button.value() == 1:
        led.toggle()
        utime.sleep(1)
```

Now you know how to create code to control LEDs and read buttons, you can add multiple ones connected to different GPIO pins. Ground or 3V3 connections can easily be shared by using the rails on the sides of the breadboard: connect the rail to 3V3 or GND and then wire your components to the rail. You could also add a piezo buzzer that beeps when a button is pressed. →



When the button is pressed, the program detects its signal and triggers the GPIO pin to make the LED light up

> "Whenever you press the button, the LED will light"

MULTICOLOURED LIGHTS

CREATING CODE TO CONTROL RGB LEDS AND NEOPIXELS

YOU'LL NEED

PROJECTS 1 & 2

- Raspberry Pi Pico with soldered headers and MicroPython installed (see hsmag.cc/PicoMicroPython)
- Breadboard
- 1 × RGB LED
- 3 × 330 Ω resistors
- Male-male jumper wires

PROJECT 3

- Raspberry Pi Pico with soldered headers and CircuitPython installed (see hsmag.cc/PicoCircuitPython)
- Breadboard (optional)
- 1 × NeoPixel strip/shape
- Male-female jumper wires (or female-female)

You can light an RGB LED in any colour by altering the red, green, and blue parameters. The same goes for NeoPixels (aka WS2812B LEDs), for which we'll use a CircuitPython LED animations library.

Light an RGB LED

An RGB LED has four legs: for red, green, blue, and ground. Depending on the manufacturer, the order of the colours may vary, but the longest leg is for the ground connection. Connect the LED to Pico as in **Figure 3**. We're using GP13, GP14, and GP15 to connect its red, green, and blue inputs, all via 330 Ω resistors to limit the current. Note that each of these GPIO pins uses a different PWM channel (Pico has 16 in total).

This time in our MicroPython code, instead of importing the full **machine** and **utime** modules, we'll just import the classes we need: **Pin**, **PWM**, and **sleep**. This means we can omit the 'machine' and 'utime' prefixes when using them.

```
from machine import Pin, PWM
from utime import sleep
```

We then set each of the LED's components to PWM on the connected pin.

```
red = PWM(Pin(13))
green = PWM(Pin(14))
blue = PWM(Pin(15))
```

PWM is short for 'pulse-width modulation', which is how we'll control the brightness of the RGB components of our colour. This is achieved by pulsing a digital GPIO output pin on and off at high frequency. The duty cycle determines how much of the time it's on, thus setting the LED brightness.

By default, the PWM frequency (of pulses) on Pico is 1907Hz; it's best practice to set a value in your program, however.

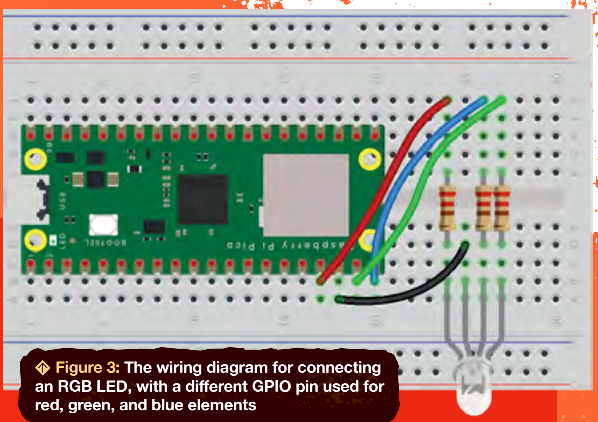


Figure 3: The wiring diagram for connecting an RGB LED, with a different GPIO pin used for red, green, and blue elements

```
red.freq(1000)
green.freq(1000)
blue.freq(1000)
```

In a **while True:** loop, we'll light up the red, green, and blue parts of the LED in turn.

```
while True:
    red.duty_u16(65535)
    sleep(1)
    red.duty_u16(0)
    green.duty_u16(65535)
    sleep(1)
    green.duty_u16(0)
    blue.duty_u16(65535)
    sleep(1)
    blue.duty_u16(0)
```

With **duty_u16**, we're setting the duty cycle with an unsigned 16-bit integer. The higher the duty cycle, the more of the pin's pulses will be on instead of off. Here, we're setting it to the maximum 65535 for full brightness.

Mix RGB LED colours

Let's create different shades by combining red, green, and blue primary light colours. We'll cycle through colours whose RGB combinations are stored as tuples in a list. This time, we'll use a function that we'll call repeatedly in a **while True:** loop. We start with the usual setup code.

```
from machine import Pin, PWM
from utime import sleep
red = PWM(Pin(13))
green = PWM(Pin(14))
blue = PWM(Pin(15))
red.freq(1000)
green.freq(1000)
blue.freq(1000)
```

Next, we create our list of shades. A standard Python list is a range of comma-separated values within square brackets. Here, we're using tuples (in parentheses) for our three RGB values for each list entry.

```
shades = [(255, 0, 0), (127, 127, 0), (0, 255, 0), (0,
127, 127), (0, 0, 255), (127, 0, 127)]
```

For this example, we're using primary and secondary light colours – red, yellow, green, cyan, blue, and magenta – adjusted for equal brightness.

We then create a function to select the colour. Its **rgb** input variable is used to select a tuple from the list – the value of **rgb** is determined when calling the function later in the code. The **[0]**, **[1]**, and **[2]** select the first, second, or third part of the tuple – Python is zero-indexed, so it starts counting from 0. We then multiply the value by 256 to get the 16-bit integer used to set the brightness level of each RGB component.

```
def set_colour(rgb):
    red.duty_u16(rgb[0] * 256)
    green.duty_u16(rgb[1] * 256)
    blue.duty_u16(rgb[2] * 256)
```

We create a **counter** variable that will increase by 1 each time in our **while True:** loop; the latter calls the **set_colour** function repeatedly with a different **counter** value to select a new shade from the list.

```
counter = 0
while True:
    counter +=1
    if counter >= len(shades):
        counter = 0
    sleep(0.5)
    set_colour(shades[counter])
```

Here, **+=** adds one to the counter. An **if** statement is used to check if it is greater than (**>=**) the length (**len**) of the **shades** list; if so, it resets it to 0. After a short **sleep** delay, we call the **set_colour** function; the contents of the parentheses apply it to the **shades** list with the current **counter** value to select the RGB colour.

> *"We create a function to select the colour"*

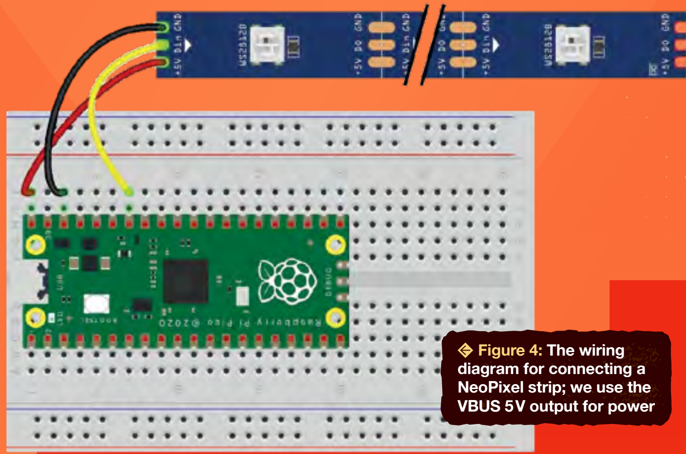


Figure 4: The wiring diagram for connecting a NeoPixel strip; we use the VBUS 5V output for power

NeoPixel lights

For this project, we're using CircuitPython (install from **hsmag.cc/PicoCircuitPython**) so we can make use of Adafruit's LED animations library. You'll also need to download the CircuitPython libraries bundle (**hsmag.cc/CPLibraries**) and copy the **adafruit_led_animation** folder and **neopixel.mpy** file to the **lib** folder on Pico.

Connect your NeoPixel strip (or shape) to Pico. Vcc goes to VBUS (5V) on Pico, GND to GND, and Data-In to a GPIO pin – we're using GP28 (see **Figure 4**). In Thonny, change the interpreter to 'CircuitPython (generic)' and enter the following program. Note how CircuitPython uses the **board** library, not **machine**, for controlling Pico's pins.

```
import board
import neopixel
from adafruit_led_animation.animation.rainbowcomet
import RainbowComet

pixels = neopixel.NeoPixel(board.GP28, 20)
pixels.brightness = 0.5

rainbow_comet = RainbowComet(pixels, speed=0.1,
tail_length=7, bounce=True)

while True:
    rainbow_comet.animate()
```

Change the 20 parameter for the **pixels** variable to how many NeoPixels you have. Run the code for a rainbow comet trail effect. This is just one example from the LED animations library. You can group animations to run them together or in sequence. For more info, see **hsmag.cc/CPLEDAnimations**. →

Running the rainbow trail animation on a NeoPixel strip

READING SENSORS

CONNECT A SENSOR AND TRIGGER AN ACTION BASED ON ITS READING

YOU'LL NEED

- Raspberry Pi Pico with soldered headers and MicroPython installed (see hsmag.cc/PicoMicroPython)
- Breadboard
- Male-male jumper wires

PROJECT 1

- DHT11 sensor, e.g. hsmag.cc/DHT11

PROJECT 2

- Moisture sensor, e.g. hsmag.cc/moisturesensor
- Piezo buzzer – active type
- Male-female jumper wires

There's a wide range of electronic sensors that you can connect to your microcontroller or single-board computer. Readings can then be viewed and also used with if conditional statements in the code to trigger an action such as sounding an alarm.

Digital sensor

First, we'll read a sensor with a digital output signal. For this example, we're using the popular DHT11 temperature and relative humidity sensor – alternatively, you could use a DHT22. Connect it to Pico as in **Figure 5**. Its Vcc pin is connected to Pico's 3V3 pin, GND to GND, and DOUT (data out) to GP15.

Our program will read the digital values for temperature and humidity from the sensor and print them in the Shell pane of our Thonny IDE. At the top, we import the `dht` module, which handles interfacing with the sensor. We also import the `Pin` class from the `machine` library, as usual.

```
import dht
from machine import Pin
```

We then define a `sensor` object for our DHT11 sensor using the `dht` module, `DHT11` method, and `Pin` class, with the pin number as 14 (for GP14). Note that if you're using a DHT22 sensor, you should replace `DHT11` with `DHT22`.

```
sensor = dht.DHT11(Pin(14))
```

We then take a reading from the sensor with:

```
sensor.measure()
```

From that, we create two variables with values set to the temperature and humidity readings.

```
temp = sensor.temperature()
hum = sensor.humidity()
```

Finally, we convert the readings into a string for ease of reading, then print them to the Shell pane.

```
readings = ("Temperature: {}°C   Humidity: {}%
".format(temp, hum))
print(readings)
```

Here, the `{}` represent placeholders to put the data, which is the `temp` and `hum` variables in the parentheses after the `format` method. We've left the placeholders empty, so the variables are taken in order, but you have the option of specifying the variable for a placeholder. You could also set the number of decimal places in the placeholder, e.g. with `:.1f` for one place, but since the DHT11 only gives whole number values, it's not needed here. Run the program and see the reading – we haven't used a loop, so the program ends. Try blowing on the sensor, then run the code again and see the difference in values. You could extend the project by using a mini LCD screen and using a loop to repeatedly update the readings shown on it.

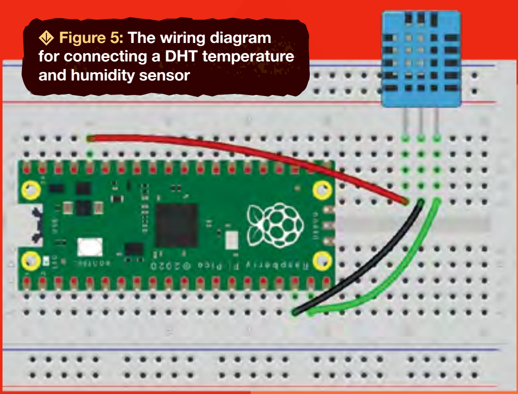
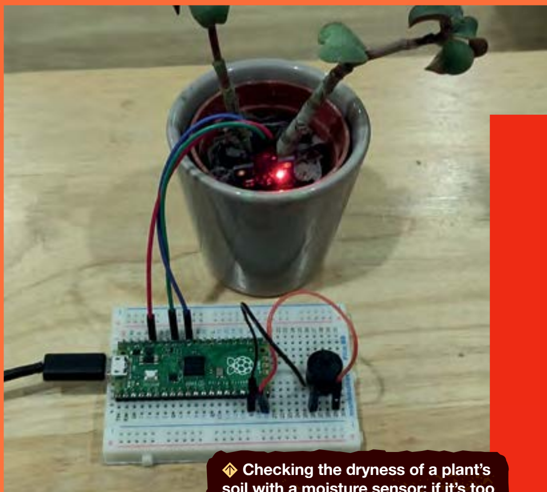


Figure 5: The wiring diagram for connecting a DHT temperature and humidity sensor



Checking the dryness of a plant's soil with a moisture sensor; if it's too dry, a buzzer alert is sounded

Plant monitor

For this project, we'll use an analogue moisture sensor. Unlike a digital sensor, this sends a variable voltage from its signal pin. To read this, we can use one of the ADC channels built into Pico's RP2040 processor. Most microcontrollers have similar analogue inputs; if you were using a Raspberry Pi computer, however, you'd need to add an ADC chip to the breadboard circuit to convert the reading to a digital value.

Connect the moisture sensor to Pico, as in **Figure 6**. Vcc is connected to Pico's 3V3 pin, GND to GND, and AOUT (analogue out) to GP26, which is one of the pins connected to an ADC channel on Pico – the others are GP27 and GP28. Wire the piezo buzzer to GND and GP15 – note that this will only work with an active buzzer, not a passive one.

Along with `Pin`, we import the `ADC` method from the `machine` module; this enables us to take an analogue reading. We also import the `sleep` method from `utime`.

> "If the voltage falls below 500 mV, it then triggers an alert"

```
from machine import ADC, Pin
from utime import sleep
```

We then define a `moisture` object set to the ADC channel on the GP26 pin, along with a `buzzer` object as an output on GP15.

```
moisture = ADC(26)
buzzer = Pin(15, Pin.OUT)
```

We create a `while True:` loop to take a reading every two seconds and print it to the Shell.

```
while True:
    reading = moisture.read_u16()
    voltage = 3300 * reading/65535
    print("{:.1f}mV".format(voltage))
    sleep(2)
```

As in our RGB LED project code, the `_u16` specifies that the converted digital value will be an unsigned 16-bit integer. To convert this to the voltage in mV (microvolts), we multiply it by 3300 and divide by the maximum 16-bit value of 65535.

Run the program to see the voltage reading to one decimal place. Try holding the sensor prongs in one hand to see it increase, since you are conducting electricity between them. Wet your fingers and it'll rise much higher. Now place the sensor's prongs in the soil around your plant.

Next, we'll add an `if` conditional statement to the loop: if the voltage falls below 500 mV (or whatever threshold you want to set), it then triggers an alert for dry soil, reminding you to water the plant.

```
if voltage < 500:
    for i in range(5):
        buzzer.value(0.5)
        sleep(1)
        buzzer.value(0.5)
        sleep(1)
    sleep(30)
```

Here, we're using a `for` loop which repeats a set number of times: five in this case. In each iteration, we turn the buzzer pin on and off every half a second. We then wait 30 seconds before taking another reading.

To extend this project, you could trigger a different action. If you have a Pico W, or another microcontroller with an internet connection, you could send an email or other type of notification. You could even set up a relay switch to trigger a pump to water the plant. →

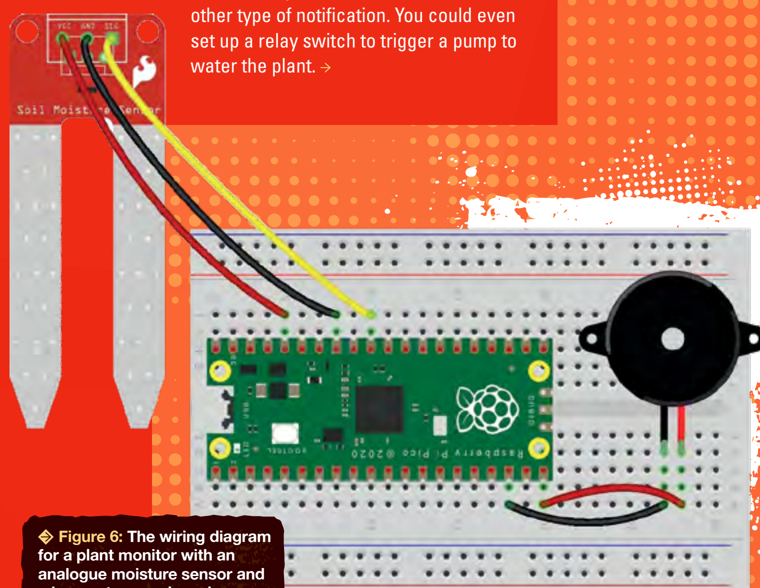


Figure 6: The wiring diagram for a plant monitor with an analogue moisture sensor and a buzzer to sound an alarm

MOTORS AND SERVOS

ADD MOTION TO YOUR PROJECTS BY CONTROLLING MOTORS WITH CODE

YOU'LL NEED

Raspberry Pi Pico with soldered headers and MicroPython installed (see hsmag.cc/PicoMicroPython)

PROJECT 1

Dual H-bridge motor driver, e.g. hsmag.cc/MotorSHIM

2 × Micro metal gear-motors (with suitable connectors, e.g. JST-ZH)

Chassis and wheels (optional)

PROJECT 2

Breadboard

SG90 micro servo 9g

1 × 330Ω resistor

Male-male jumper wires

If you want to make the wheels on that robotic vehicle go round, you're going to need some motors. We'll explore how to create code to control them, along with the precision angular movements of a rotary servo motor.

Drive motors

Motors have many uses, but here we'll use them for a simple two-wheeled robot. To be able to drive each motor in forward or reverse, we'll need a dual H-bridge driver which can reverse the polarity – depending on which of two connected GPIO pins we output a signal from.

In this example, we're using a Pimoroni Motor SHIM attached to a Pico, but the basic principles of running motors is the same for any controller. You can run them forward or backward, and adjust their speed using PWM.

For the Motor SHIM, you'll need to download Pimoroni's MicroPython firmware (**hsmag.cc/PimoroniPicoUF2**) to use the firm's Motor library – which works with any RP2040 controller. We'll start by importing the `utime sleep` method and various classes from the Motor library (**hsmag.cc/PimoroniMotorLib**).

```
from utime import sleep
from motor import Motor, pico_motor_shim
from pimoroni import NORMAL_DIR, REVERSED_DIR
```

We then set a main driving speed constant and define objects for the left and right motors. In a standard two-wheeled robot setup, one motor will have a reversed direction (as it's mounted at 180° to the other). We also set the `speed_scale` for each motor – if the robot doesn't move forward in a straight line, which is fairly common as not all motors run exactly the same, you can calibrate it by adjusting these values.

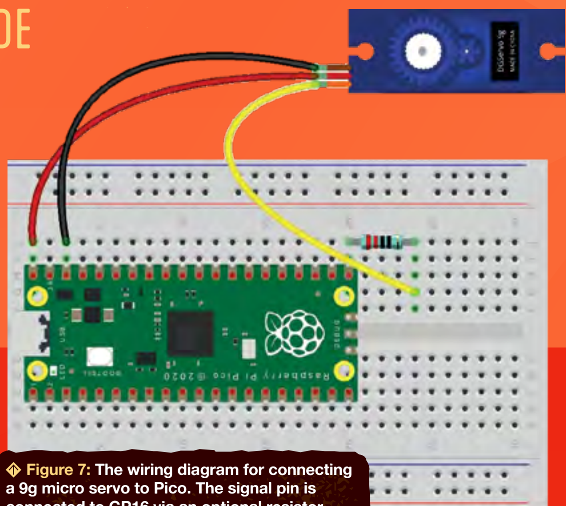


Figure 7: The wiring diagram for connecting a 9g micro servo to Pico. The signal pin is connected to GP16 via an optional resistor

```
DRIVING_SPEED = 5.4
left = Motor(pico_motor_shim.MOTOR_1,
direction=NORMAL_DIR, speed_scale=3)
right = Motor(pico_motor_shim.MOTOR_2,
direction=REVERSED_DIR, speed_scale=3)
```

Next, we define some functions for moving the robot forward and back, by running both motors together at a positive or negative speed.

```
def forward(speed=DRIVING_SPEED):
    left.speed(speed)
    right.speed(speed)

def backward(speed=DRIVING_SPEED):
    left.speed(-speed)
    right.speed(-speed)
```

To make the robot turn sharply left or right, we run one motor forward and the other in reverse.

Control a servo

A servo is a special motor with a feedback element so you can adjust its rotary or linear position – ideal for when you need precision, such as in a robotic arm. Since you don't need to reverse the polarity to operate a servo forward and backwards, there's no need for an H-bridge. You can connect it to Pico (or other microcontroller/SBC) directly and use a PWM output from a GPIO pin, varying the duty cycle to control its angle/position.

We're using MicroPython to control a rotary servo. As in **Figure 7**, the servo's red wire is connected to 5V, brown to GND, and yellow (signal) to GP16 via a 330Ω resistor (an optional safeguard). Attach a plastic horn to the servo; you can calibrate its position later. You can attach another part of a project to the horn using tiny screws or even hot glue.

At the start of the code, we import the `PWM` and `Pin` methods, along with `sleep` from the `utime` module.

```
from machine import Pin, PWM
from utime import sleep
```

We define a `servo` object to use PWM on the GP15 pin. We set its frequency to 50Hz, which equates to 20 milliseconds, or 20,000 µs (microseconds), per pulse.

```
servo = PWM(Pin(16))
servo.freq(50)
```

We create a function to set the servo angle, based on the duty cycle of the PWM signal. We obtain the latter (rounded to an integer) by dividing the `angle` input variable by 180, multiplying that by 2000 (for the range of pulse length values in µs), adding 500 for the minimum pulse length, then multiplying by 65,535/20,000 – the maximum 16-bit duty cycle value divided by the number of microseconds per pulse.

```
def set_angle(angle):
    duty = int((2000 * angle / 180 + 500) * 65535
/ 20000)
    print(angle)
    servo.duty_u16(duty)
```

To calibrate the servo, add the following line and run the code. It should cause the servo horn to point straight up (90°) when the servo is laid flat on a surface. If not, remove the horn and reattach it in as close to 90° as possible.

```
set_angle(90)
```

For this simple demo, we'll now use a couple of `for` loops to change the servo position in steps of 10, from 10 to 170 degrees, then 170 to 10. Since many servos of this type struggle to move the full 180 degrees, we're limiting it to this range.

```
for angle in range(10,170,10):
    set_angle(angle)
    sleep(0.5)
for angle in range(170,0,-10):
    set_angle(angle)
    sleep(0.5)
```

Stepper motor

An alternative to a servo is a stepper motor. Whereas a servo uses feedback to control its position, a stepper motor simply moves a precise number of steps. These are determined by the digital outputs from four GPIO pins via a stepper motor controller board; turning each output on in turn moves the motor a step.

```
def turn_left(speed=DRIVING_SPEED):
    left.speed(-speed)
    right.speed(speed)

def turn_right(speed=DRIVING_SPEED):
    left.speed(speed)
    right.speed(-speed)
```

We can turn it more gently by stopping one motor while running the other one.

```
def curve_forward_left(speed=DRIVING_SPEED):
    left.speed(0.0)
    right.speed(speed)

def curve_forward_right(speed=DRIVING_SPEED):
    left.speed(speed)
    right.speed(0.0)
```

For stopping, you can either use the library's `coast` method or the more abrupt `stop`.

```
def stop():
    left.stop()
    right.stop()
```

Those are the basic methods for controlling the motors for a robot vehicle. You can call them in a sequence for a set pattern of movement, with `sleep` delays to set the length of operation, such as in this example:

```
forward()  # Full speed
sleep(0.5)

forward(0.5 * DRIVING_SPEED)  # Half speed
sleep(1)

backward()
sleep(1)

turn_left()
sleep(1)

turn_right()
sleep(1)
```

For a more autonomous robot, you can use `if` statements to determine movement based on external inputs such as sensor readings – an ultrasonic distance sensor may be used for obstacle avoidance, or a bottom-mounted infrared sensor to follow a black line along the floor. □

Untethered robot

To run your robot untethered from a computer, you'll need to use battery power, such as a phone charger. You will also need to save your program as **main.py** so it runs automatically as soon as Pico is powered up.

DIY SMART CAMERA

WITH A RASPBERRY PI AND CAMERA MODULE, THE POSSIBILITIES ARE NEAR ENDLESS

Phil King

A long-time Raspberry Pi user and tinkerer, Phil is a freelance writer and editor with a focus on technology.

Combining the power and versatility of a Raspberry Pi single-board computer with a compact Camera Module – or larger HQ Camera with interchangeable lenses – enables you to take great pictures and videos and do a whole lot more. There's also the option of 3D-printing or modifying a case to create an all-in-one compact camera. Or, one with an eyepiece tube to fit a telescope for spectacular shots of the night sky.

Using the NoIR Camera Module variant, which omits the IR filter, you can even capture images in the dark (with IR LED illumination) – ideal for night vision, nocturnal creature photos, or even a ghost detector.

With the addition of machine learning computer vision, advanced projects enable you to identify objects and monitor your body movements during workouts.

Other fun project possibilities include a photo booth, security surveillance system, wildlife camera trap, and time-lapse photography. Lights, camera, action!...

MAKE A TIME-LAPSE VIDEO

COMBINE A SERIES OF STILL IMAGES INTO A SPED-UP VIDEO

One of the easiest, and yet most impressive, uses for a Raspberry Pi camera is to film a time-lapse video created from a series of still images.

Sometimes used for interludes in film and TV dramas, to indicate the passing of time between scenes, a time-lapse video is an impressive and versatile effect that appears to speed up time. Classic examples include clouds passing across the skyline, city traffic moving (a lot faster than in reality), star trails, a sunset or sunrise, and plants blooming/growing.

With a Raspberry Pi Camera Module or High Quality Camera, you can easily create effective time-lapses. You just need to set everything up correctly and make sure the camera doesn't move during the filming period.

STEP 1: CONNECT THE CAMERA

You will need to connect the Camera Module or High Quality Camera before switching on your Raspberry Pi, otherwise it won't be detected. If the ribbon cable isn't already connected to the camera's CSI connector, pull the plastic tab out and insert the cable (with the covered blue side facing the plastic tab), then push the tab back in. On the Raspberry Pi, insert the other end of the cable into its Camera port in similar fashion. Note that on a Raspberry Pi Zero, the Camera port is on the edge of the board and is narrower, so requires a special ribbon cable/adapter. Make sure both cable ends are snug and secure – a poor connection can stop the camera working.

STEP 2: REMOTE ACCESS

You can connect the Raspberry Pi to a monitor to check the camera is working – by entering the `libcamera-hello` terminal command for a five-second camera preview.

When filming a subject, however, you may well want to position your camera away from a monitor – in which case, you will need to access the Raspberry Pi remotely via SSH (Secure Shell) to control it. You can set up SSH in the Raspberry Pi Configuration menu's Interfaces tab, or in the raspi-config tool from the command line.

Above Make sure that the camera is connected to the correct port

> **You can connect the Raspberry Pi to a monitor to check the camera is working**

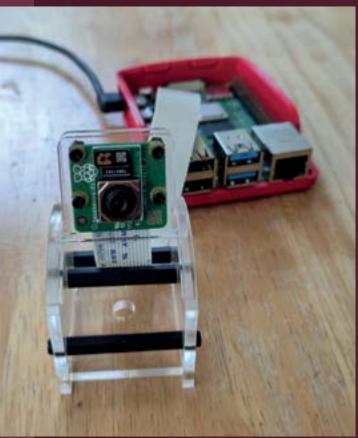
Alternatively, you can enable SSH when writing the Raspberry Pi OS image to a microSD card in the advanced options of the Raspberry Pi Imager tool (click the cog icon after choosing an OS), as well as altering the host name and/or setting up the Wi-Fi connection to your router so the Raspberry Pi automatically connects to it on first boot.

If you're only going to be using the Raspberry Pi remotely, you may want to just install the Lite version of Raspberry Pi OS, which lacks a desktop and therefore takes up less room on the card and is quicker to boot.

You can SSH into the Raspberry Pi from the terminal/PowerShell of another computer using the Raspberry Pi's IP address:

```
ssh [username]@[IP address]
```

You can find the IP address of your Raspberry Pi by entering `hostname -I` at the command line – or, →



Above If you don't have a purpose-made camera mount, you can make one

if you don't have it connected to a monitor, by checking the devices list on your wireless router web page to find it.

Alternatively, you can use the Raspberry Pi's host name instead of an IP address. It's **raspberrypi.local** by default, but can be set to another name in Imager's advanced options.

```
ssh [username]@raspberrypi.local
```

Either way, you'll be prompted to accept the connection by typing 'yes' (the first time), then asked to enter the Raspberry Pi's password. You'll then see the usual command line prompt for it.

STEP 3: POSITION YOUR CAMERA

The main thing to ensure when filming a time-lapse video is that your camera doesn't move, otherwise it'll ruin the effect. So it's best to place it in one of the many Camera Module mounts, which can be bought cheaply – or, if using an HQ Camera, you can screw it onto a standard camera tripod.

An alternative for filming exterior scenes is to stick the camera to a window using a special mount with suction cups, such as the ZeroView (**hsmag.cc/ZeroView**) which can also hold a Raspberry Pi Zero W on the rear.

At this point, you may want to take another test shot to check the focus and light level for the scene. The V1 and V2 Camera Modules have fixed focus, while the V3 has the option of continuous autofocus. The exposure mode is automatic by default, although you can use the `--ev` parameter to make it lighter/darker or use one of several exposure scenes, e.g. night, with the `--ex` parameter. Note that conditions may change during the shoot, however.

The HQ Camera's focus and aperture can be adjusted manually by rotating parts of the attached C- or CS-mount lens (see **hsmag.cc/HQCmountLens** or **hsmag.cc/HQCSmountLens**).

STEP 4: TAKE A TEST SHOT

The latest versions of Raspberry Pi OS have libcamera-apps pre-installed. While these replace the previous `raspistill` and `raspivid` commands, they work almost the same way with a similar range of options. To take a test still image with the camera, enter:

```
libcamera-still -o test.jpg
```

You'll see a stream of information in the terminal as the shot is taken. If you're using a desktop version of Raspberry Pi OS and are connected to a monitor (or are accessing the desktop remotely via VNC), you can view the photo in the Image Viewer desktop application. If not, you can copy it to your local computer using SCP (Secure Copy):

```
scp [username]@[Pi's IP or hostname]:test.jpg
 [/local/path/to/save]
```

Enter the Raspberry Pi's password to proceed. Check that the test shot looks OK before proceeding with filming the time-lapse. Otherwise you may want to reposition the camera or alter some of libcamera's settings – see the documentation for more info: **hsmag.cc/libcamera**.

Above Summer has finally come!

RASPBERRY PI CAMERAS

There is a range of official Raspberry Pi cameras available. For more details and full specs, see **hsmag.cc/RPiCameraDoc**. Third-party cameras are also sold, but may require their own software libraries.

- **CAMERA MODULE V2:** An upgrade of the original 5MP, fixed-focus Camera Module, this tiny camera features adjustable focus and an 8MP sensor. Available in standard and NoIR (for low-light shooting) versions.
- **CAMERA MODULE V3:** Upgraded with HDR (high dynamic range), a 12MP sensor, and – most notably – motorised focus which can adjust automatically when shooting video. Available with a standard (70°) or wide-angle (120°) lens, both coming in standard or NoIR versions. It can also record HD video at 50fps.
- **HIGH QUALITY CAMERA:** A larger board with a 12.3MP sensor, it enables you to attach different C- and CS-mount lenses with manual focus and aperture. There is also a version for M12 lenses.
- **GLOBAL SHUTTER CAMERA:** Similar to the0 HQ Camera, with interchangeable lenses, it has a much lower 1.6MP resolution, but a global – not rolling – shutter that's ideal for shooting fast-moving subjects.

Left
We love watching clouds move in fast-forward

STEP 5: LET'S DO THE TIME-LAPSE

You can start shooting a time-lapse with a single `libcamera-still` command with the `--timelapse` option and a few parameters. For example:

```
libcamera-still -t 30000 --timelapse 2000 -o
 image%04d.jpg
```

Here, the number after `-t` is the total duration in milliseconds, in this case, 30 seconds. The number after `--timelapse` is the interval between images in milliseconds, in this case, 2 seconds. Finally, we use `%04d` to add a four-digit number to the file name, after 'image', starting with '0000'.

By varying the timings, you can take much longer time-lapses with more or less frequent image captures. For slower scenes, such as clouds and star trails, use a longer interval between captures; for faster ones, such as traffic or people, a shorter interval is better to capture the movement without it appearing too disjointed. If you're shooting a long duration via SSH, you may want to use the `nohup` prefix before the command. Short for 'no hangup', this will ensure it continues working even if the SSH connection is lost.

STEP 6: COMPILE THE VIDEO

Once you have your sequence of stills, it's time to compile them into a time-lapse video. For this, a good option is the ffmpeg command-line tool. It should be pre-installed in the latest version of Raspberry Pi OS; if not, install it with:

```
sudo apt install ffmpeg
```

Compile the time-lapse video from your images with a command like this:

```
ffmpeg -r 10 -f image2 -pattern_type glob -i
 'image*.jpg' -s 1280x720 -vcodec libx264
 timelapse.mp4
```

Here, the number after `-r` is the frame rate, in this case, 10 fps – try varying it for different results. The next part tells it to pattern-match your set of image files using the * wildcard. The numbers following `-s` (scale) determine the video resolution. The final part uses the x264 encoder to save the video as an MP4 file.

Depending on the Raspberry Pi model and the number of images, this may take a while to complete. A Raspberry Pi 4 is pretty quick, but other models are slower (a Raspberry Pi 3 will encode around two frames per second). So you may instead want to transfer the image files to a more powerful computer via SCP:

```
scp [username]@[Pi's IP or hostname]:image\*.
 jpg [/local/path/to/save]
```

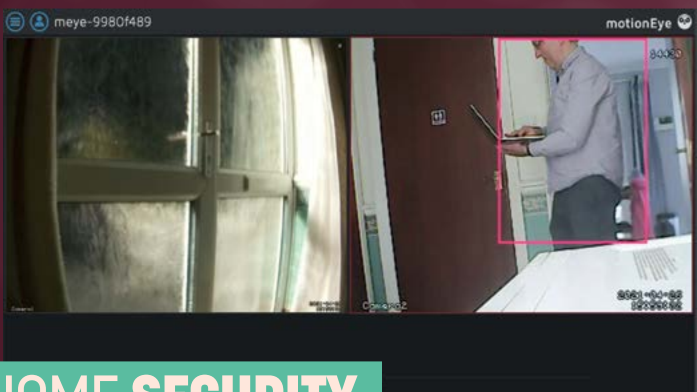
...and then use ffmpeg on that to compile the video much faster. Now you know the basics, have fun experimenting with different subjects and interval/frame rates to create impressive time-lapse videos. →

OCTOLAPSE

As well as using OctoPrint on a Raspberry Pi to monitor and control your 3D printer via a web interface, you can install a plug-in called Octolapse that enables you to shoot cool 3D printing time-lapses. See **hsmag.cc/Octolapse** for more details.

RASPBERRY PI CAMERA PROJECTS

INSPIRATIONAL IDEAS FOR YOUR RASPBERRY PI CAMERA MODULE



Above
The rectangle indicates that motion has been detected (by comparing frames)

HOME SECURITY

Low-cost and tiny, Raspberry Pi Camera Modules are ideal for home security video surveillance. There's even a special operating system, motionEyeOS (**hsmag.cc/motionEyeOS**), that makes it super-easy to set up one or more cameras which can be viewed and controlled via a web interface.

Just head over to the GitHub repo's supported devices page (**hsmag.cc/motionEyeOSDevices**) and download the relevant version for your Raspberry Pi model (or other single-board computer). The wiki has full instructions for how to configure everything, including a network of multiple cameras/devices – including USB webcams. Numerous features include motion detection to trigger photos/videos, email/push alerts, and time-lapse videos.

You could use it to catch porch package thieves. Alternatively, check out Canadian maker Ryder's custom deterrent system that uses machine learning to recognise packages and people, triggering an alarm and showering the thief with flour and water (**hsmag.cc/ThiefDeterrent**).

WILDLIFE CAMERA

A popular use for a Raspberry Pi Camera Module is for capturing close-up shots of wildlife. By adding a PIR sensor to detect movement, you can activate the camera whenever animals are in the field of view – although you may also get a few false positives triggered by people or vehicles passing by.

When placing your camera outdoors, you'll want to protect it and the Raspberry Pi from the elements with a weather-proof enclosure. This could be a transparent box or something more purpose-built, such as the NatureBytes (**naturebytes.org**) Wildlife Camera Case – there's also a fully-fledged kit, including a Raspberry Pi and rechargeable battery pack. Just point it at a bird feeder and you'll get some spectacular shots of your feathery friends.

To take photos of nocturnal creatures, you could set up a similar PIR-triggered camera trap at ground level with a NoIR Camera Module and an IR LED ring (**lisiparoi.co.uk**) that fits around its lens for non-visible flash/illumination. Alternatively, you could place a NoIR camera in an empty bird box, ready to view its occupiers when they arrive: **hsmag.cc/IRBirdBox**.

Above
Pointing your camera at a bird feeder is a sure-fire way to photograph various species, such as these long-tailed tits

COMPACT **CAMERA**

While the Raspberry Pi Camera Modules can be used in mounts or, in the case of the HQ Camera, on a tripod, you can also incorporate one into a case so it looks more like a traditional compact camera.

Among the numerous 3D-printable designs available, one project that caught our eye is PolaPi Zero: **hsmag.cc/PolaPiZero**. Not only does its case enclose a Raspberry Pi Zero and Camera Module, but there's an LCD on the rear for a live view. Inspired by old-school Polaroid cameras, it also features an Adafruit Nano thermal printer that produces a physical monochrome copy of your photo.

If you want to use the larger HQ Camera on the move, check out James Martel's project (**hsmag.cc/RPi4HQCamera**) for inspiration. He modified an existing case for a Raspberry Pi 4 and LCD to incorporate a mounting plate for the HQ Camera. Alternatively, you could hack an official Raspberry Pi case to add an HQ Camera and screen, like Richard Hayler did – see The MagPi issue 94 for a guide: **hsmag.cc/MagPi94**.

Above
Richard Hayler's modified case packs in an HQ Camera, Raspberry Pi 4, and LCD

PHOTO **BOOTH**

Above
For illumination, this booth features a strip of ultra-bright LEDs diffused by an angled piece of wood.
Credit: Jonathan Lang Photography

A fairly simple Raspberry Pi camera project is to build your own photo booth, which is a fun addition to any event. Baulking at the high hire prices for commercial booths, Jack Barker made one for his own wedding (to Pam). Building the main booth structure from plywood, he cut out a section for an LCD screen and drilled holes for a large arcade button below and the Raspberry Pi Camera Module above.

Pressing the button starts a seven-second countdown with an instructional slide, followed by a live view from the camera so you can see yourself on screen and perfect your pose.

While the wedding venue's lack of Wi-Fi scuppered Jack's plan for live online photo backup, he added a printer to the booth to produce instant copies of the photos.

If you fancy making your own photo booth, take a look at Jack's seven-part build guide: **hsmag.cc/JackPhotoBooth**. →

NIGHT VISION

All of the standard Raspberry Pi Camera Modules are available in a NoIR version which omits the infrared filter. This gives it the ability to see in the dark with infrared lighting, opening up all sorts of possibilities, such as monitoring nocturnal animals and the health of plants. Note that it is also possible to manually remove the IR filter from standard Camera Modules and the HQ and GS Cameras – see **hsmag.cc/RPiCameraDoc** for details.

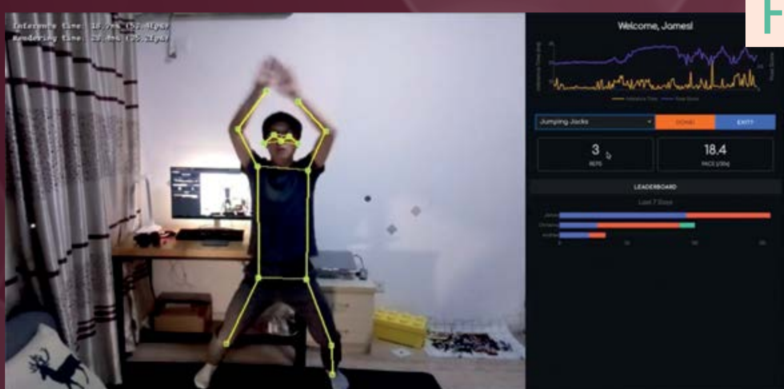
Dan Aldred built a night vision camera (**hsmag.cc/NightVisionCam**) enabling him to see in the dark, just like in the Splinter Cell classic video game. While he used a third-party camera with IR LEDs, connected to a Raspberry Pi Zero, you could instead use a standard NoIR Camera Module to creep around like a stealth operative.

Or you might want to try detecting some ghosts, like Anthony DiPilato. Not only does his handheld Ghost Detector (**hsmag.cc/GhostDetector**) incorporate a NoIR Camera Module and touchscreen for night vision, but also EMF sensors with twin antennae, mic, compass, temperature sensor, and even a Geiger counter.

Above See in the dark with a night vision camera, perfect for a midnight feast

Left I ain't 'fraid of no ghost! Scan for paranormal presences with this Ghost Detector

FITNESS TRAINER



Above HIIT Pi: each video frame is analysed using the PoseNet model in TensorFlow Lite to detect and track the user's pose

A positively healthy use for a Raspberry Pi camera is to monitor your body movements while you're exercising. With the use of computer vision, you can then analyse how well you are doing during your workouts.

This is especially important if you're engaging in high-intensity interval training (HIIT) on your own, to make sure you're doing it right for optimum results. Using a Raspberry Pi aided by Google Coral USB Accelerator, James Wong created a machine learning Raspberry web app to track his movements and poses and then score them based on a set of pre-defined standards. Find code and instructions on his GitHub repo: **hsmag.cc/HIITPi**.

If you prefer your exercise to be a little less intense, Salma Mayorquin and Terry Rodriguez's YogAI (**hsmag.cc/YogAI**) combines a Raspberry Pi magic mirror with AI yoga pose recognition to help guide you.

ASTROPHOTOGRAPHY

Due to its compact size, a Raspberry Pi Camera Module is ideal for using with a telescope to take images of the night sky. David Palmer's PiPiece project (**hsmag.cc/PiPiece**) is based around a 3D-printable case for a Raspberry Pi Model A and Camera Module with an eyepiece tube design available in various sizes to fit most amateur astronomy telescopes.

The project does involve removing the tiny lens of the Camera Module, which should be done carefully to avoid damage to the sensor. Once it's fitted to the telescope, you can adjust the latter's eyepiece to focus, as you would normally. The field of view is small, so fine alignment of the telescope is required to observe a planet or other celestial body.

Alternatively, if you lack a telescope, a Raspberry Pi HQ Camera fitted with a suitable zoom lens should enable you to take detailed shots of the moon – it requires good aperture selection and careful focus. For more on astrophotography, check out the feature in The MagPi issue 128 (**hsmag.cc/MagPi128**).

Above The PiPiece attached to a telescope, enabling the Camera Module to capture the night sky

VISION AID

For blind and partially sighted people, a Raspberry Pi and camera can provide a way of seeing by analysing the view and turning it into audio. Inspired by a blind cousin, Robert H'obbes' Zakon built a Seeing Wand that can speak the name of whatever it's pointed at. When the user presses the button on the prototype device, the Camera Module attached to its Raspberry Pi Zero takes a photo; this is sent to Microsoft's Cognitive Services Computer Vision API to obtain a description, which is then spoken using the eSpeak text-to-speech tool. Find the build details at **hsmag.cc/SeeingWand**.

Alternatively, the All-Seeing Pi (**hsmag.cc/AllSeeingPi**) is worn on a partially sighted user's head and sends a video feed from the camera to a display right in front of their eyes to give them a more detailed, close-up view of the environment.

Above It may look a little makeshift, but the Seeing Wand is easy to build and the code is on GitHub: hsmag.cc/SeeingWandGH

HackSpace
TECHNOLOGY IN YOUR HANDS

REVIEWS

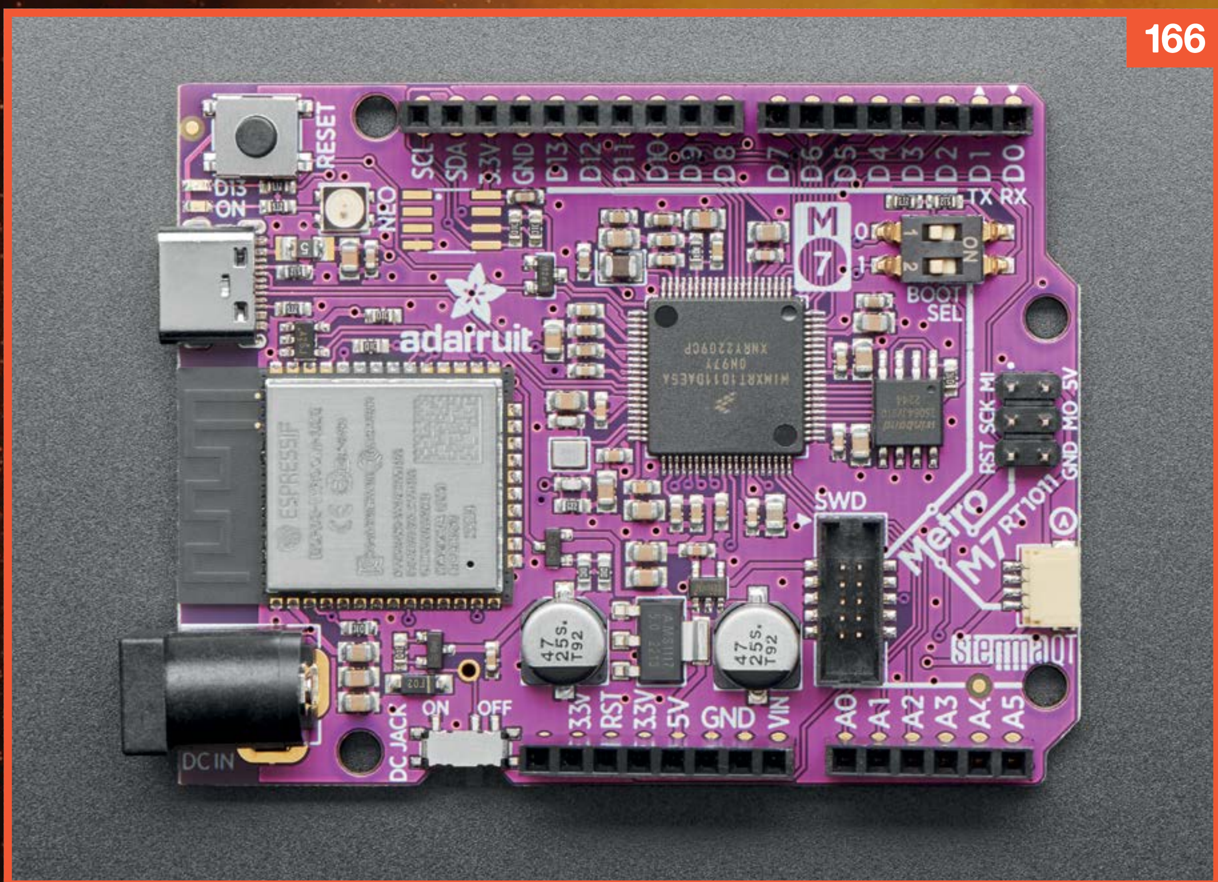
HACK | MAKE | BUILD | CREATE

Don't be shy – tell Santa what you want for Christmas

PG 166

ADAFRUIT METRO M7

Enormous processing power on a sensible budget



166

PG 168

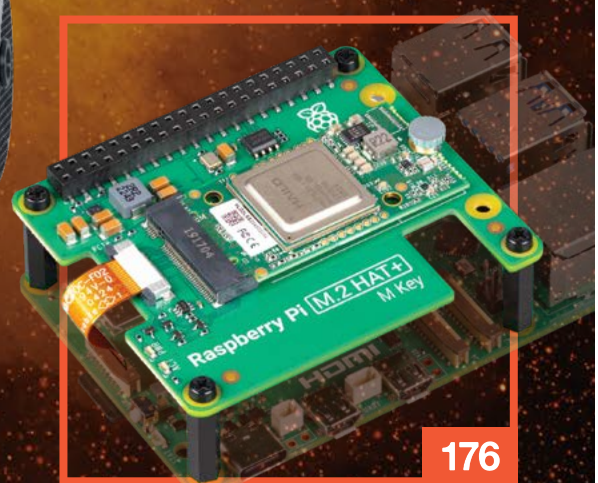
XTOOL LASER CUTTER

Lasers aren't just for industrial use: we could do with one of these at home

PG 176

RASPBERRY PI AI KIT

Give your project brains with an artificial intelligence add-on board



PG 170

PRUSA XL

The fastest, most efficient way to 3D print in multiple filaments – you'll love it!

REVIEW

Metro M7 with Airlift

Who needs efficient code when you've got more clock cycles?

ADAFRUIT ◈ **$29.95** | **hsmag.cc/metrom7**

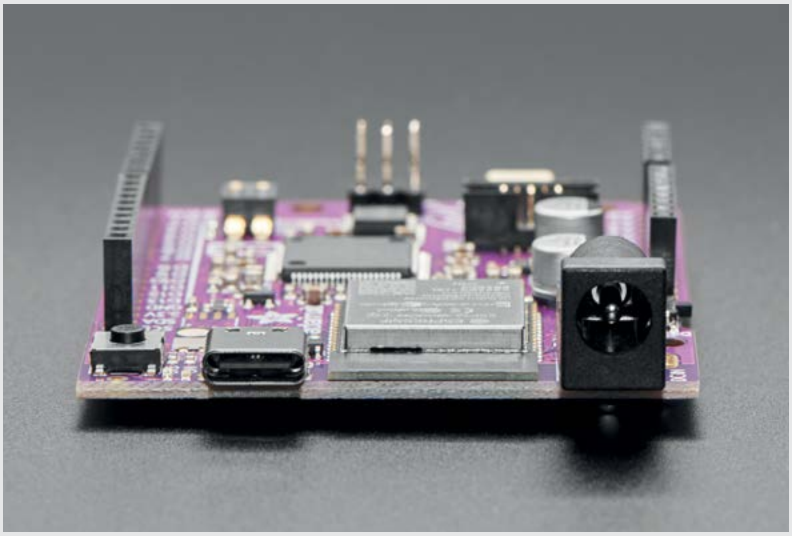
By Ben Everard @ben_everard

The NXP iMX RT1011 that sits at the heart of the Metro M7 is, frankly, a ridiculously powerful microcontroller. It's based on the Arm Cortex-M7 core and runs at 500MHz.

Twenty-two of the microcontroller's GPIO pins are broken out in the classic Uno (what Adafruit calls Metro) style. This means that there's already an ecosystem of shields that can go on top to provide additional hardware, though the majority of these shields come with support for the Arduino programming language rather than CircuitPython, and many are 5V, while this board runs at 3V. If you're planning on using this with a third-party shield, make sure they will work together, not just physically fit.

Hardware doesn't have to be slotted on top, though. The M7 Metro also comes equipped with a Qwiic port for attaching I2C hardware – there's a huge range available from Adafruit and other suppliers.

Below ◈
The ESP32 Wi-Fi module includes an on-board antenna



Alongside the main powerful microcontroller, there's a second microcontroller – an ESP32 that's used for wireless networking. In theory, this can do both Wi-Fi and Bluetooth Low Energy, but at the moment, there's only support for Wi-Fi. Adafruit calls this setup, using a secondary ESP32, Airlift.

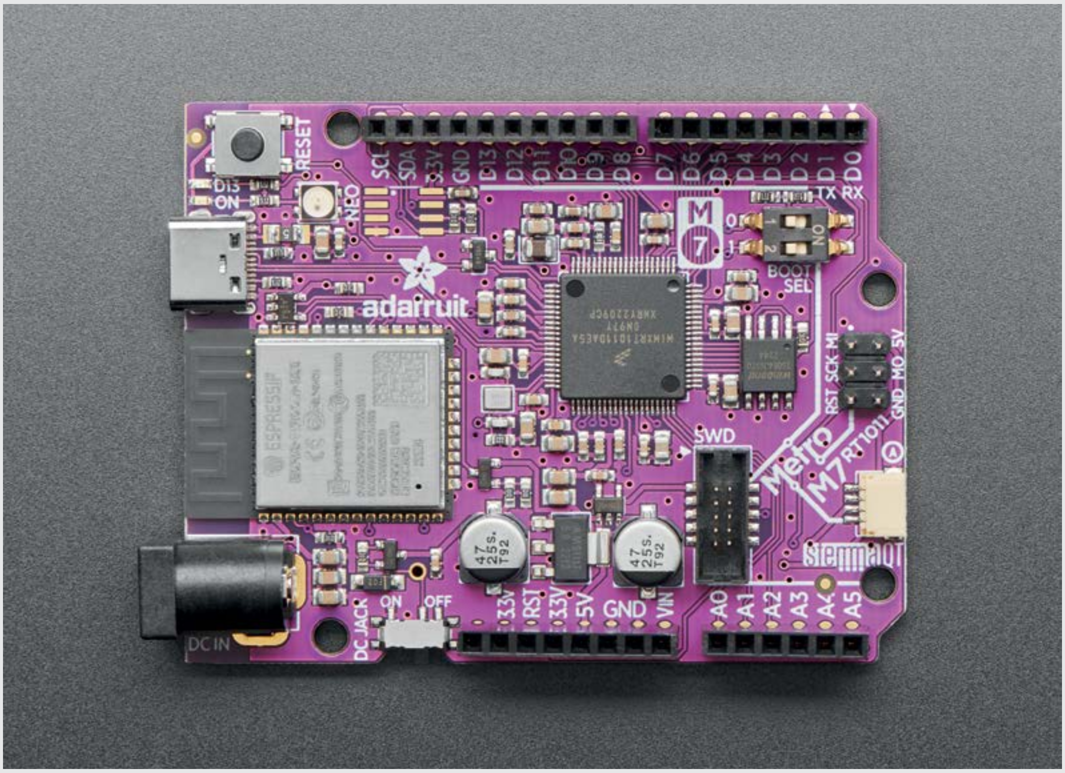
The Airlift networking setup offloads most of the work onto the secondary microcontroller. This means that your main processor isn't burdened with the various issues of keeping connected and shuffling data in and out. That's perhaps less of an issue on this beast of a processor than on some others, but it does mean that your code's performance should be far more predictable.

We tested the Metro M7 Airlift that includes wireless connectivity and costs $29.95, but there's also a version without wireless (and with an SD card port) that comes in at $19.95.

Perhaps the most unusual thing about this board is that – unlike almost all of Adafruit's other boards – you can't use it with the Arduino IDE. You can program it with CircuitPython or the MCUXpresso IDE created by NXP (the microcontroller's designers). For most people, that's likely to mean that this is a CircuitPython board.

IN USE

We tested this out with some audio code. Not so long ago, we were pretty happy if we could make a microcontroller go beep while also doing something else. With this, we were able to play ten WAV files and dynamically adjust the volume simultaneously, and make it Wi-Fi accessible. What's more, we were able to do all of this in Python. Some of this is, of course, down to improvements in hobbyist microcontroller software over the years, but it's also due to the fact that this is almost as powerful as the PC we used to use to program microcontrollers.



Left With two powerful processors, there's a lot packed onto this board

We also speed-tested the Metro M7 against the Adafruit Grand Central M4 Express and the Metro ESP32-S2. These are two of the fastest CircuitPython boards from Adafruit, running an Arm Cortex-M4 at 120MHz, and a 240MHz Tensilica core, respectively.

We found the M7's performance to be about five to six times faster than the M4 across a range of different areas, including integer and floating-point maths. This is down to both the higher clock speed and the fact that the M7 core can do more computation in each clock cycle. When compared to the ESP32-S2, performance was a bit more varied, but the M7 always came out on top. GPIO access and floating-point arithmetic was about twice as fast, and integer arithmetic was about 4.5 times the speed.

You might think that more computing power is always a good thing, but it does have a drawback. It needs more electrical power to keep it running. Given modern batteries, this is less of a problem than it used to be, but if you need something to run off-grid, you probably want to think a bit about whether you really need this amount of processor power.

The Arm Cortex-M7 is a powerful microcontroller core, but the Metro M7 Airlift isn't the only high-speed Arm Cortex-M7 board, so it's not just a matter of choosing a fast microcontroller – it's a question of whether you want *this* M7 microcontroller. It's reasonably chunky, but whether this is a plus or minus is down to your particular project. Given that this isn't compatible with the Arduino IDE, and that the Uno hasn't been the dominant form factor for microcontroller add-ons for over half a decade, it's unlikely that this form factor is going to be important to you. That said, we're quite fond of this size. It's not too fiddly to work with, but still small enough to fit most spaces, and we prefer socket headers to the more popular pin headers. The Metro M7 Airlift is the only Wi-Fi-enabled M7 board that we're aware of, so if you need both oodles of power and network connectivity, then this is a good choice. CircuitPython support is great, as you would expect of a board from Adafruit.

"Having power to spare can make the build go a bit smoother and lets you worry about optimisation later"

This is the sort of board we like to use when prototyping projects. We might not need the raw performance or the dedicated networking hardware in the final build, but it's good to have it there while testing everything out and getting it all working. Yes, this is ridiculously powerful for a microcontroller, and yes, few of your projects really need this much grunt, but having power to spare can make the build go a bit smoother and lets you worry about optimisation later. □

VERDICT

A powerful board with Wi-Fi and great CircuitPython support.

10/10

xTool S1

A little laser with a lot of features

XTOOL ◆ **£1799** | **hsmag.cc/xtools1**

By **Ben Everard**

Laser cutters are great tools for makers. They let you cut shapes out of – or engrave images into – sheet material quickly and accurately. By piecing together different shapes, you can build up 3D objects much faster than you can with a 3D printer.

If you're after a laser cutter, there are a few things to consider. Let's take a look at some of the key things you have to think about, and take a look at how the xTool S1 stacks up.

The first big decision to make is safety. Many budget machines aimed at makers are unenclosed. That means there is nothing to stop the laser beam from being reflected back towards the operator. Exposed lasers are dangerous. Very dangerous. A beam powerful enough to cut wood is easily powerful enough to blind you instantly. Yes, you can wear laser goggles, but this isn't a good solution, and when your sight (and the sight of anyone else who might happen to open a door) is at stake, then we want the best possible protection.

Below ◆
The enclosure keeps fumes and laser beams contained

No tool is perfectly safe, but the xTool S1 comes packed with a lot of safety features. It's fully enclosed, so it should be impossible for any lasers to leak out. If you open the door while it's in use, it'll automatically stop.

FLAME OFF

Another significant safety risk when using a laser cutter is fire. The S1 has a flame detector which will stop it if what you're cutting catches fire. You can also plumb in a fire extinguishing system, but we didn't have a suitable system to test this with. Finally, you need to consider the fumes given off by the material as it burns or vaporises away. Being fully enclosed, the S1 catches the fumes but it will need to vent them. There's a hose that you can poke out of a window or plumb into an extraction system. If this isn't possible, an air filter is available separately.

Once you've looked at safety, you might want to consider power. Laser power isn't completely comparable between different systems: a diode laser will cut materials differently to a CO_2 laser of the same wattage; different wavelengths of light are absorbed by materials differently; and a laser cutter that requires mirrors will lose some power as the light is reflected.

The xTool S1 has a choice of three different power options: 2W infrared laser, and 20W and 40W blue laser modules are available. We tested the 20W version and found that it worked well with wood (both MDF and construction plywood) up to about 10mm

thick, and worked excellently with laser ply. This cutting power is comparable with CO_2 lasers that we've used. For wood and plastics, the S1 should serve hobbyists well.

Perhaps the biggest selling point of the S1 is its ease of use, and on this, it really is excellent. There's autofocusing (with adjustable depth offset for cutting through thick materials) and a system for locating your objects on the print bed. Both of these work in slightly unusual ways. The autofocusing works with a rod attached to the side of the laser. This then probes the surface. Once it's probed, the print head whizzes off to the side to push it up and out of the way so it doesn't interfere with any of the engraving. Should the probe hit something, it's magnetically attached, so it can pop off without damage.

LASER LOCATOR

The system for locating your items uses a visible laser that's in the same place as the laser cutter. You physically place the laser head in one point, then press the button to mark the point, then move it to another point, press the button, and so on until you've marked out the necessary points. The software then converts these points to a rectangle, circle, polygon, or other shape that lets you place your work in the correct position relative to this shape.

All of this needs software to support it, and in the case of the S1, that's xTool Creative Space.

Our only gripe with the software is that it needs an internet connection to work. This is the case even when connected to the laser cutter over USB (you can connect over Wi-Fi). We can't see why this is the case, and it does make us concerned about what would happen if xTool went out of business, or shut down this particular service.

xTool Creative Space has a few features, both useful and odd. It can automatically generate QR codes for you. It also has an AI system for generating artwork that we're still getting our heads around. On this, you have to pay for each set of generated images using credits. You get a set of credits when you create your account, but after that, you have to generate more by sharing projects on the xTool website.

The S1 is also compatible with LightBurn, but not all of the features are available in this software.

Overall, we've been really impressed with the xTool S1. However, there are two areas that are potential pitfalls. The first is size. It can only cut 498 × 319 mm. You can fit material up to 600 × 400 mm, which might be useful if you don't have a good saw for cutting up sheet material, but you won't be able to cut against the edge of the sheet which, depending on what you're cutting, might mean material waste. The other thing we can't ignore is the price. The setup we tested cost £1799. You can get laser cutters cheaper than this, and you can get bigger, more powerful laser cutters for around the same price. However, given the full range of what's on offer with this machine – the safety features, ease of use, and the power – we think it represents excellent value. □

Above Left
This pin is used to autofocus the laser

Above Right
You can quickly and easily swap between different laser modules

VERDICT

An easy-to-use, fully enclosed laser cutter at a reasonable price.

9 /10

Prusa XL

A big printer with big potential

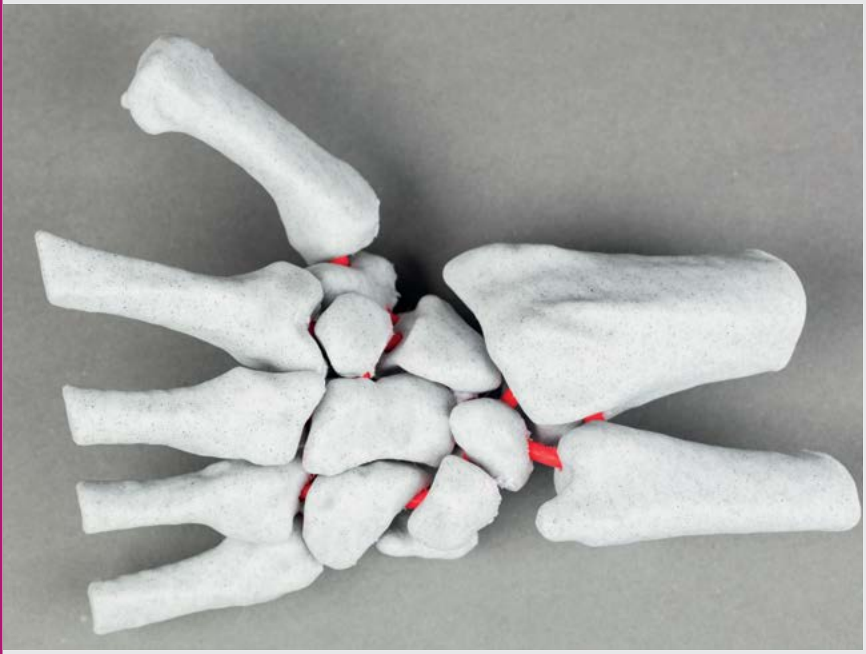
PRUSA RESEARCH ◈ £1798.80 | prusa3d.com

By **Ben Everard**

The XL is Prusa's first large format core XY printer, and this alone would be enough to make this a special printer. However, the XL has a much bigger feature tucked away at the back of the bed: a five-tool tool changer.

Let's take a step back and look at what a tool changer is in general, before looking at this printer in more detail.

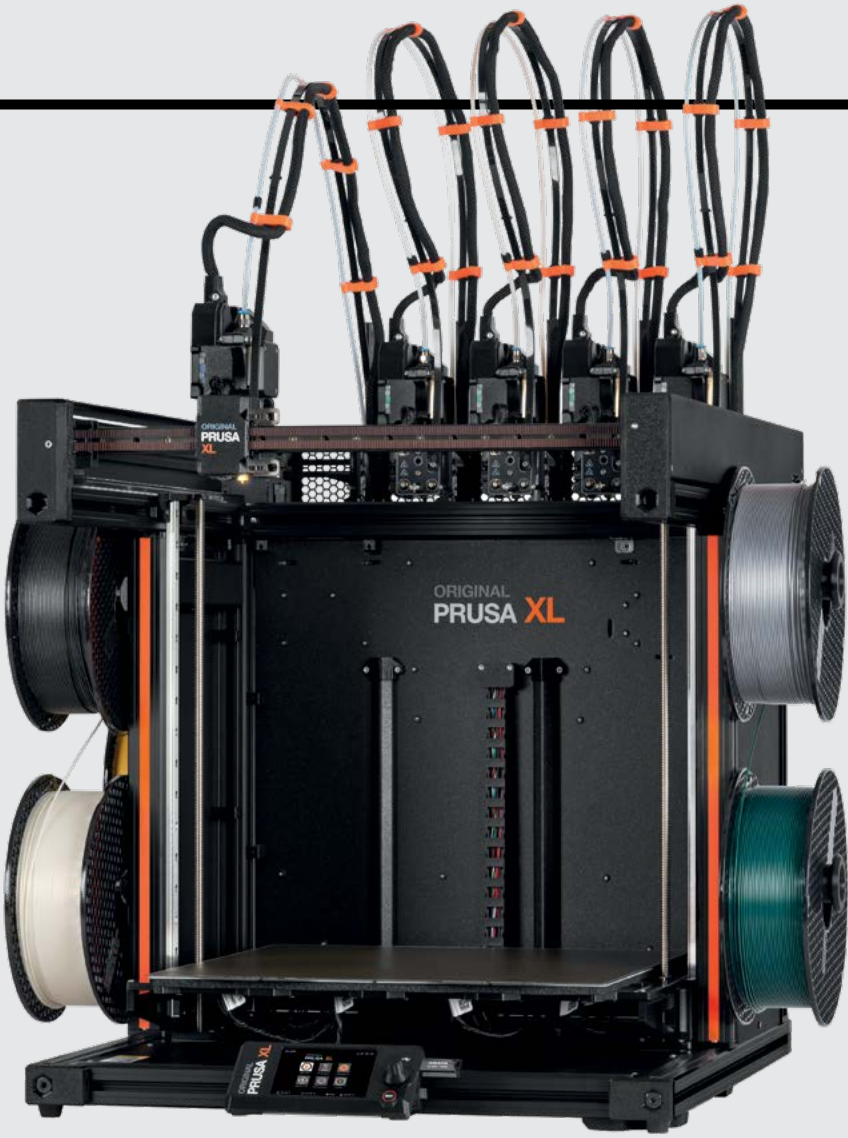
Below ◈
This print is based on a CT scan of a wrist. The bones are printed in PLA and they're joined by sections of flexible filament. The result is a print that moves like a real wrist



The vast majority of multicolour 3D printers work by loading and unloading filament into one nozzle. This works to an extent, but it has a few limitations: it's slow, wastes a lot of filament, and only works well if all the filament is the same type of plastic. You can also get printers with more than one extruder that moves about the bed on the gantry. Sometimes these extruders are fixed together and sometimes they can move independently, in which case they're known as independent dual extruder or IDEX. These can print different filaments but, again, they have limitations. In this case, it's that you're mostly limited to two extruders because otherwise it gets very crowded. The final option for multi-material printing is tool changers. With these, there are multiple extruders, but only one extruder is engaged by the moving gantry at a time – the others are parked out of the way. The gantry should be able to quickly and automatically change to a different extruder. On paper, these are the most powerful option because they can print more different filaments faster than the alternatives. However, historically, they've been the least common because they are significantly more complex than the other options.

Let's get back to the Prusa XL, then. You can load it with up to five extruders (also known as tool heads).

One problem testing out this printer is that it has capabilities that no other printer on the market has, so you can get caught between two problems. Firstly, it's very easy to get caught up in hype about things that later turn out to be just gimmicks. And



secondly, it can be hard to see the full value in features until a wide range of people have had the chance to flex their creative muscles and really find great uses for them. We'll do our best to steer a path between these two issues.

The five extruders can each be loaded up with a different filament. These can be different colours of the same filament, in which case they can print multicolour objects in a very similar way to other multicolour printers. However, it can do more. Because each extruder is entirely separate, you can use different types of plastic. There are some limitations – for example, the print bed will be the same temperature, which might cause a problem for some mixtures of filament.

The XL can mix rigid plastics, such as PETG, with softer plastics like TPU. We got a couple of test models from Prusa with which to try this out. One was a print-in-place box that included a flexible, water-resistant seal. The other was a CT scan of someone's wrist with the bones printed in PLA and the bones joined with flexible tendon-like rods printed in TPE. Both of these prints printed excellently, and have a wow factor that, honestly, we've not seen in 3D prints for a long time.

> **It is probably the most expensive printer** we'd still consider fitting in the 'prosumer' hobbyist category

While the test prints designed by Prusa are undeniably impressive, we wanted to see how hard it was to conceive and design something ourselves that made use of this mixture of materials. The two things that we came up with are a keyboard support with integrated rubber feet, and a phone case. Both turned out well, though we would like some more powerful tools in PrusaSlicer for placing filament. →

Above You need quite a bit of desk space for the XL – the spools mount on the sides, and you need to be able to reach the top of the printer to access the tool heads

> **The Prusa XL can print our multicolour cube about** eight times faster than the Bambu Lab X1-Carbon

Above Multicolour prints use far less material than traditional colour-changers

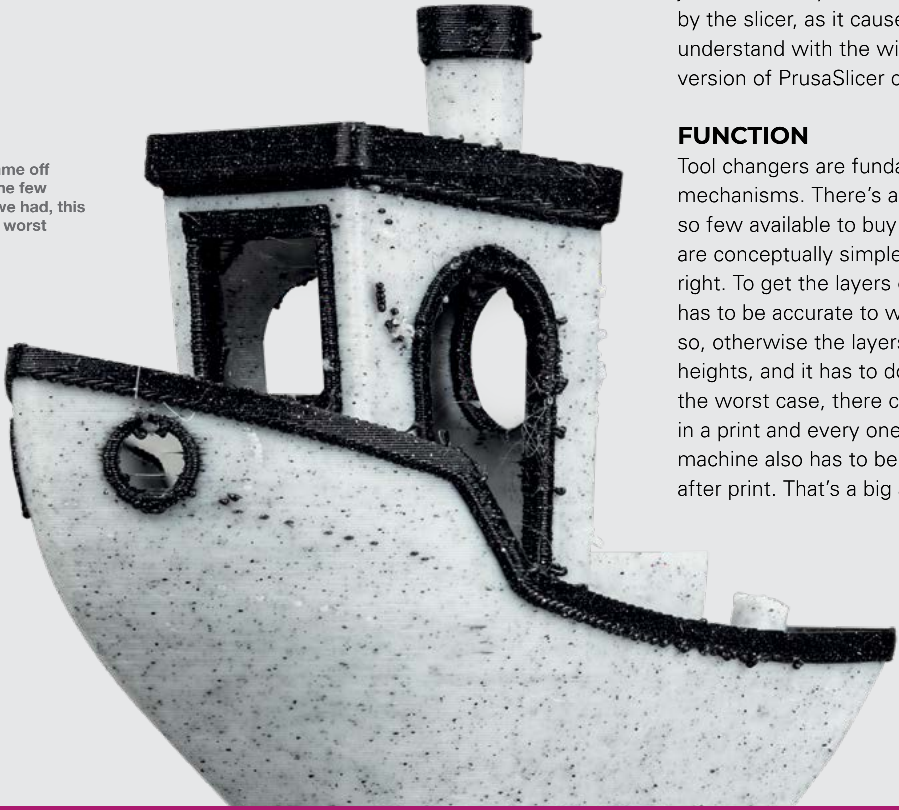
The keyboard support could, we'll be honest, have been a print entirely in rigid plastic with rubber feet glued on. This would have worked perfectly well, but we wouldn't have been able to get feet in the shape of a lizard this way.

The phone case is another matter. We've tried 3D-printed phone cases a few times on other printers and never had much success. We've found that any filament that is flexible enough to provide some impact resistance is also stretchy enough to fall off the phone. We combined rigid and flexible material so that there was a sturdy frame around which we put soft flexible filament (40D on the Shore hardness scale). The result is a case that fits securely and snugly, while still providing a lot of impact resistance. We think there's still a little way to go before we've nailed this case design, but it's already by far the best 3D-printed phone case that we have had.

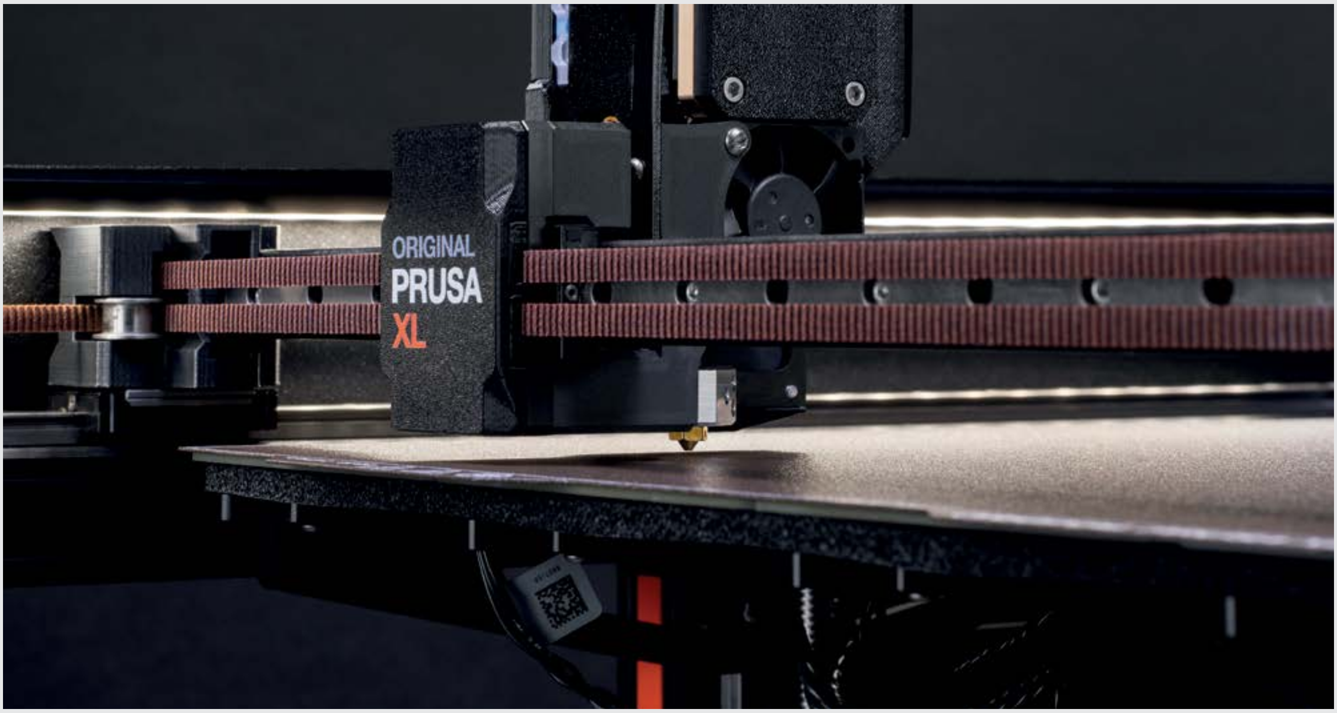
Another option is to mix different materials for supports and the main print. This can be as simple as using a cheaper material for supports – or more advanced like using two plastics that don't stick, to create supports that are easier to remove (such as PETG and PLA). There's even the option of using a soluble material such as PVA for supports that just wash away. This isn't currently well-supported by the slicer, as it causes a problem we don't fully understand with the wipe tower. Hopefully, a future version of PrusaSlicer can help with this.

FUNCTION

Tool changers are fundamentally complex mechanisms. There's a reason that there have been so few available to buy over the years. While they are conceptually simple, there is a lot of detail to get right. To get the layers consistent, each tool change has to be accurate to within a tenth of a millimetre or so, otherwise the layers will be at slightly different heights, and it has to do this over and over again. In the worst case, there could be 20,000 tool changes in a print and every one has to be spot on. The machine also has to be able to keep doing it print after print. That's a big ask.



Right Most prints came off clean, but of the few messy prints we had, this was by far the worst



Above The Core XY motion system has two drive belts, both of which are needed to move the print head in either the X or Y direction

Initially, we did have a little difficulty with the different extruders having slightly mismatched Z-heights, which resulted in some filaments being too squashed into the print bed on the first layer, and others being too far away. We followed the calibration guide and still had the problem. We solved it by both lubricating the tool changer, and unloading the filament before doing the tool head calibration.

The only other issue we've had is getting some large blobs of filament protruding above the top of the wipe tower. This seems to happen when the extruder primes but doesn't move, so squirts a blob in place. The next time it uses the wipe tower, it then has to bump over this blob. It makes a bit of a clatter as it does so. At first, we were so paranoid about this that we kept checking on the printer and snipping off any big blobs with a pair of side cutters. However, after a while, we just got used to them being there. It looks pretty horrible, but it's on the wipe tower and doesn't seem to cause any problems.

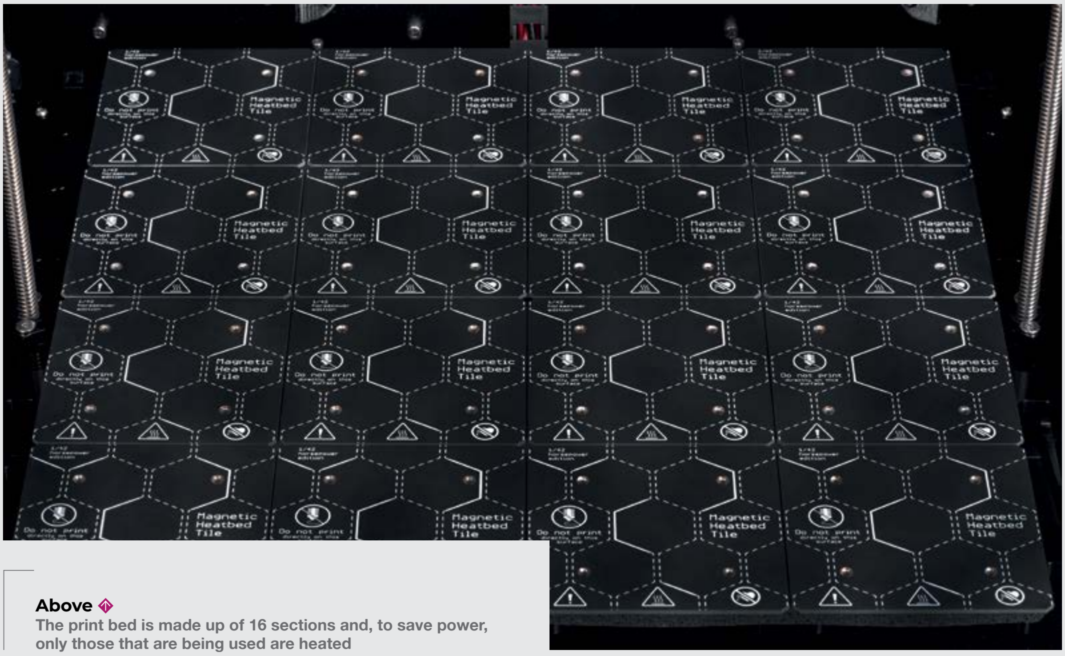
SPEED

It felt like 2023 was the year that 3D printing became fast. Almost every 3D printer was judged on its ability to create a Benchy in minimal time. The XL is actually a really hard printer to judge on speed. When it comes to straight one-filament speed, it's reasonable, but not spectacular. There's a lot working against it. Frames with larger print volumes inevitably have less rigidity, and the tool changer both increases the mass of the print head and places it further away from the axis. The end result is a single-colour speed that's slightly slower than the MK4.

A Benchy is probably about the worst case for speed on the Prusa XL, as it contains a lot of small moves that benefit smaller machines that won't flex as much under high accelerations. The best case is something that has a lot of colours on every layer because the tool changer can switch filaments far quicker than most other printers.

For an example of this, we took a one-inch cube and coloured each face differently (other than the top and bottom). The Prusa MK4 isn't yet compatible with the colour changer (though it is in the works), so we'll use the Bambu Lab X1-Carbon as a comparison. This printer is no slouch and is probably the fastest off-the-shelf hobbyist printer. However, the XL can print our multicolour cube over eight times faster than the X1 with the same layer height. This is a phenomenal speed-up, but even this is understating it – we limited it to four colours because that's as many as the X1 can do. The XL can do an additional colour.

The only honest answer we can give about speed on the XL, compared to other fast high-end →



Above
The print bed is made up of 16 sections and, to save power, only those that are being used are heated

hobbyist machines, is that it's somewhere between slightly slower and eight or more times faster, depending on what you want to create. Therefore, if you want a printer for speed alone, it's really important to look at the actual models you intend to print, not just isolated benchmarks cooked up by reviewers. However, we can say that if you do plan on printing with multiple colours, you are likely to get a significant speed-up, particularly if there are a lot of colours on each layer.

Another significant speed-related issue is the ability to print lots of things at the same time. While the time to print one Benchy on the Prusa XL is about the same as printing it on the MK4, you can fit nearly three times as many on the print plate. This might seem a bit surprising, because a 36 by 36 cm print bed doesn't seem that much larger than a 25 by 21 cm print bed, but it really is. The speed-up here isn't from the printer taking less time per Benchy, but from you, the user, not having to come and take things off the print bed as often.

Obviously, this does come with the risk of one failed object on the print bed damaging all the others, but you can cancel an individual item within the print without having to cancel the entire lot.

QUALITY

The quality of prints we've had from the Prusa XL is good, but not perfect. It's not uncommon for multicolour prints to get a bit of the wrong colour filament in the wrong place. This is due to a bit of stringing, or a blob from the wipe tower, being picked up by the nozzle. It doesn't happen much, and most of the time it's easy to clean up.

SOFTWARE SUPPORT

Overall, we'd say that the Prusa XL is as well-supported by the slicing software as most existing printers. However, the tool changer opens up some whole new possibilities that we're only just getting our heads around. For example, embedding text or a logo in one colour onto a surface of another colour looks great, and you can do this in PrusaSlicer, but it's a bit clunky as it involves creating a new part and merging them. It'd be far easier if you could just do this in the same way you can emboss text or SVGs onto a part.

We also had an issue in one print where we used the colour paint tool to apply colour to the surface of an object but, on a curved surface, the colour didn't go deep enough and the underneath colour shone through a light-coloured upper layer.

Another minor problem encountered was when an object is made of multiple different parts in CAD and then merged, it can become impossible to properly select some of the parts to paint them.

In all these cases, the problem can be solved by going back to the original CAD files and manipulating them there before going into the slicer. However, we've gradually moved away from CAD to the

PrusaSlicer as it has been upgraded with more and more powerful tools for manipulating 3D shapes, and we'd like to keep moving in this direction.

These are all relatively minor issues, but we'd hope that the colour and filament selection options in PrusaSlicer get some attention soon, as it feels like there are some improvements that could really help people get to grips with this printer.

WASTE

We can't ignore the fact that 3D printers use plastic, and plastic pollution is a major source of pollution for the planet. Despite some claims to the contrary, PLA is not biodegradable in any realistic way. It is possible to recycle it (and other 3D printer filaments), but it's not particularly easy and few recycling centres will accept it. That doesn't mean we can't enjoy 3D printing as a hobby, but we need to think about our impact on the world.

The Prusa XL uses dramatically less plastic when printing in multiple colours than traditional multicolour printers. The exact amount will depend on the model, but it can easily end up using less than a quarter of the filament, especially for small models and those with lots of colours. Potentially the savings are even greater if you don't use the wipe tower, (though this option relies on having very dry filament). Not only does this save plastic, but it also saves you money. A heavy user of multi-material prints could easily find that the XL is actually a cheaper option compared to a classic multicolour printer, when considering the saving in filament over the life of the printer.

> **We did have a little difficulty with the** different extruders having slightly mismatched Z-heights

The Prusa XL is a hard printer to review, because there's so little to compare it against, especially now the E3D tool changer has been discontinued. The large print volume and weight of the tool changer slow it down, so it's not the best for large, single-colour prints, and it is probably the most expensive printer we'd still consider fitting in the 'prosumer' hobbyist category. However, it can create prints that no other printer can. This will probably prove true in ways we can't yet predict as people get their hands on the machine and start experimenting with creative ways to use the capabilities. On top of this, for multicolour prints, it's massively faster than the competition, and the lack of waste is both an environmental win and a money saver. This isn't a printer for everyone, but for some people, it's absolutely the perfect printer. □

Above The Prusa XL has the same control interface as the Prusa MK4. We found it reliable and easy to use

VERDICT

Pushing the boundaries of what 3D printers can do.

10 /10

Raspberry Pi AI Kit

Putting the ghost back in the machine

RASPBERRY PI ◈ **$70** | **hsmag.cc/aikit**

By **Ben Everard**

In case you've missed the news, AI is poised to be the next big thing in tech. Actually, scratch that, it's already the current big thing in tech. The only slight problem is that no one can quite agree what it is.

While the latest headlines are being grabbed by large language models, including ChatGPT, which have a habit of lying to users and writing uncompilable code, AI models have been quietly working away in the background. They generate captions for our videos, help us take better photographs, help scientists identify things in photographs, improve quality control in factories, and generally help make our lives progress a little smoother. The neural networks underpinning these are running everywhere, from server rooms to the phones in our pockets.

Neural networks have two stages – first, they must be trained. This is where you define the structure of the network, and run training data through it (typically large amounts of training data). While a lot depends on the particulars of the model you're training, this usually takes a huge amount of computing power and is only done rarely. In fact, the majority of people using AI don't train their own models. Instead, they use pretrained models that are available from a variety of sources (there's a wide range of models for the Hailo-8L – the accelerator at the heart of the AI Kit – available at **hsmag.cc/Hailo8L**).

Once you have a model, you can then run it – this is where you use it to analyse real-world data. Running a model takes a much more modest amount of computing power, and it's this that the Raspberry Pi AI Kit is designed to do.

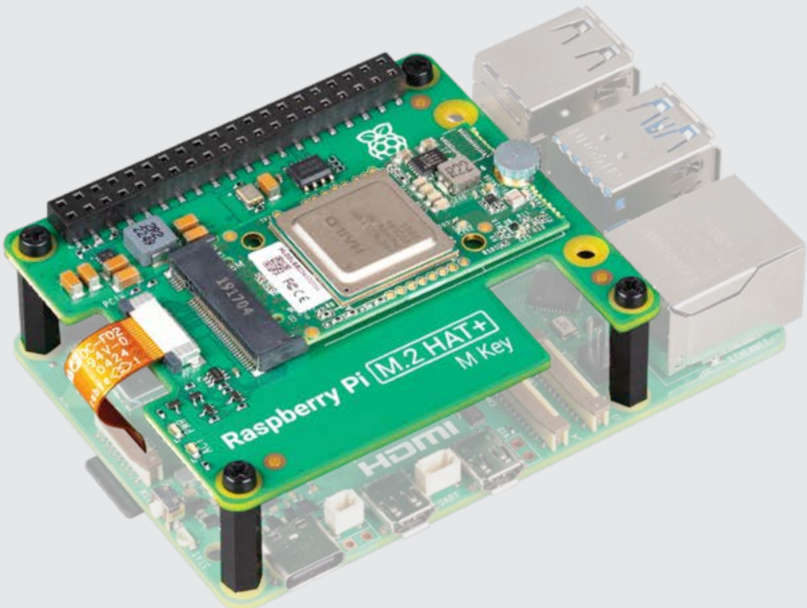
The Hailo-8L accelerator can perform 13 trillion operations per second (aka 13 TOPS – though the T in this case stands for Tera). That's obviously a big number, but to put it in context, Apple's M3 processor's neural engine can perform 18 TOPs, while its A15 SoC (from the iPhone 13) can perform 15. Meanwhile, an NVIDIA A100 GPU can perform 1248 TOPS.

Models are getting faster and more accurate all the time, so it's hard to say exactly what this is capable of, since it will probably be able to run better models in a year's time than it can now. However, to give you an idea, the YOLO models can distinguish between about 80 different types of object (person, car, bicycle, etc.), and they can run quickly in real time on the AI Kit.

Similar models can detect someone's pose. Take a look at the model zoo (in previous link) for a fuller breakdown of the different models and their performance, but broadly speaking, the sorts of models this can run can differentiate between around a hundred types of object and find them in a scene.

Just as executable files have to be compiled for the particular processor you're using, neural network models have to be compiled for the particular accelerator you're using (as well as the framework they are running in). Hailo has a Dataflow Compiler that accepts models in many common formats including TensorFlow, PyTorch, and Keras. The compiler converts these input files into HEF files that can be loaded onto the AI Kit.

That's a lot about what the Raspberry Pi AI Kit is meant to be, so let's now take a look at what it is.



Above
Everything is held together securely, so it's easy to embed this in other hardware

Inside the kit itself, you'll find an M.2 HAT+ and a Hailo-8L board. These two plug together and then into a Raspberry Pi 5 – because it connects to the PCIe port, earlier versions of the Raspberry Pi won't work. This is all detailed in the Getting Started Guide (**hsmag.cc/AIGettingStarted**).

Once the hardware is connected and the dependencies are installed, you can start on the software. While the Raspberry Pi AI Kit isn't explicitly a vision product, we suspect the vast majority of its use will be in vision. That's just the area where neural networks of the sort of size this can run are most useful.

At the moment, you can run the Hailo models within Raspberry Pi Camera apps by passing a suitable value for the --post-process-file parameter. There are also examples created by Hailo on its GitHub at **hsmag.cc/RPI5Examples**.

We suspect, though, that most people want to use the models in their own software. This is possible at the moment with Hailo's TAPPAS framework, but it should soon become far easier when support for the Picamera2 Python module is released.

Neural acceleration on small computers has lagged for a long time, so we're really excited to see development in this area. The Hailo-8L is powerful enough to let many vision processing tasks run in real time, while still leaving the CPU mostly free to do whatever other processing you need.

We've said plenty of times in this magazine that the products that excite us the most are the ones that open up new categories of project, and this is one such example.

It's not the first AI accelerator for small computers, but it's the first one we're aware of with this level of performance at a hobbyist price point, and it should really open up the field of embedded AI.

VERDICT

A new product that opens the door to many potential AI projects.

10 /10

SUBSCRIBE TODAY FOR JUST £10

Get 3 issues + FREE Pico W

+

Subscriber benefits

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 23% compared to stores

Subscribe for £10

- Free Pico W
- 3 issues of The MagPi
- £10 (UK only)

Subscribe for 6 Months

- Free Pico W
- 6 issues of The MagPi
- £30 (UK) $43 (USA)
  €43 (EU) £45 (Rest of World)

Subscribe by phone: **01293 312193**

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

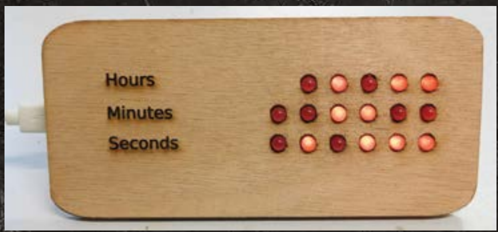
Email: **magpi@subscriptionhelpline.co.uk**

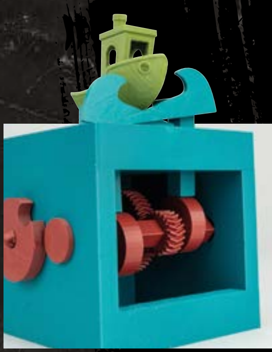
Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A free Pico W is included with a six-month subscription in USA, Europe and Rest of World.

BOOK OF MAKING 2025

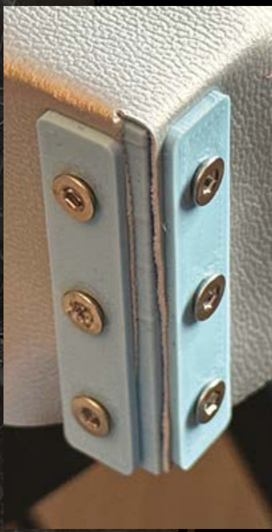
Technology doesn't have to come in a sealed box. With a bit of know-how you can start building your own DIY devices to tell you the weather, to water your house plants, to make music – anything you put your mind to. Whether you're already comfortable with microcontrollers or you're just getting started, there's a project in here for you – get ready to build a better world!







180 PAGES OF HACKING & MAKING




- Step into the wonderful world of making
- Be inspired by amazing community projects
- Get your hands dirty with 3D printing, microcontrollers, and more
- Discover projects that you can do in an hour, afternoon, or weekend