

MAKE | BUILD | HACK | CREATE

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc

July 2018

Issue #08

ARDUINO

THE NEXT GENERATION

WIFI UNO

IDE 2.0



Plus

▶ **The Open Space Agency**
Explore the cosmos from home

▶ **Restore your watch**
Spruce up your wrist bling

▶ **Get started with AI**
Make your hardware adapt to your habits

EXCLUSIVE
ACCESS ALL AREAS

Discover the new boards



GLASS BLOWING

Learn the ancient art of inflating baubles

Jul. 2018
Issue #08 £6



NOISY TOYS | TAPE MEASURES | CRICKIT | AUDIO MODULES

CanaKit Raspberry Pi 3 Ultimate Starter Kit

Model B | 1 GB RAM | 1.2 GHz | Quad-Core CPU



- > Learn to Code
- > Explore Computing
- > Get started with Electronics

KIT INCLUDES RASPBERRY PI 3 AND ...

PREMIUM CASE & HEAT SINKS



2.5A POWER ADAPTER



32 GB CLASS 10 MICROSD CARD



PRE-LOADED WITH OPERATING SYSTEM

USB MICROSD CARD READER



PREMIUM HDMI CABLE



QUICK-START GUIDE



GPIO TO BREADBOARD INTERFACE BOARD



RIBBON CABLE



FULL-SIZE BREADBOARD



JUMPERS



MALE TO MALE & MALE TO FEMALE

LEDs



RESISTORS & PUSH-BUTTONS



Available for worldwide shipping at:

WWW.CANAKIT.COM

Raspberry Pi Zero W
Available at CanaKit!





Welcome to HackSpace magazine

Like many of you, my first introduction to physical computing came courtesy of Arduino. It wasn't much, admittedly – I played with a series of LEDs, getting them to change configurations as I pressed a few buttons. However, that was, for me, the first time my software had broken away from the screen. Suddenly I wasn't limited to 'a computer', and could program anything, just attach an Arduino and whatever hardware I wanted, and anything could be controlled with code.

On a personal level, their new products will make some projects I'm building easier, and I'm excited to get some boards, and get stuck in

It's been a rocky few years for Arduino, where legal issues have overshadowed their technical achievements,

but it's fantastic to see these problems all sorted and the organisation getting back to doing what they do best – making exciting hardware and software.

On a personal level, their new products will make some projects I'm building easier, and I'm excited to get some boards, and get stuck in. Maybe they can help you too. Head to page 32 for a rundown of what's coming out.

BEN EVERARD

Editor ben.everard@raspberrypi.org

GET IN TOUCH

hackspace@raspberrypi.org

[hackspacemag](#)

[hackspacemag](#)

ONLINE

hsmag.cc



EDITORIAL

Editor

Ben Everard

ben.everard@raspberrypi.org

Features Editor

Andrew Gregory

andrew.gregory@raspberrypi.org

Sub Editors

Nicola King, Jem Roberts

DESIGN

Critical Media

criticalmedia.co.uk

Head of Design

Dougal Matthews

Designer

Lee Allen

Photography

Brian O'Halloran, Paolo Tangari

CONTRIBUTORS

Lucy Rogers, Andrew Huang, Matt Bradshaw, Mayank Sharma, Cameron Norris, John Parks, Graham Morrison, Sophy Wong, Dave Astels, Zack Akil, Ricardo Caja, Cameron Fraiser, Marc de Vinck, Richard Smedley

PUBLISHING

Publishing Director

Russell Barnes

russell@raspberrypi.org

DISTRIBUTION

Seymour Distribution Ltd

2 East Poultry Ave,

London EC1A 9PT

+44 (0)207 429 4000

SUBSCRIPTIONS

Select Publisher Services

Ltd, PO Box 6337, BH1 9EH

+44 (0)1202 586 848

Man Enterprises Ltd,

Unit E, Brocks Business

Centre, CB9 8QP

hsmag.cc/subscribe



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

HackSpace magazine is published by Raspberry Pi (Trading) Ltd., Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0). ISSN: 2515-5148.

Contents

50



06

SPARK

- 06 Top Projects**
The Rijksmuseum of DIY projects
- 16 Objet 3d'art**
Function, meet form; form, function
- 18 Manchester Robot Orchestra**
Meet the new Ed Sheeran, with extra hard drives
- 22 Columns**
Streamline life like Jeff Goldblum in Jurassic Park
- 24 Letters**
A PhD researcher needs your help. So help her!
- 25 Kickstarting**
Transport for the kids, hackable by the kids
- 26 Hackspace Artisan's Asylum**
This is what maker heaven looks like

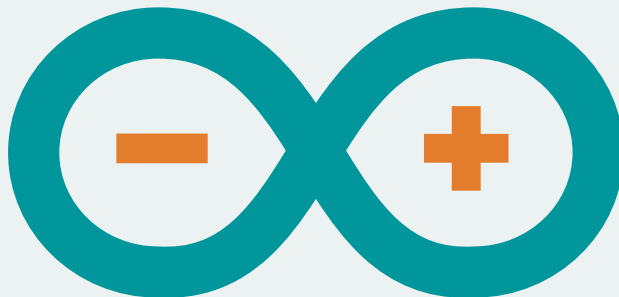
31

LENS

- 32 Arduino: the Next Generation**
New boss, new gear, new software. Forza Arduino!
- 46 How I Made: Guitar Synth**
Make sweet music inspired by our favourite Iclander
- 50 Open Space Agency**
Seek out new life and new civilisations, with DIY gear
- 54 Interview: Noisy Toys**
On analogue audio and nasal integrity
- 64 Improviser's Toolbox** Measuring tape
Home projects, made-to-measure
- 68 HackSpace learns: Glass-blowing**
We try not to burn ourselves on hot silica. And succeed!

Cover Feature

ARDUINO



The world's favourite maker board is getting a boost – find out how, when, and what you can do with the new stuff

32

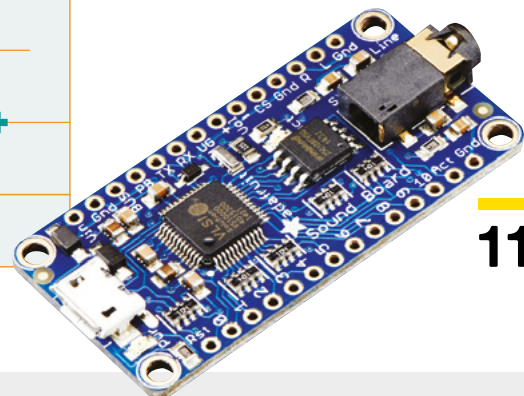
Tutorial

Dry ice ice cream



104

Science is fun, dangerous, and full of saturated fat flavour



118

Direct from Shenzhen

LED panel



116

All the colours of the rainbow, in one cheap package

75

FORGE

- 76 SoM Tap your own screws**
Add thread to make bespoke parts for your builds
- 78 SoM Arduino**
Go beyond 7-segments to add a proper display
- 84 SoM Reciprocating saw**
When you want to destroy, accept no substitute
- 88 Tutorial Wearables**
Match accessories to the world around you
- 94 Tutorial Z80**
Program your retro computer with assembly
- 100 Tutorial Machine learning**
Teach an Arduino to make decisions for you
- 104 Tutorial Dry ice ice cream**
Tasty treats made with solid carbon dioxide
- 108 Tutorial Standalone Arduino**
By-pass the board and program the chip directly
- 110 Tutorial Watch renovation**
Treat your timepiece to a bit of tender loving care

Interview

Noisy Toys



54

Carrots, dogs, and cornflour combine to teach kids about sound

How I Made

Guitar Synth



46

Top Projects



06

122



115

FIELD TEST

- 116 Direct from Shenzhen LED Panel**
Add super happy rainbow colours to everything you touch/make
- 118 Best of Breed**
Audio add-on boards for when a buzzer just won't cut the mustard
- 122 Can I Hack It?**
How can we improve on a standard mini arcade machine?
- 124 Review Adafruit Crickit**
Add a big bunch of functionality to your Adafruit CPX
- 126 Review Line-us**
Draw pretty things with this WiFi-enabled plotter
- 128 Review Simba-Pro**
Add Bluetooth to your builds with a tiny input module
- 129 Book Review Hippo**
Not the river horse of antiquity, but the brain of modern humans

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

Articulated sparrow

By Laura Mathews

 @lauramathewsart

A

lthough the sparrow is the only one I've made, I still love my childhood collection of animal marionettes, which I played with endlessly. I


was always intrigued by biological mechanics – I remember my mum having to repeatedly hide the dead blackbird I'd found in the garden because I

kept finding and examining it, opening and closing the wings, and marvelling at their beauty and precision. We also raised a tiny pink baby sparrow, which had fallen from its nest, into a fully fledged adult. Watching the pin feathers slowly unpeel into a perfect flying machine was incredible, and hugely influential on my work.


For my previous articulated animal static sculptures, I've studied skeletons and musculature in depth. I aim to create a truly convincing impression of each creature, and that requires understanding the way they work.

Laser cutting is a perfect method for me: the speed and precision mean I can duplicate complex shapes easily, without having to be too precious about testing and destroying prototypes in the development process.

In October 2017 the Dinosaurs of China exhibition came to Nottingham, which closely examined their development into birds. I spent hours with the skeletons, attended many lectures and films, and was inspired to create a truly realistic bird marionette. I made a lot of mistakes but I was obsessed, perpetually daydreaming about pivots and strings. You can watch the development, from wobbly 'bones' into the flying sparrow, on Instagram [@lauramathewsart](#).

I'm currently working on a puppet horse, with individually controlled legs for creating the full range of gaits, which will hopefully evolve into a Pegasus! 



Right  Laura makes other mythical beasts, including dragons and unicorns



Clockwork submarine

By Neil McLaren

mclarenclockworksubmarines.com

The submarines and boats that I build are made entirely from everyday tin cans, from baked bean tins to biscuit and chocolate tins. There are no heavy press machines, it's all done by hammer and hand – the most sophisticated tool I use is a drill.

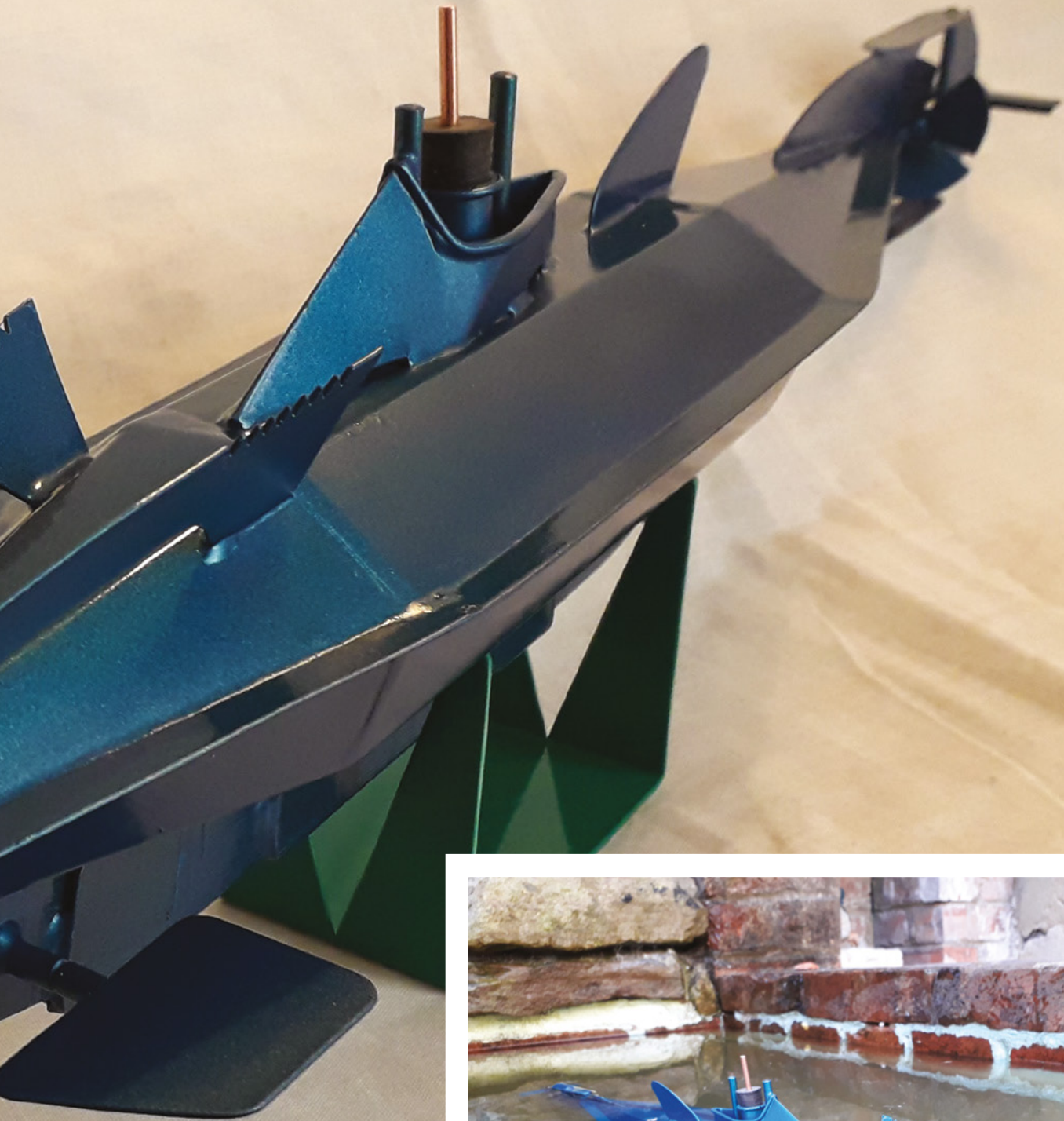
The clockwork motors are made from old alarm clocks and broken old toys available from charity shops for a couple of quid – and again, each one I hand-build.

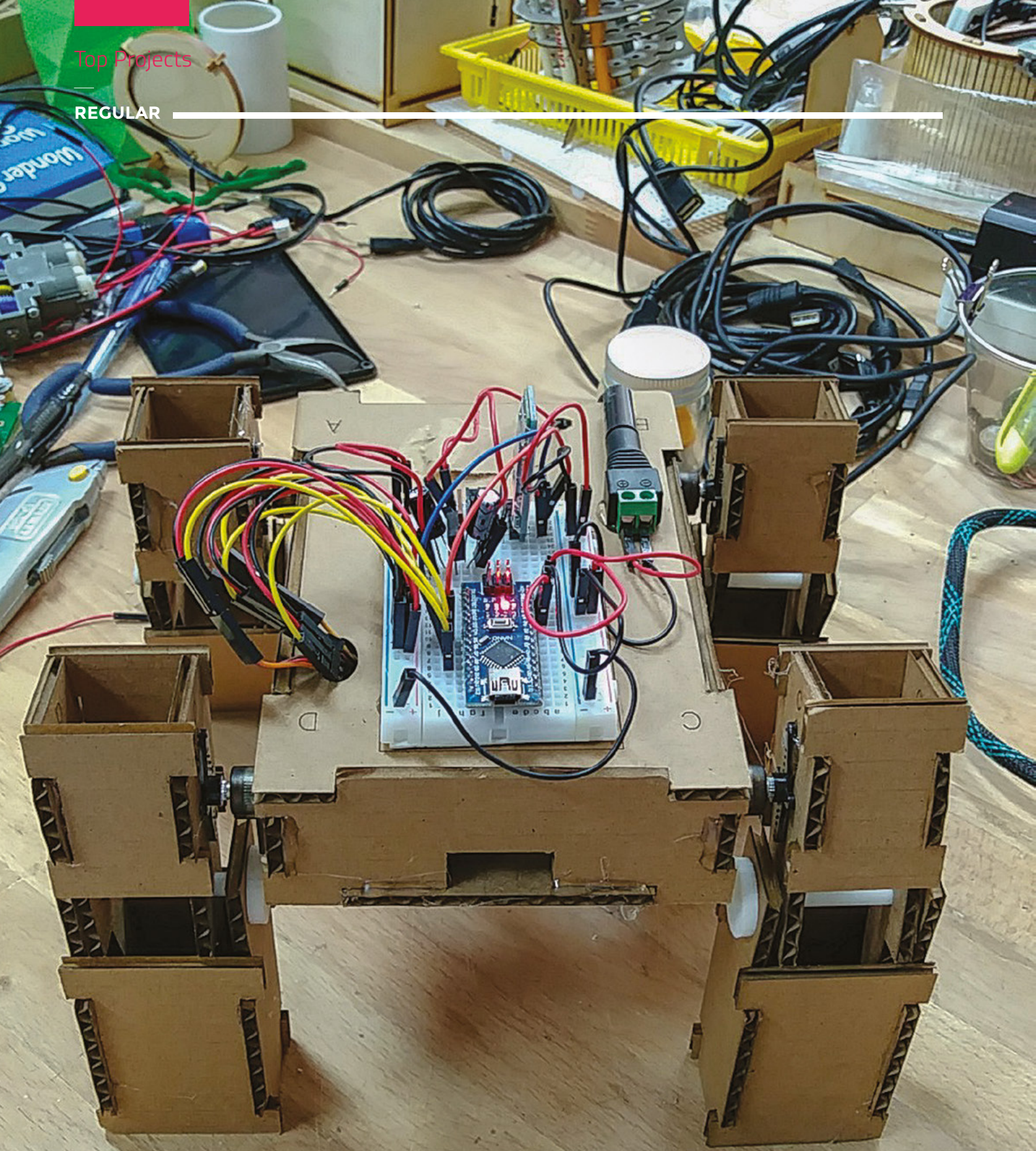
The reason I make these things is because, firstly, I get a real kick from turning junk into something which will still work in 100 years' time, and still do exactly what it's supposed to. Modern mass-produced stuff has no soul, it's tossed aside for something new with hardly a thought. This is not good – so perhaps my interest could possibly inspire a younger generation? Who knows. What I do know is that generations of androgynous drones hell-bent on pseudo-stardom is not what made Britain; it was engineers, physicists, scientists, mathematicians, and inventors from Darwin to Brunel, Newton to Turing. These are the people we need, not dancers! ▣

Right ▣

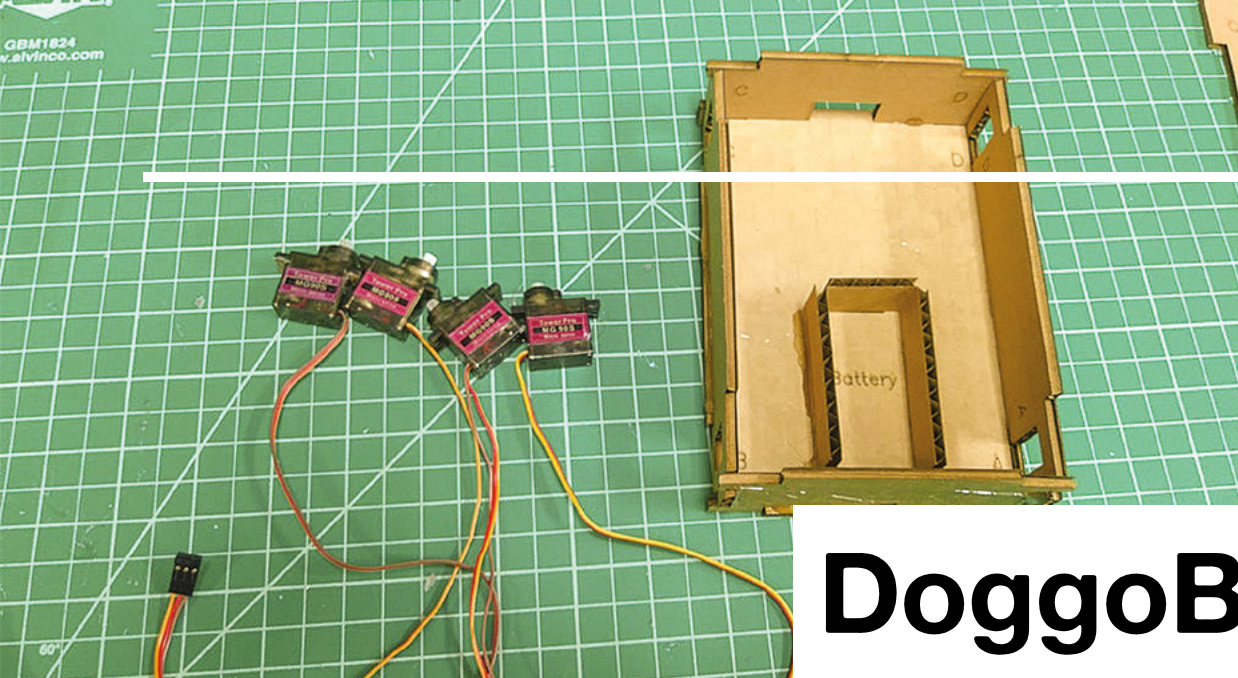
This model was inspired by Captain Nemo's Nautilus and Gerry Anderson's *Stingray*





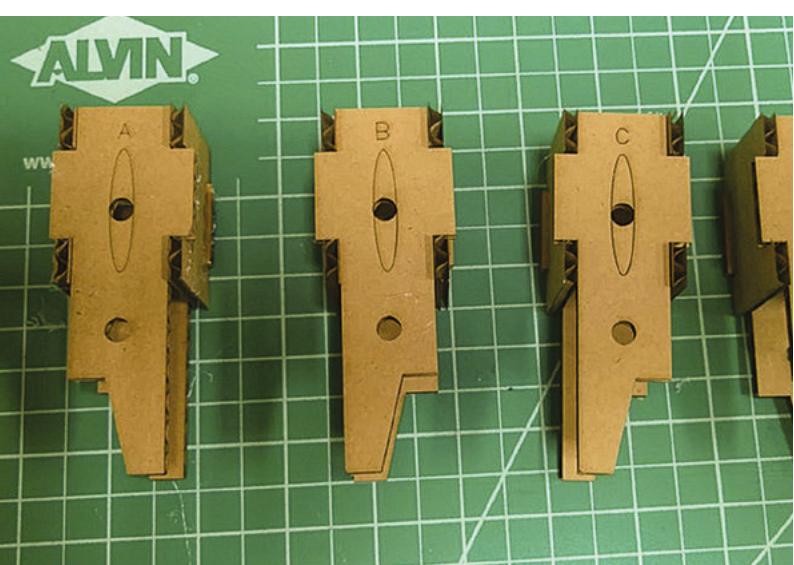


Above ↗
For roughly \$150, you too can build your own Boston Dynamics-style killer robotog



DoggoBot

By Justin Kirk skillmillnyc.com

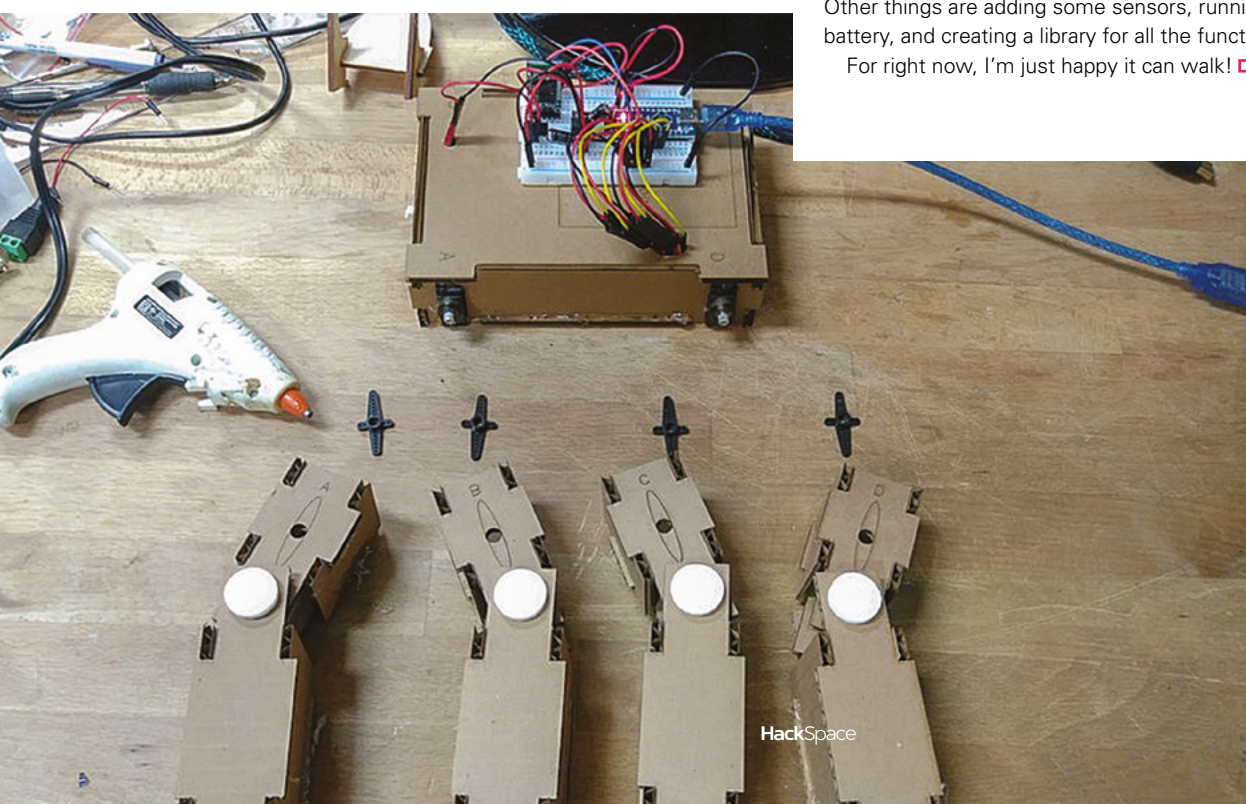


Skill Mill NYC is the makerspace we manage in New York City. We teach classes on 3D printing, laser cutting, computer coding, robotics, and sewing. We also offer design and production services for 3D printing, laser cutting, and Arduino projects. When I'm not teaching robotics or handling our laser cutting orders, I keep myself busy tinkering with various Arduino projects. The DoggoBot is one of those.

The DoggoBot is the result of months of tinkering and problem-solving using the Arduino microcontroller. The DoggoBot uses an Arduino Nano to control four servo motors and take input via either a Bluetooth or serial connection. In its current version, the bot can stand, sit, and walk. I plan to add more, so I wrote the code in such a way that it would make adding more actions and movements fairly easy.

There's definitely some room for improvement with the DoggoBot. One of them is to make it turn. It's harder than it seems! Other things are adding some sensors, running everything from one battery, and creating a library for all the functions of the bot.

For right now, I'm just happy it can walk! □



Rombus pinball machine

By Matt Brailsford

 @mattbrailsford

By day, I run my own digital agency called **Outfield Digital**. However, I'm also the main organiser behind Barnsley.IO, the maker group in my home town in South Yorkshire, as well as co-organiser of The Things Network Barnsley, a community group bringing free IOT connectivity to the area. Oh, and I also like to hack retro tech for fun.

For the pinball build, I went with a LattePanda board. I would have preferred to go with a Raspberry Pi, as it's what I'm familiar with, but there were a couple of reasons the LattePanda made more sense. The first and main reason was that there just weren't many pinball emulators to choose from on the Linux platform. All the main pinball emulators seemed to be Windows-based, which the LattePanda is. Secondly, the LattePanda natively supports dual monitors, as you can use the official 7" display that attaches via ribbon cable to a dedicated connector, plus a regular HDMI monitor via the HDMI port, and it just works. For both these reasons it just made most sense to go with the LattePanda for this build.

Right now, I've just loaded three games onto it to test it out, but there are whole heaps of them available online, so I'll definitely be filling it up when I get the chance. For now, I'm kinda digging the *Terminator 2* table I have on there, which is a pretty fun one.

The main screen uses the official 7" display for the LattePanda, with the second 'back glass' display being just one of those cheap 5" HDMI displays you can find on Amazon or your friendly online auction site. The speakers are from Pimoroni and were a lesson in perfect timing as they just got released as I started the project, and they were just perfect for it. In addition, I'm using an Adafruit 3.7W stereo amp, and some arcade buttons from Arcade World. In terms of hardware used to create it, it was all designed by hand in Inkscape, and cut out on my laser cutter with a few small elements created on my 3D printer (such as the little feet).

All in all, the build took a little over a month. I try to make my builds go pretty quickly, as I like to ride the wave of motivation and I know if I stall, I'll end up putting it on a shelf and not finishing it. □

Right 

The body of this project was all made on a laser cutter, so should be within the reach of most makers





Organ donor beats and pedals

By Charlie Williams

 [c.harl.ie](https://github.com/c.harl.ie)

This project recycles the guts of an old 1970s electronic organ into two standalone instruments: a MIDI controller, and a rhythm-generating box.

An old organ was starting to fail, but the pedals and beats both worked and were my favourite parts of the instrument. Bass pedals are always useful – you can sing, play ukulele/guitar/piano, and accompany all that with bass, all by yourself! And the rhythm chip from the organ was fantastic: for some reason, it was designed so that you can superimpose beats. So if you have, say, 'bossa nova' playing, and also put down the 'tango' switch, it doesn't stop playing bossa nova, it puts the two beats on top of each other.

There are three PCBs inside the beats box: one is from the original organ; it's large, single-sided, and hand-traced. You can hold it up to the light and see all the copper connections through it! The other two PCBs are a tube amp and power supply. I added the tube amp specifically (rather than just a solid-state amp) to put some subtle 'stereo' action in the audio; the beat generator circuit outputs a mono signal, but putting it through the subtle distortion of the tube means that the two sides of the signal are slightly different in a pleasing way. Also of course, having the tube sticking up out of the top of the build looks cool!

The pedals are running a Teensy 3.2 on a custom PCB – my first one! I didn't do any fancy multiplexing, I just have one pedal per pin. No other components, except for some resistors and a status LED. Both projects involved a fair bit of woodworking, but Beats was my first time working with hardwood. Not only that, but the design is curved in all three dimensions, so in many cases I couldn't easily clamp it down to work on it! That really made me appreciate why so many things are built into rectangular boxes – but all in all I'm glad I did an unusual shape, because I'm very happy with the way it looks and feels. ▣



Right ▣

Charlie made the Beats housing hand-held, curved like an mbira or djembe



Objet 3d'art

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



Head to 3dhubs.com/book to check out the #1 3D printing book on Amazon


Having a shiny laptop means you can work wherever you want, but when you're working on a cramped laptop keyboard for a long period

of time, your wrists will let you know about it. That's why separate keyboards and laptop stands are useful; for raising the screen to your eyes and giving you the space to use a proper keyboard to get more mileage out of your fragile human joints.

This MacBook Pro stand is by Paul Markham, who based it on an existing design by Nicolas Iragorri. The reason for the new version is that Apple has changed the hinge on its laptops, reducing the angle that you can open them; this made Nicolas's original design obsolete, as it set the laptop at a steep angle. Thanks to open-source, sharing, and the power of 3D printing, Paul could tweak the design and come up with his own, which everyone can download. [□](#)

hsmag.cc/oBqwdN





T here's no law to say that ordinary objects can't be good-looking. Take this peg for instance: it's a simple design, perfectly functional, and you can choose whatever colour you like to suit the rest of your minimalist home decor.

Nice things don't look nice by accident: this is the work of Ocet Design, aka Tomasz Golánski and Julia Grochal. You can find more of their work at www.ocetdesign.com, and the files to download your own industrial chic peg below. □

➔ hsmag.cc/YbmZYH

Manchester Robot Orchestra

Why learn a musical instrument when you can build a bot to do it for you?

By Ben Everard

 @ben_everard

Most computers make noise, whether that's the whirr of the fan or the 'grrr-chunk' of a hard drive. Older readers will remember the 'cluck' of a floppy drive and the 'burdung-burdung-beep-chrrrrrr' of a modem connecting to the internet. In general, these noises are unfortunate side effects of the computer doing its work. However, it doesn't have to be like this – the whirrs and clanks can come together to make music. Manchester Robot Orchestra is doing just this, with a collection of instruments made from recycled PC parts, adapted instruments, and just about anything else that can be pressed into service.

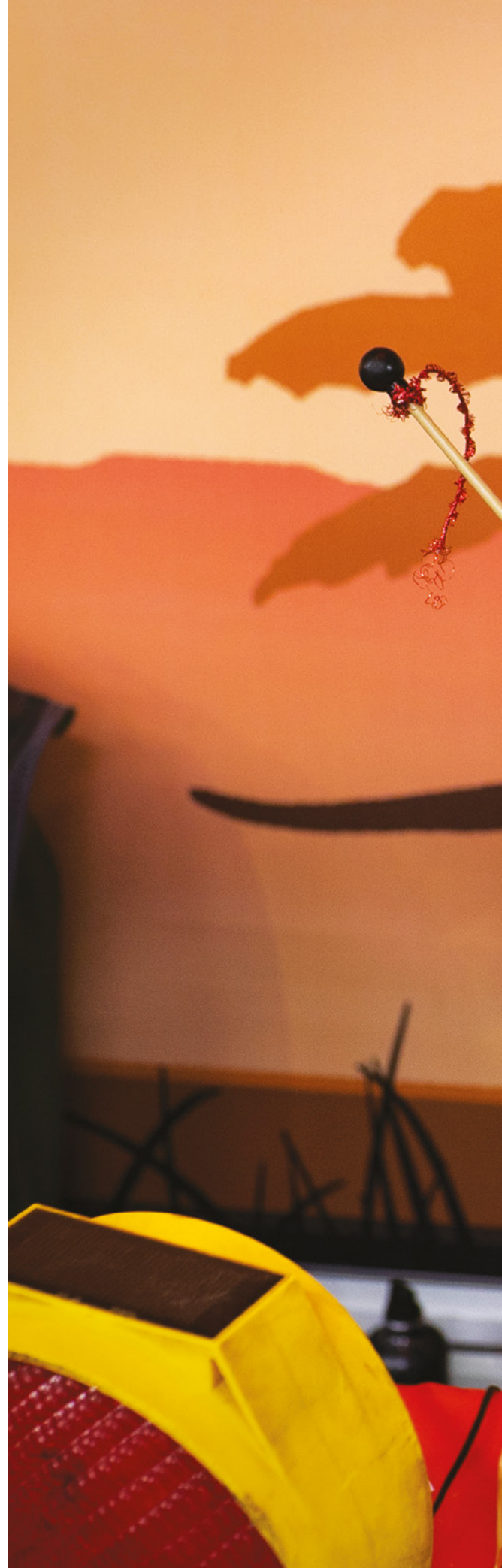
We spoke with Dr Will McGenn to find out how this set of instruments came into being.

"It started initially when Professor Danielle George did the Royal Institution Christmas Lectures in 2014. Their big finale was to have a robot orchestra.

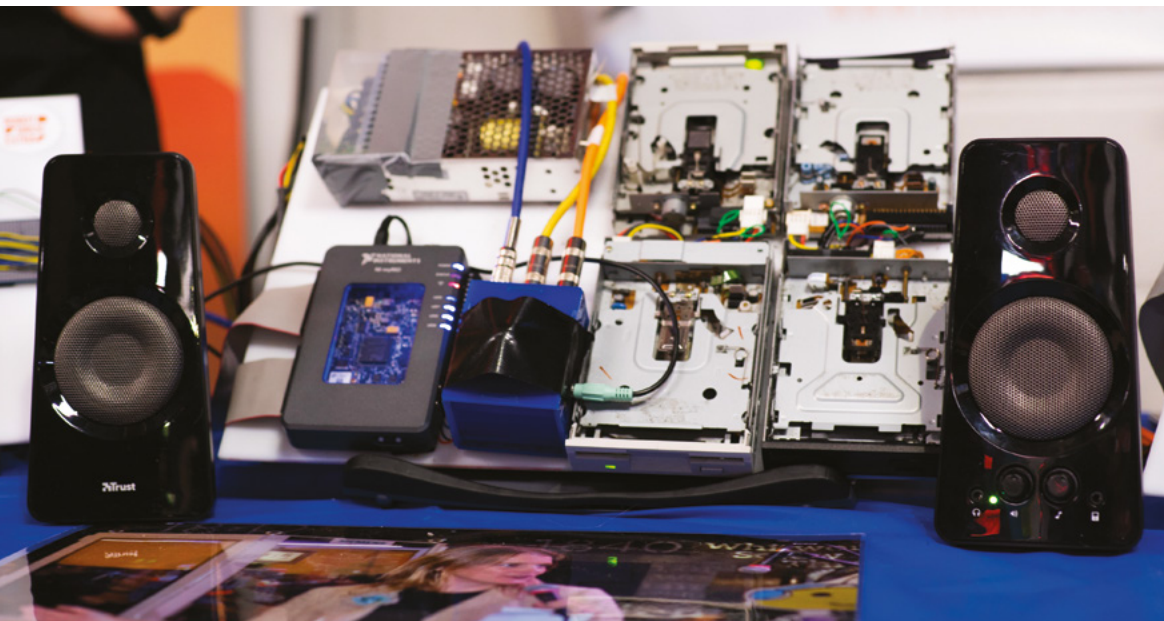
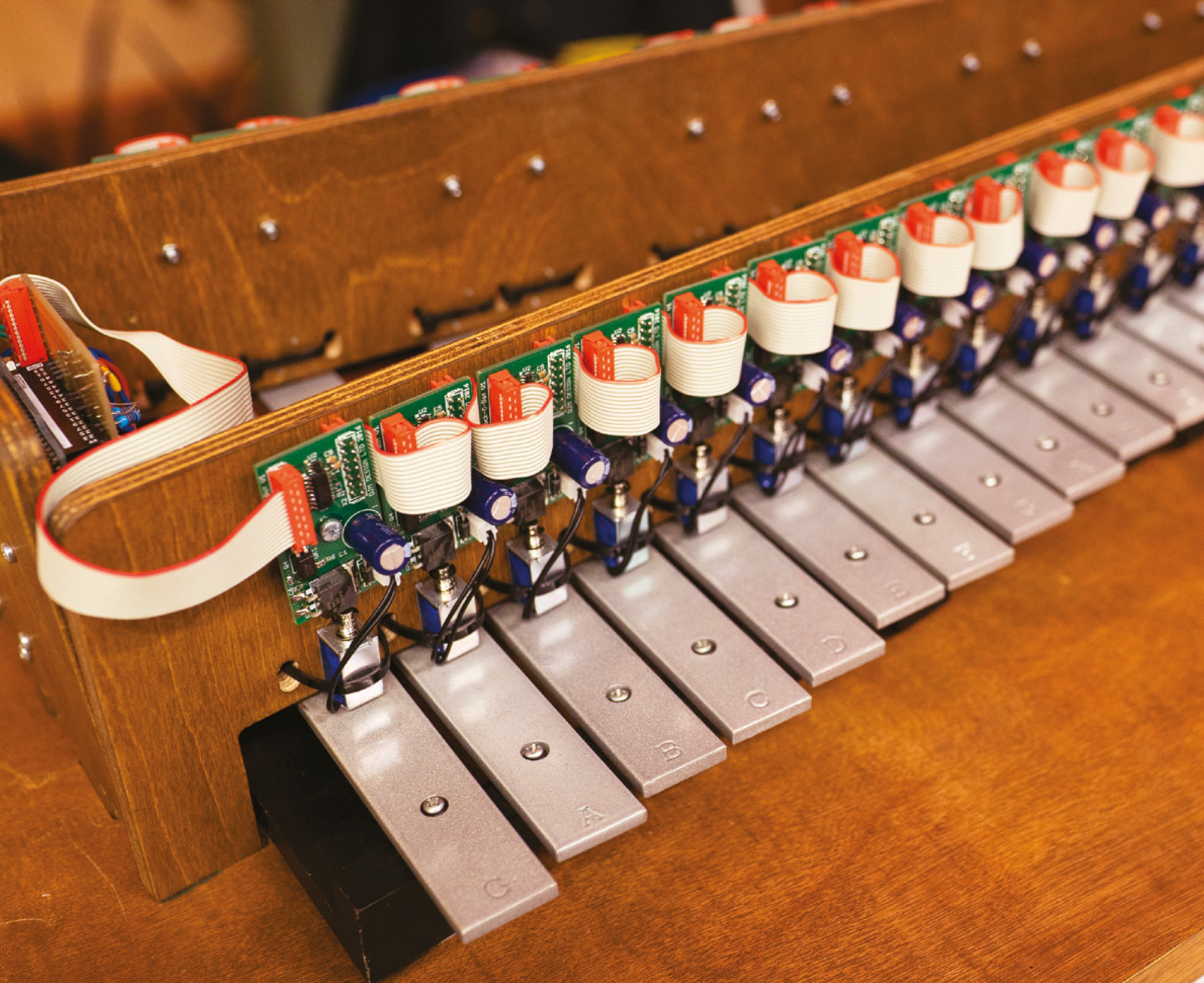
"The robots for that came together from various research groups and universities, so it wasn't something that could be kept together – lots of it was actual research hardware and things like that. It had to go back after it was finished. In 2016, Manchester was European City of Science, and that was when the Robot Orchestra really started up in its current form. Initially it was meant as a one-off thing – just for one performance, which we did at the Museum of Science and Industry.

"We played quite a lot of songs with human musicians that day, and since then it's just snowballed. People have been grabbed by it and are really keen. People keep on writing to us: 'come to our event'; →

Right ♦
Musical robots don't have to be complicated







Left ♦
The motors in these floppy drives are rigged up to play tunes

Right ♦
Almost anything can become a robotic instrument, if you have enough imagination





Left ■ Robots aren't limited to just two hands, so can play music of almost any complexity



“ When the orchestra started a couple of years ago, one of the big themes we wanted was **reusing, recycling, upcycling** ”

‘can we do stuff with you?’; ‘can you do this?’. Now it’s just trying to work out how to keep it going, without it still being so intensive from a time commitments [aspect for] volunteering on this.

“We’ve got a team of Masters students working on this Orchestra this year. They’re building a new core of the Orchestra – a new conductor, and a few new instruments and the improvements that’s going to provide will make the Orchestra so much easier to use, and we’ll be able to pass it out to more people. More people will be able to take it out on the road to events like this. Hopefully that will start a whole new cycle of getting kids building new instruments.

“[You program the instruments in] quite a few different ways. We’ve got robots based around quite a lot of different controllers. We’ve got some with Raspberry Pis, some with Arduinos, some that use the little Crumble controllers, which are really simple things based on the Arduino. The Raspberry Pis and Arduinos can take a MIDI file, so a little bit of manipulation to it and [you can] program it onto the Arduino or the Pi. The Crumbles are a lot simpler, so they’re controlled a lot more from our robot conductor. Each time they are needed to play, the conductor will send them a signal.

“When the Orchestra started a couple of years ago, one of the big themes we wanted was reusing, recycling, upcycling. Pretty much all the instruments have been made by school kids – both primary school and secondary school. We’ve got a couple made by university students, but pretty much everything else was made by school kids.

“We’re on social media – just search for ‘robot orchestra manchester’. We put pictures up; we’ve got lots of videos of the different songs as well.” □

Above ◆ When not making robots play music, Will works on radio telescope receivers



Reducing mental load

Keep your head clear for better making



Lucy Rogers

🐦 @DrLucyRogers

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

Do you sometimes get that horrible feeling of being overwhelmed by life? For me, it's usually by the little things that I have to think about before I can even start

to think about the big things. What shall I wear today? Is there enough toothpaste left in that small tube for five days away? Where's my travel pass? This ALWAYS adversely affects my making. Or, to put it more bluntly, I stop making.

Over the last few years of reading and listening to people, I have found seven hints and tips that help me – I've shared an overview of them here. I know not all will work for everyone all of the time, but please help yourself to any that do appeal.

Seven 'R's' of reducing mental load:

Reduce

- Reduce the number of decisions and choice.
- Reduce the amount of physical things I own.
- Reduce information – mailing lists, post, television.
- Reduce people stress (e.g. I mute people on Twitter).

Routine – don't think, just do.

- I have a 'uniform' of jeans, black T-shirt, a jacket, and boots. I don't have to decide what to wear.
- Favourites in my online shopping are automatically added.
- I do a 30 minute Pomodoro before breakfast, and one straight after – one hour's work done without thinking about it (or procrastinating).

Right place for everything

- Easiest example is a cutlery tray – knives, forks, spoons always in the same place.
- For this to work, the things must go BACK in the right place too!

Replicate

- I have a laptop power lead in a bag ready for trips, and one for everyday use.
- Phone charger upstairs and downstairs.
- I keep a spare phone battery, cables, business cards, pen, and notebook in a pencil case – I just grab it when going out for a day.

Readiness

- Get my clothes out night before
- Buy train tickets the day before

Write it down – paper lists, apps such as WorkFlowy, Trello, or pictures such as Pinterest.

Relax

- Bubble Bath
- Reading
- MAKING!

Note that some of these ideas:

- Cost money, and I know that this is not an option for everyone.
- Aren't environmentally friendly.
- Do not sit comfortably with everyone's idea of what is right.

These are the things that help me do more making. I'd love to hear what works for you. □

Turn your TV into a home assistant using a Raspberry Pi

When legal confusion causes technical problems



Bunnie Huang

[@bunniestudios](#)

Andrew 'Bunnie' Huang is a hacker by night, entrepreneur by day, and writer by procrastination. He's a co-founder of Chibitronics, troublemaker-at-large for the MIT Media Lab, and a mentor for HAX in Shenzhen.



One of my favourite aspects of the Raspberry Pi is its rich ecosystem of applications. A good example of this is Alasdair Allan's demo (hsmag.cc/MXDCSf), combining a magic mirror and Google AIY to create a voice-activated home assistant. I especially like the mix because then I don't have to listen to the long-winded responses of audio-only home assistants. This example really shows off the Pi's application ecosystem.

One downside of the magic mirror is the need to dedicate a whole screen to the application.

Even if you had a spare TV laying around, finding a spot to put it, and wrapping it in a custom frame can take a lot of time and effort. Wouldn't it be nice if you could somehow merge the magic mirror's output onto the TV screen you already have?

Fortunately, there's a partial solution to the problem – it's technically feasible to encrypt the HDMI video coming from your Raspberry Pi so the TV can understand it. Once encrypted, portions of the screen can be cherry-picked to come from either the Pi or your existing home entertainment system. I've built a system called NeTV2 that can do this. It's

basically a Raspberry Pi HAT that sports an FPGA, wrapped into a custom plastic case and, as of the time of publication, it's crowdfunding over at hsmag.cc/Hbragk. Although a programmable video overlay seems like a blindingly obvious widget that should be readily available off the shelf, the manipulation of encrypted data is legally challenging. You can read more about the subtle legal challenges on the Crowd Supply page linked above.

Now, one can technically merge a magic mirror onto any TV screen by just plugging NeTV2 in-line with the main video cable and loading up the application. Combine

it with Google AIY to make a voice-activated home assistant, or perhaps just open a small browser window and connect to your baby monitor, or track how long it is until your pizza arrives while

One downside of the magic mirror is the need to dedicate a whole screen to the application

you enjoy the game. These are just a few examples of what's made possible with video overlay, and we're just starting to scratch the surface of the possibilities. If the legal challenges around processing encrypted video can be resolved, NeTV2 can also power even more exciting applications, utilising AI on video feeds to help with everything from real-time translations to fantasy sports to health and assistive care. □

Letters

ATTENTION ALL MAKERS!

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

AMBITION

Do you reckon I can fit a forge in my third-floor flat? I want one now, especially if I can have a basketball hoop and a load of cool flags.

Michael

Berlin

Ben says: I assume you refer to our interview with blacksmith, philosopher, and all round good egg Alec Steele? Go for it. It's just a fancy oven, so put it in the kitchen. Sell all your possessions and start learning how to make Damascus steel.



DIVERSITY

Jenny List's feature on diversity in makerspaces hit the nail on the head, not least in the opening assumption that most diversity 'training' offered by employers is a dull waste of time. It's one thing to sit in a classroom and be told that people are different; it's another to hear simple tips about how to make those different people feel more welcome.

It shouldn't be up to excluded groups (for want of a better phrase – nobody's excluded, but some people feel excluded) to explain how they want you to change. A little bit of thought can go a long way, such as having social events in pubs. Pubs are off-putting for a lot of people, for a lot of different reasons. Some people don't drink for whatever reason, or don't like noise, or large groups of

DEAR HACKSPACE READERS!

My name is Elisabeth and I need your help: I've been a maker for many years and I am now doing research on the Maker Movement. I've prepared a survey that I'm sending to all makerspaces in Germany, Austria, and the UK. I'm particularly interested in finding out more about potential environmental aspects and female makers (but I need responses from all genders in order to compare!). It would help me a lot if you could take a few minutes to answer the



Credit: US Air Force. (CC0)

people. The answer isn't to ban pubs, but to mix it up – try a community centre instead, for example.

David

Sunderland

Ben says: Legal precedent is such that organisations find it easier to fire misogynists/racists/otherwise intolerant folks if they've gone through some 'training'; makerspaces should do it because we want to see more people enjoying good things.

questions. I want to capture as many voices as possible to get a detailed picture of us! Either find the link on my blog at makingdiversity.co.uk, or type it in directly: hsmag.cc/XzUtGw.

Thank you!

Elisabeth

(PhD student, University of Glasgow)

Ben says: Consider it done Elisabeth. It took me about 10 minutes, or roughly the amount of time it takes to have a tea break.

CROWDFUNDING NOW

BUYER BEWARE

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

Infento

Kickstarting creative kids' kits


From \$179 | kickstarter.com | Delivery: September 2018

Infento gives kids the ability to hack their own transport.

It's a set of parts that's designed to be built into different modes of transport to get children around. You need just an Allen key to put everything together, so it's great for a parent-child building project.

There are different Infento kits, with different capabilities and different price ranges. The Explorer (ages 0–8), and Pioneer (ages 6–14), are the basic kits at \$179 each. They offer six and seven different buggies respectively. Beyond this, there are four larger kits that give you more possibilities, including the Volt (ages 4–14, \$799) which adds an electric motor and control gubbins. As well as the kits, you can get add-ons for snow and LEDs.

The great thing about the kits is that, while there are some recommended ways of building them, they're still flexible enough to enable you to hack them in different ways. See the community showing off their builds at infentorides.com/community.

Infento aims to walk a fine line of being easy to get started, fun for children, and still hackable enough to let your creative spirit shine through. We haven't got our hands on one of the kits, so we can't say if they achieve this or not, but if they do, it'll be a great product to get a new generation making. 



Right

The small wheels can be locked in one direction, or allowed to pivot freely for performing stunts



Space of the month: Artisan's Asylum



Artisan's Asylum

Artisansasylum.com

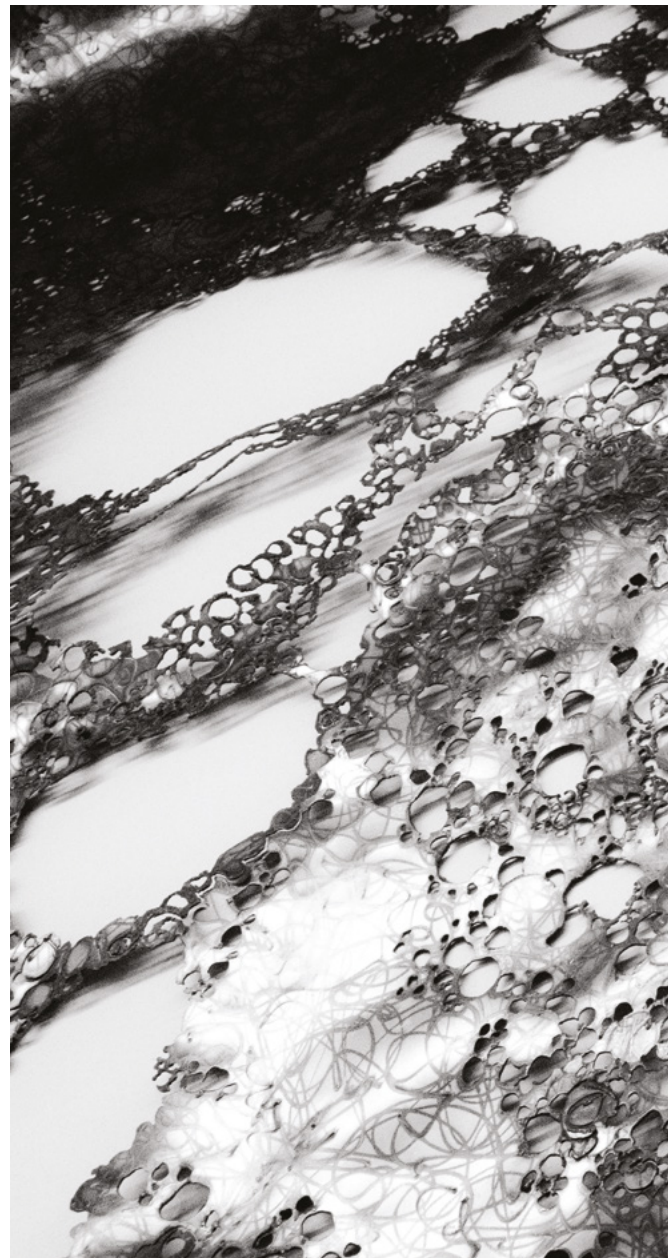
@artisansasylum

Last month we featured **Urban Hax in the heart of England's historic maker country**. Geoff, one of the space's co-directors, cited Artisan's Asylum as an inspiration for Urban Hax, so this issue we're featuring this huge makerspace in Somerville, Massachusetts.

The story of Artisan's Asylum began in 2010, when Gui Cavalcanti, a robotics engineer at Boston Dynamics, and Jenn Martinez started looking around for a makerspace. Recent graduates of Olin College of Engineering and MIT, they were used to having a university makerspace, and the outside world didn't have one at the time. So, they decided to create their own.

Gui bought a set of metalworking tools and the pair found a 1000 square foot industrial unit, and they put out a call to local makers, hoping to get maybe 20–30 people. When 100 people showed up, they knew they were on to a winner. They moved twice in the first year and a half, into 9000 and 25 000 square foot locations, and now Artisan's Asylum has 40 000 sqft of space and 300 members. Artisan's Asylum members have received \$5 million in Kickstarter funding, and \$4 million in venture capital funding, and the businesses associated with the space have doubled the number of manufacturing companies in the city.

Artisan's Asylum has a huge range of facilities, including (deep breath): a ventilated room for working with spray paint; a bike shop; dedicated digital fabrication room with 3D printers, laser cutters, a vinyl cutter, and then there are jewellery, welding, electronics, and woodworking rooms, and a whole bunch more.



Gretchen Green, steel sculptor ♦

“For me, the Artisan’s Asylum has been an incredibly positive transformational force in my life.

“One day two years ago, walking to my job as a corporate tax lawyer ... I read an article about Stompy the Hexapod. With that article, I discovered the Asylum.

“I took a beginning metalworking class. In four Sunday afternoons that summer, I learned to weld. I learned to use the plasma cutter. Amazing things happened.

“I made work unlike any I had made before, and yet in that work, I could feel all the things I’ve done before. Everything fit.

“Brooklyn Boulders commissioned a large wall sculpture. I was filmed by The Economist. My work appeared in **Boston.com**. I performed on-stage in New York. I started showing and selling in Massachusetts, California, and New York. I served on the board of directors for the Asylum.

“For the last year, I’ve been working full-time as a professional sculptor, something I never imagined I’d be doing, and something I don’t believe I ever would have done if I hadn’t found the Asylum.

“The Asylum has given me the training, the tools, and the space to create and display large steel sculptures. Our community has provided resources and encouragement, both of which have helped me launch and grow my business.

“I love the diversity, the knowledge, and the generosity I have found. I love coming to work every day.”

➔ kgretchengreene.com



Left ♦ Gretchen’s work has been exhibited around the USA, as well as in Australia and South Korea



Sal Mancini, woodworker

"I've always found the term 'maker' strangely sterile. I like to use the word 'artist', but I do realise that it doesn't cover everyone. This place is where MassArt meets MIT. There's a lot of really good cross-pollination there. I always took the word 'asylum' to mean a safe place to be yourself, instead of 'the loony bin', which is often what the term is associated with.

"In regard to how it's impacted me, it's a nice validation of my skill set on a daily basis. It's really nice to have so many intelligent and creative people around who inspire me and who I can learn from. I go almost every day because it's kind of like my second home."

Above ♦ As well as a makerspace, Artisan's Asylum runs open days catering for adults, and smaller adults



CONTACT US

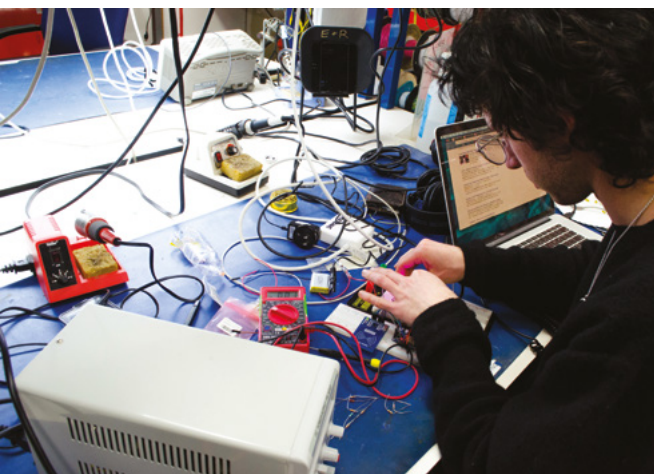
We'd love you to get in touch to showcase your makerspace and the things you're making. Drop us a line on Twitter [@HackSpaceMag](#) or email us at hackspace@raspberrypi.org with an outline of what makes your hackerspace special and we'll take it from there.



One of the courses on offer at Artisan's Asylum is 'How to Run a Makerspace', which is sorely needed, as many in the UK will attest

Bronwen Senhouse, maker and instructor

"This place gives me the feeling of visceral joy. It's working together with other people to build such a healthy community that lets people take the things they've imagined and make them real. It's what humans do that is different from all other critters – we get this idea and the idea can become very fully formed without any physical manifestation. This place lets people manifest their ideas physically."



Over 10,000 Different Boards and Modules In Stock

NO ONE KNOWS BOARDS LIKE DIGI-KEY!

- Open Source Community Boards
- Evaluation and Development Boards
- Wireless Modules
- Single Board Computers
- Reference Designs
- Programmers, Emulators and Debuggers

DESIGNS START HERE!
DIGIKEY.COM/BOARDS



#MAKEWITHDIGIKEY

Digi-Key is a franchised distributor for all supplier partners. New products added daily. Digi-Key and Digi-Key Electronics are registered trademarks of Digi-Key Electronics in the U.S. and other countries. © 2018 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA



LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG
46

HOW I MADE: GUITAR SYNTH

A unique instrument inspired
by a unique artist

PG
50

OPEN SPACE AGENCY

Its continuing mission: find cool
stuff in space using homemade
science equipment

PG
54

INTERVIEW NOISY TOYS

We've learned a new phrase
this month: science busking.
It sounds like fun, and it is!

PG
64

MEASURING TAPE

It's not just for finding
out how long things are:
you can build with it, too

PG
68

HACKSPACE LEARNS: GLASS-BLOWING

Another string to our
maker's bow, as we try
this ancient craft

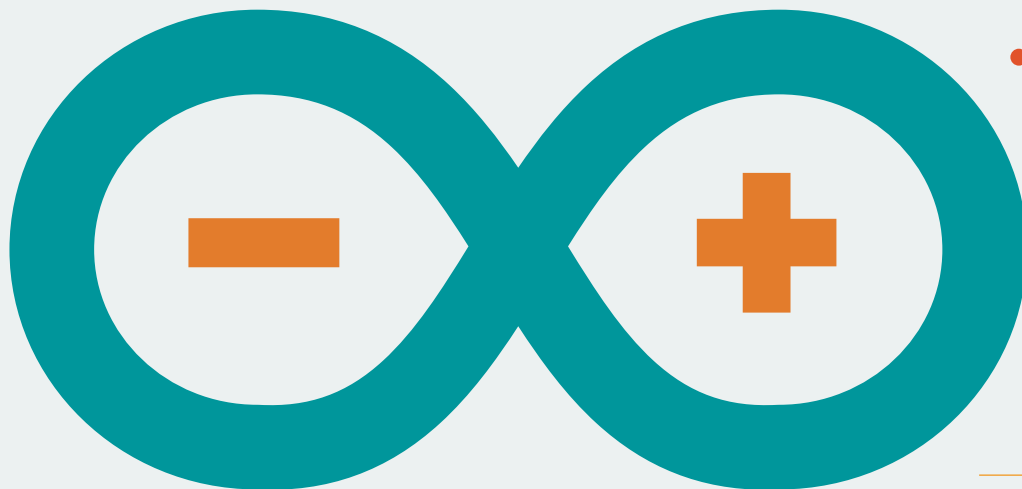
PG
32

ARDUINO THE NEXT GENERATION

The inside story of the world's
favourite maker board

ARDUINO

THE NEXT GENERATION



Behind the scenes with the ubiquitous microcontroller

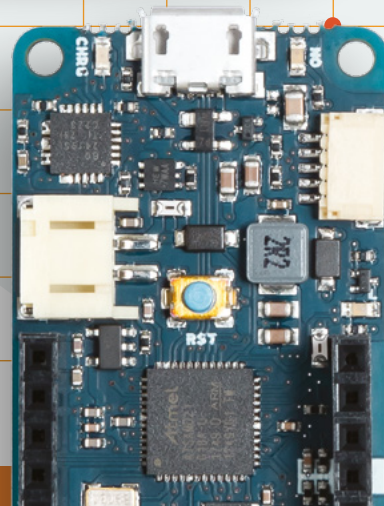
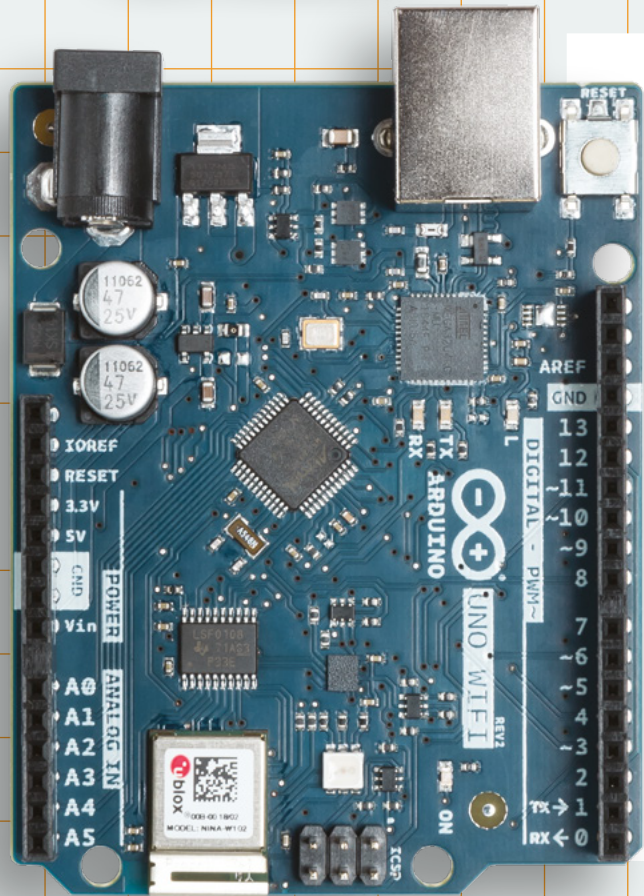
The now ubiquitous Arduino sparked a revolution in digital making when it was launched in 2005. It's gone through a few revisions since then, but the basic board of the flagship model (currently the Uno Rev 3) still retains a similar look and form factor.

It was cheaper and easier to use than any other embedded platform at the time, and it came from an organisation committed to making it a great platform for designers and artists.

Just as Arduino looked like it was about to achieve world domination, a rift between the organisation that designed the boards and the company that manufactured them (at this point both organisations called themselves Arduino) erupted into a legal battle that confused customers and took the team's attention away from their core focus of making great hardware and software.

After a tumultuous few years of legal battles, Arduino emerged united once again at the end of 2017, with a new CEO and new backers (in the form of ARM Holdings). Since then, they've been hard at work on the hardware and software offerings. Now, the newly reinvigorated Arduino is set to release a whole host of new boards and software.

We spoke with new boss Fabio Violante to get the inside story on these new releases and find out what Arduino has in store for the future. →



NEW CEO

Meet the man in charge



Massimo Banzi had been the public face and CEO of Arduino for as long as there had been an Arduino, but in November 2017 he stepped down as part of a partnership with ARM

Holdings that also ended the lawsuit between two organisations, both called Arduino. Up stepped Fabio Violante to become the new CEO. We caught up with Fabio to find out about the new technology, and how he got started with Arduino.

HackSpace You're the new company CEO, but what's your personal relationship with Arduino?

Fabio Violante It's a long story! It goes back to the origin of Arduino. I know Massimo because we were working together and we were friends in normal life. When I finished my PhD in Computer Science at Politecnico di Milano – the main technical university in Milan – I was doing a PhD in human-computer interaction, but doing boring stuff. Human-computer interaction for safety-critical systems, so, theory and doing a lot of it.

I went to visit the Interaction Design Institute of Ivrea – a school that was started just six months before I went to visit them – and they asked me if I knew someone who could teach electronics to designers and to ask this question to my colleagues at the Politecnico.

I went back and they said "No! Teaching electronics to designers? For us?" Those were guys working on highly sophisticated FPGAs, so they didn't care about designers. I thought about Massimo – he had a real passion for electronics and he worked as a CTO for an internet provider at that point in time. I said, "Massimo, you could be the right person for this type of engagement – they're designers, you love design, and you know electronics." I introduced Massimo to the school and they hired him. That's how the story started. When he was teaching at the Design Institute of Ivrea, they started the Arduino project as a way to

standardise the electronics projects the students were doing. I introduced Massimo to the school and they invented Arduino, so I'm sort of the great-grandfather to some extent.

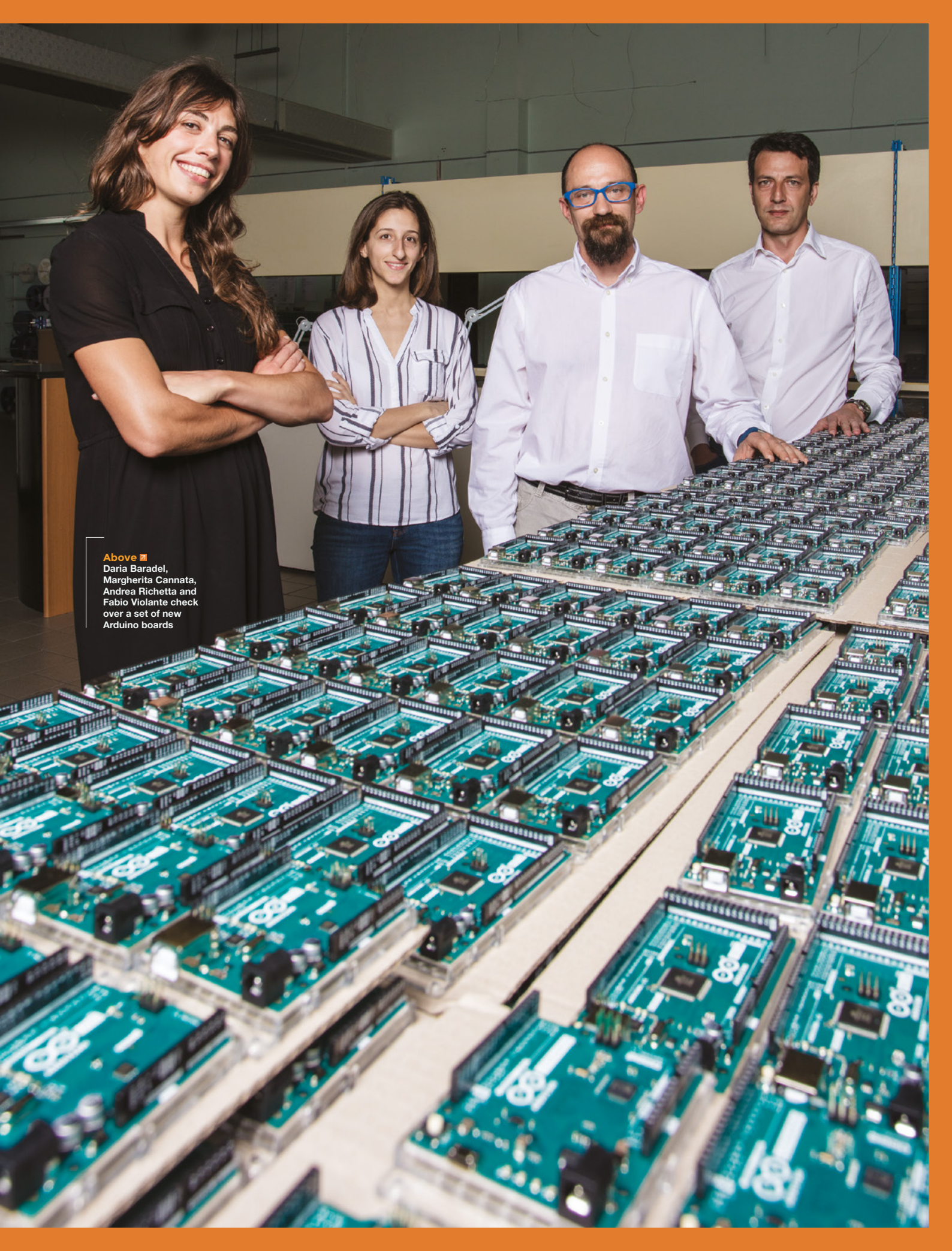
I was in touch with Massimo occasionally – once or twice a year as a sort of mentor – but I had my own company. In 2010 I sold my company to BMC Software (the eighth largest software enterprise in the world). I then became CTO of BMC Software for five years until 2015, but I was commuting from Italy every other week, so I decided to resign. At that point Massimo reached out to me and said, "Now that you have more time, could you help me with Arduino?"

This was more or less the starting time of the legal mess between the other party and us, so I started just advising and then step-by-step I got more engaged because there was a lot of work to do – to transform Arduino into a company and solve the legal situation and, from there, more and more engagement. It became like a 200% occupation of my time, and in 2017 we resolved the situation with the help of ARM and I became the CEO, and we made Massimo focus more on the technology side as CTO of the company. It's a very long story – I tried to compress it!

HackSpace It feels like there's a lot more coming out of Arduino at the moment.

FV When you don't have to cope all the time with lawyers and you have this community ... for me the innovation was the simplest thing. The more complex thing was transforming this group of ultra-smart people into something that can deliver. There was a change between the past and now. In the past there was this announcement and like one year later or two years later [the product came out]. Now I put a rule in Arduino that, when we talk about something, we should have this thing almost ready to ship. All these products that we are announcing will be available between the end of June and the beginning of July. →

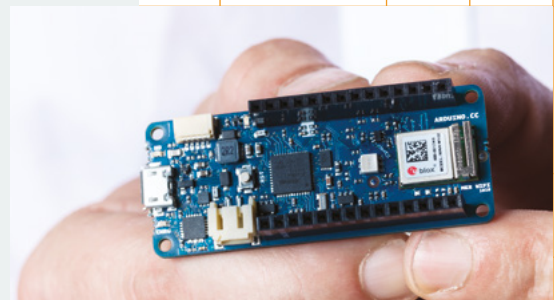
Above ♦
Fabio Violante is justifiably proud of the new Arduino products



Above ■
Daria Baradel,
Margherita Cannata,
Andrea Richetta and
Fabio Violante check
over a set of new
Arduino boards

BOARDS FOR MKRS

Hardware designed for professionals



A whole host of digital makers got started using the Arduino Uno, and it's still a great board for beginners, but perhaps a little lacking for more advanced projects. It's quite a large board, there are limited power management options, and connecting to a network can be a trifle awkward. Enter the MKR series of boards that first launched in 2016. 2018 has seen the line bloom to eight boards, all in the same form factor, but each with different connectivity options, ranging from widely used protocols such as WiFi and Bluetooth, to more niche options such as Sigfox and LoRa.

HackSpace What are the key problems that the MKR boards are designed to solve?

Fabio Volante The idea with the MKR form factor is to respond to the needs that small and medium enterprises were asking for... basically, the concept of the MKR is not only the form factor but also having a microcontroller, a secure crypto element (with hardware that can handle certificates), and also power management, both for battery-powered devices and also low-power management and connectivity.

HackSpace In the past Arduino has made a range of boards, but generally left creating add-on boards (known as shields) to third parties. You've now announced a series of MKR shields. Is this a change in tactics?

FV [We have created] some professional shields that can be used even in production. Some of the shields that we release have the RS485 connectivity, they

Above The MKR boards are small enough for most embedded products but still pack plenty of processing power

have the CAN bus interface. There is an Ethernet shield – we designed them because we saw people wanted to retrofit their equipment, and send signals to the cloud directly from the board.

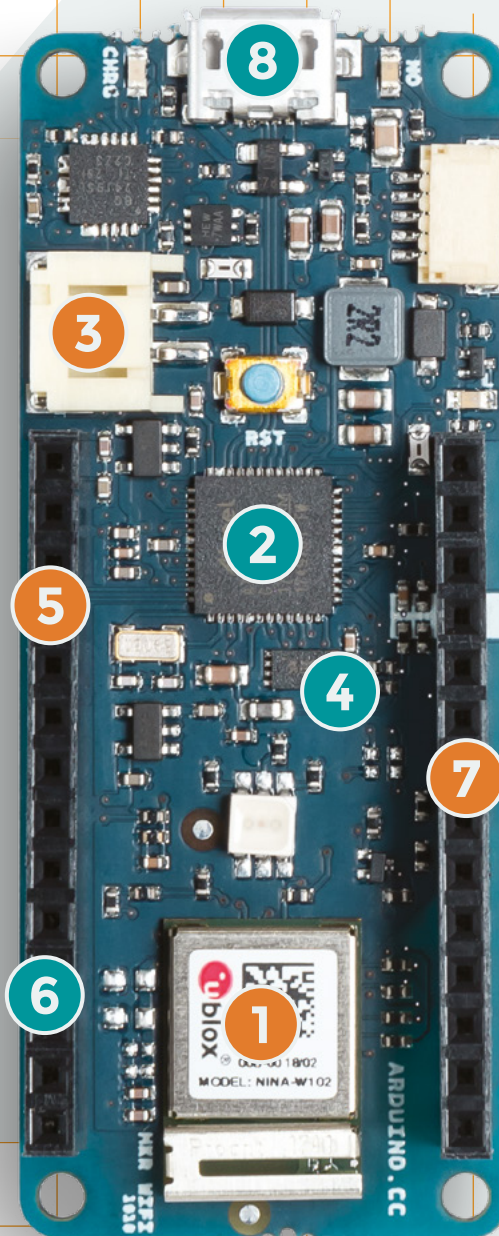
We created these to respond to the need for professional shields, because in some cases the [third party] shields were high quality, but in some other cases they were made for hobbyists and just throwing Chinese technology onto a shield. So, all the shields we have provided, such as the CAN one or the RS485, they have galvanic isolation, they use high-quality components, high-quality terminals, a high-quality PCB, and so on and so forth. We wanted to have a flagship range of shields.

We will be releasing other shields from now until the end of the year. We're working on eleven or twelve more that solve specific problems, not only for the professional market, but also for home usage of the Arduino, and we are working with partners as well that are developing other shields.

We want to have people into the MKR format so we are also trying to do a better job of releasing more specifications so people can start developing shields. We would like to create a great ecosystem around this MKR form factor, and we are looking for makers and also partners to develop professional, hobbyist, and educational shields. →

INSIDE THE ARDUINO MKR WIFI 1010

- 1 ESP32 WiFi and Bluetooth module
- 2 ATMEL SAMD21 Cortex M0+ SoC running at 48MHz, with 32kB RAM, and 256kB flash
- 3 LiPo port (with charger) 3.7V, 700mAh min
- 4 ECC508 CryptoAuthentication
- 5 Seven analogue pins (one output)
- 6 Eight digital GPIOs
- 7 I²C, SPI, I²S, UART
- 8 Micro USB for programming and power

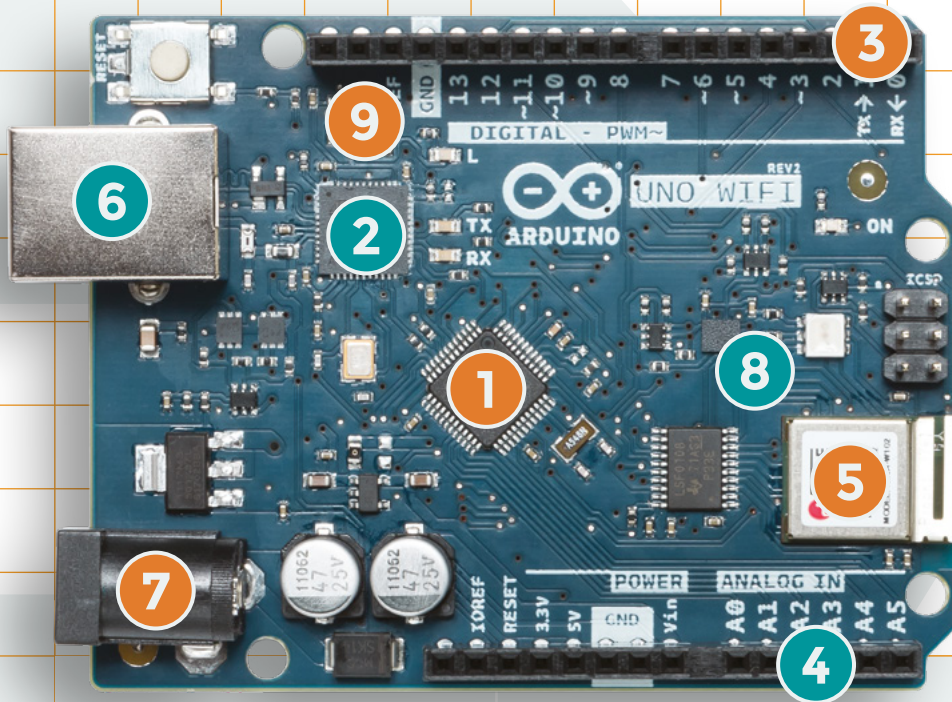


THE HACKSPACE MAGAZINE VERDICT

Cryptography support is a must for everyone today – and especially so in the professional world. Microcontroller support for cryptographic protocols is often limited, or omits some checks because of the limited processing power and RAM available. The addition of a crypto chip makes the MKR boards particularly suited to IoT devices.

The stand-out feature of the MKR line is their ability to interface with almost any network, so if you're interfacing with a legacy network or looking to take advantage of the latest long-range wireless technology, they're a particularly great option.

INSIDE THE ARDUINO UNO WIFI REV 2



1 The main processor – an ATmega4809 running at 48MHz, with 6kB SRAM and 48kB flash

2 A secondary Atmel microcontroller that handles communication over USB and programming the main chip

3 14 digital input and output pins (six with PWM)

4 Six analogue inputs

5 U-blox ESP32 WiFi that can connect to a WiFi network or create an access point. This module also supports over-the-air programming

6 A USB connection for power and data transfer

7 Port for 7-12V DC power in

8 Inertial measurement unit

9 ECC608 crypto chip accelerator

UNO WIFI REV 2

The classic board gets a revamp

Since 2010, the Arduino Uno has been the go-to board for anyone who wants to learn to use microcontrollers. It's not very powerful by modern standards, but its ease of use and huge amount of online support make it easy to get started.

As you may guess from the name, this isn't the first time Arduino has added WiFi to the Uno, but the previous version wasn't supported as fully, and got caught in the rift between the two Arduino organisations. This new board will be fully supported by the Arduino ecosystem, and work with existing Uno shields.

HackSpace Why did you develop the Uno WiFi?

Fabio Violante The idea was that we have a ton of users of the Uno board, but in many cases people want to connect [but still] leverage the ecosystem of shields that are available for the traditional boards. So, we decided to combine a new version of the AVR microcontroller – the 4809 from Microchip (that has a little bit more RAM and flash, which is very important for a connected application) – with a module from U-blox that uses the ESP32. It's a very reliable WiFi module – and we also added an IMU. In many cases, both professional and educational users can take advantage of having an embedded IMU for doing simple stuff. This is a product that will be useful to prototype a number of situations. We are always committed to supporting the traditional format of ours. This is one of a number of innovations that we will be introducing in the future on the traditional form factors.

HackSpace The Uno WiFi revision 2 uses a Microchip AVR, but Arduino is now in a relationship with ARM – is this a problem?

FV No, absolutely not. We're a sort of Switzerland. The company is a Swiss company! But besides this joke, we are trying to be the Switzerland of hardware so we can work with Intel, we can work with ARM, we can work with Microchip/AVR technology and to be honest, the relationship with ARM is really amazing, and they are really inclusive as well, so there is no influence on our strategy. Even the latest announcement [the Vidor FPGA], we announced this Intel/Altera part. There's a lot of knowledge in our community about the AVR parts – probably more so than the 32-bit Cortex technology – so we wanted to reward this loyalty and allow them to do more stuff with the knowledge they have. Creating a path for them to evolve, but also creating a platform for what they know.

HackSpace There's a wide range of connection options available on the MKR boards. Are we likely to see more of these come to the Uno?

FV For [many types] of connectivity they require a little bit more power and resources on the microcontroller side, so to make them compatible with the Uno format, the most appropriate choice is to have them on a shield; it's also the problem of power management and security. >

“ THIS NEW BOARD WILL BE FULLY SUPPORTED BY THE ARDUINO ECOSYSTEM, AND WORK WITH EXISTING UNO SHIELDS ”

MKR VIDOR 4000

Design your own hardware

Up until now, Arduino has focused on creating microcontroller boards.

On these, you upload a program into the board's storage and the on-board processor runs this code. The MKR Vidor 4000 is Arduino's first foray into a whole new class of technology – field-programmable gate arrays (FPGAs). FPGAs contain a large number of logical bits of hardware that can perform logical actions, and you can connect them together in different ways.

A specification for how to connect the different logical elements is known as a bitstream, and it contains intellectual property (IP). This IP could, for example, contain a specification for a USB port, or a circuit to manipulate an image in a particular way.

The main difference from a user's point of view is that, while a microcontroller and an FPGA can both be programmed to perform a specific action, the microcontroller does it by stepping through a program one line at a time, while the FPGA does it in a circuit that typically can do a large number of calculations at the same time. In many cases, the FPGA processes data much, much faster than a microcontroller can.

The downside of this, in general, is that historically, FPGAs have been a great deal harder to program than microcontrollers. However, of course, Arduino has a plan... →

THE POWER OF FPGA

HackSpace What sort of applications do you have in mind for the FPGA?

FV Basically, this is FPGA tech in a small form factor and this can be used for some video applications (but it isn't a powerful video adapter like the Raspberry Pi has). The idea is for people who are familiar with the Arduino language, but want more capabilities in terms of signal processing or creating a different set of digital inputs and outputs ... take tools for doing precise motor control: this requires a kind of real-time requirement and high parallelism, which is the kind of thing we have in mind. In 3D printing, this kind of solution could be very important because it can very simply increase the precision of motor control by orders of magnitude.

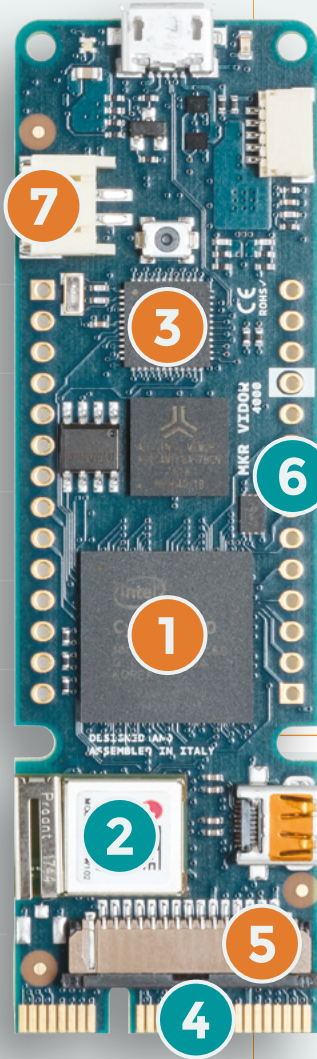
The nice thing about this is that you don't have to know FPGA technology in order to use it. The idea is to release a development environment to allow the users to just drag and drop the predefined blocks of intellectual property – a sort of environment like Blockly – and they can create their own peripherals this way.

HackSpace Will people be able to write their own IP to add functionality to the FPGAs?

FV If they know how to do that and if they know the Intel toolchain, they can program whatever they want with the FPGA. It's quite a complicated kind of technology, but if they know how to, for sure.

We think that – especially in universities – this is also a way we can increase the quantity of IP that is available for the board. We are committed to developing a number of intellectual properties, but of course we are looking for the creativity of the community as far as open-source IP is concerned.

- 1 Cyclone 10 FPGA which includes 16K Logic Elements, 504kbit of embedded RAM and 56 18 x 18-bit HW multipliers. Each pin can toggle at over 150MHz and can be configured for functions such as UARTs, (Q)SPI, high-res/high-freq PWM, quadrature encoder, I²C, I²S, Sigma Delta DAC, etc.
- 2 U-blox module for WiFi/Bluetooth Low Energy
- 3 SAMD21 Cortex-M0+ microcontroller that can be used in the same way as traditional Arduinos
- 4 Mini PCI Express port with programmable pins
- 5 MIPI camera connector
- 6 MKR form factor headers. Each pin can be controlled by both the microcontroller and the FPGA. With the latter, each can be configured with functions such as UART, SPI, high-resolution PWM, or any many other hardware protocols as the pins can toggle at over 150MHz
- 7 LiPo battery connector



THE HACKSPACE VIEW

Bringing together a microcontroller with an FPGA in an established (well, establishing) form factor opens up a lot of potential. The FPGA enables you to do things in orders of magnitude faster than the microcontroller, but FPGA programming is a very different prospect to microcontroller programming, and currently there are few hobbyists working with FPGAs.

The development environment Arduino is creating will make it much easier for people to configure their Vidor 4000, but only with pre-created blocks of IP. The success or failure of this board depends entirely on what IP is created for it. If hobbyists with the skills (or corporations with the money) to develop useful IP get behind this, then it can open up a whole new chapter in maker electronics.

Getting a critical mass of IP for the Vidor is a challenging prospect, but 15 years ago people might have said the same thing about creating a microcontroller environment that works for artists and designers. If anyone can do this, a newly invigorated Arduino can. We wish them luck because this could be a fantastic new ecosystem.

SHARING FPGA CODE

HackSpace This environment for dragging and dropping IP around – will it work on a wide range of FPGAs?

FV At the beginning it will only work on the Vidor family of boards because there is an underlying compilation process that is required – which we will do in the cloud – using a proprietary toolchain from Intel Altera (the provider of the FPGA that we use – the Intel Cyclone 10 in this case). Initially this environment will be tailored to the environment we use, but in the future it can be expanded.

At release, we will have a simplified version. A predefined set of IPs will be made available at the commercial launch in the second half of June where people can just include an Arduino library, and the IDE will run all the tasks that are needed to load the image for the FPGA, and we provide the library to use the intellectual property with the Arduino, so it's extremely simple.

People don't even have to know any details about the FPGA. They just say "I want four USB ports" and they just include the USB IP and then they can use it very simply.

ARDUINO SOFTWARE

Better tools makes for better developers

The hardware is only part of the Arduino story. They're just as famous for the integrated development environment (IDE) that makes it easy to develop software for a wide range of microcontroller boards – not just ones

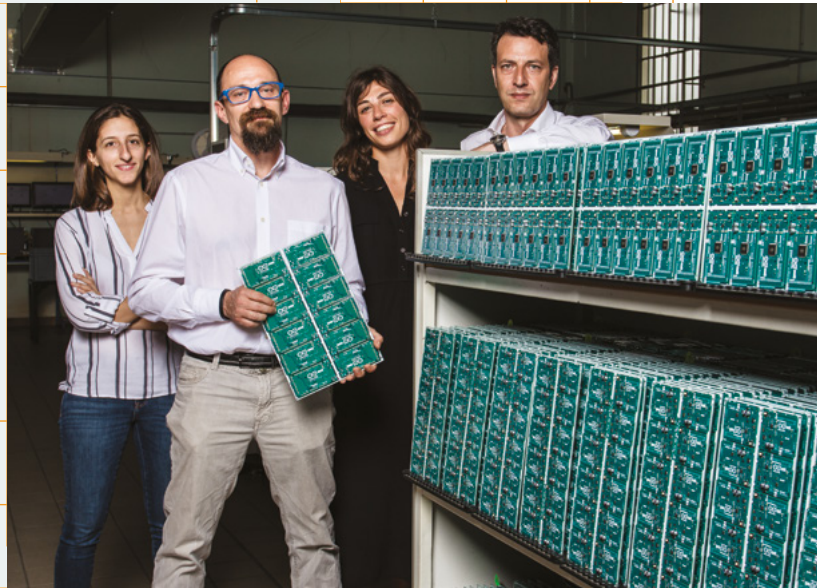
they make. Despite being around for 15 years, the IDE is only in version 1. However, the Arduino team are rapidly working towards version 2.0, which is bringing in some pretty big changes.

The headline feature for version 2.0 will be a debugger. This is a tool that enables you to peek into the code as it's running and see what's happening. It's a bit like a supercharged version of using serial.print() statements to send debugging data over the USB connection.

As well as the traditional development environment for microcontrollers, there's a new web-based environment which combines a code editor with a communication channel that lets you move data between different devices or aggregate information from lots of nodes in a single place. It will also have the ability to create web-based dashboards to display information from your Arduino boards.

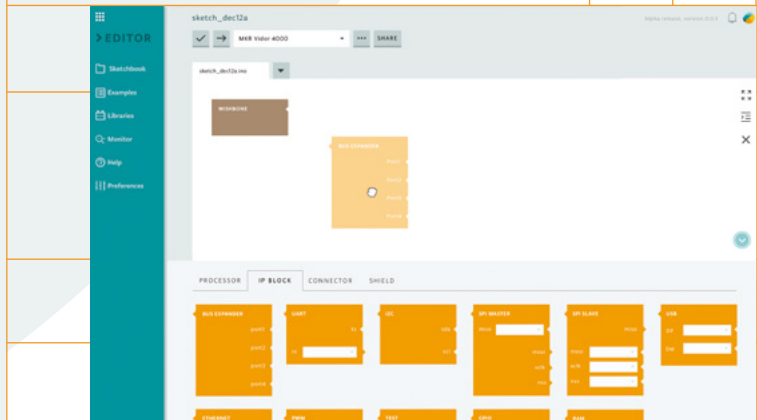
With the release of the MKR Vidor 4000, Arduino needs a whole new style of development environment which lets you build the bitstream for the FPGA. The tool allows you to drag-and-drop predefined pieces of IP into your project. This works alongside the standard Arduino IDE where you can include the software that's needed to utilise the IP.

These three new features are a really wide-ranging set of improvements which will impact almost everyone who uses the Arduino IDE.



Above ◆
Arduinos waiting to make the world a more connected place

Below ◆
An early version of the FPGA environment with drag-and-drop IP



DEBUG HAPPY

HackSpace Can you tell us about the Arduino debugger?

FV We worked quite heavily on the debugger side and now we have a working back end for that. We're working with our UX designers to create a simple front end, to make it accessible for people who are not really sophisticated code developers. I'm pushing quite a bit on this because it's an important development and Arduino was missing this for a long time.

HackSpace Will it work on older boards?

FV We're trying to make it more comprehensive – we have some boards that are very limited, but we're trying to do a good job on that. The idea is to be as inclusive as we can. In some cases there are limitations on the hardware that make it impossible to do that, but we are trying to do whatever we can to cover the widest range of use cases.

The other thing we are doing, as far as development is concerned, is that today, many Arduino users are forced to go and connect to different clouds, so we worked heavily on what we called the Arduino Things Cloud, which is part of our IoT strategy. In that area we made a lot of

progress in making a super-stable back end that can connect the end nodes to the cloud, and we're going to release some of these capabilities in June. We are also working with the user experience of data in the cloud, for users to build dashboards and applications – they can be simple applications but also more complex ones. This is a path that is also part of our strategy.

We are putting a lot of effort in this area as well – allowing people to connect to third-party clouds – but we are trying to remove the trouble of connecting devices to the cloud from end-node security, up to having the data on the cloud and creating dashboards. This is an important part of our strategy.

We were a little bit late to the game, but we're trying to make it simple for users and this is probably the most important differentiator for [Arduino] users, especially the professional ones. I think the community from Raspberry Pi is more inclined to using cloud services and high-level complex software. The Arduino community comes more from the embedded world, or they are designers and makers – even professional users in SMEs don't have all the knowledge to use the cloud, so we would like to simplify the path for that so it's complementary to the software.

THE HACKSPACE VIEW

The Arduino development environment hasn't changed a lot over the years, and while this stability is great, it can leave it a bit sparse. The debugger will make it much easier to find out what's going on in your code, and is, frankly, long overdue. The IDE will still split opinion – the simplistic interface remains, which still lacks features for managing larger projects, but that's OK: there will probably never be a development environment that will please everyone and you can develop for Arduino from other environments.

FORZA ARDUINO!

After a bit of a slow start, Arduino is finally catching up with the modern, connected world, and it's great to see this supported fully from encryption chips on the boards to a back end that's easy to use. Up until now, the whole solution has been bitty, requiring users to use, and understand, quite a range of different platforms. With the new Arduino Things back end, users will just be able to get a board, connect it to the local WiFi network and their online account, and then push data back and forth. It should make it much easier to create connected devices.

In the last year or so, we've seen the first real challengers gain a foothold against Arduino's dominant position in the field of hobbyist microcontroller development environments. Circuit and MicroPython, MakeCode, and others are now becoming popular (at least in some fields). We're now seeing Arduino respond to this challenge, and it's making a strong case for itself, particularly in connected devices.

From both a hardware and software perspective, there's never been a better time to get into embedded computing. The hardware and software feel fantastically powerful compared to where they were just a few years ago. Watch this space – Arduino is taking off. □



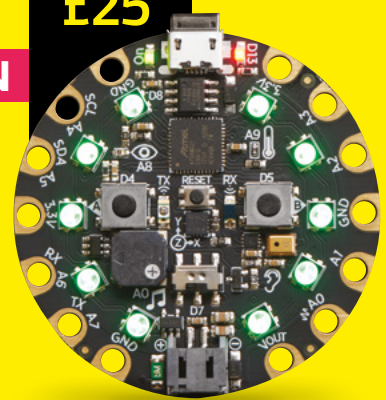
Right ■
Andrea Richetta,
Product Manager,
shows the MKR
Wifi 1010



CIRCUIT PLAYGROUND EXPRESS

WITH 12-MONTH PRINT SUBSCRIPTION

WORTH
£25



FROM JUST
£55

**12-month
subscription
from £55:**

- UK: £55 per year
- EU: £80 per year
- US: £90 per year
- RoW: £95 per year

Visit: hsmag.cc/subscribe

SUBSCRIBER BENEFITS ↓

SAVE UP TO 35% ON THE PRICE
FREE DELIVERY TO YOUR DOOR
EXCLUSIVE OFFERS AND GIFTS
GET YOUR COPY BEFORE STORES

OTHER WAYS TO SUBSCRIBE

Rolling subscription from £4 a month:

- Quick and easy to set up
- Cancel any time
- No long-term commitment
- No large up-front cost

Digital subscription from £2.29 a month:

- Direct to your mobile
- For both Android & iPhone
- No delivery fees
- Back issues available



Visit: hsmag.cc/subscribe

How I Made ELECTRIC SYNTH GUITAR

Electric dreams do come true, with this unique creation. Cue Daft Punk!

By Matt Bradshaw

The electric guitar has remained largely unchanged since its invention. However, as modern recorded music becomes increasingly electronic, the guitar as a live instrument is in danger of becoming obsolete. Many musicians now base their sets around laptops which generate complex layers of synthetic sound, but there is something unsatisfying about a live 'performance' which consists largely of someone operating a computer. When a musician's physical actions are not obviously correlated with what you are hearing, you're never quite sure if they are just hitting 'play' and then spending the rest of the set checking Facebook.

After going to a gig by the Icelandic musician Björk, I was inspired by seeing how she used spectacular instruments that she had commissioned for the show, including a 'singing' Tesla coil and a giant pendulum which plucked harp strings. Thus was born the idea for my instrument Bjarkardóttir (Icelandic for 'Björk's daughter'), a synthesizer that can be played like a guitar.

While there have been various attempts in the past to push the guitar into the realm

of electronic music, they have tended to focus on controlling an external synthesizer. What I wanted to do was to build a standalone instrument that could be turned on and played instantly, with every element of the sound controllable from within the guitarist's reach.

HOW IT WORKS

My synth-guitar can be played approximately like a regular guitar: the left hand plays chords and notes, while the right hand plucks the strings to trigger the sound. However, unlike a regular guitar, there are no physical vibrations creating the sound. Instead, the notes a user plays are converted into digital information and fed into an on-board synthesizer, which generates the sound.

To detect chords played on the neck, a small voltage is passed across the six metal strings. Unlike a normal guitar, each fret is split into six metal contacts, with each contact acting as a digital input. If a section of the fret is in contact with a particular string, a circuit is completed and the processor knows which note to play.

To detect plucking with the right hand, a similar (but simpler) circuit is used.



Above ♦
Bjarkardóttir



Left ♦
Building electronics into the neck of the guitar

Six separate strings (actually lengths of threaded metal bar) are fixed to the guitar's body, each one again acting as a digital input. When a string is touched with the metal plectrum (which is part of the circuit), a voltage is detected and the note is triggered.

While this playing mechanism takes some time to get used to, it is perfectly possible to play both chords and melodies accurately with practice. However, generating musical data is only half the battle – we also need sound...

A TEENSY BRAIN

To produce sound (and manage the data from the guitar's many inputs), I needed a microcontroller, and decided on a Teensy 3.6 with an audio add-on board. For the uninitiated, the Teensy is roughly like an Arduino but with more processing power, more inputs, and a brilliant audio library which allows you to build a fully functioning

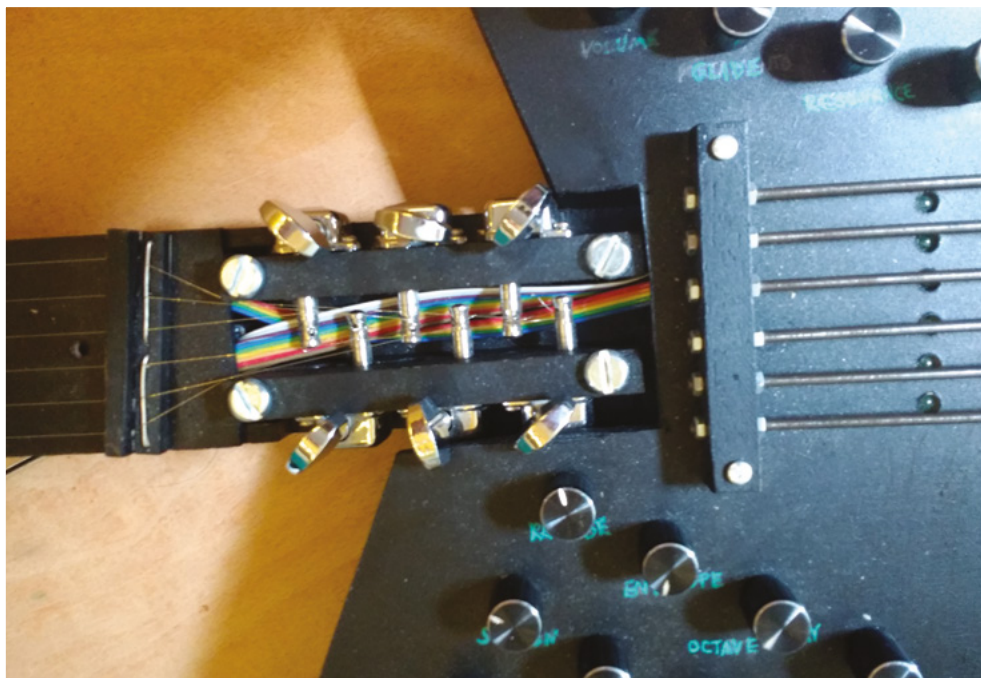
synthesizer from within the Arduino IDE. I did consider using a Raspberry Pi, but the near-instant boot time of the Teensy was the deciding factor.

I was able to write most of the code (and test it fairly extensively) before the project started to look anything like a guitar. Although the Teensy is pretty fast, I still had to do a lot of optimisation before I could get the latency (the time between

plucking a note and hearing it) down to an acceptable value. The sheer number of digital and analogue inputs being read made things difficult: there are 72 fret contacts, 12 'strings', 28 potentiometers, five toggle switches and an arcade push-button, not to mention the LEDs.

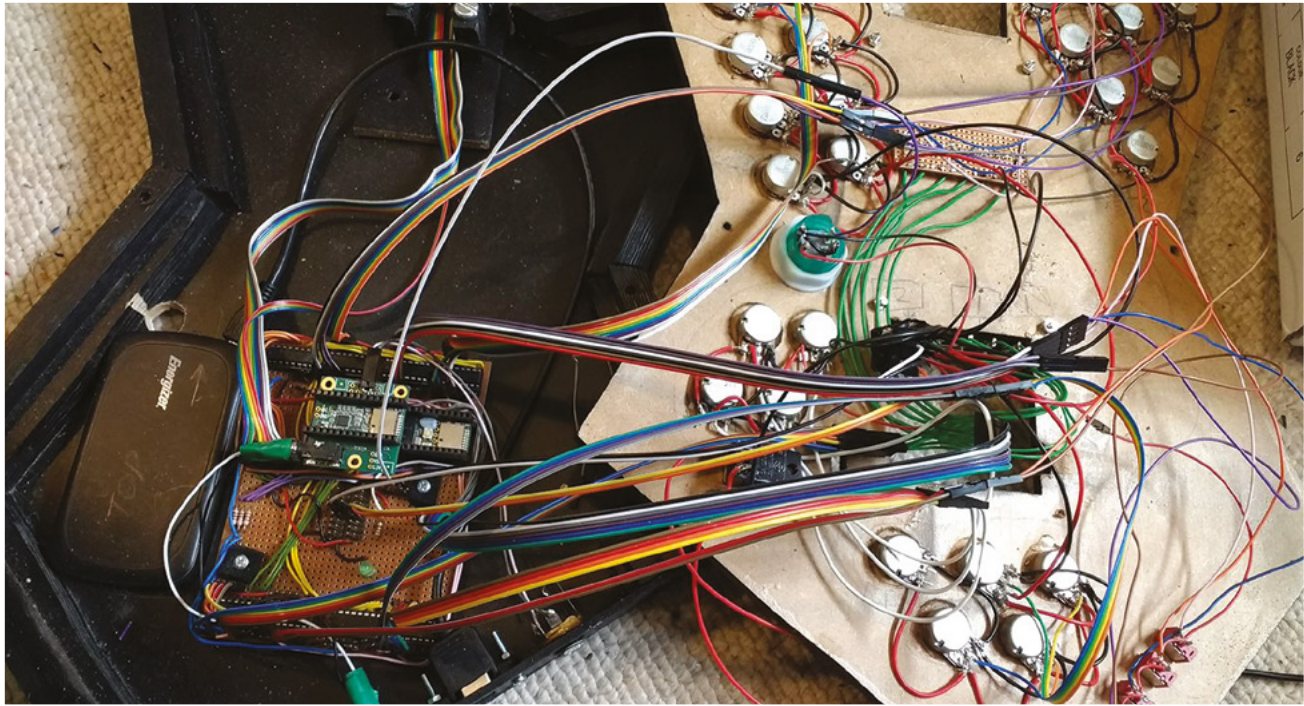
One sacrifice I had to make during this breadboarding stage was capacitive touch. My original idea was for everything to work by just touching the frets and strings, allowing naturalistic left-hand playing and finger-picking with the right hand. Sadly, however hard I tried, I couldn't get satisfactory speed or sensitivity using this method. It's possible that different materials could have helped, or that my code or wiring was inefficient in some way, but there came a point (while hacking the Teensy's core libraries!) that I decided I had spent enough time on this one feature, and gave up. I added a metal plectrum to the body and modified the design of the neck so that both would simply read conductivity >

Below ♦
The join between neck and body; ribbon cables carry fret data to the main board



Teensy audio library

The guitar's synthesizer code features a huge number of oscillators, filters, envelopes, and other effects, all connected together with a complex web of virtual audio cables. Luckily, there is an intuitive drag-and-drop interface for the Teensy audio library, which allows you to create your audio setup in flowchart form, then click 'export' to generate code which you can copy and paste straight into the Arduino IDE.



instead of capacitance. It's the part of the project that most frustrates me (particularly because I have since seen a similar project with capacitive touch working fine), but it was necessary in order to push the guitar towards completion.

BEYOND THE ALTOIDS TIN

My previous musical electronics projects have always suffered in both their

aesthetics and reliability. I have shown up to gigs with Altoids tins stuffed with badly soldered wires, and I have had a large audience wait patiently for me to debug a robotic glockenspiel with a fried stepper motor. This time, I told myself, I wanted things to be different.

My plan was essentially to build a strong box in the shape of a guitar, with as much room as possible for circuitry inside. I sketched a design that evoked a Fender Stratocaster using only straight lines, partly for the angular eighties aesthetic but mainly for ease of construction. With the hazy recollection of my A level in woodwork, I constructed the design from MDF and scrap wood, attempting to over-engineer wherever possible with judicious use of wood glue, screws, and bolts.

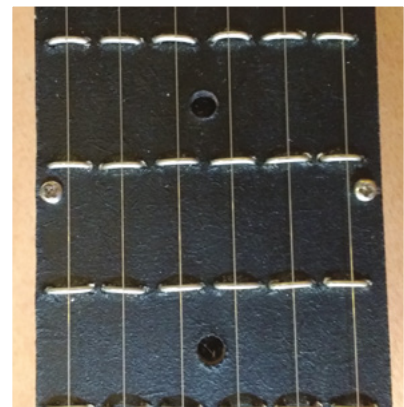
Having built a sturdy body, I wanted to make the electronics similarly reliable. Realising that I would need to tweak or repair the circuitry at some point, I tried to make the system as modular as

Above Inside the body, including a battery which wasn't used in the final version

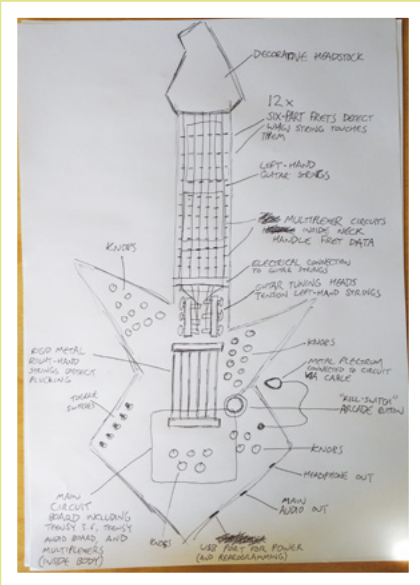
possible, using headers to attach elements such as the potentiometers to the main circuit board. While I think this approach was broadly correct, I made one major mistake: I designed the main circuit board to be mounted to the bottom part of the guitar, while the potentiometers, buttons, and switches were mounted on the top surface. This meant that both assembling and troubleshooting the circuit were tricky tasks, with tangled wires spanning two

An accidental remake

Halfway through this project, I stumbled across a very similar 'guitar' from the eighties: a MIDI controller called the SynthAxe. It too had six-part frets and separate strings for the left and right hands, and initially I was annoyed that 'my' idea had already been done (in my own county of Oxfordshire, in fact!). However, the SynthAxe originally retailed for £10000 and failed to catch on, despite later developing cult interest, so I felt justified in pushing on with my much cheaper design, hoping that it might allow some people who wanted to own a SynthAxe to build their own custom version from cheaper parts; I estimate that I spent about £100-£200 building Bjarkardóttir.



Right The fretboard, with six-part conductive frets to detect fingering



Above ♦ All good projects start with a diagram

parts of the guitar. In hindsight, mounting everything on the top surface would have made a lot more sense. Another decision I would change in retrospect would be the circuitry for the neck, which I squeezed inside the neck itself, making the fretboard wide and hard to play. Giving the guitar a bigger body in the first place would have made more room for circuitry and allowed me to have a less chunky neck.

One of the final parts of the build was to decide the function of each knob. I squeezed as many potentiometers onto the body as possible, knowing I would want control over a wide range of parameters in the Teensy audio code. After many happy hours experimenting, I settled on a layout where the more frequently adjusted (i.e. coolest) controls were within closest reach. It's nice to know that changing the layout in

Right ♦ The guitar is played with a conductive plectrum

Far Right ♦ Are you ready to rock?

future will require only a few lines of code and some new labels.

CONCLUSIONS

I have become too attached to my guitar to think of it as a prototype, yet I suspect it is; it contains both too many flaws and too much potential for there not to be another iteration. I was proud to be able to show off my guitar at Maker Faire UK this year (and prouder still that it survived a full weekend of testing by enthusiastic

children), but I learnt a lot about what I would change next time.

While the sounds the guitar generates are unique and interesting, the playing



style is definitely unintuitive. I had hoped that any guitarist would be able to pick up Bjarkardóttir and play it unthinkingly, but Maker Faire disproved that. That said, there is nothing stopping me from refining the design, perhaps using something closer to an existing guitar neck, and I would like to have another go at making the right-hand strings touch-sensitive. I would also like to add a battery, possibly even a speaker, and expand the range of sounds available, all of which should be relatively simple.

Overall though, I'm just happy to have made a guitar that – despite its flaws – looks and sounds unique, and won't break halfway through a gig. I hope that this project will

inspire people to have a go at making their own instrument, because the technology available today makes it easier and more rewarding than ever to do so. □



THE OPEN SPACE AGENCY

Meet the team that wants to give us the power to search the depths of space.
Jean Luc Picard would be proud of them



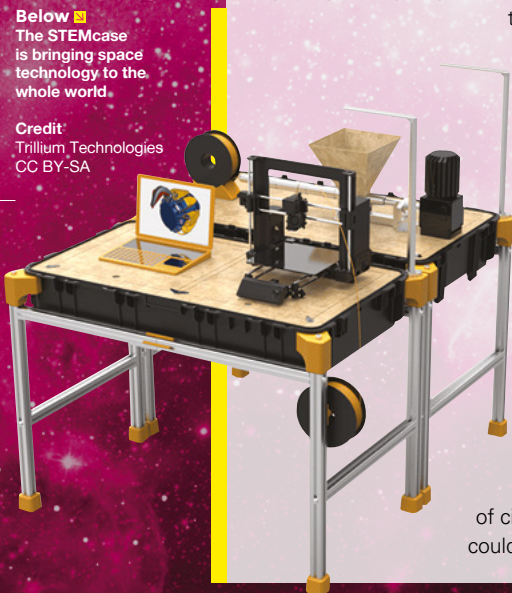
Cameron Norris

@cameronnorris

Cameron is a technology and communications specialist, passionate about the use of open-source hardware for social innovation.

Below ■ The STEMcase is bringing space technology to the whole world.

Credit ■ Trillium Technologies CC BY-SA



We're used to seeing space as something untouchable. To get into space you need billion-dollar budgets, decade-long development cycles, and the sort of organisational clout that only PayPal squillionaires and nation states can muster.

But if you can't take yourself into space, you can at least bring space to you. Today, the emergence of low-cost 3D printers and laser cutters – paired with microcontroller platforms such as Arduino and Raspberry Pi – have driven down the cost of space technologies to a point where individuals with just enough engineering and fabrication know-how can launch their own space exploration program.

London-based inventor James Parr founded the Open Space Agency (OSA) to enable citizen scientists

to meaningfully contribute to space exploration from here on Earth. The Open Space Agency developed the Ultrascope (hsmag.cc/IWYqvc), a professional-grade, asteroid hunting, automated robotic observatory (ARO), able to conduct celestial photography and photometry to help NASA detect and track near-Earth objects, to see just how far the idea of citizen space exploration could be pushed.

James describes how the rapid evolution of consumer technology had made him wonder whether it was possible to replicate the achievements of the space program using off-the-shelf technology. "The concept for the Ultrascope arose when I realised that the tools and technologies to pull it off – cloud computing; high-speed phone networks; low-cost, high-performance chips and CCDs [charge-coupled device – see page 52]; 3D printing; and the maker movement – had all matured around the same time, enabling a new era in citizen science."

Today, millions of CCDs are manufactured each year to be used in smartphone cameras. Thanks to CCD technology, the quality of a modern smartphone camera is essentially the same as those found on NASA's Mars Rover.

By leveraging these powerful new technologies, the Ultrascope can communicate with satellites to determine its exact location on Earth, enabling the 3D-printed telescope to automatically target and photograph celestial bodies like stars, planets, and asteroids in the night sky. The resulting images can then be uploaded to OSA's cloud database for post-processing and analysis. The idea is that, by creating a large, distributed network of these telescopes all over the world, the Ultrascope community will be able to make scientifically valuable observations by combining data from multiple locations.

CROWDSOURCED ASTRONOMY

One of the first members of the OSA community was Matthew Nelson, a sales rep for a radio controls company in the United States. "The design really lends itself to be understood by a complete novice

“James hopes the Ultrascope and its users will play a significant role in NASA’s Asteroid Challenge Lab, which encourages citizens to assist in seeking out asteroids that could pose a potential threat to Earth”

with no technical background,” he explains. “It might look complicated, but once you have printed the pieces, it goes together like LEGO or a puzzle.” Matt describes how he built his Ultrascope for around £200, making it many times cheaper than existing, commercially available AROs. The design and software itself is open-source, enabling anyone to create and distribute improvements or modifications to the system under the CERN Open Hardware Licence.

James and his team have always been interested in the collision of ‘DIY engineering’ and the emerging ‘space entrepreneur’ movement. They wanted to create a space-themed project that would appeal to established makers and amateur communities around the world, while also proving that it’s possible to develop open hardware that is able to contribute and scale citizen science with a level of data accuracy that is valuable to researchers.

“Citizen scientists can also make the tools for research,” says James, “which we believe is the next chapter of citizen science: huge science projects that are conducted by armies of highly skilled, data-gathering individuals who build the tools to do the work. The Ultrascope project is this – a passionate, highly skilled, data-harvesting army who build their own devices to help save the world.

James hopes the Ultrascope and its users will play a significant role in NASA’s Asteroid Challenge Lab, which encourages citizens to assist in seeking out asteroids that could pose a potential threat to Earth. NASA estimates that less than 10% of potentially hazardous objects smaller than 300 metres in diameter, and less than 1% of potentially hazardous objects smaller than 100 metres in diameter, have been discovered.

The Ultrascope’s primary purpose is to “close the gap in the Southern Sky” by providing an ultra-low-cost robotic observatory for citizen scientists to support the work of professional astronomers in the Southern Hemisphere. →

HOW DO YOU MAKE A WHISKY GLASS FOR SPACE?

Preventing extinction-level events and planetary destruction is all very well, but the Open Space Agency has been working on an even more important problem: how to drink whisky in space.

After the success of the Ultrascope project, James Parr and the Open Space Agency were commissioned by blended Scotch whisky company Ballantine’s to create a 3D-printed whisky glass that works in zero gravity.

The OSA team studied how whisky behaves in zero gravity as they built a series of prototypes, which included rotating discs and spinning bases that closely resembled laboratory centrifuges. The prototypes were then tested using the ZARM Drop Tower in Bremen, Germany to simulate weightlessness, while the results were captured in super slow motion.

“In space, you can’t sip whisky out of a glass because it floats around. You can’t pull air; by sucking you will compress the air around the liquid, but you won’t move anything. To solve the problem of drinking in space, we needed to find a way of actually capturing the liquid and then to allow capillary forces to pull the liquid up into the mouth,” explains James.

Sipping whisky in space

The Space Glass is made from the same medical grade 3D-printed PLA used for heart valves that go inside human bodies. The glass has a spiral convex base plate that creates surface tension to hold the

whisky down in a reservoir at the bottom of the glass. The base plate is stainless steel coated in rose gold, to prevent the glass from affecting the whisky’s taste.

Filling the Space Glass with whisky and drinking it has been engineered according to a four-step process, explains James. “Motion one pulls the whisky into the base of the glass; then motion two is to roll the whisky in your hand and let the heat transfer through the metal base into the liquid itself. Step three involves then moving the glass down before moving your nose into the area where the vapours are resting. The final motion is to move the glass upwards to capture the liquid in the base plate and let it enter your mouth.”



Below There is no word yet on a space martini glass for orbital 007s

Credit
Open Space Agency
(CC-BY-SA_)

“If you’re a maker, DIY engineer, citizen scientist, or just a long-time aspiring astronaut with stars in your eyes, then we’d love to hear from you”

Below ♦
The dust-, water-, and shock-proof protective case

Credit
♦ Trillium Technologies CC BY-SA



GETTING THE PICTURE

A CCD, or charge-coupled device, is a light-sensitive integrated circuit that captures light and converts it to digital data. CCDs can produce an image in extremely low light, which has made them a transformational technology for astronomers by providing a far more sensitive light sensor than had ever previously existed.

Development and South African Astronomical Observatory in Cape Town. The ‘Africa team’ flew out to Cape Town and spent six weeks working on developing and testing the Ultrascope, while conducting community outreach activities on the ground. The result was a series of incredible photographs of the night sky, Jupiter, and the Galilean moons.

Over the course of these six weeks, the team conducted live-streaming tests to various events in the US, testing remote access, the web app, global broadcasting and streaming, and most of all the capabilities of the scope to identify objects in the Southern Sky. The team successfully sent images directly from South Africa over 4G via the cloud to major public events, including two events held at the WIRED conference in NYC and one at Re/code in LA.

“If you’re a maker, DIY engineer, citizen scientist or just a long-time aspiring astronaut with stars in your eyes, then we’d love to hear from you,” says James.

ASTEROID HUNTERS

While the technology to support the work of professionals is already available, the cost of conventional kits is far beyond the reach of most. However, by using 3D printing, along with low-cost microcontrollers and smartphones, the costs associated with scaling up the number of observations can be radically reduced.

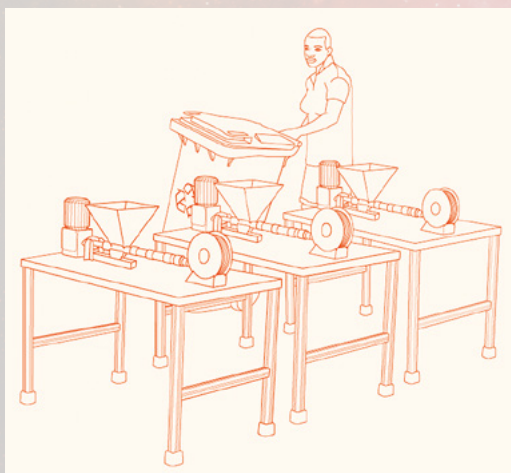
The first proof of concept was to see whether an Ultrascope could be built in South Africa and used to live-stream a series of images of the Southern Sky to the USA. The Ultrascope team had the opportunity to work with the experts at the Office for Astronomical

MARTIAN MAKER MOVEMENT

However, James’s interests aren’t just limited to planet Earth; he aims to redesign all of the things we take for granted here on Earth so that they can be replicated in space. “We want to be able to create things off-planet; in space jargon, this is called in situ manufacturing,” he explains. The benefits of enabling astronauts to manufacture on-site are painfully obvious; it currently costs around £15000 to put a single kilogram of payload into space. “When we get to Mars, many of the things that we build and manufacture, from the very large to the very small, will be done by 3D printing, using Martian resources. So we were very interested in the challenge of being able to use 3D printing to create something that could ultimately be manufactured in situ – either in orbit, or one day on Mars, without having to be transported millions of miles,” he says.

Below ■
A full maker lab fits into just two cases

Credit
Trillium Technologies CC BY-SA



AN OPEN-SOURCE PORTABLE MAKERSPACE FOR THE DEVELOPING WORLD

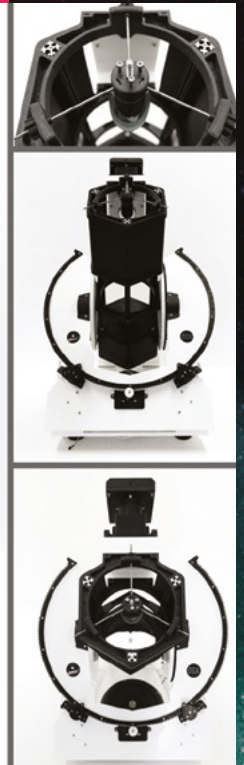
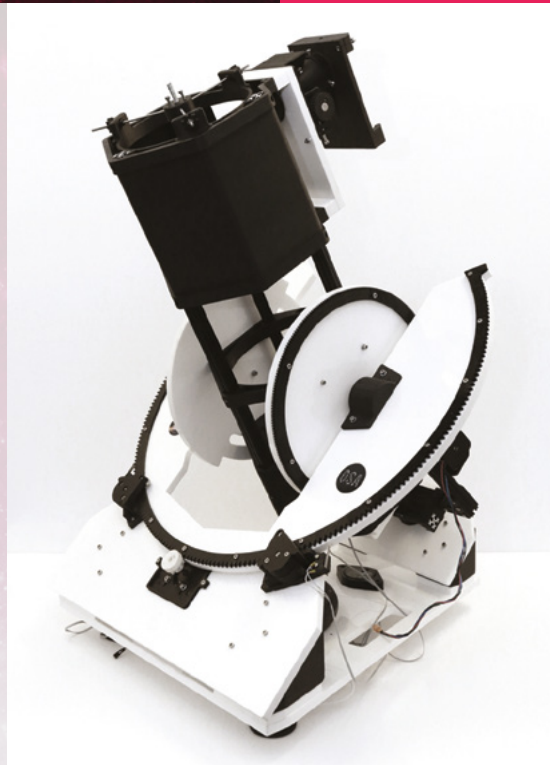
That's the plan, at least. But in practice, a 3D-printed space telescope, no matter how much cheaper it is than established hardware, is never going to be built unless people on the ground have the skills to build them. To solve this

problem, the Open Space Agency has also developed the STEMcase, an open-source portable makerspace, designed to facilitate the growth of maker skills in the developing world back down on planet Earth. Marième Eve Jamme, CEO of SpotOne Global Solutions, is a strong advocate of improving STEM-related educational policies in Africa by utilising the maker movement. She strongly believes in supporting initiatives that channel incoming foreign aid and investment into the right educational programs to upskill the youth, as the vast amount of STEM-related job opportunities in Africa are currently being outsourced. She views the STEMcase as one such possible solution.

The STEMcase consists of a set of two dust-, water-, and shock-proof Pelican cases designed to deliver STEM education to remote locations. The complete kit includes a fully equipped electronics prototyping bench, a low-cost and repairable Raspberry Pi-based laptop, a 100% recycled filament extruder, and an open-source Prusa i3 MK2 3D printer, which is considered by the 3D Hubs community to be the best low-cost desktop 3D printer currently available. The final part of the kit is an open-source SatNOGS ground station, which provides essential satellite communications to remote locations.

SatNOGS provides the plans to build satellite tracking stations, and a protocol and framework to share the satellite data with those that cannot afford or lack the skills to develop their own tracking station. The SatNOGS hardware consists of readily available materials, off-the-shelf components, and 3D-printed parts.

The Open Space Agency believes that the argument for STEM is not just an economic one; the democratisation of technical skills will aid everything from climate change to the control of epidemics.



OPPORTUNITY TO ACT ON IDEAS

The STEMcase comes with a syllabus designed to help educators focus on stimulating interest in STEM development within the local community. The idea is to work directly with keen locals to eventually take OSA out of the loop entirely, leaving a self-sufficient makerspace ready to propagate ongoing maker activity in a developing area.

"By guiding the flow of [overseas aid] to projects and initiatives that will allow the recipient country to flourish, they will be the creators of their own independence, be it economic, social, or health-related," says James. "The educational aspect that comes with the STEMcase is just as important as the capabilities of the case itself." You can find out more about the STEMcase at hsmag.cc/bSQyQL. □

Above 📷
The Ultrascopes –
celestial photography
from home

Credit
Open Space Agency
(CC-BY-SA)

Below 📷
Print an Ultrascopes and
join the community





Above ■
"People come round the back and expect some complex system, but most of it is a rubber band and a microphone"

HackSpace magazine meets...

STEPHEN SUMMERS

Making toys noisy

W

hy do you make things? Some people do it for the pure fun of creation, others do it to gain a better idea of how things

work. We met up with Stephen Summers of Noisy Toys, who combines making and performance art to help youngsters get a better understanding of the science of music. Electronic conduction, sound waves, and upcycling are all mashed together to create an interactive experience that helps people learn by discovery.

We chatted to Stephen to find out what exactly science busking is, how he makes musical instruments, and what you need to test noses for. →

HACKSPACE How did Noisy Toys get started?

STEPHEN SUMMERS It pretty much started as an educational project and it still is, at the heart of it. I'm a teacher and the initial idea of Noisy Toys was going into schools and doing workshops with kids about sounds, crossing over between music and physics – trying to get the exact middle point of those two subjects, which is often missed with the crossing over between creative and technical subjects. Since then, it's shuffled into a few different things. We still do the workshops, we do larger-scale installations, and science busking.

HACKSPACE Science busking?

SS Yeah, it's not a term I was familiar with myself, it's just something events use. I don't know about it though, because I've mentioned it to some people and they say, "are you charging on the spot?" and I say "no." It's a walkabout, basically. Interactive walkabout... we stop somewhere, we do nose testing. It's not a performance in that the audience is passive – it's something they partake in.

HACKSPACE Nose testing?

SS Ha, yes. Nose testing is a simple conductivity experiment. There's lots of them around, but ours is the best! You basically hold one contact, another person holds another, you squeeze their nose, and the system goes 'brrrrrup!' Some people are a bit scared to do it, and we play with that. If they're confident, we tell them that it hurts a bit... you know, there's a bit of psychology in it.

You get whole groups – family groups. You can extend the circuit and get everyone holding hands and squeezing noses in turn. You can go into basic circuitry: the finger on the nose is a switch and we usually have a few spoons – a wooden and a plastic spoon. A carrot

ideally. If it's an urban space, we try and test dogs as well. The muzzle and the paw works, but not the fur. It's a bit of fun with some conductivity in it.

We came up with that because the big installations are a pain to set up – they take all day. The [nose-testing] buggy – we lift out of the van, switch it on, and we go. We really like doing that. It's the smallest thing we do.

Here at Maker Faire UK, every day we're doing Bass In Your Face shows. This is something I developed for the Big Bang Faire. I'm only doing a mini-show. It's ground-level, and again it's not a passive show. People come up at various stages and put their hands in a

//

There's very little digital technology. It evolves. We keep it as simple as possible. It's pretty much all analogue acoustic

//

speaker cone, and feel 1000 watts of bass vibrations through their fingers. We do it with corn-flour, and people can trigger the bass with the hard drives. It's a good show, and it's also quite interactive.

There's Scavengers, [a programme] in schools. It's about unmaking computers – taking the electronics apart, and then upcycling the bits to, in our case, little noise machines or acoustic instruments – learning about upcycling in the process. Looking at what's inside the magic boxes (such as computers), rather than just seeing it as a magic box that does stuff. You take things apart and learn how they work. There's also a lot of sustainability with that one. You learn about where the materials come from, and where they go to afterwards. There's a focus on that.

The big installation is called the Acousatron. That's something we've been doing for years. We developed some of

that with the Manchester Robot Orchestra guys [as seen on page 18]. We've got mechanised acoustic instruments, like pianos and violins. You press buttons and spin little fans, and it strums the strings. Those are the four main things really.

HACKSPACE How do you build your stuff?

SS It's always as straightforward as it can be, so it's usually much, much more simple than people expect, especially the installation. People come round the back and expect some complex system, but most of it is a rubber band and a microphone. That's my ideal level to go for – a rubber band with a contact


microphone plugged into a PA system. That's great. If I can get away with everything at that level of tech, I will. The moment I want to do something and that level won't do it, I might introduce a little more, but I will only introduce more when I have to do something that I can't achieve otherwise. There's very little digital technology. It evolves. We keep it as simple as possible. It's pretty much all

analogue acoustic. Amplified acoustics, mechanical using a lot of upcycled bits and bobs.

HACKSPACE Have you got any tips for finding bits to upcycle into musical instruments?

SS Yes! Especially the bass drives! I've been promoting these things for years, and I can't believe I don't see them everywhere! I've still never seen anyone else use them, but apparently someone has. A hard drive is stripped down, and some leads are soldered onto it, and it makes the most amazing bass generator. I've done a YouTube video on how you can do that (hsmag.cc/WCuUAm). I love 'em. You just spin them, and amazing bass comes out.

The one thing musically most useful for me is the contact microphone. I use them a lot – I'm currently studying →

Left  A properly tested and calibrated nose





for a Masters in electro-acoustic composition. Pretty much everything we use is acoustic-found sounds, so contact microphones – I love 'em. They're an endless source of fun.

HACKSPACE How have things changed over the years you've been running Noisy Toys?

SS We've got better at it! It's evolved. When we started Noisy Toys, we had little boxes and theremins and light-sensitive boxes on tables. One of our first big gigs was in Newcastle at the Discovery centre and it was literally that. We had them on tables and had a bit of a talk, "this is de-de-de." When we started doing music festivals, we got more of a performance element. At Glastonbury, we had to build it all into a large console and make it look a bit more steampunk and think about it visually.

HACKSPACE What did you do when you were at Glastonbury?

SS That was our first interactive experiment. It was in the Bassline Circus tent. Not a tent, it was a geodome – it was Europe's largest geodome at the time. Bassline Circus do night-time cabaret, sort of techno-drum and bass shows, but nothing in the daytime. It's a huge space.

We knew a couple of those guys, and we spoke with them and said, "can we come and do some family shows in that amazing space in the daytime?" We did a combination of the interactive installation, which is large consoles of hard drives, contact microphone stuff, and controllers. Things that people can just come up and play with no explanation – we might help them if they don't know what to do, but there's no signs. No 'do X', or written explanations. It's fun, it's creative, it's exploration. We also had a performance going on at the same time.

We also got given a load of lab coats. We were wearing the lab coats and because of the photos from that, some

science events booked us. I had no idea what science communication was. It all happened because we got given some lab coats! We've ditched them now (we're in boiler suits), but that just sort of happened.

HACKSPACE Casting your mind back, do you have a favourite event?

SS There's lots. I really like Maker Faire [Newcastle], we do it every year. I really like seeing what everyone else does – the makers exhibiting, and the hobbyists.

The Festival of Thrift in Middlesbrough – they're in a good place ethically. What puts me off about mainstream music festivals is that they're very much money-making events. Festival of Thrift is a small one, but it's one to look out for.

I like doing STEM events – I haven't done any music festivals for several years now. I would consider doing some smaller ones again now, but I really like doing these STEM events.

HACKSPACE Do you have any advice for people getting started in performance science?

SS It's hard to say because I didn't start out doing science. I have no science background – I'm a musician and a music teacher. I've learned the science before I started doing this because I was interested, but I'm self-taught. I only found out that science communication was a thing – and a job – about three or four years ago. I don't know. I feel like I'm someone who's come from a creative background in performance and music, with a lot of experience there, and I've come across into the science world, and that gives us a different angle.

I see a lot of really good content at science events and STEM events. Often, what people in that field could learn from the arts and the performance side is engagement. Often, people with an arts background, that's what they think – that's what it's all about. It's about how to engage people on an emotional level. If you combine those two, I think it works →

INTERVIEW

really well. I say the same thing to artists who complain about the fact that there's no funding any more. Well, you don't have to do something creative in the creative sector. You can do something creative in any sector. It works the other way around as well. If you're doing something science-based, you can do it in a creative way. I still see a lot of people doing STEM and science communication in a very formulaic way – disseminating information: 'I'm science, I'm showing you XXX' – try and cross over the different fields.

HACKSPACE Is there anything that you'd like to get into a show that you've not been able to yet?

SS We've been doing Bass In Your Face for a while, and I'm going to be creating a new show for next year. There's two shows that I'd like to do. One combines music and maths – I'm a musician (a cellist). I want to use an electric cello to look at the links between Bach and techno. There's a lot of cross-over there. There's so many cross-overs between music and maths,

and the cross-overs can get complicated quite quickly, and it's not my strong point. We'll be looking at some of the rhythmic devices that cross over completely different music genres, like baroque music to modern electronic music.

The other one will be something like DJ vs Sound Engineer. My background, before I did this, was hiring out PA systems, and I often worked as a sound engineer, and I used to be a DJ as well, so I've been on both sides of that timeless war between those two roles. So, it'll be looking at what happens to signals, why speakers blow, what happens when you overheat amplifiers – signals, square waves, that kind of thing. Basically, it's always the DJ's fault – that's what it comes down to! What kills speakers? DJs do.

HACKSPACE Is there any message you'd like to leave our readers with?

SS Take things apart! They probably already are, your readers. Some of the best inventors started by taking their radios apart. □



Above ■ Water and cornflour combine to make one of the most magical materials

Left ◆ Feeling the music has more than one meaning



£12.99
200 pages of
Raspberry Pi

THE *Official*

RASPBERRY PI PROJECTS BOOK

VOLUME 3

**Amazing hacking
& making projects**

from the creators of
The MagPi magazine

Inside:

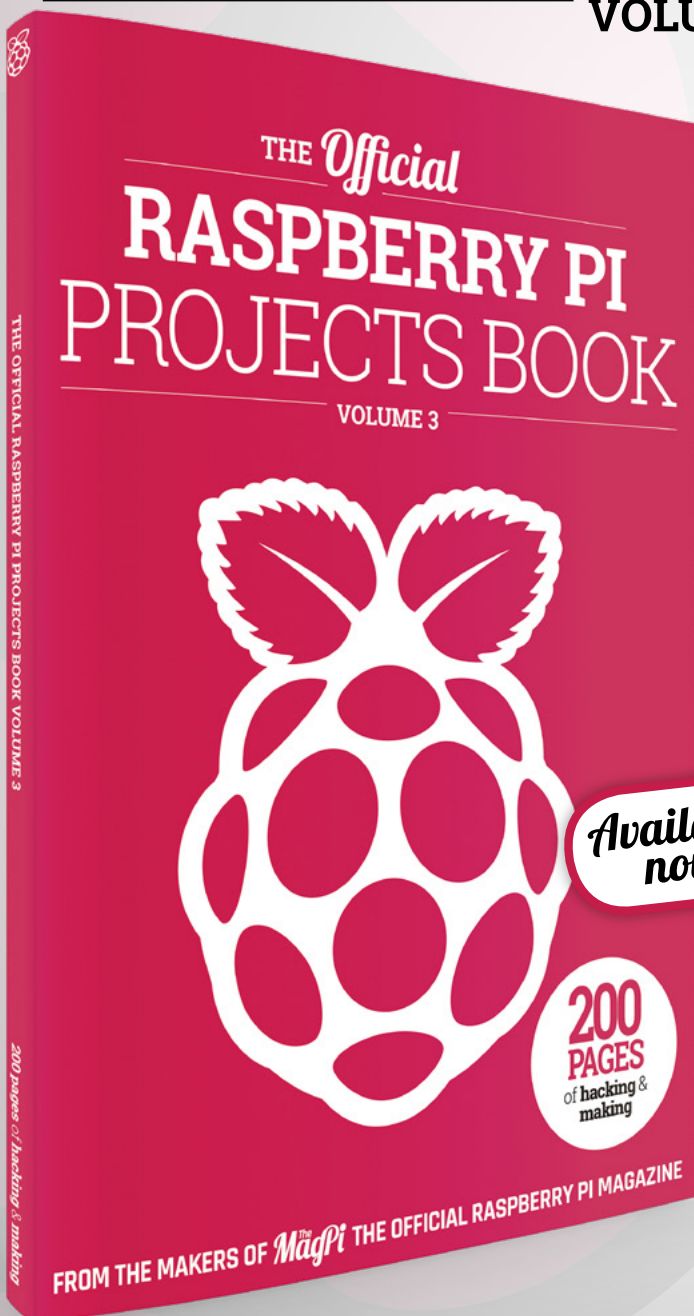
- How to get started coding on Raspberry Pi
- The most inspirational community projects
- Essential tutorials, guides, and ideas
- Expert reviews and buying advice

**Available
now**

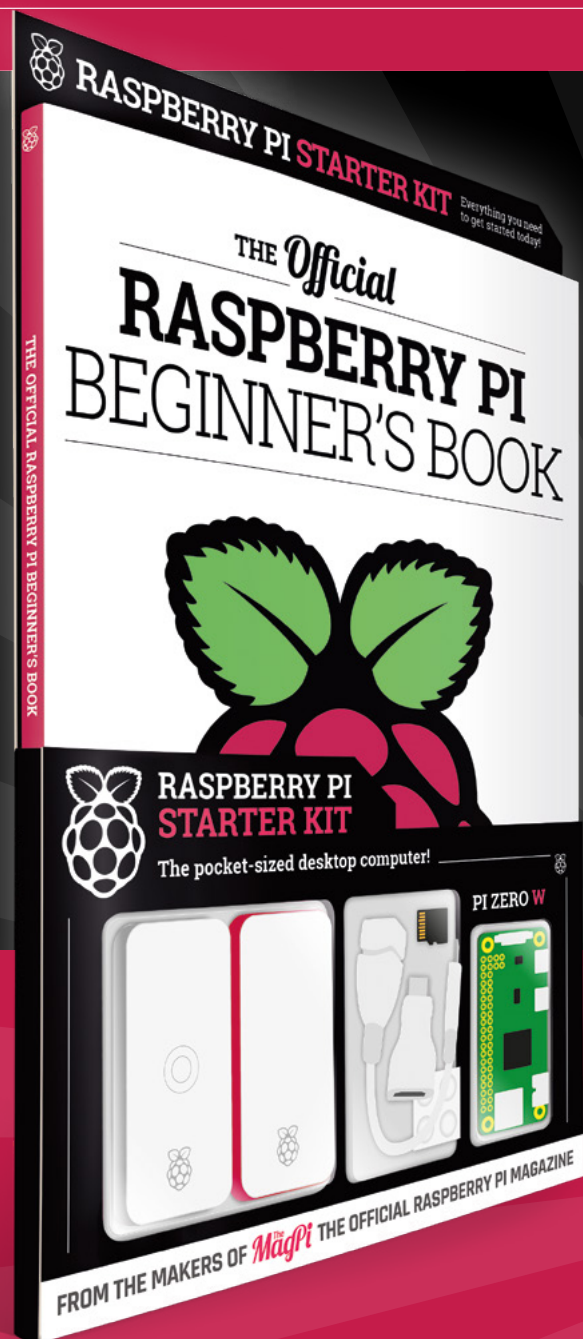
store.rpipress.cc

plus all good newsagents and:

WHSmith **BARNES & NOBLE**



THE *Official* RASPBERRY PI BEGINNER'S BOOK



LEARN COMPUTING THE EASY WAY!

Includes

- Pi Zero W computer
- Official case with three covers
- USB and HDMI adapters
- 8GB micro SD card
- 116-page beginner's book

*Available
now*



Buy online: store.rpiexpress.cc

MEASURING TAPE



Mayank Sharma

[@geekybodhi](#)

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

They are good for measurements, and innovative makers can use them for good measure

A

tape-measure has a couple of qualities that make it a good choice for an improviser's toolbox. First, it can measure a significant length, but can still fit inside your pocket, which makes it very portable. Secondly,

it's very dexterous as it helps you measure around curves, corners, and edges. Measuring tapes can be classified into two broad categories, dictated by how they're made. Those designed for sewing are made of either cloth, plastic, or fibre, while the ones intended for carpentry, or other types of craftsmanship, are usually made of a slightly curved metal strip that can coil into a small box.

The history of measurement runs concurrently with ours. We've been measuring stuff in one form or the other ever since we started possessing things. It's in fact very difficult to separate the exact origin of the tape-measure from the history of measuring distances, in general, by other means such as a ruler. All we can say for sure is that it was developed long before the first modern tape-measure device was patented in the US in 1864.

William H. Bangs Jr. received a patent for the first spring-return tape-measure. His tape could be stopped at any point and made to return to the case by sliding a button on the side of the case, which triggered the spring to pull the tape back into its case. Bangs's device was actually an improvement on an earlier design patented by James Chesterman of Sheffield, England, in 1829. Chesterman was making tapes he dubbed 'flat wire' for dressmakers, who used it to hold the shape of the crinoline hoop skirts. When the skirts went out of vogue, Chesterman repurposed the wires as long steel tape measures, with etched length markings, and marketed it to surveyors as a lightweight alternative to the bulky chains they were using back then.

Despite being mass-produced, the early tape-measures were still expensive and a novelty item. In fact, the tape's container has long been an avenue for beautiful craftsmanship and later as a popular medium for advertisement. There were containers in the shapes of animals and various objects like violins, each with its own unique winding device. Some were also made with precious metals and are a sought-after collectible. It wasn't until the early 20th century that the retractable tape-measure, now offered in celluloid containers, overtook the wooden folding carpenter's ruler.

TAPE-MEASURE ANTENNA

To help his eight-year-old nephew get a technical class amateur radio license, Chris went looking for a project that was “easy, fast, and required few tools or skills”.

He adapted the measuring tape antenna, designed by ham radio operator Joe Leggio (hsmag.cc/QcfEZL), to help his nephew understand the fundamentals of antennas and other associated topics.

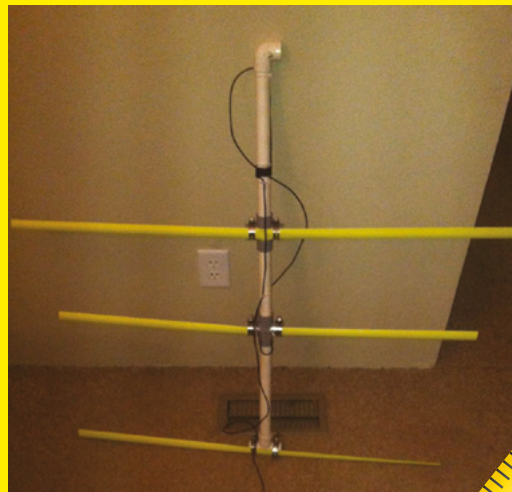
On his Instructables page, Chris details the process of building the different booms and the frame of the antenna from PVC pipes. He then cut various lengths of a 1" wide measuring tape, and used them as the director, driven, and reflector elements. The end of the cut tapes are very sharp, so make sure you carefully sand them. Also sand off the ends of the bottom side of the tape that'll act as the driven elements for soldering wires. Use the hose clamps to secure the elements to the frame. Follow Chris's nicely explained and illustrated details to strip an RG58 cable, and solder its wires to the driven element. He also shares technical notes on fine-tuning the antenna, and the page has useful discussions with other people who have replicated his project. □

Project Maker
CHRIS ORMSBY

Project Link
hsmag.cc/mVBhdJ

Below ♦

The antenna is built with \$20 worth of PVC pipes, hose clamps, measuring tape, and other materials



WINE TOTE

Project Maker
PEPPER
JORDAN

Project Link
hsmag.cc/PvcdFF

Right ♦
It took Pepper over two hours to tediously weave the tote. She says it makes for an unique gift, (but remember to throw in a bottle of wine too)

Pepper's husband, Ryan McFarland is an avid maker who, by his own admission likes to salvage at the town's dump "almost every weekend". On one such

outing, he scored 75 feet of measuring tape, but struggled to create something with it. Pepper took over the reins, and set about to weave the metal strips into a tote bag. She first created the base of the tote with 14 pieces of two and a half feet strips, with seven strips placed horizontally, and seven vertically. In hindsight she suggests others don't try to make a perfect square for the base. She wove them through each other alternatively, going under and over, and pushed them together to make a tight weave. Pepper has detailed the entire process of weaving the body of the tote, which is the most time-consuming and physically exhausting part of the process. Along the way, she also reflects on the different steps, and how you can learn from her mistakes. When she was done, Pepper trimmed the rest of the strip and used epoxy resin to seal the ends. To create the handle, she punched a hole in a measured length of binding strip, and then connected it to the bag using a rivet. □



MEASURING TAPE BRACELET

Ariana is a 27-year-old, multi-talented maker from The Bronx, New York, with backgrounds in graphic design, painting, digital photography, and performance art, as well as face painting, sewing, and crafting. She was reading Mark Montano's *The Big-Ass Book of Crafts 2* when the chapter title 'Wristful Thinking' gave her the idea of repurposing an old tape-measure as a piece of jewellery: "I have a sewing machine, so that actually gave me enough practice working on my stitches, as well as upcycling something old into something new." She first cut various pieces of the tape-measure after measuring them around her wrists, plus an extra 1". Ariana then sewed the strips together, folding and stitching the ends to seal. Finally, Velcro sticky dots attached to both ends of the strips act as the closure for the bracelet. □



Project Maker
ARIANA
S. LABUS

Project Link
hsmag.cc/yMvySx

Left ◆
Ariana says the bracelet is the perfect accessory for anyone into sewing, and will complement any type of outfit

SNAP BAG

Soon after a friend informed Julie that the closure of her favourite snap case was made from old tape-measures, she found a broken tape-measure while going through her father's tools. This was enough incentive to get her to use the discarded tape-measure to make her own snap bags: "My parents never threw out anything, and they would be thrilled to see that their frugality and belief that all things could find a use somewhere, ultimately proved to be right, at least for the broken tape-measure!" Her Instructable is the result of a fair amount of trial and error and help from several online sources. She cut two 4" pieces of tape, and covered their edges with duct tape. Follow her Instructable to

create and sew the body of the bag, and then insert the tape in the channels. Lock the mechanism by sewing the second side seam. □



Project Maker
JULIE JAI

Project Link
hsmag.cc/kixkYU


Left ◆
Julie used her 16" broken measuring tape to make different sized snap bags, with different fabrics, to store everything from safety pins to eyeglasses

HackSpace learns...

GLASS-BLOWING

HackSpace makes a Christmas bauble – we're going to need a bigger tree...

By Andrew Gregory

 @andrewgregory83

Humans have been creating with glass since ancient times. Not for as long as we've been using wood, stone, or metal, but they're naturally occurring, and our ancestors could get them from the world around them.

Unlike those materials, glass needs to be made before we can make anything out of glass.

It needs high temperatures, the right materials, a load of skill, and a combination of things to go right if you're not to end up with a load of, well, broken glass.

We were lucky enough to have a go at glass-blowing, thanks to Hen Ogledd Glass (henogleddglass.co.uk), a team based in Cumbria who are keeping this ancient craft alive, taking it to the people, and creating some unique handmade pieces along the way. Josh and Ann started their own studio only recently, after training at the University for the Creative Arts, and working with master craftspeople in Somerset and Devon.

WE ARE GLASS

We start with the raw materials. In this case, that's premade glass pellets, as Hen Ogledd's Ann explains: "We buy in our raw, plain glass in pelletised form. When the glass is made from its raw chemical constituents, it takes around three days of

heating at 1600°C. Then you've got to let that sit so all the bubbles rise out, to ensure it's in the same condition throughout. We have a mobile furnace, so if we did it that way we'd have to be here three days in advance, burning gas 24/7." This speeds up the work, but glass is such a good insulator of heat that, even at 1200°C, it takes three hours to get the glass up to temperature.

After the furnace, the tool you'd most obviously associate with glass-blowing is a metal tube, called the iron. The first step is to dip the iron into the molten glass that's sitting at the bottom of the furnace. This comes out glowing reddish-orange, and you have to keep twisting the iron in your hands to stop the molten glass dripping onto the floor, like honey on a spoon. We take a deep breath and try to squeeze some air down the tube and into the glass. Then we try again, but this time harder; the glass is molten, but it's still viscous and heavy, and the first breath in takes a lot of effort to break the surface tension and introduce a bubble into the glass.

Now it's time to add the colours. These come as granules of broken glass, into which we dip the molten glass in the same way you'd put breadcrumbs on a bit of meat to make schnitzel, first on one side to add the yellow granules, then on the other to add the green. This isn't as simple as it looks, however... >



Above ■ Glass has been found dating back from the Minoan Bronze Age, Ancient China, Rome, and Israel. This example was made in Newcastle in 2018, by me



Right ♦

The furnace reaches 1200°C, but it's cool enough to touch on the outside – perfect for warming up a pasty for breakfast

“ EACH DIFFERENT TYPE OF GLASS HAS A DIFFERENT CHEMICAL COMPOSITION, SO IT EXPANDS AND CONTRACTS AT A DIFFERENT RATE ”

Below ■

There are other ways of adding colour, other than granules. Solid rods of different colours are also used, but, as they're so dense before they've been worked, they all look the same – black

“To make colour, they add metal oxides to clear glass,” says Josh, the other 50% of Hen Ogledd. “The pink colour is made with gold oxide – it's really expensive. White used to be made with arsenic but it's now made with tin; blue comes from iron oxide. When you add the colours, it changes the quality of the glass, so the stuff we buy to add colours has to have guaranteed compatibility.

“Each different type of glass has a different chemical composition,” explains Josh, “so it expands and contracts at a different rate. We work with glass that's 96 coefficient of expansion (COE), which is 96×10^{-27} .

“If we were to mix this with bottle glass, that resultant mixture would break, no matter what the rate at which we cooled it down. Because the glass we use contracts as it cools at 96. This contracts as it cools at something like 30, so as the bottle glass molecules contract, they move; these stay in the same place, and that causes internal stress. The forces get so much that it will just explode. If you hold a piece of glass up to a polarising light, you can sometimes see the stresses inside, like petrol on the surface of a puddle of water.”

The granules stick to the molten glass, but they need to go back into the furnace so that they themselves can melt onto the surface of the glass blob. Still turning it round so that the glass doesn't drip, the iron goes back into the furnace, then comes out when the coloured granules have reached the same consistency as the rest of the glass.


GLASS OF MANY COLOURS

At this point, all the yellow is on one side, and all the green is on the other side; now it's time to swirl the colours around to get a pattern that will look good on your Christmas tree.

Hen Ogledd uses a chair with a flat arm, which can be used to roll the iron back and forth at a constant speed. Keeping the glass moving, we spin the iron with two hands and grab hold of the end of the glass with a pair of tongs (yes, this takes three hands – beginners always need help with everything). The twisting motion gives the glass →





Above  The first bit of air is the hardest to get in to the glass – once that's in there, you can put your thumb over the end of the iron and let the hot air expand



Right ♦ We used this tool, called a block, to shape the glass; earlier in the day Josh was using a wad of damp newspaper

the swirls and mixes up the colours. It's a hypnotic motion that looks like toffee or fudge being folded, and looks like we can lean forward and lick it. Obviously that would result in significant injuries, so we resist.

So, the glass literally looks good enough to eat, but it's still not right:

"With the colours being based on metal oxides, they change colour when they get hot, which is really annoying. So oranges, yellows, and reds all go to a red/black. Blues tend to go to pink and greens stay the same, funnily enough. Whites go clear, and blacks just get blacker and redder. So you can't actually tell the colour of the piece until it's cooled down."

KEEPING THE SKILLS ALIVE

Given Britain's role in the development of glass, it may surprise you to learn that the raw materials and the gear used by today's glass-blowers come from all over the world – anywhere but the UK, it seems. The plain pelletised glass Hen Ogledd uses comes from New Zealand, and the coloured granules are made in Germany. The specialised glass-blowing tools are from Sweden, and the hand tools are from the USA – all over the place, as Ann explains:

"We used to have one of the best toolmakers in the world in the UK, but unfortunately he never took on an apprentice, so when he died, his skills went with him.

"That's one of the reasons why we want to share what we're doing as much as possible. We've got this situation where we're losing techniques because people haven't shared them. We're of the view that we may as well share the knowledge and then it will live longer."

Right ♦ You can see the colours swirling through the glass as we hold on to it while twisting the iron

Another quick blast in the furnace and it's time to give the bauble its final shape.

GREAT BALLS OF FIRE

Now comes our favourite bit: smoothing the glass and shaping it to a perfect orb. This uses a specialised tool that's a block of wood with the shape of a round cavity carved out of it, with a handle, and it lives in a bucket of cold water, for reasons that will become obvious. The hot glass goes in the wooden cavity, the iron spins, and a big cloud of steam comes up from the damp wood:

"When the wood burns it creates a carbon layer, which doesn't stick to glass at all. And because it's wet it creates a kind of cushion around the glass. For more complex shapes, we'll use wet newspaper,





Left ♦
Ann demonstrates proper use of the jacks, a tool for cooling specific areas of the hot glass

which gives you more control, but as it's only a little touch here and there it doesn't burn – it just gets a bit warm."

HOT AIR

This next bit requires someone who knows what they're doing in charge, so Ann takes over spinning the iron. We just get to do the essential step of filling the bauble with air. One deep breath and the bauble grows to grapefruit size as Ann spins it to make sure it stays round. Now the thing that we're making looks like other things that people have made, which is a good sign. It's time to get the bauble off the iron, but working with glass requires its own logic. Rather than cut the bauble off, we use a tool called a pair of jacks – they look a lot like a pair of shears, with a handle in the middle and two blades that cut into the glass. This looks like it's cutting, but in fact it's cooling the glass at a place we can control. This makes it brittle at that point, which means we can break it off.

"To take the piece off the iron we have to make what's called a necking line, which is where the glass will break away. If we don't put that line in, it either stays attached to the iron, and then it ends really badly, or it will break off in completely the wrong place and the piece that you've been just been working on for an hour and a half will break right down the middle."

Ann adds a blob of molten glass to the top of the bauble to make a loop and that's the thing complete. Sadly, I can't put it in my pocket and take it home, because it's still soft, and it's still hot enough to melt my skin.

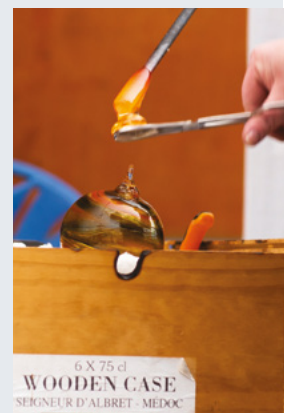
“ WHEN THE WOOD BURNS IT CREATES A CARBON LAYER, WHICH DOESN'T STICK TO GLASS AT ALL, AND BECAUSE IT'S WET IT CREATES A KIND OF CUSHION AROUND THE GLASS ”

"Once the glass is made, we have to put it away in the kiln, overnight. Even though it looks hard, it's still around 500 degrees. Between 500 and 380 is where glass actually forms its structure; it's called a neolith. If you wanted to blow some glass and then just leave it out in the air, it would be very susceptible to temperature changes, because the structure of the glass would have stresses in it.

"If you imagine a bauble, inside it's going to stay hotter than the outside. The outside is going to cool faster than the inside, it's going to contract while the inside is going to stay the same, which will introduce stresses into the glass. So we have to bring it down nice and gently so it forms properly. We bring it down by around 30C an hour, which means it cools nicely, the glass forms with little to no stress in it, it's stronger and it can take temperature changes."

Our final piece is surprisingly heavy, and seems to look different every time we pick it up. It's not perfect but it's ours, and it's a living link back to methods that have been used for thousands of years. □

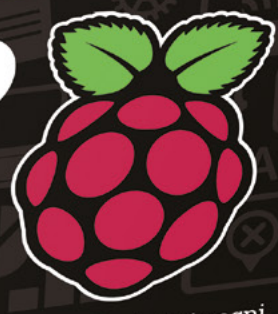
Below ♦
An extra blob of plain molten glass to form a loop, and our bauble is complete. Not bad for a first attempt



DON'T MISS THE **BRAND NEW** ISSUE!

YOUR OFFICIAL **RASPBERRY PI** MAGAZINE

The *MagPi*



The official Raspberry Pi magazine

Issue 71

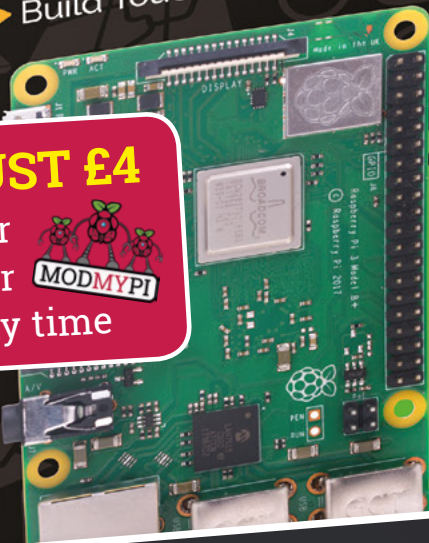
July 2018

rasberrypi.org/magpi

RUN ANDROID ON RASPBERRY PI

▶ Use Android Apps ▶ Build Touchscreen Devices ▶ Control Electronics

MAKE YOUR OWN



SUBSCRIBE FROM JUST £4

- ▶ FREE! £5 ModMyPi Voucher
- ▶ FREE! Delivery to your door
- ▶ NO OBLIGATION! Leave any time



CARD COOK...
Using AI to evaluate cards



PLUS! FREE
CASE, THREE
COVERS &
CABLES

HERE COMES

FREE PI ZERO W

With your 12-month subscription to the print magazine

magpi.cc/Subs1

Buy online: store.rpipress.cc

FORGE

HACK | MAKE | BUILD | CREATE

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

PG
88

WEARABLES

Make colour-changing accessories so your outfit never clashes

PG
94

Z80

Put some life into retro hardware with Z80 assembly language

PG
100

AI

Don't program everything: get your hardware to adapt to your needs

PG
104

ICE CREAM

Use frozen carbon dioxide to create delicious frozen treats

PG
108

ARDUINO

Ditch the prototype and program chips directly

PG
110

WATCH REPAIR

Restore your timepiece and let your wrist sparkle

PG
76

SCHOOL OF MAKING

Start your journey to craftsmanship with these essential skills

76 Tapping screws

78 Arduino programming

84 Reciprocating saws



WORKSHOP BASICS

Creating screw threads

Bolt straight into metal using taps



TOOLING UP TIPS

- Taps are spiralled cutting tools that are fluted to allow cut material to pass.
- The body of the tap has a tapered lead with cutting teeth, followed by teeth that keep the tool positioned properly as the lead is screwed down into the pilot hole.
- There are flutes between the toothed 'land' sections to allow for chip removal.
- Tap shank has a squared end for securing into the tap handle (i.e. tap wrench).
- Major diameter measures the widest depth of the threads being cut.
- Minor diameter is the narrowest section of the cut, slightly wider than the pilot hole.
- Thread pitch is used to measure the distance between thread crests, and corresponds to the coarseness or fineness of the screw threads you'll use.
- Tap handles hold the tap securely and give you leverage to turn the tap in the hole while you're cutting threads.
- A note about dies: you'll often hear the words 'tap and die' used together. A die is in inverse of a tap – instead of cutting the threading into a hole as with a tap, the die is used to cut threads onto the outside of a rod, thus turning it into a machine screw.

Above ♦
A tap and hole ready to add a screw thread



John Park

johnedgarpark

John Park is a maker who builds creative technology project videos and tutorials for Adafruit Industries.

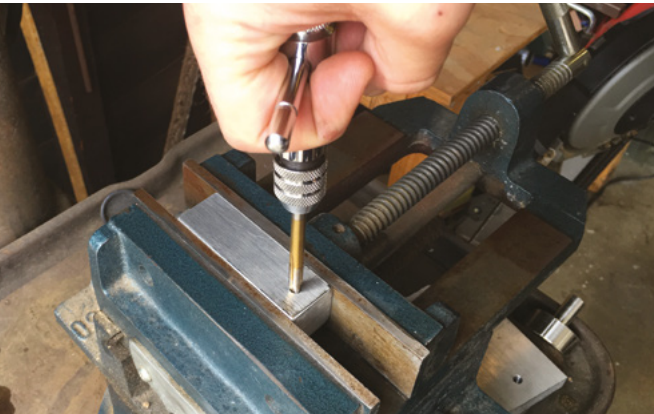
You can use a hand tap to easily cut machine screw threads into metals and plastics, providing yourself with a great way to fasten parts using standard machine screws. While

there are lots of ways to join materials such as rivets, screws, and nuts, these can be difficult to use with thick materials or enclosed parts where drilling a hole all the way through the work is impractical or impossible. Tapping a threaded hole internally into a pre-drilled pilot hole in the work is a terrific alternative that enables easy assembly and disassembly. Thanks to the high tolerances of taps and screws, you can then match this hole with the same size machine screw and you'll have a perfect fit, making it easy to securely fasten your parts and materials. Plus, it's very satisfying to make and use professional-looking precise threaded holes you've cut for yourself!

A GENTLE TAP

It's good to get a small set of taps in the sizes you'll most commonly use, such as M3 × 0.5 (this second number specifies thread pitch), M4 × 0.7, M5 × 0.8, and M6 × 1. You'll also need tap drill bits that correspond to each tap. These are used to cut a pilot hole slightly smaller than the minor diameter of the tap's teeth. Store your taps and tap drills together so they're clearly labelled and easy to find. Extra points for making your own index with cardboard and gaffer tape.

You can use a hand tap on different grades of metals and plastics. Aluminium, brass, and softer grades of



Above ♦ Go slowly and carefully when tapping a hole, to avoid jamming

steel are all good material choices for hand-tapping metals. Plastics that tap well include ABS, HDPE (cutting boards are made of this), Acetal/Delrin, and even Perspex/acrylic if your tools are sharp and you're careful to go slowly to avoid cracks.

TURN ON THE TAP

Remember to go slowly! Tapping isn't difficult, but it's easy to break off taps if you go too fast.

The basic steps for tapping are:

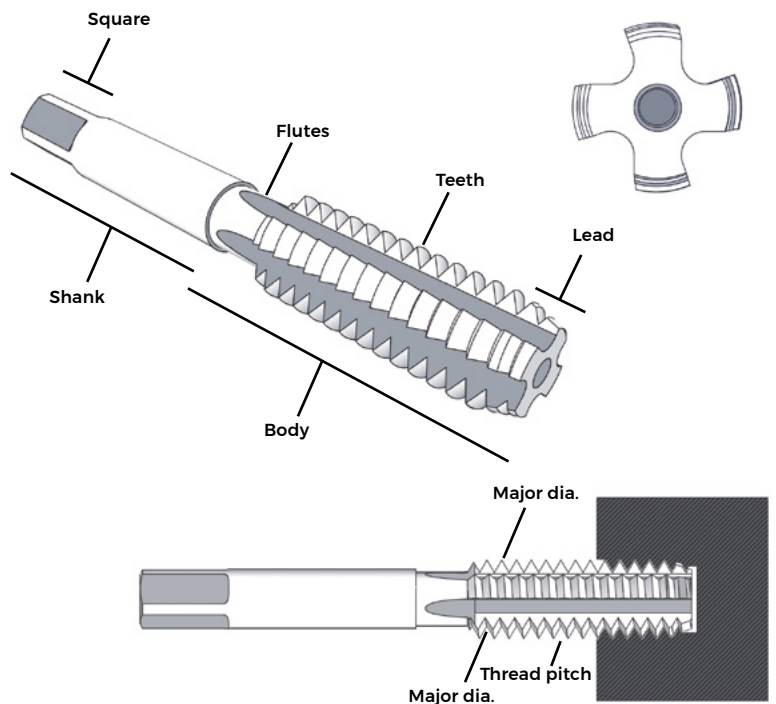
1. Measure, mark, and drill a pilot hole with the proper bit for your desired tap. The angle of this hole will determine the angle of the tapped hole, so make sure to drill true and square. A drill press works best, but you can certainly use a hand drill if you keep the drill square to the surface. Drill the hole a little bit deeper than your desired final depth – it's good to reference the machine screw you'll be using as a guide. You can mark the bit with a piece of tape to make sure you drill to the correct depth.
2. Clean out the hole with some compressed air to get rid of any chips left behind by drilling.
3. Add a bit of cutting fluid to the hole if you're cutting metal – plastic doesn't require any as most machinable plastics are self-lubricating. Tap Magic is a good general cutting fluid, but

you can get away with using WD-40 if you don't have a more specific cutting fluid available.

4. Put the tap into your tap handle, align it square and straight to the hole, and begin screwing it in a half turn. This first cut is critical to the straightness of your final tapped hole, so take your time and make sure you haven't introduced an off angle. A bit of pressure is helpful, just don't use too much! Back out the tap a quarter turn to release any chips from the tap's teeth. You're now on your way, cutting the threads.
5. Proceed with another half turn, followed by a quarter turn back. Continue repeating the half turn forward, quarter turn backward process. You need to go steadily, and slowly, and never force the tap if it becomes stuck – this is a sure way to break off a tap inside the hole!
6. Repeat until you've reached the bottom of the pilot hole, then unscrew the tap from the hole.
7. Use compressed air to flush the hole of chips. You've tapped your first hole! You can now thread your machine screw into place to test it.

Now that you can tap holes, you can fasten a wide range of materials together. Get out there and get making! □

Below □ The structure of the tap allows plenty of space for waste material to come out



PICKING A TAP

The following common taps and drill sizes work together:

METRIC TAP DRILL SIZES	
Tap	Tap Drill
• M3 × 0.5	2.50 mm
• M4 × 0.7	3.30 mm
• M5 × 0.8	4.20 mm
• M6 × 1	5.00 mm

Arduino programming: Stacks, classes, and scrolling displays

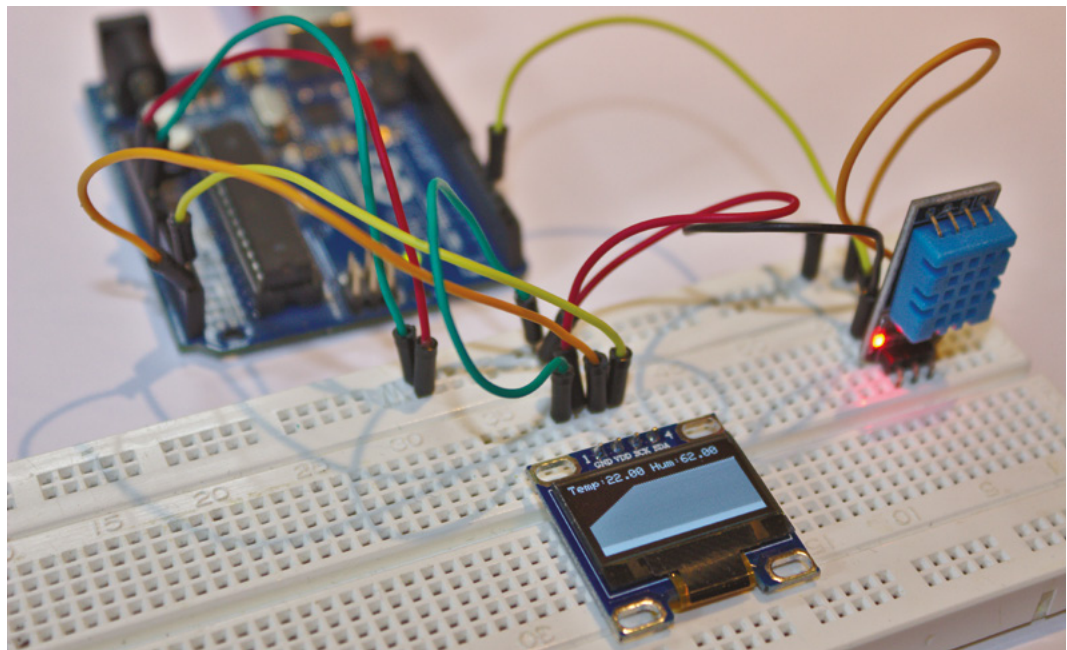
Learn new code skills and impress your friends with the coolest looking thermometer in the land



Graham Morrison

@degville

Graham is a veteran Linux journalist who is on a life-long quest to find music in the perfect arrangement of silicon



Right

The completed project shows both temperature and humidity, alongside a chart for recent temperature changes

YOU'LL NEED

- SSD1306 monochrome 0.96" 128 x 64 OLED graphic display
- DHT11 digital temperature and humidity sensor

In our previous Arduino tutorial, we expanded both our C programming knowledge and our data visualisation potential by using a library – we stood on the metaphorical shoulders of giants and imported code written by other developers. Rather than being a cheat or a lazy option, this is how nearly every project is developed. Libraries, and their close relation, the API, allow programmers to utilise all kinds of advanced functionality without having to constantly reinvent the wheel. Not only that, but you also benefit from the programming wisdom that goes into the development of a library, wisdom that can sometimes stretch generations when dealing with

old system libraries, more true when programming with C than many modern languages.

We're going to use a couple of new libraries in this tutorial to do some magical stuff that would otherwise take a year's worth of tutorials. We'll use the same temperature and humidity sensor from before, but we're moving on from the hipster austerity of seven-segment displays to a whole new world of usefulness – a real bona fide screen. The screen we're using is known as an SSD1306. It's commonly available and costs very little, and yet has a bright OLED display with a resolution of 128 x 64. It's also tiny, making it perfect for embedded projects where you need to output a few more details than a couple of numbers. In fact,

we're going to use this display to create a real time side-scrolling histogram, so you can see changes in temperature over time with a simple glance.

CODE

We're going to use two new libraries. The first is the equivalent of the DHT library, only for the screen. This allows us to easily access the hardware without needing to understand or reverse-engineer the protocol it uses to speak to the Arduino. The wonderful Adafruit provides this library, and it's called `Adafruit_SSD1306`. The second library is also from Adafruit, `Adafruit_GFX`, and provides a collection of graphics 'primitives' for drawing things like lines, rectangles, and text without needing to write the algorithms ourselves. Both libraries can be installed by opening the library dialogue from the Arduino IDE (Sketch > Include Library > Manage Libraries...), searching for the library names, and clicking 'Install' from the correct result.

// We're going to use a couple
— of new libraries in this tutorial
to do some magical stuff that
would otherwise take a year's
worth of tutorials

Before we dive into writing our own code, we need to edit the header files of the `Adafruit_SSD1306` library. Without this edit, our screen would only display every other line, and this is because the header is hard-coded to use a display resolution of 128×32 rather than 128×64 . To change this, open `Adafruit_SSD1306.h` (usually found in `Arduino/libraries/Adafruit_SSD1306`) and uncomment `#define SSD1306_128_64` by removing the first two forward slashes (on line 73 in our version). Add two slashes to the beginning of the `#define SSD1306_128_32` line to comment out the old resolution and save the file. Your code should look like the following:

```
#define SSD1306_128_64
// #define SSD1306_128_32
// #define SSD1306_96_16
```

With that out of the way, let's start our own new project. Although the skeleton of the code is similar to the previous tutorial, we're going to be changing most of the implementation. At the top of the file, →

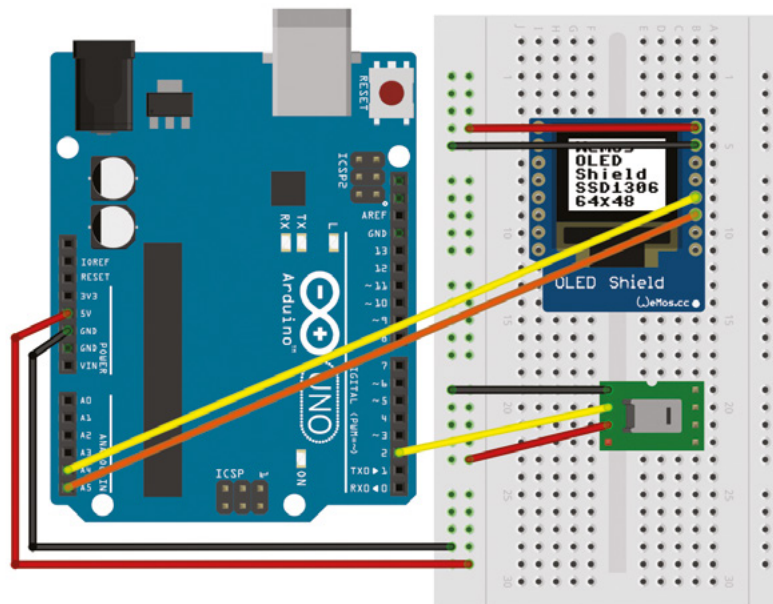
WIRING

One of the nice things about the `SSD1306` display we're using, and many of its derivatives, is that it pushes straight into your breadboard without requiring any additional jumpers. The signal carried by each of its four pins is annotated across the top of the screen, and this means you can still see which pin does what, even with the board plugged in. This is particularly important because you need to pay attention to which pin carries the power (usually labelled `VCC`) and which is for ground (`GND`). Get these the wrong way round and you may break the screen, your Arduino, or both. You also need to check that power requirements for your board match the Arduino – ours is 3V ~ 5V DC. Power needs to be connected directly to 5V on the Arduino and ground to the `GND` adjacent to this on the Arduino.

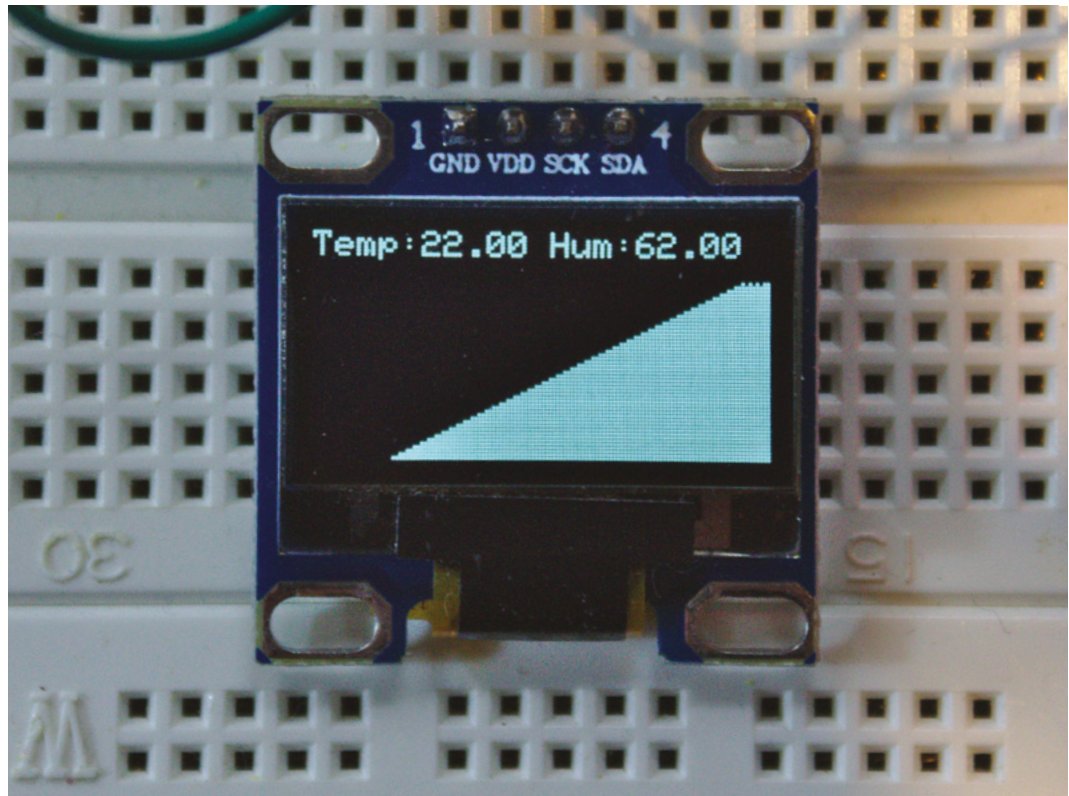
However, we also need to connect the temperature and humidity sensor to the same power pins. The best way of doing this is to use the power and ground 'rails' on a typical breadboard.


Two are usually found on the outer edge of each long side of the breadboard, and connecting 5V from the Arduino to one of these and `GND` to the other will deliver the power and ground to any pin connected across the length of the rail. With those connections in place, it's then as simple as making one connection from the 5V rail to `VDD` on the screen and another from the 5V rail to `VCC` on the sensor, and the same must be done for both `GND` pins.

The screen and the Arduino talk to each other using the I²C protocol, and this requires the use of specific pins on the Arduino. These two pins, normally labelled `SCL` and `SCA` on the screen, need to be connected to the corresponding pins on your Arduino, and these can be different depending on which Arduino you're using. As we're using an Uno R3, `SCL` is analogue pin 5 and `SCA` is analogue pin 4. Finally, the data pin on the temperature and humidity sensor is connected to digital pin 2 on the Arduino, as it was in the previous tutorial.



Above ♦ The screen and sensor share the same 5V and `GND` rails on the breadboard



Right  The display we're using is less than an inch across, which is ideal for tiny IoT installations and self-contained devices

we want to include the two new library header files alongside **dht.h** for the sensor:

```
#include <dht.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h>
```

Beneath these lines, we're going to use three **#define** statements to bake-in system-wide values that save us from changing the actual code to accommodate hardware differences:

```
#define DHT11_PIN 2
#define SCREENADR 0x3C
#define MAXSTACK 128
```

The first line sets the pin connected to the temperature and humidity sensor, the same as in the previous tutorial. The second line is the I²C address of the screen. The screen and the Arduino talk to each other using the I²C protocol, and because you can connect multiple devices over I²C, each is differentiated with an address. Ours is 0x3C. This should be included in your screen documentation, or even burnt into the PCB, but you can also run a script to probe any connected I²C devices and return the address of each device if you need to (hsmag.cc/kigPet).

The third statement in the above code is a precursor to a new and important concept we're going to introduce in this tutorial, and that's

something called a 'stack'. We're going to use a simplified stack to hold 128 separate temperature measurements, so that we can draw a histogram of changes in temperature over time. You might wonder why we don't use a simple array to hold these values, but this is because we want the histogram to scroll in real time as temperatures are added. If we were to simply update the values in an array sequentially, the histogram would draw itself across the screen, left to right, and then simply reset to the left border of the screen again, as you see in many such implementations. But a stack allows us to have a sliding window of values that follow a leading edge, effectively creating a scrolling histogram of temperature data. This all sounds more complicated than the actual code, so let's take a look:

```
class Stack
{
private:
    int ourList[MAXSTACK];
    int top;
public:
    Stack() {
        top = 0;
        for (int i = 0; i <= MAXSTACK; i++)
            ourList[i] = 0;
    }
}
```

QUICK TIP

Using the text function requires a foreground and background colour. Without a background colour, when the text updates it will look corrupted, but it's because old text pixels are still there in the background.


```

void push(int item) {
  if (top == MAXSTACK)
    top = 0;
  ourList[top++] = item;
}
int peek(int x) {
  return ourList[(top + x) % MAXSTACK];
}
};

```

This stack is a list of data that we can keep pushing data to, and peeking at data in. It will always hold the most recent 128 datapoints pushed into the stack.

Our stack is implemented within a class. We discussed classes in the previous tutorial when we used one to access the DHT11, but in the above code we're creating our own. Classes, a little like stacks, are a huge subject that can even dictate the design of an entire programming language, but they're basically just a way of co-locating data with the functions that use the data. In our case, that means the data is the value for each temperature reading, and the functions add and read values from the stack. If the data and functions are solely for the use of the class, they're defined beneath a 'private' specifier, and won't be accessible outside the class – this helps hide the complexity and avoids erroneous access from outside the class. Conversely, for data and functions intended to be accessed by you, the programmer, we use the 'public' specifier. In our above class, the **push**

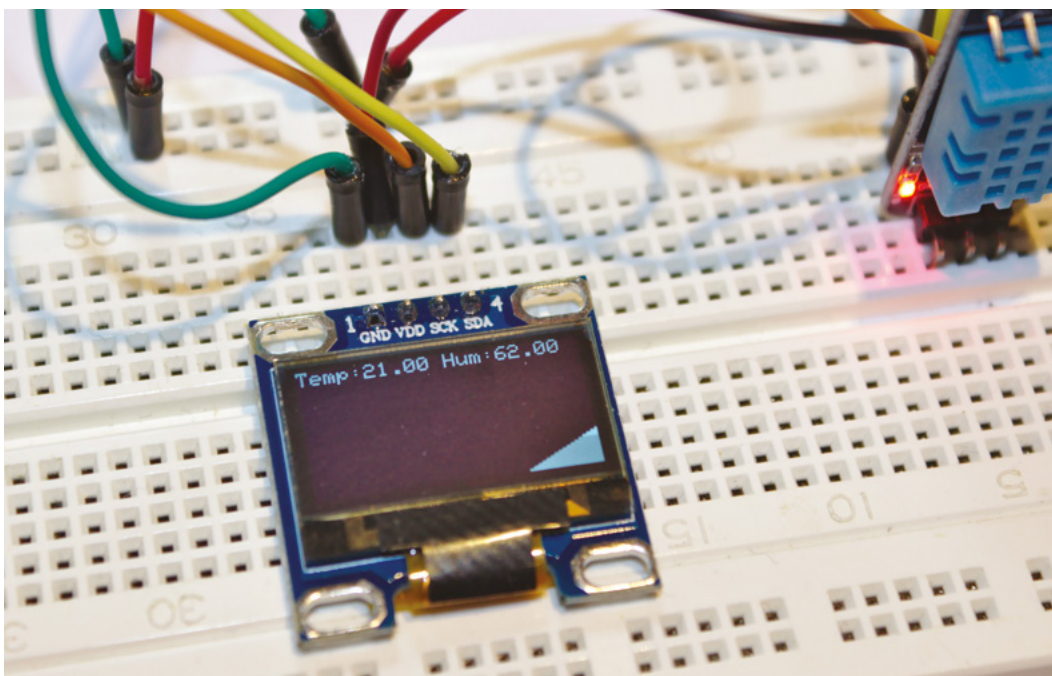
and **peek** functions are all public, as we'll be using these to create and view our stack. The array that holds the temperature readings, **ourList**, is private, as too is an integer that holds the current top array position of the stack.

There are three functions that are members of this class. The first is special because it takes the name of the class itself – **Stack()**. This is the constructor, and like **setup()** in an Arduino project, it runs automatically when a class is instantiated. We use this instantiation to set the internal values to zero, including every element of the array. This safeguards against wayward values being left over in memory or a previous execution. Although we've not used it here, the opposite function to the constructor is the destructor, written as **~Stack()** in a class definition, and this function is run when a class is deleted. As our code only quits when the Arduino is reset or powered off, we're saving space and not →



Above ♦
SSD1306-compatible screens are cheap and readily available, and can even be found in different colours and in multiple colour configurations

Left ☑
The OLED display and temperature sensor in situ on the breadboard



QUICK TIP

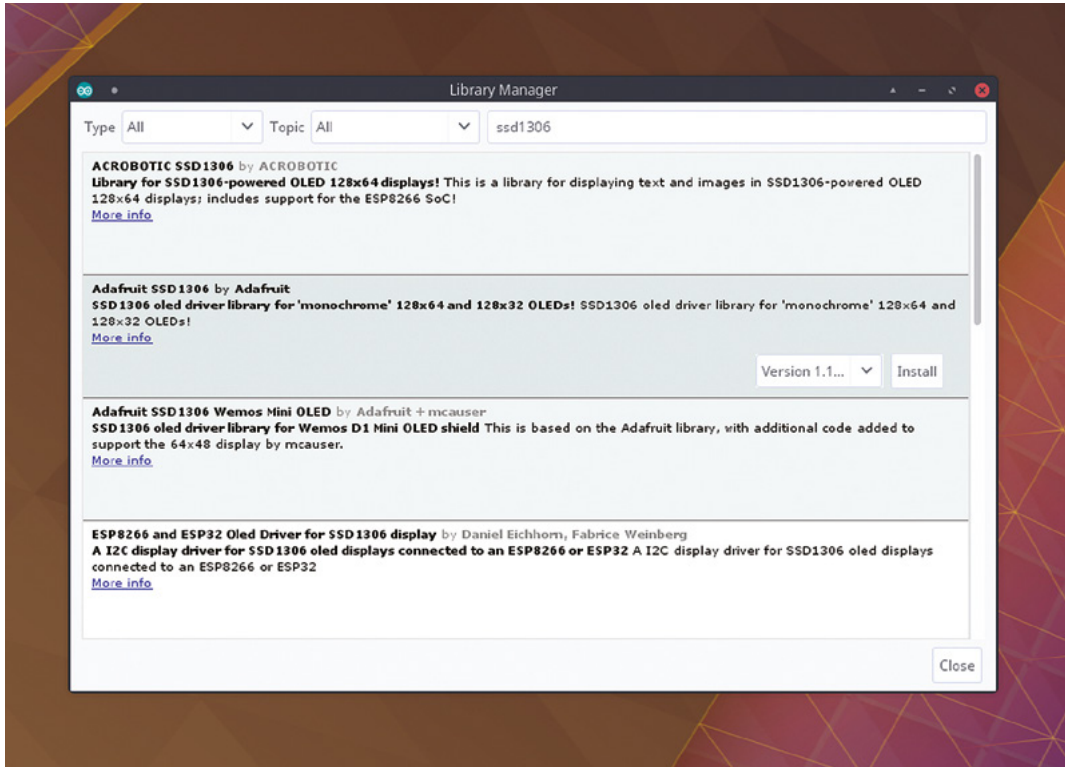
If you experience display problems, you may need to use an external 5V power source for the screen, connecting the common ground to the Arduino.

QUICK TIP

The term 'stack overflow' actually refers to when you try to write to the stack and the stack is full. Fortunately, as ours is fixed in size, this won't happen.

Right

Libraries can be downloaded and installed manually, as discussed last month, but it's much easier to just use the Arduino IDE



adding a destructor, but a good programmer uses the destructor to free up any allocated memory and generally clean up after themselves.

The `push` function simply checks to make sure the top isn't yet at the maximum stack size value, and enters the `item` value at the current top position before incrementing `top` to the next array location. We haven't implemented `pop` because it's not needed – we're simply overwriting previous values in the array. Instead, we have `peek` to return the

STACK OVERFLOW

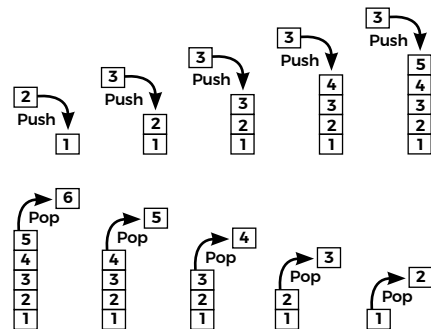
A stack is a data structure, and that just means it holds data in a specific way. The most common stack holds data in the same way you create a deck of cards, putting one card on top of the next and removing cards from the top of the pile. In stack terminology, this is a LIFO stack – the card that was last in is first out. FIFO (first in, first out) is another common variant, and this operates as a basic queue. Alan Turing even coined the terms 'bury' and 'unbury' in 1946 to describe the process of adding and removing data from a stack, but we now use the terms 'push' and 'pop' for the same thing. Additionally, 'peek' is often used when you want to take a look at the top card, rather than remove it, or examine another card in the pack. Just like in 1946, however, stacks are ideal when you only have a limited amount of memory.

// The module itself combines both temperature and humidity sensors, and the great thing about this module is that it's incredibly easy to use

`item` value at `x`. The tricky part is that because `top` is always changing, `x` is an offset from the value of `top`, which we modulo against the maximum stack size, to make sure it's both within range and loops over when higher. Modulo is very useful for such a simple operator!

DRAWING LINES

The next chunk of code instantiates three types for the sensor, our new `Stack` class and for the screen, before filling out the Arduino's `setup` function. This



initiates the display and runs a function to clear the display of the noise that typically accompanies the display turning on:

```
dht ourDHT;
Stack temperature_stack;
Adafruit_SSD1306 display(4);

void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, SCREENADR);
  display.clearDisplay();
}
```

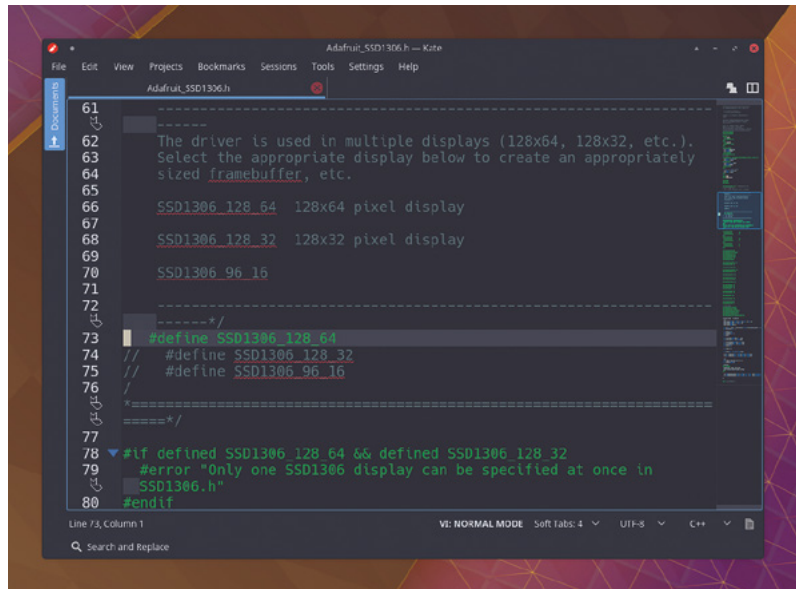
The next piece of code is all that's needed to draw the histogram. Thanks to Adafruit's graphics library, we call its `display.drawLine` function to draw a line from one set of coordinates to another, and we do this to first black out a column (the same x value) and then to draw a white line up to the temperature value in that column. We get the value from our stack using our `peek` function.

```
void displayChart() {
  for (int x = 0; x < MAXSTACK; x++) {
    display.drawLine(x, display.height(), x,
display.height(), BLACK);
    display.drawLine(x, display.height(), x,
display.height() - temperature_stack.peek(x),
WHITE);
  }
}
```

For good measure, we're also going to add text to show the current temperature and humidity readings. This is just as easy as drawing a line, although we do pull the readings directly from the sensor rather than our stack:

```
// Function to display a character
void displayNum() {
  display.setTextSize(1);
  display.setTextColor(WHITE, BLACK);
  display.setCursor(0, 0);
  display.println("Temp:" + String(ourDHT.
temperature) + " Hum:" + String(ourDHT.
humidity));
}
```

All that's now left to do is write the main `loop` function. This simply pushes a new temperature value onto the stack, runs both the text and histogram generation functions, and finishes up with the `display.display()` function to update the display. We then add a delay in milliseconds to wait until we repeat the sequence. Changing this will affect the duration between each reading, altering the scroll speed from seconds to hours if you so wish,

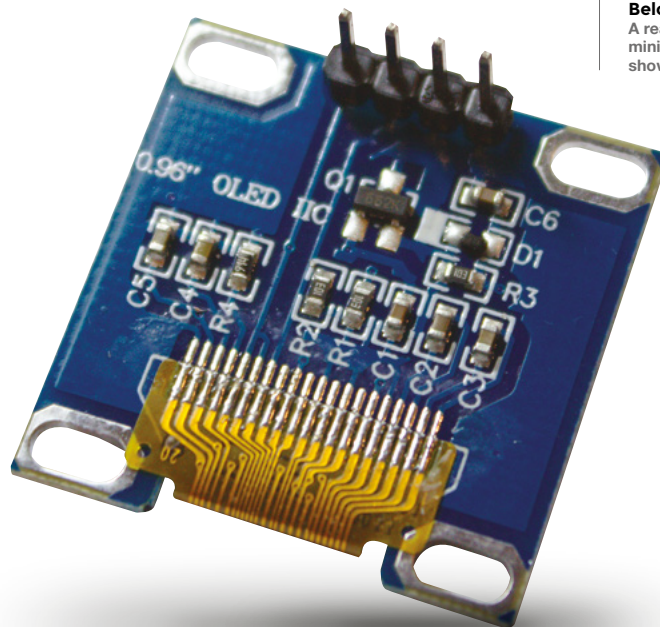


which is great if you want to monitor the change in temperature over an entire day – try `delay(86400000)`.

```
void loop() {
  int chk = ourDHT.read11(DHT11_PIN);
  temperature_stack.push(ourDHT.temperature);
  displayChart();
  displayNum();
  display.display();
  delay(100);
}
```

The code for this tutorial can be found at git.io/vh4x9.

Above ⚠️
You need to edit the screen driver header to make sure it uses the correct resolution for your display




Below ⚠️
A rear view of the mini OLED display, showing its four pins

Maker's Toolbox: Reciprocating saws



Whatever needs cutting, this'll (probably) cut it



Above 
The Bosch GSA1100-E is a tough saw, suited to heavy use



Ben Everard

 @ben_everard

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

One power tool epitomises the word 'hack' in the physical sense, then it's the reciprocating saw. It'll tear through most materials pretty quickly, with enthusiasm rather than accuracy. Want to hack through metal pipes? Go for it. Does a tree need hacking down? This'll do it. Wood with nails in it? Hack away.

When thinking of power saws, you might automatically think of circular blades of whirling doom, or the endless spin of a band-saw, but this one is far simpler. Reciprocating saws (sometimes know as a sabre saws) are a type of electric saw mimics the action of a manual saw by repeatedly pushing, then pulling, a blade through your material. Depending on the type of blade, they can cut through a wide variety of materials, including wood and metal.

Because it's so similar to manual sawing, it's one of the easier power saws to use – you line it up as you would a manual saw, ensure that the foot is on the material, and start cutting. Almost all reciprocating saws have a two-handed setup, where one is on a vertical handle, while the other grips the body of the

saw. Typically, these saws have a variable speed trigger on the vertical handle, so you can start slowly at first, then build up. You can even start your cut in the middle of a sheet of material with a plunge cut (start with the saw roughly parallel to the material, and slowly lever the blade downwards).

Reciprocating saws are powerful and can often cut wood up to 30cm thick. Many reciprocating saws are able to cut through wood with nails in it (though check that your blade can handle it), which adds to the speed of taking things apart. However, they are a very rough-and-ready tool, and they're not designed for millimetre-perfect accuracy. That said, with a little care, they're not far off.

These saws are probably most famous as tools of destruction – their portability and ability to cut through a wide range of materials make them perfect for taking things apart. However, these same traits make them a versatile tool for building as well.

If you're after a powerful, quick saw and prepared for the fact that accuracy is a flexible concept, then a reciprocating saw could be what your toolbox needs.

SAFETY

Like all saws, reciprocating saws can't tell the difference between what they're meant to cut and what they're not (such as your fingers) – if it gets in the way of their blade, they'll do their best to get through it, and they may well succeed. Keep fingers and limbs well out of the way when cutting!

Wear eye and (if necessary) ear protection when cutting, be aware that the saw can 'kick back' if it encounters something solid, don't meddle with the blade while the power source is connected, and, as with all power tools, follow the manufacturer's instructions.

PICKING YOUR SAW

There's a wide range of reciprocating saws available at a wide range of prices, from around £50 to several hundreds of pounds. In general, you get what you pay for, but it's important to match your needs with the tool you want to buy.

Weight Reciprocating saws can get up to about 5 kg – which is a hefty tool to work with – down to around 1.5kg. A heavy tool might be fine for occasional use, but if you're going to use it regularly, then a lighter tool might be worth the investment.

Cutting depth This varies between 15 cm and 30 cm for wood, and 1 cm upwards for steel.

Orbital action Some saws move the blades in a slightly elliptical motion, rather than linearly. These cut faster, but messier.

Blades You'll need different blades for different materials. Check which blades are included and available for your saw. Third party blades can be cheaper than the manufacturer's, but are often of a lower quality and can dull quickly.

Stroke length This is the distance the blade travels on each stroke. The longer the stroke length, the faster the saw will cut at a given speed. This varies from around 15 mm to 30+ mm.

Speed Typically measured in strokes per minute, this is the speed the tool runs at when unloaded. The faster the saw blade moves, the faster it will cut.

Power source Batteries are more portable than mains, but also more expensive, and will drain in time.

You might also want to consider the other tools you have (and if they can share batteries with a saw), and whether you want a brushless motor (which are more power-efficient, but more expensive).

We looked at two saws, to see what you can get at the entry and mid-range levels of reciprocating saw: the Erbauer ERB373RSP (£89.99), and the Bosch GSA1100-E (£119.99). In many ways, these are similar tools – they both run at 2700 strokes per minute, with a stroke length of 28 mm, use 1100 watts of power, and weigh 3.5kg. The Erbauer tool is rated to cut through thicker wood (300 mm as opposed to 230 mm).



We looked at two saws to see what you can get at the entry and mid-range levels of reciprocating saw



Primarily, these are two tools that can do similar jobs. The difference isn't really in their capabilities, but in their use. According to their manufacturers, they both have 'soft grip' handles. However, this is a flexible term, and the Bosch is easier on the arms. It's generally tougher as well – the plastic casing of the tool feels like it will live up to more abuse, and comes with a longer warranty, and a solid case as well.

Overall, if you need something to cut through material now, then either tool will do the job. If you're after something that will put up with years of use, either by yourself or in your space, then the extra money on a tougher tool, such as the Bosch, will pay off. □

Below ♦
The Erbauer ERB373RSP is a bit more budget-friendly, and sturdy enough for hobbyists



WIN!

1 of 10

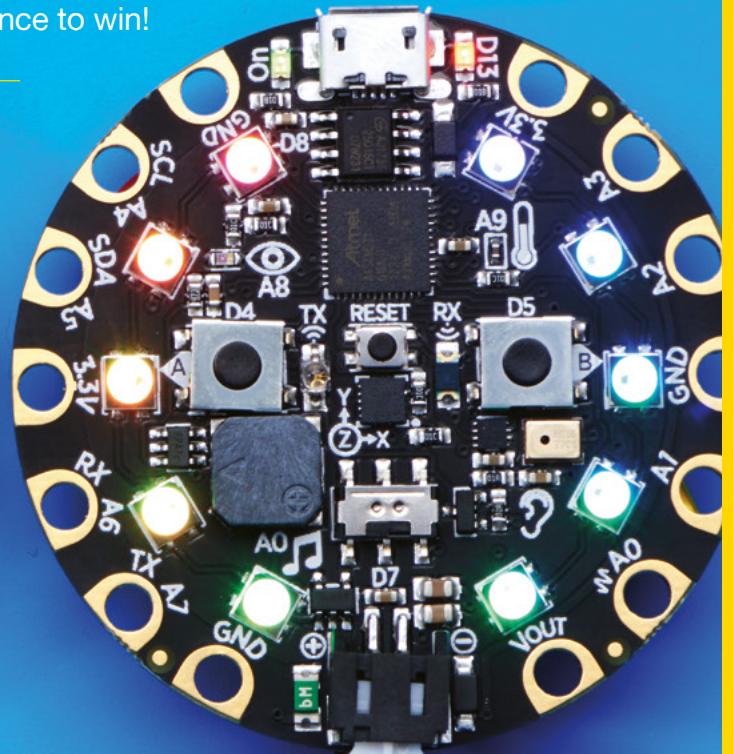
Adafruit Circuit Playground Expresses

signed by [Limor 'LadyAda' Fried](#)

WORTH
£25

Head to hsmag.cc/win

Enter by 30 June
for your chance to win!



Terms & Conditions

Competition opens on 21 May 2018 and closes on 30 June 2018. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Twitter or Facebook.



HackSpace

TECHNOLOGY IN YOUR HANDS

Download the app

Out now for smartphones & tablets



**SAVE
25%**
with an annual
subscription

£2.29

rolling subscription

or

£26.99

subscribe for a year



Colour-sensing clutch with Circuit Playground Express

SCHOOL OF MAKING

Sample and process colours in real time

Use a Circuit Playground Express to create the ultimate fashion accessory



Sophy Wong

@sophywong

Sophy Wong is a designer, maker, and avid creator. Her projects range from period costumes to Arduino-driven wearable tech. She can be found on her YouTube channel and at sophywong.com, chronicling her adventures in making.



It's no secret that at HackSpace, we are hooked on the Circuit Playground Express. With so many on-board sensors and lights, we keep thinking of new ways to use it!

— This project is inspired by Angela Sheehan's fairy wings project – she uses a colour sensor in a magic wand to 'collect' colour from an object, while the lights in her wings change to match that colour – brilliant! In our build, we'll mimic this by using the built-in light sensor on the Circuit Playground Express, and a strip of NeoPixels inside a clear handbag. Change the colour of the lights in your bag to match your outfit, or a found object while you're out and about!

The circuit for this project is simple, and can be used in any project where you want to connect an external strip of NeoPixels to your Circuit Playground Express. Instead of sewing the circuit, we'll do some light soldering to connect everything together. Coding is easy with MakeCode, and once your project is complete, you can quickly modify and develop your code further, as the CPX remains easy to access on the front of your bag. The NeoPixel strip will be on the inside of your bag, so we'll need to pass wires through holes punched into the front. Choose a bag that you are willing to dedicate to this project for good. (Your author purchased the bag shown for about £25.)

The transparent 'holographic' coating on this bag makes a great surface for reflecting the NeoPixel

LEDs, and it really shines in low-light conditions. Adafruit's NeoPixel strips are available in several different options, with varying densities of LEDs along the strip. The 60 LED per metre option (shown) is easy to work with, with bigger pads and easy-to-read markings. Advanced makers can intensify this project by sizing up to 144 LEDs per metre! NeoPixel strips are sold by the metre, but for most handbags you'll only use about 25cm. Let's get started!

LET'S MAKE SOME HOLES

First, we'll mark the placement of the holes we need to make to pass wires through the front of the bag. Place the Circuit Playground Express on the front of your bag, with the USB port at the top and the JST port at the bottom. Use a Sharpie to mark holes that the wires will go through: GND and VOUT on either side of the JST port, and A1. We'll also need some holes for mounting the CPX, so mark these four holes we won't be using in our circuit: GND and 3.3V at the top, and A0 and TX/A7 at the bottom.

Set the Circuit Playground Express aside, and use an awl to make the holes you've marked. Align the awl precisely: there's no going back once you've made a hole in plastic. Twist the awl and slowly press to make holes about 1 mm in diameter. Take your time so as not to crack the plastic. →

YOU'LL NEED

Materials

- ◆ Clear handbag with a flat bottom
- ◆ Circuit Playground Express
- ◆ NeoPixel strip
- ◆ Silicone-coated stranded wire 30 AWG
- ◆ 3 x AAA battery pack with on/off switch and JST connector
- ◆ Clear packing tape or Gorilla tape
- ◆ Small cable ties, 101.6 mm length

Tools

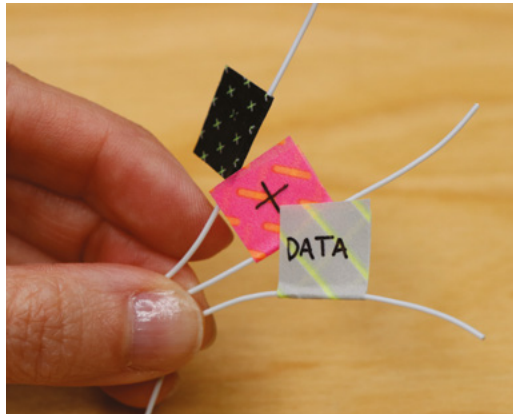
- ◆ Awl
- ◆ Craft knife
- ◆ Basic soldering supplies
- ◆ Masking tape or colourful washi tape
- ◆ Scissors
- ◆ Third helping hand

Left ◆ Gradual, even pressure is the key to making clean holes in plastic



Colour-sensing clutch with Circuit Playground Express

SCHOOL OF MAKING



Right ♦ Labels don't have to be colourful, but why not make life a bit brighter?

Made a hole in the wrong place? No problem! Make a few more holes and create a pattern – turn the mistake into a decorative feature!

WIRING UP

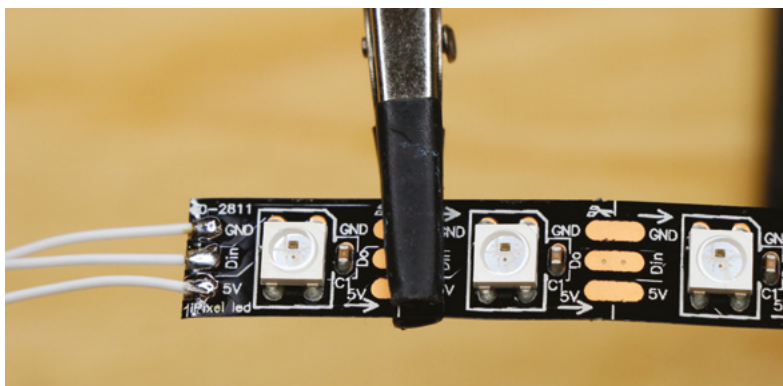
Cut three lengths of wire – long enough to reach from the holes you just made, straight down to the bottom edge and over to the bottom corner of your bag. Cut the wires a little longer than you think you'll need; we'll cut them down later.

Using one colour of wire gives a clean look to this build, and is a nice option if the wires in your project will be visible, as they are on this bag. To avoid confusion while you're working, mark the wires temporarily with tape. Colourful tape is useful here, but a Sharpie and regular masking tape work just fine too. Set aside your prepared wires, and let's work on that NeoPixel strip.

NeoPixel strips come in one-metre lengths (or more), so you'll need to cut it down with a pair of scissors. Hold the NeoPixel strip against the flat

Below ♦ A third-hand-style holder makes it easier to solder the wires in place

“ **The circuit for this project is simple, and can be used in any project where you want to connect an external strip of NeoPixels to your Circuit Playground Express** ”



bottom of your bag, and determine how many NeoPixel LEDs will fit on your bag. Allow at least 1 cm of space on each end so that the strip can be completely covered with tape later. Cut the strip through the midpoint of the conductive pads, after the last NeoPixel you'll be using. Remove the protective silicone sleeve from your cut piece.

Note the arrows printed on the NeoPixel strip: this is the direction that data flows through the strip. You'll need to solder your wires to the beginning of the strip, the end the arrows point away from. Strip and tin the ends of your wires, and tin the pads of the NeoPixel strip. Solder the wires to the strip: power to 5V, ground to GND, and data to Din.

MAKING CONNECTIONS

Insert the NeoPixel strip into your bag, and feed each wire through its hole to the front of the bag. Refer to the Circuit Playground Express to make sure the wires are going through the correct holes. Once your wires are fed through their holes, mark them again with masking tape on the outside of your bag to make doubly sure they'll be connected properly to the CPX.

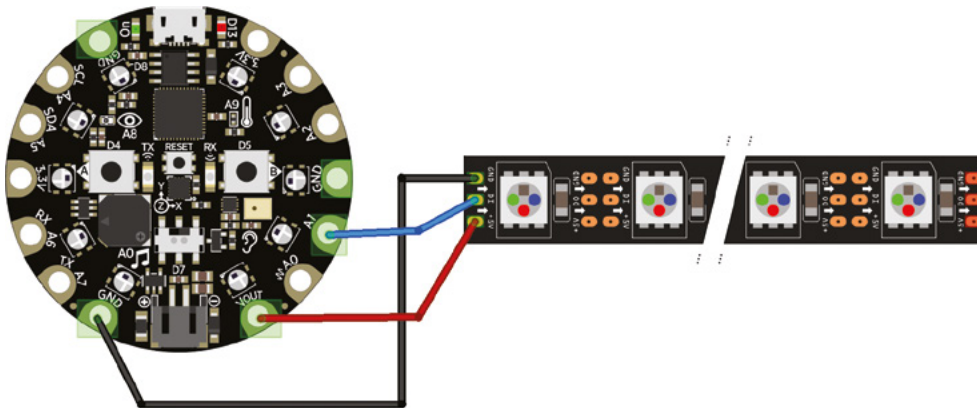
Secure the Circuit Playground Express in a set of third helping hands, to keep it from moving around while you solder. Tin the three pads you'll be using: GND and VOUT on either side of the JST connector, and A1. Strip about 5mm of insulation off the end of the wires, and insert each wire through its hole on the CPX from back to front. Refer to the circuit diagram to make each connection correctly. For a secure connection, wrap the bare wire around the edge of the hole and solder in place. Trim off any protruding solder or wire. You don't want any pokey bits on your wearable tech!

ENGAGE CPX

We'll use cable ties to make fasteners for attaching the board to the bag. Small black cable ties work well for this and won't be very visible when the lights are on. They're also easy to remove if you need to remove the CPX from this project later. You'll need eight cable ties in total.

Snip the eyes off four of the cable ties, and save the long tails for another project (they're quite useful for applying tiny dabs of glue on things). These eyes will lock the other four cable ties in place.

Pull the slack in the wires to the inside of the bag, and lay the Circuit Playground Express in place against the front of the bag. Align the CPX with the holes you punched for GND and 3.3V at the top, and A0 and TX/A7 at the bottom. From the inside of the bag, feed a cable tie through each mounting hole. Slide one eye



Left ♦ The NeoPixels are really simple to wire. There's just power and one data connection

Below ♦ The battery should push into the JST socket

onto each cable tie, and secure tightly. Make sure you are putting the eye onto the cable tie in the correct orientation, or the eye will slide right off. You can look at a fresh cable tie for reference. When your Circuit Playground Express is secured in place, use angle cutters to snip the end off each tie.

ATTACH THE NEOPIXEL STRIP

Now, let's secure the NeoPixel strip to the bottom of the bag, and give it some protection at the same time. Place the strip on the bottom of your bag, close to the side that the Circuit Playground Express is mounted to. Ensure that the wires flow neatly next to the strip and up the middle of the front of the bag to the Circuit Playground Express. These will be visible, so make them as tidy as possible.

Cover the NeoPixel strip with a strip of clear, heavy-duty packing tape or clear Gorilla tape. Press the tape down well on all sides of the strip and be sure to press out any air bubbles. The clear tape disappears almost completely when the lights are on. Use another piece of the clear tape to cover the wires running up the front of the bag, and keep them from getting jumbled up and snagged on things inside your bag.

INSTALL THE BATTERY PACK

It's time to add the battery holder! The battery holder will live inside the bag, and the Circuit Playground Express is mounted to the outside, so we'll need to cut one more hole in the front of the bag for the battery holder's wires to pass through. Using a craft knife, cut a slit about 1 cm long below the CPX's JST port. With the battery pack on the inside of the bag, push its JST plug through the slit, to the front of the bag. Depending on the thickness of your plastic, this could be tricky, and you may need to cut a fully fledged hole to get it through.

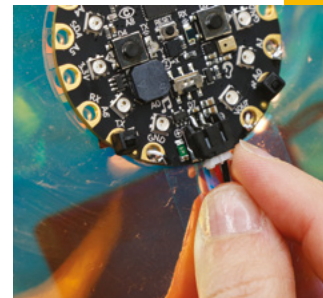
Insert batteries into the holder, and plug the JST connector into the port on the Circuit Playground Express. Use the on/off switch on the battery holder to turn your circuit back on and off.

Now let's make everything light up with code! This code is a little more complex than our previous projects, but by the end you'll know how to extend the abilities of your Circuit Playground Express by adding external outputs, like NeoPixel strips, to pins. To get started, connect the CPX to your computer with a micro USB cable. Then open your browser and navigate to makecode.adafruit.com, and start a new project.

SET UP

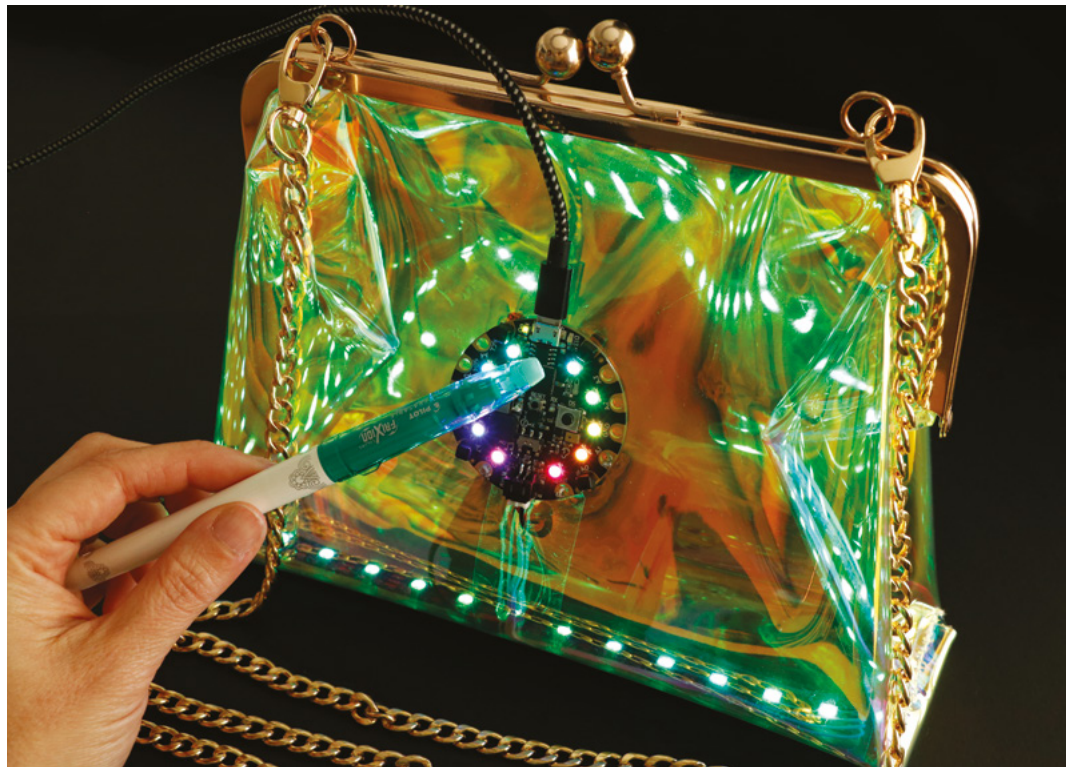
We won't be using the Forever loop in this program, so delete it from the workspace. However, we will need to set some actions to happen at the beginning of the program, so open the Loops menu and drag an On Start loop onto the workspace.


The light sensor and built-in LEDs are set up by default in MakeCode, but we need to create a new variable for our NeoPixel strip so we can work with it in our program. Go into the Variables menu, click 'Make a Variable' and name it 'strip'. You'll now see that a few new items have appeared in the →



Below ♦ The LED strip fits unobtrusively in the bottom of the clutch





Right 
Match your clutch bag with clothes or accessories

Variables menu. Drag the 'Set strip to 0' block into the On Start loop.

Click on the Light menu to reveal the nested NeoPixel menu. Click on NeoPixel to view more options for strips. Find the round module labelled 'Create strip on A1 with 24 pixels' and drag it onto the workspace. Drop it onto the 'Set strip to 0' block, directly on top of the '0'. From now on in our code, the variable 'strip' is the placeholder for the NeoPixel strip we have connected to pin A1. Change the number 24 to the actual number of LEDs on your strip (ours was 12).

// Depending on the amount of light in your environment, you may need to experiment with these settings to get a good reading from the light sensor

One other setup item we need to address is setting the thresholds for the light sensor. Depending on the amount of light in your environment, you may need to experiment with these settings to get a good reading from the light sensor. Even if yours works great right from the start, it's good to know where these settings are, because chances are you'll need to tweak them eventually.

The light sensor is an input, so head into the Input menu and scroll down to find the 'Set dark threshold value to 0' block. Drag this block into the On Start

loop, and just leave the value at 0 for now. Duplicate the block by right-clicking and selecting 'duplicate'. Drag the new block into place right underneath the first one, and click on the drop-down menu to change 'dark' to 'bright'. Leave this one at 0 for now also. Later on, if you've downloaded the code to your Circuit Playground Express and the sensor doesn't seem to be responding correctly, you can adjust these threshold values until the sensor works well in your particular lighting conditions.

MAKE A STARTUP SEQUENCE

Now let's create a startup sequence so that we know the lights are working when we turn on the Circuit Playground Express. Head into the Light menu and grab the 'Show rainbow animation for 500 ms' block. Add it to the On Start block, and click on the drop-down menu to change the duration from 500 ms to 1 second. In the Circuit Playground Express simulator, you'll see that we just programmed the on-board NeoPixels to show a rainbow animation. Let's also make the NeoPixel strip on pin A1 show the same animation.

Click on the Light menu again to get to the NeoPixel menu, and find the 'strip show rainbow animation for 500 ms' block. Drag it into the On Start loop under the previous animation, and change the duration to 1 second. Although you won't see it in the simulator, our NeoPixel strip will now show the same animation as the on-board NeoPixels.

Next, let's set all the pixels to black (off) and get ready to sample a colour. Go back into the Light menu, click on the NeoPixel menu, and grab the 'strip set all pixels to (red)' block. Drag it onto the workspace and drop it in the On Start loop. To change the colour setting to black, you'll need to head back into the Light menu, scroll down to the Colour section, and grab the round module labelled 'red'. Drag this module onto the workspace and drop it directly on top of the red colour block to replace it. You can then click on the word 'red' and select 'black'.

To set the on-board NeoPixel ring to black, simply duplicate this block, and drag the copy into the On Start loop under the original block. Head back into the Light menu, then the NeoPixel menu and find the round module for 'onboard strip'. Drag this module onto the workspace, and drop it on top of the 'strip' variable to replace it with 'onboard strip'. That's it for setup!

MAKE THE MAGIC HAPPEN

We're getting there! All that's left is to tell the Circuit Playground Express to read the light sensor and set the on-board strip to the same colour as it detects. Start by going into the Input menu and finding the On Light Dark loop. Drag it onto the workspace. Now, anything we put inside this loop will happen when the light sensor is blocked by an object and it detects 'dark'.

Let's have it take a colour reading and display that colour on our NeoPixel strip. Head back into the NeoPixel menu (remember, it is nested inside the Light menu) and grab the 'strip set all pixels to (colour)' block, and drop it into the On Light Dark loop. Replace the colour in the block with the 'ambient colour' module, which you'll find in the Input menu. Now, if you wave an object over the light sensor, the NeoPixel strip should change to match the colour of the object!

It's a magical effect, and we can embellish the magic by adding a few more actions. Let's make the on-board NeoPixel ring display the rainbow animation once the colour sensor takes its reading. From the Light menu, drag the 'show animation rainbow for 500ms' onto the workspace and add it to the On Light Dark loop. The Circuit Playground Express can also play sounds, so let's add a magic wand sound as well. In the Music menu, grab the 'Play sound Power Up until done' block and add it to the On Light Dark loop. Click on the drop-down menu to change 'Power Up' to 'Magic Wand'. To return the CPX to its ready state, let's set the on-board pixels

to black again. Simply duplicate the final block of the On Start loop, 'onboard strip set all pixels to black', and drop it into the bottom of your On Light Dark loop. Your program is complete!

You can test this out on the Circuit Playground Express simulator by clicking and dragging on the yellow light sensor input icon to the left of the CPX image. You should see the on-board rainbow animation and hear the magic wand sound. If everything works as expected on the simulator, click Download, and follow the instructions in MakeCode to install the code on your Circuit Playground Express.


HAVE FUN COLLECTING COLOURS!

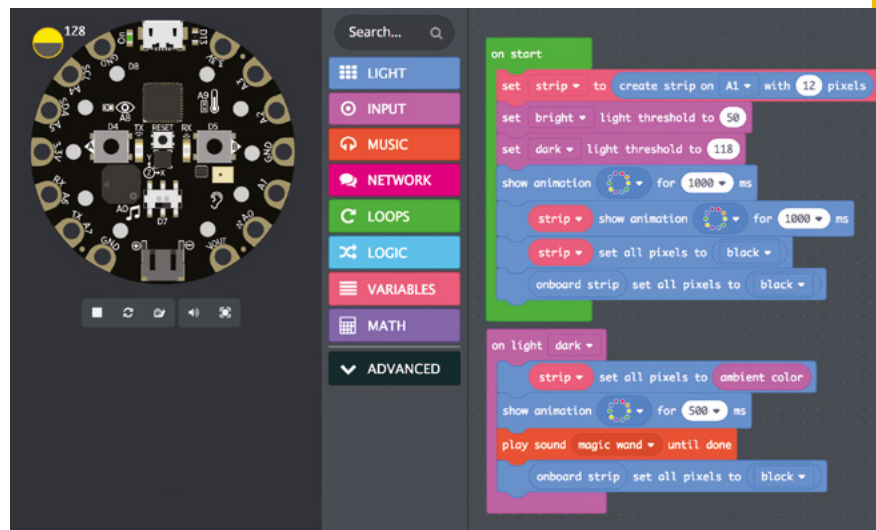
Now you're ready to sample colours from the world around you! Remember that it may take some experimenting with the light sensor threshold values to get a solid reading at first. Look for the eye icon on the Circuit Playground Express – hold the object in that area a few centimetres away from the board. When the sensor takes a reading, you'll hear the satisfying magic wand sound, and see your bag change colour. It's like magic, but better, because you made it!

We've just scratched the surface with sensors in this project, and because the Circuit Playground Express is easy to program on the front of your bag, you can modify this project in lots of ways. In addition to the light and colour sensor, the CPX also has an accelerometer, a temperature sensor, a sound sensor, and an infrared receiver and transmitter on board. How are you using these sensors in your projects? Show us at [@HackSpaceMag!](https://hackerspace.org/mag)

QUICK TIP

Don't carry a clutch? You can apply the same circuit and code to almost any clothing or fashion accessory.

Below  The MakeCode editor is easy to use and powerful enough for our needs



Getting retro with the Z80: software

Dive into Z80 assembly language



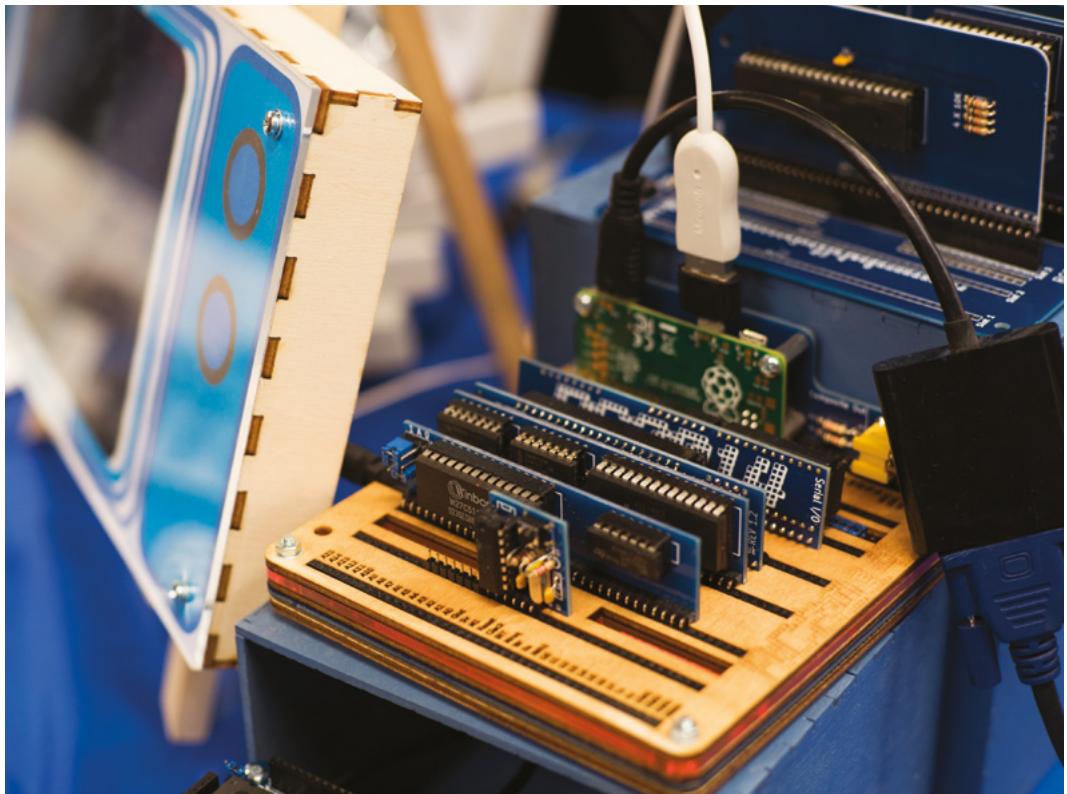
Dave Astels

@dastels

Dave's career started in the 8-bit days, with the Z80 and 6502, and he's been working with computers ever since. He does some writing at daveastels.com and learn.adafruit.com.

YOU'LL NEED

z80pack
hsmag.cc/SWCRQK



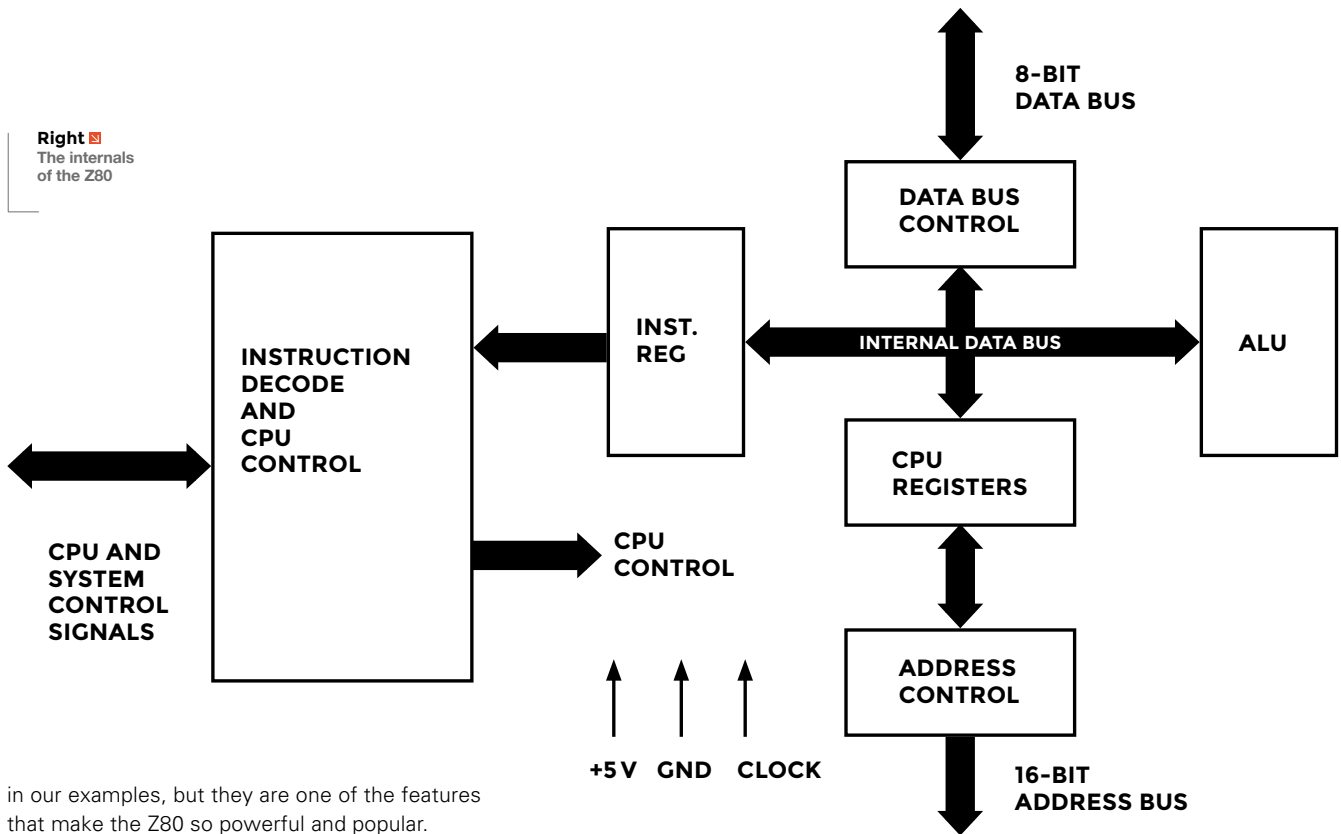
Last month, we built a Z80 computer on a breadboard. This month, we're going to take a look at how to program it using assembly language. Assembly language is a symbolic version of machine code. It uses names for the instructions, registers, condition flags, etc. It also allows you to label addresses in the code or data and refer to the labels in the code rather than the numeric addresses. The idea behind assembly language is to provide a one-to-one mapping to machine code, while being far more readable and writable than a series of bytes.

TEMPORARY STORAGE

Every CPU has registers to store values that can then be operated upon. The Z80 has a fairly rich set of registers for an 8-bit CPU. It has an 8-bit accumulator (called A), as well as six other general purpose 8-bit registers: B, C, D, E, H, and L. These six have the added utility that they can be used in pairs (BC, DE, and HL) as 16-bit registers. That allows working directly with addresses inside the CPU in a very flexible and powerful way. Two other important registers are the 16-bit index registers, IX and IY. These let us do powerful array and structure type operations very easily. We won't be using them

Above You can build your own system or get a kit such as this RC2014

Right 
The internals
of the Z80



in our examples, but they are one of the features that make the Z80 so powerful and popular.

In the hardware article, we saw the address and data buses. Recall that they are the way the CPU interacts with the rest of the system: the address bus identifies where in memory (or which I/O port) the CPU wants to read/write data from/to, and the data bus moves the data between the system and the CPU.

The buses show up inside the CPU as well. Various registers have their contents routed to/from one or the other buses in order to move data around, as defined by our code.

The ALU (arithmetic logic unit) is where maths and logic operations are performed.

The instruction register, decode, and control blocks are where the instruction being executed is stored, and processed. Each instruction causes a specific sequence of events to occur in the CPU. The exact sequence will depend on the instruction and will move data around the CPU and between CPU and memory (or I/O), as required to implement the instruction.

So, registers hold data being worked on, the ALU does the work, the buses communicate with the rest of the system, and the instruction decode/control module orchestrates everything.

HELLO WORLD

'Hello World' is the canonical first program, so why don't we start our exploration of Z80 assembly

// The idea behind assembly language is to provide a one-to-one mapping to machine code while being far more readable and writable than a series of bytes //

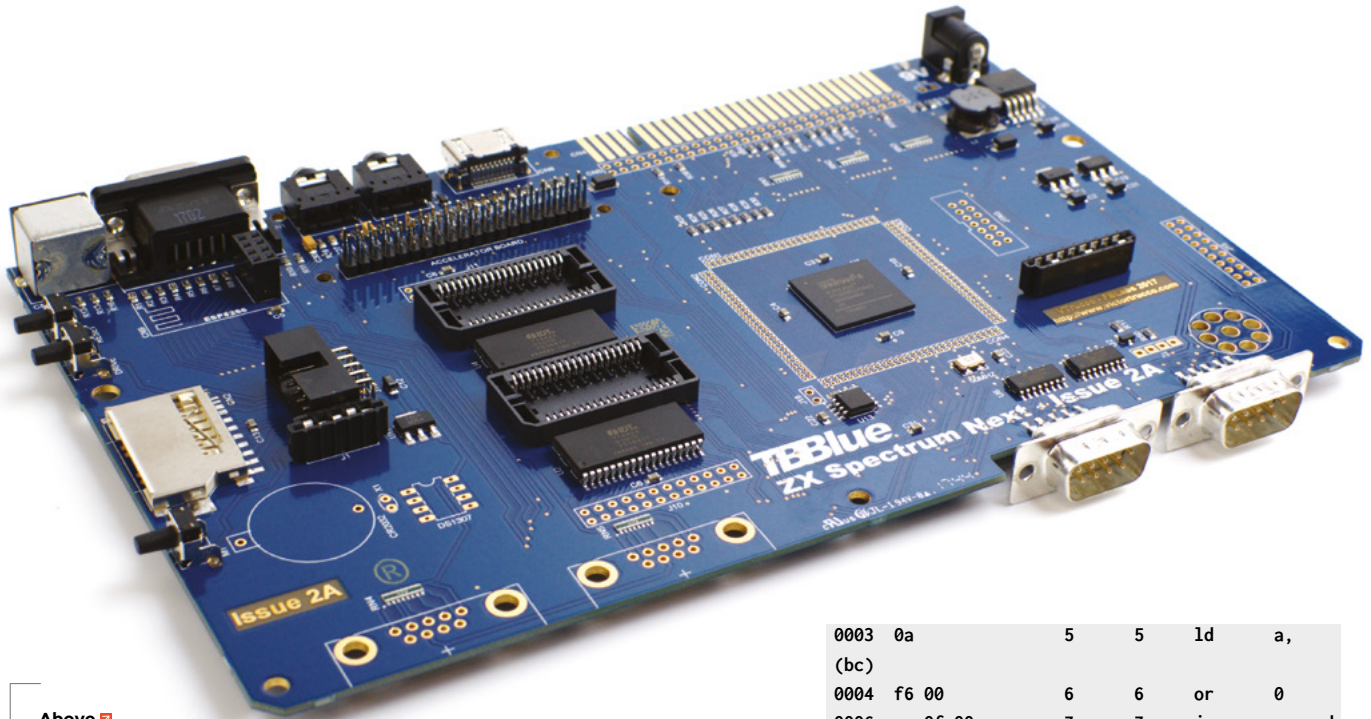
language with it? We're going to dive in head first and then walk through and explain the code.

We want a program to output 'Hello, World!', followed by a new line. We put the code below in a file called **hello.asm**. Each line is a single instruction that is made up of an opcode (what to do) and zero or more operands (what data to use).

```
title 'Hello, world'

ld    bc, hello
loop:
ld    a, (bc)
or    0
jp    z, end
out   (1), a
inc   bc
jp    loop
end:
halt
```

TUTORIAL



Above The ZX Spectrum Next includes a Z80 implemented in an FPGA

```
hello:
defm 'Hello, World!'
defb 13,10,0
```

We assemble this using z80asm:

```
>:z80asm -l hello.asm
Z80 - Assembler Release 1.8, Copyright (C) 1987-2017 by Udo Munk
0 error(s)
```

Assembling is much like compiling, in that it converts our human readable program into computer executable code.

We can see the result of this conversion by looking at the **hello.lis** file which was generated because we gave the **-l** option to z80asm.

```
>:more hello.lis

Z80-Assembler      Release 1.8      Mon Apr 23
11:47:26 2018     Page 1
Source file: hello.asm
Title:             Hello, world

LOC  OBJECT CODE  LINE  STMT SOURCE CODE
'Hello, world'
0000 01 10 00     3     3     ld     bc,
hello
                                4     4 loop:
```

QUICK TIP

The simulator has the pleasing feature that anything output to port 1 is sent to stdout.

0003	0a	5	5	ld	a,
	(bc)				
0004	f6 00	6	6	or	0
0006	ca 0f 00	7	7	jp	z, end
0009	d3 01	8	8	out	(1), a
000b	03	9	9	inc	bc
000c	c3 03 00	10	10	jp	loop
		11	11	end:	
000f	76	12	12	halt	
		13	13		
		14	14	hello:	
0010	48 65 6c 6c	15	15	defm	'Hello, World!'
0014	6f 2c 20 57	15	16		
0018	6f 72 6c 64	15	17		

WHERE IN THE WORLD?

There are different ways to specify where data is located. These are called the CPU's addressing modes. The **ld** instruction on line 3 loaded the literal value 0x0010 into BC. This is called immediate addressing; the value to use is included in the machine code immediately following the opcode. The **or** instruction on line 6 also uses immediate addressing mode. The **ld** on line 5 is different than the one on line 3 in a couple of ways: 1) it references the BC register pair as the source of the data to be used, and 2) **bc** is surrounded by parentheses. This means that the value to load is read from memory, specifically by using the contents of BC as an address, and reading the byte at that location. This is the register indirect addressing mode. It uses a register, not as the data, but where the data can be found.

In total, the Z80 features ten different addressing modes.

001c	21	15	18	
001d	0d 0a 00	16	19	defb
	13,10,0			
		17	20	

Look at line 4:

0000	01 0d 00	4	4	ld bc, hello
------	----------	---	---	--------------

We can see that the assembly language instruction **ld bc, hello** translates directly to the bytes 0x01, 0x10, and 0x00. 0x01 is the opcode for loading the BC register pair with the two bytes immediately following the opcode. B is loaded with 0x00, and C is loaded with 0x10. If we look later in the listing to line 13 and 14, we can see that the label **hello** corresponds to address 0x0010.

// The assembler also provides some pseudo-instructions such as title, defm, and defb that are part of the assembler, not the CPU instruction set //

So, let's have a closer look at this listing file. There are five columns, four of which are interesting:

LOC: the memory address of the start of the line/instruction

OBJECT CODE: the bytes that make up the assembled instruction, the machine code

LINE: the line in the source file

SOURCE CODE: the code that was in the .asm source file

Each source instruction is converted into one or more bytes of machine code. The assembler also provides some pseudo-instructions such as title, defm, and defb that are part of the assembler, not the CPU instruction set.

SIMULATION

Before we go any further, let's execute the assembled program, using z80sim as shown in the 'Running the code' box'. We load the bin file with the **r** command, giving it the name of the file. We can also provide the memory address at which to load the machine code. As we said in the hardware article, the Z80

RUNNING THE CODE

```

>:z80sim

##### ##### ###          ##### ### # #
# # # # # # # # # # # # # # # #
# # # # # # # # # # # # # # # #
# ##### # # ##### ##### # # # #
# # # # # # # # # # # # # # #
# # # # # # # # # # # # # # #
##### ##### ###          ##### ### # #

Release 1.36, Copyright (C) 1987-2017 by Udo Munk

CPU speed is unlimited
>>> r hello.bin,0000
Loader statistics for file hello.bin:
START : 0000
END   : 001f
LOADED: 0020

>>> g 0000
Hello, World!
HALT Op-Code reached at 000f

PC  A  SZHPNC I  IFF BC  DE  HL  A'F' B'C' D'E' H'L' IX  IY  SP
0010 00 010100 00 00  001f 1f94 38a3 d20b f21c c6c8 adb7 ae55 2b53 1261
>>>

```

starts executing at address 0x0000 and z80asm/z80sim assumes that's where you want it to start putting the machine code it generates/loads.

Once the code is loaded, we can execute it with the **g** command (short for go). We can provide the address to start executing at, in this case 0000.

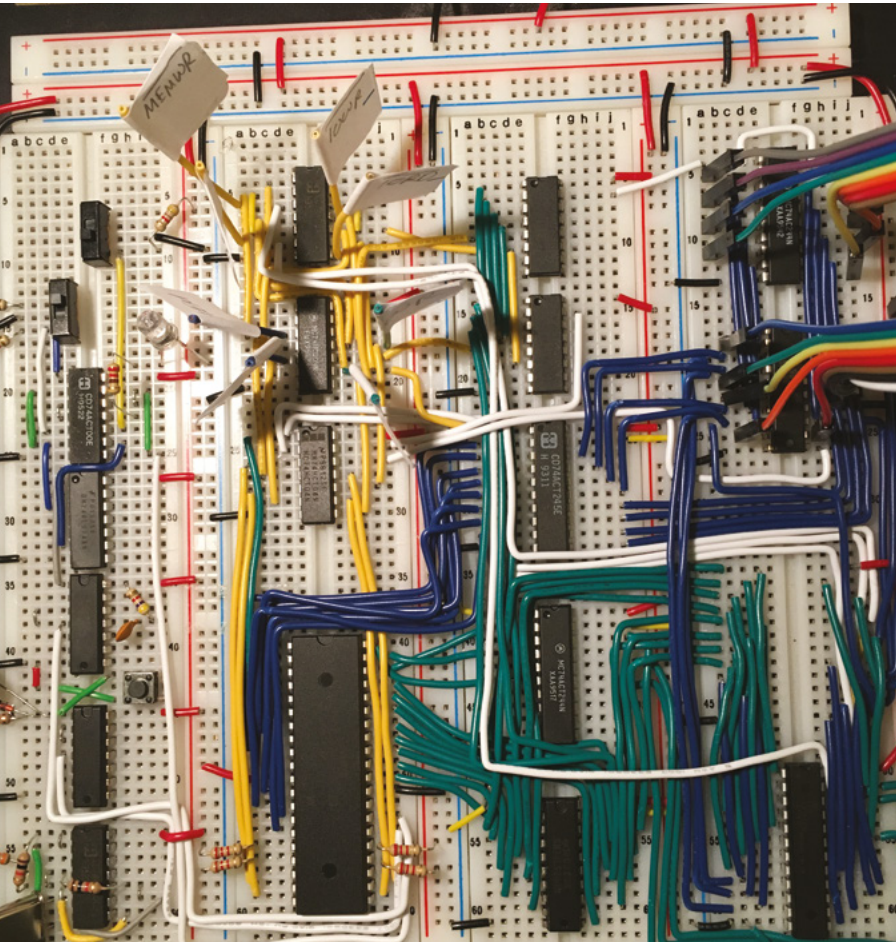
We can then see it print 'Hello, World!' and stop due to the **halt** instruction.

In the simulator we can examine the contents of memory. We can look at it as code using the **l(ist)** command.

There are many more commands available in the simulator, including debugger features like single stepping, breakpoints, etc. You can see the list using the **? command**. →



Below Originally, code was loaded into EPROMs such as this



The first thing the code does is load the address of the string (which we labelled with **hello**) into the BC register pair using the **ld** instruction. We'll be using BC as a 16-bit register, rather than two 8-bit ones. This is simply a matter of using instructions that use it that way, and that is just a matter of referring to **bc** in the assembly code rather than **b** and **c** separately. Keep in mind that you can mix and match how you use the registers; the assembler doesn't care, and the CPU (or the simulator) doesn't either.

Next, we have the start of the loop that outputs the string, a character at a time. We label this point in the code so that we can refer to its address later; **loop** seems like a reasonable name. If you look at line 10, you can see the label being used in a **jp** (aka jump) instruction.

The loop starts by loading A from where BC specifies. Next, we want to check if what was loaded into A was a zero. Every CPU has a flag register that includes bits indicating if the last operation resulted in a zero value (very commonly used for decision making), or overflowed, or generated a carry or borrow, etc. Here we're looking for the 0 byte at the end of the string, but loading a value in the Z80 doesn't change those flag bits, so we have to force the zero bit to be updated. A simple way to do that is to OR the A register with 0. That will put the result (which will be the same as the value previously in A) back into A and, more importantly, update the zero flag. Then we use a conditional jump instruction that will jump to the specified address when the zero flag is set. Here, the zero flag is set if the number in A is zero. In that case, we jump past the end of the loop and stop.

If A didn't contain a zero, we output its value to port 1. After that, BC has 1 added to it, i.e. it is incremented, so that it will contain the address of the next character (or the zero terminator). Adding and subtracting one are such common operations that they have special instructions: **inc** and **dec**.

Now we've reached the end of the loop and simply jump back to the start.

IN A ROUTINE

One important programming concept is the subroutine, aka functions. Assembly language provides two instructions that implement this concept: **call** jumps to a subroutine, saving the program counter (PC) on the stack; **ret** pulls the return address off the stack and puts it into the PC. Both **call** and **ret** have conditional versions.

We can rewrite our code using a function to output a string, and use relative jumps as a demonstration:

// **The first line simply gives the program a name.** This will appear at the top of each page of the listing file, along with the file name

//

THE CODE

Now we're ready to walk through the Hello World code.

The first line simply gives the program a name. This will appear at the top of each page of the listing file along with the file name. This is optional, but a good idea.

As we saw earlier, the string we want to output is stored in memory starting at the location labelled **hello**. The **defm** on line 15 sets aside and initialises memory for the string we want to display, one byte per character. The **defb** on the next line does the same for a series of 8-bit numbers. Here it's used to add the carriage return and line feed (characters 13 and 10, respectively) as well as a 0 to mark the end of the string.

Above ♦
The Z80 board we built last issue. Head to hsmag.cc/issue7 for details

JUMP AROUND

The code uses **jp** instructions. These jump to a literal address in memory. There's another way to do this: the **jr** (jump-relative) instructions. **jr** takes a relative value, i.e. how far to jump backwards or forwards. This is done with a single byte, so it's limited in how far it can jump to between 126 backward, to 129 forward (relative to the address of the **jr** opcode). That's a small limitation, and the advantages are that it takes one less byte of machine code, and it doesn't use a literal address, so the code can be loaded into different places in memory. That can be quite handy at times.

```
title 'Hello, world 2'
```

```
ld bc, hello
call outstr
ld bc, world
call outstr
halt
```

```
hello:
```

```
defm 'Hello, '
defb 0
```

```
world:
```

```
defm 'World!'
defb 13,10,0
```

```
outstr:
```

```
ld a, (bc)
or 0
```

```
ret z
out (1), a
inc bc
jr outstr
```

Now we can load the address of a string into BC and call the **outstr** subroutine. As shown in the code above, having a subroutine means that we can use it for different strings without having to repeat the code each time.

The subroutine is just the code we had before to output the string. The big difference is that, instead of jumping past the end of the loop when the zero byte is found, it returns.

GOING FURTHER

The Z80 has 158 different instructions and ten addressing modes available; there's a lot more than we've seen here.

Remember, every CPU has an assembly language, including the CPU cores in modern MCUs. It's very possible to write MCU code in assembly. With modern C compilers, it's not necessary to work in assembly language to get good performance, but it does give a more detailed understanding of how a particular MCU or CPU works.

If you want to explore the Z80 and its assembly language with some real hardware and don't want to build your own Z80 computer, you can get RC2014 on Tindie. The project has a GitHub repo with plenty of information and tools: github.com/RC2014Z80.

There are a variety of sites dedicated to information on the Z80 and its usage. The [z80pack](https://z80pack.com) site has plenty, and z80.info is a great collection of online material on the Z80 to boot. ▣

QUICK TIP

The CPU has a program counter (PC) register that contains the address of the next instruction to be executed.

Below ▣

Despite being over 40 years old, Z80 chips are still made today



Make your Arduino learn

Machine learning doesn't have to require data centres: all you need is know-how and an Arduino!



Zack Akil

ZackAkil

Zack is a data scientist focusing on the big problems, such as the proper filming of rugby matches.

Below ♦
Our final circuit lighting up our dark evenings

You've probably seen countless news stories about new artificial intelligence (AI) systems that can learn like humans do. At the core of these AI systems is something called machine learning. Machine learning is a technique that allows the internal rules of a system to be automatically learnt using data, rather than a programmer hard-coding in all of the rules. There is a misconception that you always require a lot of data and computing power to build these machine learning systems. For example, it's common to hear about teams using millions of data points and powerful servers in their machine learning efforts. For these larger machine learning jobs, they need those resources because they are trying to solve very complex problems, like learning how to drive a car around a busy city. For simpler applications such as 'when to turn on a lamp', the amount of resources required can be relatively minuscule: so minuscule that you can fit every aspect of the machine learning process onto a single Arduino.

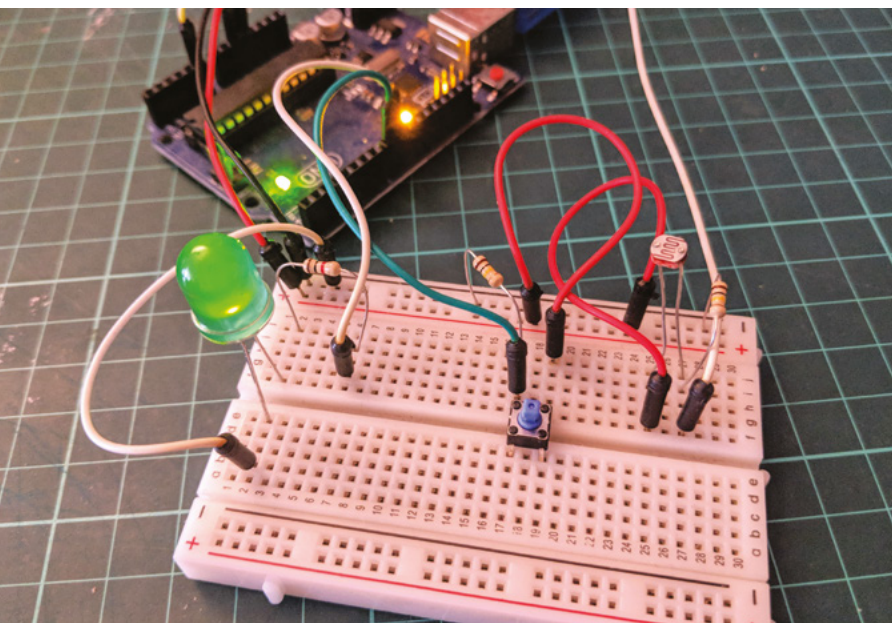
We'll walk through how to build a simple 'automatic night light' system (a system that measures the ambient light in the room and turns on an LED, based on some defined threshold) with a machine learning twist; rather than hard-coding the ambient light threshold for which to turn on the LED, the machine learning system will learn when to turn on the LED based on data collected regarding when the user has manually turned the LED on or off.

You will start off by toggling the LED on or off, as you normally would by simply using a button, and then the system will begin to learn and perform the task automatically. What's especially powerful about this system is that it can be given to many different people, and it will learn each of their individual preferences without them changing code or fiddling with a settings knob, as in traditional night light systems. As well as the system's adaptability to different people, as your own preferences change over time, so too will the system's. Behold, this is the power of machine learning systems.

There are three main processes of machine learning systems: data collection, training/learning, and prediction. Larger-scale applications would normally have these different tasks done separately (e.g. collect and store user data using a web form and database, then use servers to perform model training, then finally save the trained model into a web application that can be asked to make predictions on new data). Our system will have all of the same processes, but running on just one Arduino. Although these processes run in the order: 1) collect data, 2) train model, 3) make predictions, it is easier to explain the development of these processes in reverse order.

PREDICTION

In order to solve a problem with machine learning, you need to 'phrase' the problem with a set of variables that need to be optimised. For example, in the real world you might want to boil an egg perfectly – to do this you need to find the optimal heat level and cooking time. So, if we were using machine learning we would need a variable for 'heat_level' and 'cooking_time', and it is these variables that will get automatically



optimised. The problem we are trying to solve now is when to turn on an LED, based on ambient light – i.e. a night light system. So we are looking for the optimal threshold of when to turn on the light (see pseudo code below):

```
IF light sensor reading < optimised threshold {
  turn on led
} ELSE {
  turn off led
}
```

Just optimising a single value is a bit boring, so we'll make it a little more complex. In the classic 'night light', the light is programmed to come on when it gets dark and vice versa. However, we also want our system to automatically learn if the user wants the light to turn on if the ambient light is bright and vice versa (see pseudo code below).


```
IF light sensor reading < optimised threshold {
  IF optimised flip behaviour flag == TRUE{
    turn on led
  } ELSE{
    turn off led
  }
} ELSE {
  IF optimised flip behaviour flag == TRUE{
    turn off led
  } ELSE{
    turn on led
  }
}
```

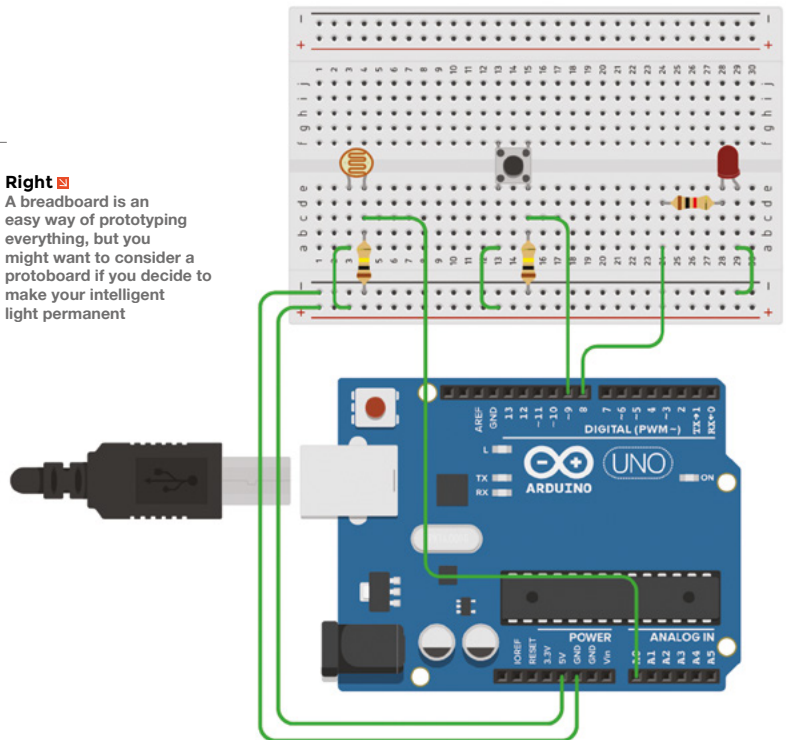
So, we have two variables in our program that we want to optimise: a sensor threshold variable (stored as an int), and a behaviour flip flag variable (stored as a Boolean). So the Arduino code for our 'model' that predicts if the LED should be on or off based on these variables is as follows:

```
bool predict(int sensor_val, int thresh, bool flip_
behaviour_flag){
  if (flip_behaviour_flag){
    return (sensor_val < thresh);
  }
  else{
    return (sensor_val > thresh);
  }
}
```

TRAINING

Because we are only trying to find the optimal values for a single integer between 1 and 1024 (range of light sensor readings), and a single Boolean which is either

Right  A breadboard is an easy way of prototyping everything, but you might want to consider a protoboard if you decide to make your intelligent light permanent



TRUE or FALSE, we can simply test every combination of values to see which one performs best, but how do we know if a combination of values is good or not?

We score the performance of a combination of variable values by using collected user data. In our case, whenever the user toggles the LED manually, our system will store the current ambient light sensor value, along with the state that the LED is toggling to. For example, if the LED is currently 'off' and the ambient light sensor is outputting a value of 700, and then the user tries to toggle the LED: because the user wants the LED to be 'on' when the ambient sensor is outputting 700, we store the value of 700 alongside the value of 'on' in an array of user data. Below is an example of some collected user data:







LED | AMBIENT LIGHT

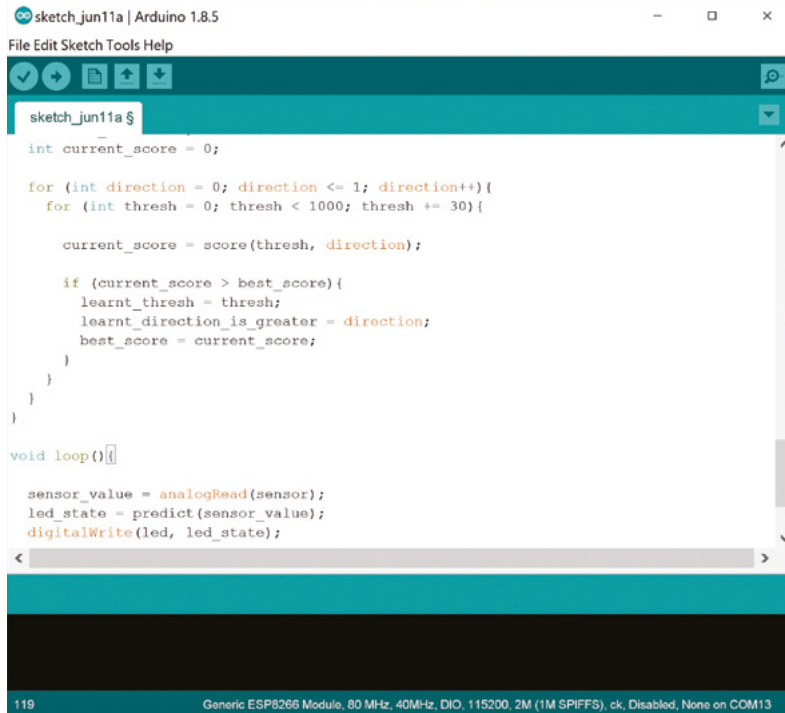
- Off – 100
- Off – 300
- Off – 600
- On – 800
- On – 850

Hopefully you can see that the user data above indicates that the user likes the LED to be on when the ambient light sensor is greater than 600.

Now that we have some user data, we can go back to the task of judging whether or not a combination of variable values for our prediction model is good or ->

YOU'LL NEED

-  **Arduino**
-  **Light-dependent resistor (LDR)**
-  **LED**
-  **Push-button (push-to-make)**
-  **2 x 100 kΩ resistors**
-  **1 kΩ resistor**



Above ♦
You can do this using either the Arduino IDE or create.arduino.cc

bad. The process involves finding out how many of the user data cases a combination would have predicted correctly. For example, here are the prediction scores for three random variable value combinations :

Flip_flag : true, thresh : 500

- 100 – on wrong
- 300 – on wrong
- 600 – off correct
- 800 – off wrong
- 850 – off wrong

Total correct = 1

Flip_flag : false, thresh : 200

- 100 – off correct
- 300 – on wrong
- 600 – on wrong
- 800 – on correct
- 850 – on correct

Total correct = 3

Flip_flag : false, thresh : 700

- 100 – off correct
- 300 – off correct
- 600 – off correct
- 800 – on correct
- 850 – on correct

Total correct = 5

Now imagine we scored the performance of every combination of threshold and flip_flag; that's roughly 2000 different combinations. The Arduino does it in a blink of an eye. Here is the Arduino code that loops through each combination of variable values:

```
void optimise(){
  // do brute force search to optimal coef and
  // intercept values
  int best_score = 0;
  int current_score = 0;

  for (int flip_flag = 0; flip_flag <= 1; flip_flag++){
    for (int thresh = 0; thresh < 1000; thresh +=
    30){
      current_score = score(thresh, flip_flag);
      if (current_score > best_score){
        learnt_thresh = thresh;
        learnt_flip_flag = flip_flag;
        best_score = current_score;
      }
    }
  }
}
```

You can see that we score the performance of each value combination and save the value combination that performs best. You may think of this as automated 'trial and error', that's because machine learning is effectively trial and error – but isn't human learning also trial and error? Try something, evaluate your performance, change what you're doing, repeat!

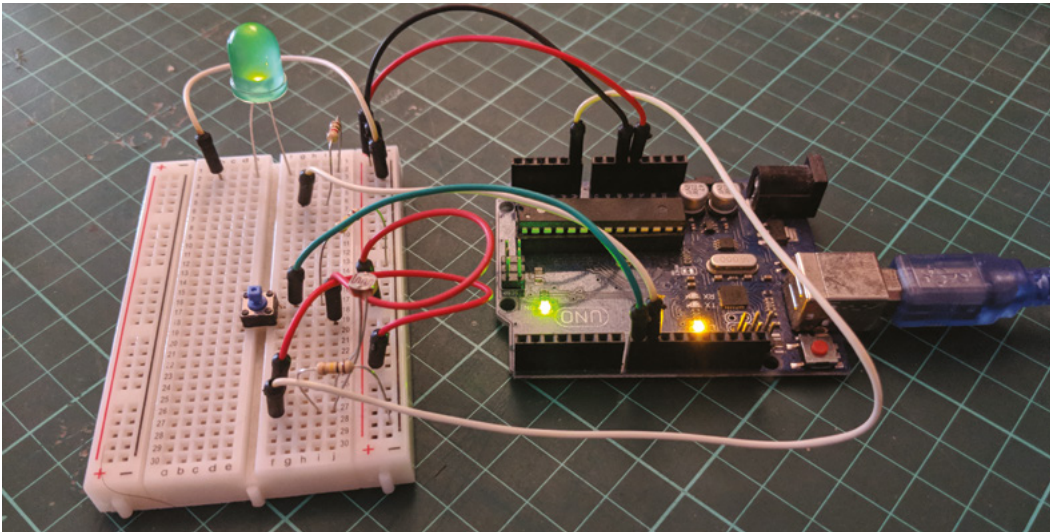
To score the performance, we are counting how many of the saved user data points the value combination correctly predicts. So if we only stored three data points, the best possible score a value combination can get is three.


DATA COLLECTION

We want this system to learn based on the most recent user input only. This will allow the system to quickly adapt to new user behaviour. We can achieve this by just storing the last ten times the user toggled

MORE ADVANCED LEARNING

Usually, machine learning systems have far more than two values to optimise (there can be millions of values in advanced image-detection systems). To train those systems it's not feasible to just try every combination, so instead they'll use something called gradient descent, which is a lot more efficient but more complex to implement.



Left  These days, marketers tell us that every item with digital electronics is smart, but can it learn? Our light can!

the LED. To achieve this with Arduino code we can create the two arrays to store ambient light sensor data and LED state data. Using a variable to track the current index of the arrays to write to next, and a variable to track if the arrays have been filled yet, we effectively create a circular array that will begin to overwrite the old data when the whole array is filled.

```
const int data_store_size = 10;

int sensor_val_store[data_store_size];
int state_store[data_store_size];

int store_cursor = 0;
bool store_filled = false;

void save_data_point(int sensor_val, bool led_state){
  sensor_val_store[store_cursor] = sensor_val;
  state_store[store_cursor] = led_state;

  store_cursor++;

  if (store_cursor >= data_store_size){
    store_filled = true;
    store_cursor = 0;
  }
}
```

The reading of the data is done when we want to score a set of prediction model variables:

```
int score(int thresh, bool direction_is_greater){

  int correct_count = 0;

  // if the store array isn't full, then only read
  upto the store_cursor
  int count_to = store_filled ? data_store_size :
```

```
store_cursor;

  for (int i = 0; i <= count_to; i++){
    bool pred = predict_with_params(sensor_val_
store[i], thresh, direction_is_greater);
    if (pred == state_store[i]){
      correct_count++;
    }
  }
  return correct_count;
}
```

Finally, we can tie all of the processes together. First we'll take a reading of our light sensor and feed it into our predictor function. Feed the output of the predictor (which will be a Boolean representative if the LED should be on or off) and set the LED to it. If the user presses the toggle button, we want to save the new data point and then run the model optimisation process (because you have just added new data).

```
void loop(){
  sensor_value = analogRead(sensor);
  led_state = predict(sensor_value);
  led_state = predict(sensor_val, learnt_thresh,
learnt_flip_behaviour_flag)
  digitalWrite(led, led_state);

  if (button_pressed()){
    debounce_button();
    save_data_point(sensor_value, !led_state);
    optimise();
  }
}
```

Play around with a night light to see how the system quickly learns whatever your night light preferences are!

The full code can be found ready for your tinkering delight over at hsmag.cc/sqSuaL. 



This is an advanced tutorial that involves handling very cold objects. Any working with a gas requires good ventilation. This tutorial is a guide only, and you should be confident in your ability to handle these substances safely before making dry ice ice cream. You are responsible for your own safety – take this responsibility seriously.

Dry ice ice cream

Apply some science to make extra-smooth frozen treats



Ben Everard

@ben_everard

Ben is busy trying to find more culinary uses for his chemistry set. Bunsen burner-grilled marshmallows for dessert anyone?

First, you need to get some dry ice. This will depend on where you live, but here in the UK it's easy to buy online. Make sure that you get food-grade dry ice to ensure that it's not contaminated with any nasty chemicals. It's also far easier to work with if you can get it in pellet form.

It'll arrive in some form of insulated box, quite possibly made out of expanded polystyrene, and it'll stay solid in here for several hours at least – if you've got a large quantity it might even survive overnight – but it's worth organising delivery on the day you'll use it, to avoid being left with an empty box (well, technically, a box of CO₂).

Before we get too bogged down in what we're doing, let's look at what ice cream is, and what makes

good ice cream. You need three basic things to create the perfect-textured ice cream: oil, water, and gas. Your ice cream should start as an emulsion of oil (or fat) suspended in water. The two usual ways of doing this are either by using egg yolks and milk to make a custard, or cream (possibly with yoghurt or milk added to lighten it). The key point is not how they freeze, but how they melt in the mouth to create that delicious feeling of eating ice cream. Many ice creams also have gas bubbles trapped in them which gives them a light and fluffy feel, even before they start to melt.

As ice cream is all about texture, it's critical to make it as smooth as possible, and this is the challenge to the home ice cream maker. As water freezes, it has a tendency to form crystals of ice. However, our emulsion isn't just water, so it won't freeze into one



RECIPES

There's a wide range of ice cream options and we couldn't limit ourselves to just one. You can be a little creative with the flavourings. Here are two bases: one for a light, fresh ice cream, and one for a richer, creamier result. On top of these bases, you'll need to add some flavouring to the liquid before you start to freeze it with dry ice. We've included some suggestions, but feel free to go off-piste.

Rich and creamy base

- 600 ml double cream
- 1 tin (397 g) sweetened condensed milk

This works well with sweet, rich liquid flavours. For example, blend 250 g of hulled strawberries and sieve to remove pits. Mix this with 60 g of strawberry jam.

Light and fresh base

- 300 ml double cream
- 200 ml natural yoghurt (full fat)
- 200 ml milk
- 300 g sugar

The slight tartness of the yoghurt comes through, but it's still got enough cream in to have a proper ice cream texture. Citrus works particularly well with a base like this. For example, add the juice and zest of two lemons.

big block of ice, but many little ones, broken up by bits of fat or oil. If these crystals are too large, then the ice cream will feel gritty before it melts.

There are two ways of preventing the crystals getting too large, and ideally you should do them both. Firstly, you should mix the liquid as it's freezing, as this breaks up the crystals as they form. This also helps to trap some air into the mixture, which improves the texture. Secondly, you should freeze the liquid as fast as possible, as this gives the crystals as little time as possible to form. It's this second process that can be challenging for people making ice cream at home. Most domestic freezers hold the temperature at -18 degrees C, but may go as low as -25. While this is cold, and easily cold enough to freeze the ice cream, it will do so quite slowly. This is compounded by the fact that air conducts heat quite slowly, so if you put a bowl of unfrozen ice cream in the freezer, the air around it won't conduct the heat away very quickly. There are some ways of mitigating this to some extent (such as putting a heavy bowl in the freezer to cool down, which will conduct the heat out of the ice cream faster than the air in the freezer). However, ultimately, the temperature just isn't cold enough to make really great ice cream.

So, we need something that can cool down a liquid quicker than a freezer. Enter dry ice, or frozen carbon




Above  Mix the flavours into the base gently to avoid knocking out too much air

dioxide, which is at least -78 degrees C. Obviously, this is much colder than the freezer, but dry ice has another trick up its sleeve that helps it make great ice cream – it's a gas at normal freezer temperature. This means that you can add the dry ice directly to the ice cream mixture and it will sublimate away, leaving only bubbles of trapped gas behind. Because of some curious quirks of chemistry, carbon dioxide



There are two ways of preventing the crystals getting too large, and ideally you should do them both



Above  We had better results starting with whipped cream than with an egg custard

doesn't form a liquid at atmospheric pressure, so it goes directly from being a solid to a gas. Both the temperature and the chemical properties of dry ice make it excellent for freezing ice cream very quickly (liquid nitrogen can also work – see box on page 107).

Those are the basics, so let's dive in and start making. There's nothing special about a recipe for dry ice ice cream, and any ice cream recipe should work but, in our experience, some work better than others. Broadly, ice cream recipes tend to fall into one of two forms: custard-based recipes that start with heating milk and adding egg yolk very carefully so that it doesn't fully cook, and cream-based recipes that are based on whipped cream. The former can work, but will work better if you have an ice cream churn to mix the liquid viciously while it freezes. →



Above ♦
An ordinary food processor will crush dry ice pellets to a fine powder

SAFETY

Minus 80°C is cold. It's easily possible to hurt yourself with this, so don't touch it with bare skin – wear a solid pair of insulating gloves (regardless of whether or not you've seen people on YouTube handle it bare-handed). We used a pair of welding gauntlets, but any solid and dry pair of gloves should work. Even with these, it's best to avoid touching the dry ice as much as possible. Move it with a scoop, rather than by picking it up by hand.

As dry ice warms up, it sublimates directly into gas. At room pressure, you won't get a liquid. This gas is odourless and invisible. If the levels of CO₂ build up in the air, it can cause problems with your body's ability to expel CO₂ from the blood, and this is dangerous. Make sure you do this in a very well-ventilated environment. Ideally, store the CO₂ outside, and only bring in small quantities as you work with them. Even with this precaution, all efforts should be made to improve ventilation and you should ensure that a good fresh air supply gets into the room you're working in. Be particularly cautious of this if you have to transport the dry ice by car.

CO₂ is heavier than air, so will sink in the room. As such, children, pets, and other creatures that breathe closer to the floor are at a higher risk than tall adults. It's best to keep them out of the area when working with dry ice.

As dry ice sublimates, its gas takes up a much larger volume than the solid form did. If you put it in a sealed container, it will increase in pressure until either all the dry ice has sublimated, or the container explodes. As such, always store your dry ice in a container with a loose-fitting lid to let the gas out.

We found that, when making ice cream by hand, whipped cream-based recipes worked best.

Start with all your ingredients chilled in a fridge (see recipes on page 104 for what you need). Starting with everything cold will mean that it will freeze faster. Then whip the cream until there are soft peaks. This will trap air inside and the bubbles will also help limit the size the ice crystals are able to get to. The next step is to add all the other ingredients (except the dry ice) and mix them in. You want to ensure that everything is thoroughly mixed, but with a gentle stirring action, rather than the more vigorous whisking you used earlier.

Now it's time to add the dry ice. Before the dry ice gets into the mixture, though, you need to make sure that it's well ground down, to a fine powder. This will help chill the liquid evenly and ensure that the dry ice sublimates quickly (rather than creating super-cooled blocks frozen around a chunk of dry ice). This is easily done using a food processor. Take a small amount (around a medium-sized mug-full) of dry ice pellets, put them in your food processor, and mix until you're left with a fine powder. Be careful when doing this because you don't want little bits of dry ice flying around, and at this point the dry ice will start to sublimate quickly – see the safety box left for details.

Add this powder to your liquid dessert a spoonful at a time. Be careful, as any dry ice spilling out could get on you. Wear gloves and other protective clothing to make sure it doesn't get on your skin. Sprinkle it

Below ♦
Decant your ice cream into Tupperware and leave it in the freezer to fully discharge any leftover dry ice



over the surface, rather than dumping it all in one spot, and mix. You'll find that the surface of the liquid freezes very quickly, but you need to mix this into the liquid. It will bubble quite violently – which is a good reason not to add more in one go! – this is the dry ice sublimating. Once the bubbling has died down and there aren't any hard spots in the liquid, add another spoonful and repeat. Quite quickly, you will find that the liquid starts to thicken as it begins to freeze. You don't need to freeze the liquid completely, just until it's frozen enough that you won't end up with large crystals as it finishes off in the freezer. Continue to add dry ice to your mixture until the liquid is almost, but not quite, too stiff to stir. You should ensure that the mix is smooth at this stage, as any lumps in there could indicate still-frozen chunks of dry ice and you want to eliminate them at this stage. Transfer to a tub and put it in the freezer to finish hardening up.

It might be tempting here to dive in and start eating your ice cream – do not. Despite ensuring that it's smooth, it might still have some bits of dry ice in that haven't sublimated yet. You need to wait long enough

LIQUID NITROGEN

Liquid nitrogen is, in many ways, a better option than dry ice. It's much colder (around -196°C) and as it's in liquid form, it mixes better. However, it's much harder to get hold of. If you can get hold of some that's food-safe, and you know how to handle it safely, it makes a great ice cream.

that all the dry ice has converted into gas. Since your freezer is about -18 degrees C and dry ice sublimates at -78 , your freezer should easily be hot enough for this to happen. There's no clear-cut way of knowing when it's happened, and the consequences of eating dry ice can be severe, so prudence is recommended. Leaving it overnight should be enough time to ensure that this has happened, but leave it longer if you're not confident all the dry ice has gone. Don't put the lid on all the way around as you need a gap to let any carbon dioxide escape.

Your ice cream is now ready to eat! ▣

Below ▣
If you've got some dry ice left, you can drop a few pellets in water to create fog for spooky effects



Build your own Arduino

Make your projects Arduino-compatible



Ricardo Caja Calleja

funwithcables.wordpress.com

An aerospace engineer by profession, Ricardo is deeply interested in robotics and automation. If there's nothing to repair at home, he'll make up some plan to build anything that includes cables or screws.

The Arduino Uno is a great board for prototyping with; however, it's bulky and a little pricey. This is because it comes with a whole bunch of stuff you might not need, such as a USB to serial converter and a large number of exposed headers. You can incorporate the core processing component from the Arduino Uno (an ATmega328P) and make it programmable from the Arduino IDE in a smaller, cheaper, and more flexible way. That's what we're doing with our 'standalone' Arduino.

The idea of the 'standalone Arduino' is to get rid of all the components included in the development board (Arduino Uno) that you just don't need. So you are left with a 16MHz crystal oscillator, a voltage regulator, four capacitors, and the ATmega328P chip, plus all the extra components that your project will need. You can build the circuit either on a solderless

breadboard, a soldered protoboard, or a printed circuit board (PCB). The last two options are better suited for final projects, as they are more durable and resistant.

Before setting up the standalone Arduino in your circuit, you will need to burn the bootloader on the ATmega328P chip and program it with the Arduino code. The easiest way to do this is by means of the Arduino Uno. The connections between the Arduino Uno and the standalone Arduino will be different for bootloading and programming, so it is recommended to perform these two tasks on a solderless breadboard.

GETTING BURNED

Table 1 shows the connections between the Arduino Uno and the standalone Arduino chip (ATmega328P) for bootloading.

First, you have to load the ArduinoISP sketch on the Arduino Uno with the Arduino IDE (File > Examples > ArduinoISP). Then you have to select Tools > Board > Arduino/Genuino Uno, and burn the bootloader onto the ATmega328P chip: Tools > Burn Bootloader. The bootloading process will take several seconds to complete.

Once the standalone Arduino chip has the bootloader burned, it is time to program it with the Arduino code. We'll do this via the Uno, but in order to make sure that we program our separate chip, not

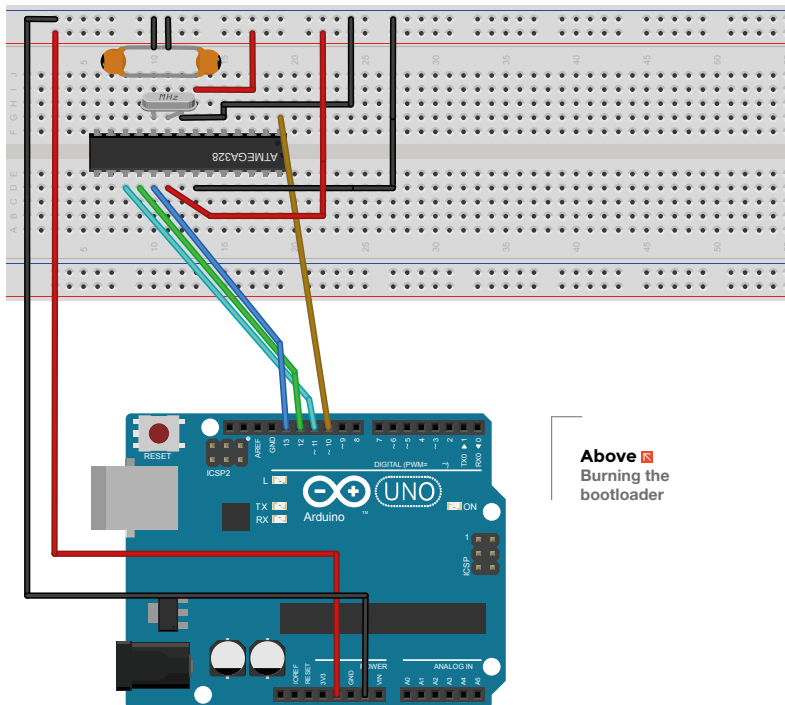


Table 1 ♦ The connections between the Uno and the raw ATmega328P. Chip pin numbers are counted anti-clockwise, starting with the pin immediately anti-clockwise of the U-shaped notch

ARDUINO UNO PIN	ATMEGA328P CHIP PIN
5V	7, 20
GND	8, 22
10	1
11	17
12	18
13	19

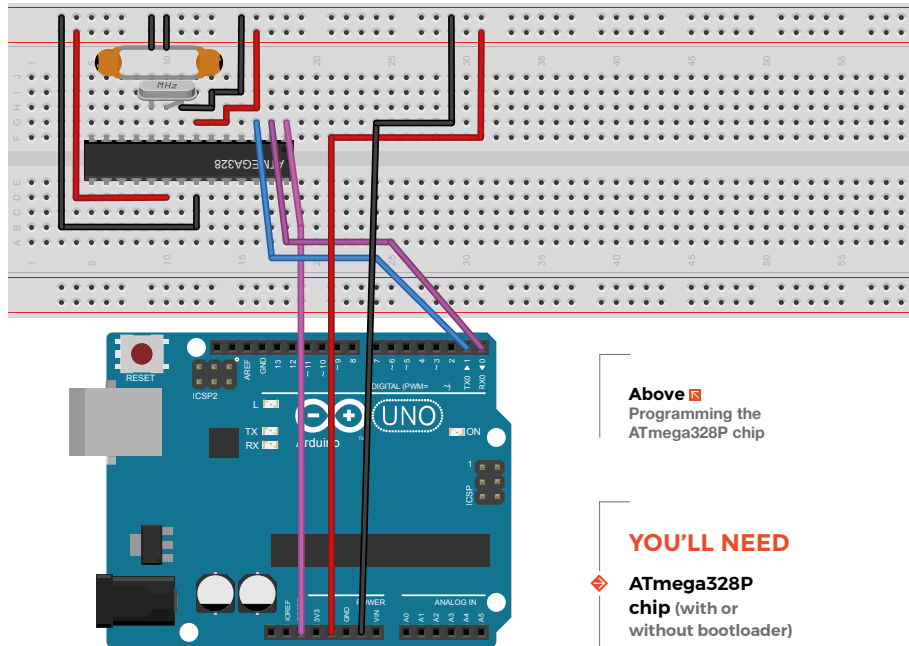
ARDUINO UNO PIN (WITHOUT CHIP)	ATMEGA328P CHIP PIN
5V	7, 20
GND	8, 22
RESET	1
Rx	2
Tx	3

Table 2 ♦ The connections for programming the ATmega328P

the main chip on the Uno, we first need to carefully remove the latter. There's one large chip in the middle that's pushed into place and can be removed by pulling. It may take a little force, but be careful not to damage the chip.

Reconnect the Arduino Uno with the standalone Arduino according to **Table 2**.

You will upload the code to the standalone Arduino from the Arduino IDE as usual (Sketch > Upload, or simply clicking on the Upload button). Now you can disconnect the Arduino Uno, put its chip back, and reuse the board for your next project.



Above □ Programming the ATmega328P chip

YOU'LL NEED

- ♦ ATmega328P chip (with or without bootloader)
- ♦ 16MHz clock crystal
- ♦ 2 × 22 pF capacitors
- ♦ Arduino/Genuino Uno
- ♦ Jumper cables

BOOTING UP

What is a bootloader? It's a piece of code that runs in the chip as soon as it is turned on (like the BIOS on your computer). If the user is trying to program the chip (with a computer), the bootloader will upload the program to the chip memory. This enables you to program the chip through the serial port, just with a USB cable.

The standalone Arduino can work with just the 16MHz crystal oscillator and the two 22 pF capacitors. However, using a voltage regulator in your circuit is highly recommended, as any voltage higher than 5V could damage the chip. This circuit can be implemented on breadboard, protoboard, or a PCB.

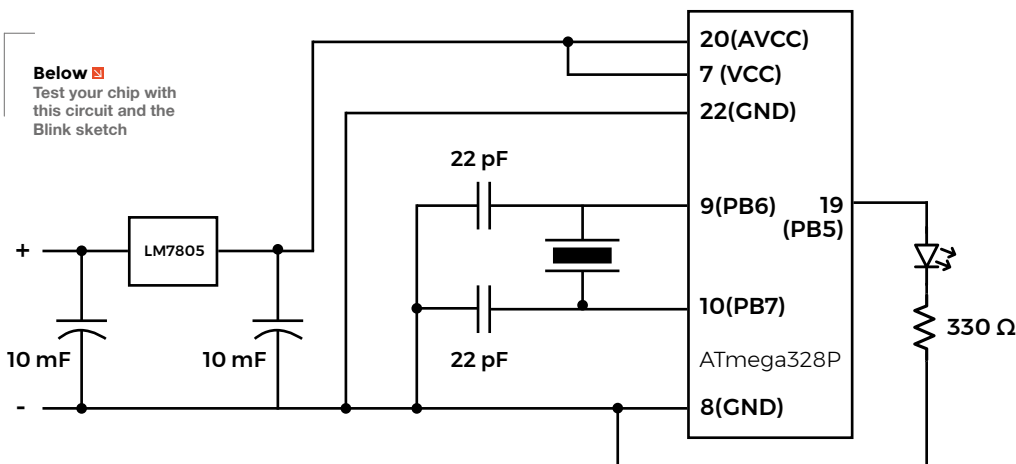
You can add all the extra connections that your project needs in the same way as you would connect them to an Arduino Uno. The datasheet (hsmag.cc/exdrFL) for the chip gives details of which physical pins relate to which internal pins, as well as other information such as the voltage and current tolerances. The internal pins are split into three banks: PB, PC, and PD. PD is digital pins 0–7, PB is digital pins 8–13, and PC is analogue pins 0–5.

Now, go out and shrink down your projects, but don't forget to let us know what you create. □

ISP FLASH

What is ISP? ISP stands for 'In-circuit Serial Programmer', and it's a device that can program the flash memory of the microcontroller, being connected to some of its pins. In this case, the Arduino Uno acts as the ISP.

Below □ Test your chip with this circuit and the Blink sketch



Restore your watch's original look

Use a few household products to make your wrist-bling shine like new



Cameron Fraser

watchtoolkit.co.uk

Cameron is a Newcastle-based watchmaker. He blogs about the tips from the trade at watchtoolkit.co.uk



Over time, watches can become scratched, chipped, and dented.

This article will describe the methods that watchmakers use to restore watches to a 'like new' condition.

Restoring a watch is a simple process. You do not need any expensive tools or supplies (although these can speed things up), as there is a good chance that you will already have most of the materials to hand.

Often when perfectly serviceable watches become a little lacklustre, they are discarded and replaced, which is bad news for your wallet and the environment.

All watches can be restored, providing you have the knowledge. This article will discuss the procedure used to restore the most common type of watch: a modern one made from stainless steel.

This method can also be used to restore watches made from other metals such as gold, titanium, and platinum, although I would recommend practising on watches made from stainless steel before progressing to working on more expensive gold and platinum watches.

These techniques cannot be used to restore watches that are plated or coated (such as gold plated, platinum clad, or black PVD), or watches that are made from ceramic, plastic, or wood.

It's usually very simple to identify the material that the watch is made from, as it is generally marked on the back of the watch or in the owner's manual. As a general rule, any modern watch made from white metal, which cost between £50 and £10 000, will be made of stainless steel, but it is best to be sure before you start.



Right Almost all metal watches can benefit from a bit of attention



Figure 1 ♦
The polished finish has an even shine that reflects images

On most stainless steel watches you will find two types of finishes. These are:

- Polished: highly reflective like a mirror (see **Figure 1**)
- Satin 'brushed' finish: a matte finish with a linear grain (see **Figure 2**)

You may find that your watch has one of these finishes, or a combination of both polished and satin surfaces.

Before you start restoring your watch, it is important to take note of how the watch was originally finished. This can be done by carefully inspecting it. If the watch is badly scratched, it may be helpful to reference a photo of the watch when it was new, and these can usually be found online.

It is important to identify the primary and secondary finishes. The primary finish will be the finish that is applied to the majority of the surfaces, and the secondary finish will be the finish that occupies the smaller surface area. The reason for this will become clear later in the article.

The first stage of the restoration is to remove the metal bracelet or strap from the case of the watch. On the majority of watches, the bracelet/strap is secured using a spring bar (**Figure 3**). The bracelet/strap can be removed by inserting a small flat-head screwdriver, a pair of tweezers, or a spring bar tool between the



Figure 2 ♦
The satin finish has a more diffuse look that doesn't show reflections

edge of the bracelet/strap and the lugs of the watch. After inserting the tool, you then need to compress the spring bar by moving the tip of the tool away from the lug of the watch. This can be a bit fiddly at first, but it's an easy technique to master.

Now that you have separated the bracelet or strap from the watch case, you will need to clean the parts. The metal bracelet can be cleaned using a toothbrush and warm soapy water, and so can the case of the watch if you are certain the watch is water-resistant. If it is not, use a clean microfibre cloth.



Before you start restoring your watch, it is important to take note of how the watch was originally finished



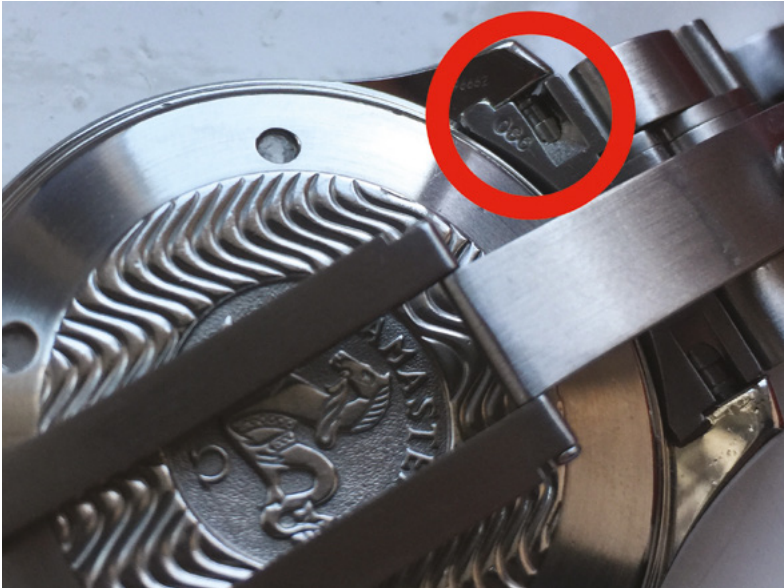
Leather and fabric straps can be difficult to clean, so it may be worth considering replacing the strap with a new one. Straps of all shapes, sizes, materials, and colours can be found online for as little as £2 a strap. All you need to know is the width of the strap, and where it connects to the watch case, and you will be able to find a replacement strap to fit.

Now that the parts are cleaned, it is time to apply the primary finish. When applying the primary finish, you should be targeting the surfaces of the watch case and bracelet that require this type of finishing, but you do not need to be concerned if the surfaces that require the secondary finish type are accidentally finished, as these surfaces will be addressed after the primary finish has been perfected.

APPLYING POLISH

A polished finish can be achieved using a variety of different methods. Typically a buffing wheel is →

TUTORIAL



“ **Abrasives are used to apply a satin ‘brushed’ finish to the surfaces of a watch, and different abrasives will give different results** ”

Figure 3 ♦ A spring bar secures the strap/bracelet on most watches

used to achieve a high lustre in very little time. A buffing wheel does have its drawbacks: using a high-speed buffing wheel is very difficult to perfect, as often an amateur polisher will remove more metal than necessary and even change the shape of the watch!

If you are new to restoring watches, I recommend polishing by hand. Hand polishing is a gentle process and there is next to no chance of causing damage to the watch.

There are dozens of metal polishes available that will produce excellent results, and I prefer to use diamond polishing paste. These can be purchased online for around £10, and a little goes a long way. A 1-micron

Below ■ To get the right finish, you need to start with the right products



grit diamond paste will usually be sufficient, although you may want to experiment with other polishes.

To use the polishing paste, apply a small bead of polish (about the size of a peppercorn) to a microfibre or a Selvyt cloth, and work it into the weave of the cloth. The cloth can then be wrapped around the tip of your finger and used to buff the scratches out of the watch. Depending on the condition of the watch, this can be a time-consuming process, but the lustre that this method produces is far greater than what can be achieved using a buffing wheel.

Some areas can be difficult to polish with a cloth. The technique most watchmakers use to finish these areas is to shape the tip of a wooden dowel to fit the area, and apply polish to the tip of the dowel.

It is important, when using any type of polish, not to accidentally polish the glass. Many watches have a thin transparent coating on the glass that reduces reflections, and polishing will remove this coating.

You are looking to achieve a smooth, highly polished surface. It is not always possible to remove all of the scratches using this method, particularly deeper imperfections. I usually stop polishing when the majority of the scratches and imperfections are removed and the surface is highly reflective.

SATIN FINISH

Abrasives are used to apply a satin ‘brushed’ finish to the surfaces of a watch, and different abrasives will give different results. Watches often have complex 3D surfaces that can make it tricky to achieve a flawless satin finish.

Scotch-Brite, like the kind used to clean pots and pans, is perfect for the task, as its sponge-like texture will conform to the shape of the watch, resulting in an even finish.

You will find that there are three common types of Scotch-Brite: white, that is non-abrasive; red, that has a fine grit for a more matte finish; and green, which has a coarse grit that produces a brighter finish.

Both red and green Scotch-Brite will produce an attractive finish.

Start by securing a large piece of Scotch-Brite to the workbench, or to a board, using carpet tape. You may want to secure a few Scotch-Brite pads end-to-end to make a long strip of Scotch-Brite.

Pass the parts over the pad at a brisk pace, in the direction parallel to the original grain. You do not need to use a lot of pressure, just enough to partially compress the Scotch-Brite and ensure full contact between the surfaces.

Check your progress after every five to ten passes. You are looking to remove the majority of

the scratches. Deeper scratches will be difficult to remove, and attempting to remove them can result in a significant amount of metal being removed from the watch. Stop when you have achieved a finish that you are happy with.

Be extra careful when removing scratches from areas such as the buckle, where there are engraved or raised logos or text, as Scotch-Brite is an abrasive and can remove these markings. Usually, I will make no more than 10 to 20 passes on areas with engravings, to prevent accidentally removing them, although it is to be assessed on a case-by-case basis.

Some areas of the watch can be difficult to reach with a Scotch-Brite pad. These can be addressed using a small piece of Scotch-Brite secured to the end of a dowel, with a drop of superglue.

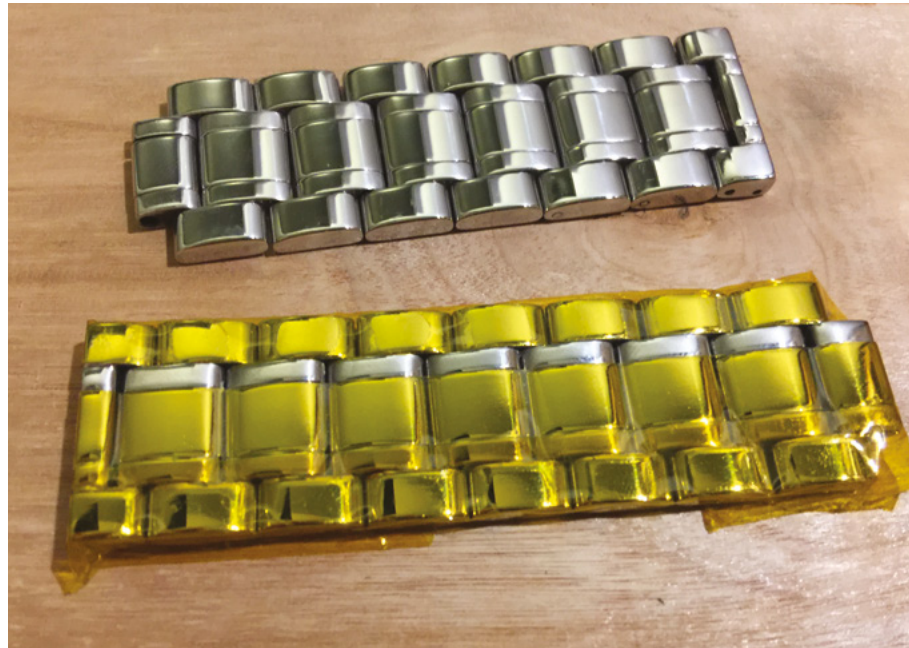
THE SECONDARY FINISH

If your watch has a secondary finish, you will need to follow these next steps.

Clean your watch again, using the method previously outlined; this is to remove any abrasive or polish residue.

The next stage is to apply masking tape to all of the areas that feature the primary finish, leaving the surfaces that require the secondary finish exposed.

Most types of tape will work as a suitable masking tape. A high quality, clear tape will work perfectly well, but an even better option is to use polyimide 'Kapton' tape. This is a special type of abrasion and heat-



Above Use tape to protect part of the surface, while you work on another area

resistant tape, designed specifically for watchmakers. It is yellow in colour so that you can clearly see what is covered by the tape and what areas are exposed. Simply put, it is Sellotape (known as Scotch tape on the other side of the Atlantic) on steroids and will make the job a little easier.

Masking a watch can be a fiddly process, but it is worth spending the time doing this stage properly. Like many things, preparation is key. Apply the tape carefully so that it closely follows the transitions between the primary and secondary finish. A razor blade can be used to cut the tape to shape.

After the watch has been carefully masked up, it's just a case of applying the secondary finish using the methods previously described. You must also periodically check that the masking tape is doing its job, and has not peeled off or worn through, applying more tape as necessary.

After you are satisfied that the secondary finish has removed all of the scratches from the surfaces, the watch is almost complete!

Now it is just the simple task of removing the masking tape, cleaning the watch again, and refitting the bracelet.

Refitting the bracelet/strap is done by positioning one end of the bracelet in between the lugs (the part of the bracelet with the clasp is the side that goes to the set of lugs at 12 o'clock) and compressing the spring bars in a similar way to how the bracelet was removed, then slotting the end link into position between the lugs.

These finishing techniques can be adapted and applied to just about any watch and to other metal items to produce a unique aesthetic. □

Left Ordinary household Scotch-Brite can give a satin finish





Build a Makerspace for Young People

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

Sign up today: rpf.io/makerspace

FIELD TEST

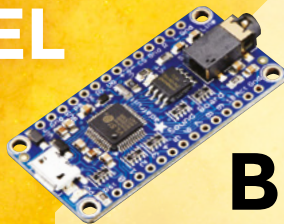
HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG
116

DIRECT FROM SHENZHEN: LED PANEL

Make everything sparkle
with a wrap-around



PG
118

BEST OF BREED

Are you ready to rock?
Add sounds to your makes with our pick
of audio add-on boards

PG
122

CAN I HACK IT?

A mini arcade machine gets
pulled apart for improvement



REVIEWS

124 **Cricket**
Interface with the world

126 **Line-us**
A sketchbook for your computer

128 **Simba-Pro**
Bluetooth gets really small



129 **Hippo**
Computing a better future

DIRECT FROM
SHENZHEN

LED screen

Lighting up the world 256 pixels at a time

By Ben Everard


@ben_everard

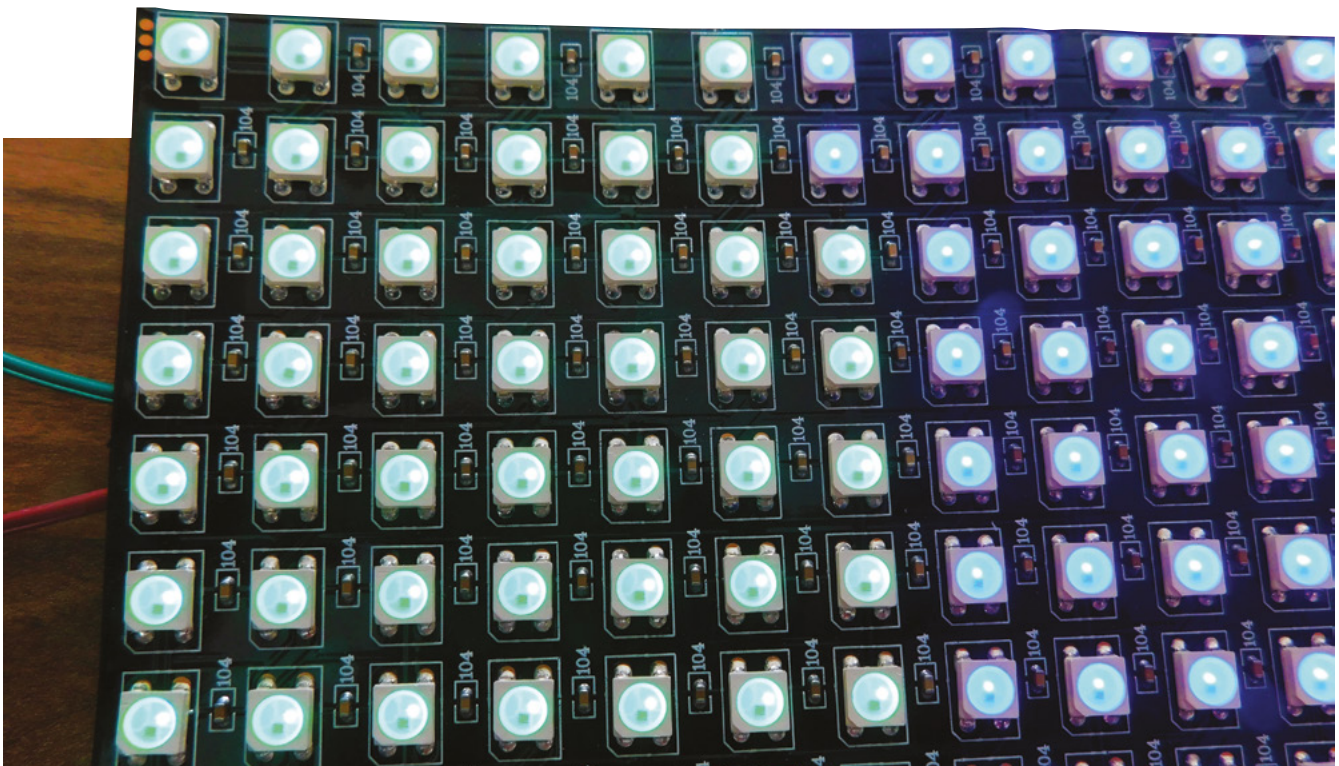
RGB LEDs are a wonderful invention that allow us vast amounts of visual creativity in our builds. You can light things up as subtly or as garishly as you like. In reality, these lights aren't a single thing, but three tiny LEDs mashed together in one package. This means that to control them you need three pins – one for each colour. This quickly adds up and, even with multiplexing, it's hard to control much more than a grid of ten by ten RGB LEDs. Fortunately, help is at hand in the form of the WS2812B, which is an RGB LED with embedded microcontroller. These have a really simple networking protocol, so that each has an input pin and an output pin. Attach a

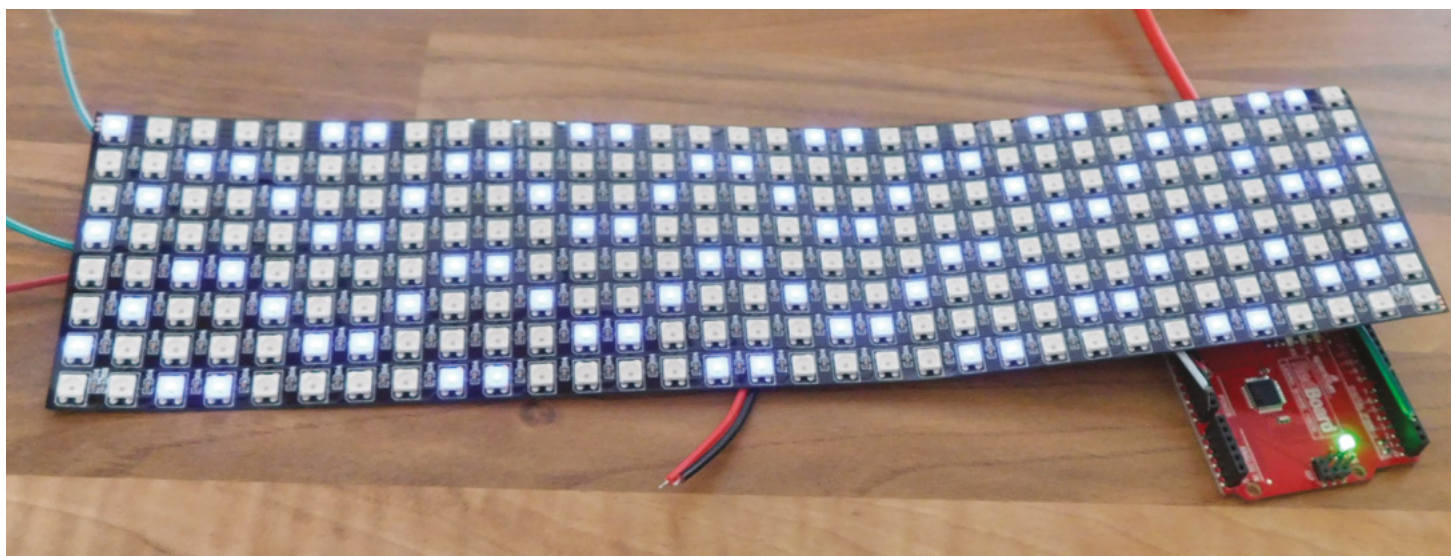
microcontroller to the input pin of one WS2812B, then attach its output pin to the input pin of the next, and you can daisy-chain a string of these LEDs almost as long as you like, (the only limits being power and microcontroller memory).

These little LEDs are best known as Adafruit's NeoPixels, but the same part is available in a lot of forms from a lot of sellers – and crucially, they all use the same protocol, so you can use the Arduino NeoPixel library to drive them.

We ordered an 8 × 32 pixel flexible display for £25.61 from AliExpress store Ledworld. That's 256 LEDs. The first thing you need to realise when using this number of LEDs is that you're not going to be able to use a simple power supply. At full brightness,

Below  Even at low brightness, the colours stand out and are easily visible





Each LED uses up to 20 mA of current, and each pixel has three LEDs (a red, green, and blue), so that's potentially over 15 amps! Of course, this is the maximum current if you've got all your LEDs on white, at maximum brightness. If you know that you're not going to hit this, you may be able to get away with a lot less. Even with them all on full brightness of a single colour (red, green, or blue), this comes down to five amps.

DANGER!

Our screen came with three points to supply power, which should reduce the amount of current through each point. All three are connected on the PCB though, so you shouldn't connect them to different power supplies, just three times from the same one. This will reduce the resistance between the power supply and the LEDs. We do have some concerns about the quality of the joints joining the leads onto the PCB. The wire around the soldered blobs seemed prone to fraying.

We wouldn't be comfortable using this display at a high current because of this fraying, and the potential fire risk this entails. That's not to say that there isn't a safe way of using these displays (after all, anything sets on fire if you put enough current through it). There are plenty of applications where you can use this sort of display without using the high current levels that could cause risks – if you don't need to illuminate all the LEDs, or don't need them all bright, you can run this safely. Make sure you're familiar with the issues before going down this route.

If this screen isn't big enough for you, it does come with a connector to daisy-chain multiple ones together (in this case they can be powered by separate power supplies, as long as you tie the grounds together).

As well as power, it takes memory to run these as the microcontroller has to be able to manipulate the graphics data and send it out. You need three bytes of RAM per pixel, so our screen will use 768 bytes of RAM. An Arduino Uno has 2kB of RAM, so there's enough to fit our screen in, provided we don't need too much additional memory.

Our screen came on a flexible PCB, but had no specifications attached, so we don't know how tight it's supposed to bend before breaking, or how many times it's supposed to withstand bending. It creaks a little while bending, so we suspect the best answer to the above questions is not very tight, and not very often, however ours didn't actually break during our testing.

// **Our screen came with three points to supply power, which should reduce the amount of current through each point** //

Addressable RGB LEDs are a great way of making your builds stand out. Generic WS2812B LEDs can be cheap, but come with no support or documentation, and are of variable quality – for experienced makers, this may not be a problem but it might lead to confusion for beginners. It's best to start small to make sure you know what's going on before diving into a display this big. While it's great to get cheap bits, the extra money on a reputable manufacturer may be worthwhile if you need support or parts to handle high currents. □

Above □
With careful control of the brightness, you can run this off a single USB connection

DIRECT FROM SHENZHEN

When beep, boop, beep just won't do

BEST OF BREED

ONLY THE BEST

When beep, boop, beep just won't do

Adding high quality audio feedback to your next project

By Marc de Vinck

@devinck

Sometimes your project would be better, or at least more fun, if you could just add some audio feedback that's a little more sophisticated than just a beeping piezo or small speaker.

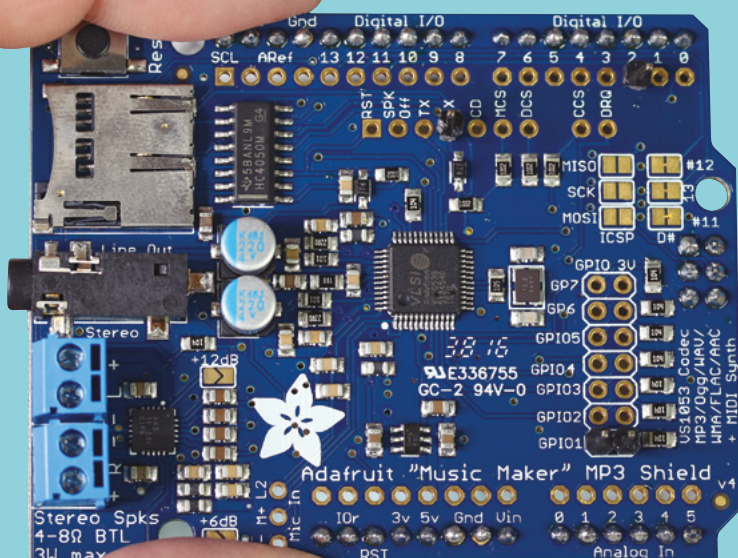
We're not talking about building your own audio amplifier, because we know that would just be a topic on which we would all end up debating best practices for decades, if not

longer. We're talking about adding a little audio flair to your project by playing back pre-recorded audio.

You might think that playing an MP3 audio file is easy, but as soon as you start looking into the actual requirements of encoding and decoding WAV, MP3, or other audio file formats, you'll soon come to the conclusion that many times it's just easier to pick up a shield or board to handle all the heavy lifting. In this Best of Breed article, We'll be looking at a range of solutions for adding simple tones, recorded messages, or even full HD audio playback to your next project.

There are so many options out there, as we quickly found out when doing a deep dive into the seemingly never-ending abyss of audio add-on boards available online for this review.

Even several of the boards we've selected to review include multiple variants that feature different capabilities or board compatibility, so be sure to head on over to the manufacturer's website for even more options before making a selection. Hopefully these reviews will inspire you to add a little more than just a simple beep, boop, beep in your next project. Don't get me wrong, we happen to love the sounds of R2-D2, but sometimes C-3PO is a little easier to understand.



SparkFun MP3 Trigger

Triggers MP3s, exactly as it claims

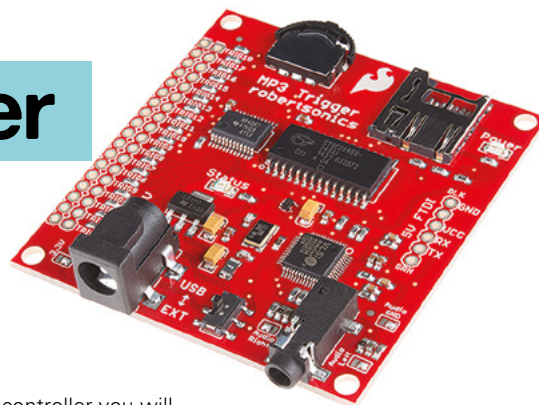
SPARKFUN ◆ \$49.95 | sparkfun.com

Adding audio playback in any project can be a bit difficult, especially decoding MP3 files. Using the SparkFun MP3 Trigger, with its on-board Cypress PSoC CY8C29466-24SXI microcontroller, it couldn't be easier. The board features 18 external trigger pins that can trigger a variety of preselected MP3 tracks; or, by using the serial control port, you can control up to 256 tracks. All your MP3 files are stored on a microSD card, making it easy to change and edit them as needed from your computer.

SparkFun has configured the board to work in several different ways, depending on how you

Right ◆
A go-to board for adding MP3 audio to your project

Credit
SparkFun Electronics CC BY 2.0



like to work and what microcontroller you will ultimately use, if any. The product page has more detailed information about volume controls, playback methods, and an interesting 'Quiet Mode' which basically reports back, via a serial message, what track should be played, allowing for a lot of flexibility in choosing what microcontroller you prefer to use in your audio project.

The comments on the product page suggest using it for Halloween props, custom doorbell chimes, and museum exhibits. What you ultimately use it for is up to you, and thanks to the way it's configured, you should be able to implement MP3 playback in your next project. □

VERDICT

An easy-to-use, but expensive, way of triggering sounds stored as MP3 files.

8/10

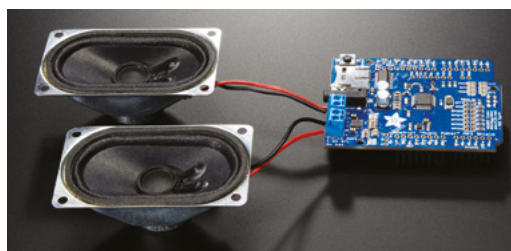
Adafruit 'Music Maker' MP3 Shield for Arduino

Expand your Arduino

ADAFRUIT ◆ \$34.95 | adafruit.com

The Adafruit Music Maker shield for Arduino features a VS1053 encoding and decoding chip that can decode a wide variety of audio formats, including the very popular MP3, AAC, Ogg Vorbis, WMA, MIDI, FLAC, and WAV files. You can also record audio in PCM (WAV) and Ogg Vorbis formats.

All of the board's functionality is handled through an SPI interface, which enables the Arduino to easily play files directly from an SD card. And since the board has a built-in MIDI synthesizer and drum machine,



Left ◆
Audio playback on an Arduino made simple

Credit
Adafruit licensed under CC

you can easily play a variety of built-in drum and sample effects.

The board is packed with a lot of audio goodness, but how do you get those sounds from a bunch of bits to something you can hear? No worries: there is an included 3-watt amplifier built in, which can power 4 or 8Ω speakers directly. Some soldering is required, but once it's completed, you will have a great audio board to use on your next Arduino project. □

VERDICT

An integrated 3-watt amp and multiple format support make this a great choice.

9/10

When beep, boop, beep just won't do

BEST OF BREED

IQaudio Pi-DAC+

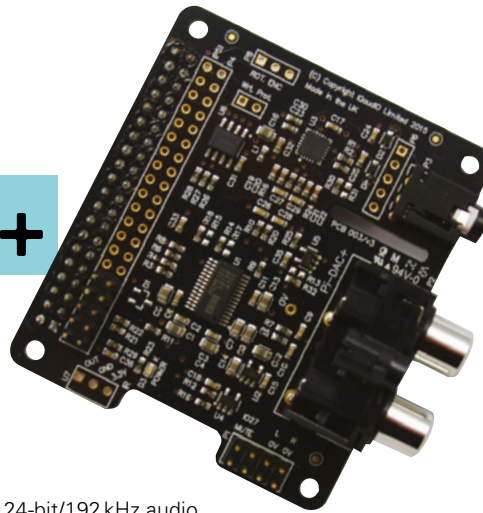
Upgrade your Pi's audio

IQAUDIO ♦ £33.89 | iqaudio.co.uk

We all know that audiophiles tend to be very particular, and extremely passionate, when it comes to the quality of their audio playback, and the IQaudio Pi-DAC+ audio card aims to please even the most discerning of those makers. The board can deliver a full HD audio experience thanks to the on-board Texas Instruments PCM5122 digital-to-analogue converter (DAC). This enables a Raspberry Pi A+, B+, Pi 2 or 3,

to output 24-bit/192 kHz audio playback, which is much higher than many other DIY audio boards.

The Pi-DAC+ also features a headphone amplifier and industry-standard phono/RCA connectors to allow for easy integration into your existing setup. All the power requirements are supplied from the Raspberry Pi, and it comes fully assembled and ready to use. And since no soldering is required, you'll be up and listening to HD-quality audio on your Raspberry Pi in just a few minutes. □



Left □
Add high-quality audio to your Raspberry Pi

Credit
Iqaudio Limited

VERDICT

Don't underestimate the power of this high-quality audio board. It's really good!

9/10

Adafruit Audio FX Sound Board – WAV/OGG Trigger with 16MB flash

An all-in-one solution

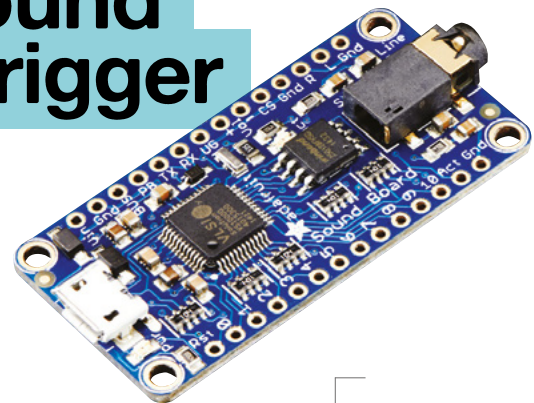
ADAFRUIT ♦ \$19.95 | adafruit.com

Sometimes you just want to add sound to your project and don't need or want a microcontroller involved. There are people that (gasp) don't know or even like to code! If that's you, or even if you love to code, the Adafruit Audio FX Sound Board might be a perfect fit for adding all sorts of sounds and noises to your build.

What's interesting about this board is you don't even need an SD card. All files are stored directly on the board via built-in mass storage. Just drag and drop your files on the board after plugging into your

computer. There is 16MB of storage, which is plenty of space when it comes to WAV/Ogg files.

Once you upload the files, you can simply add up to eleven buttons or switches to trigger them to play. And how they play, and on what pin, is determined by how you name the files. You can have the files be activated with a simple trigger defining the pin, or hold and loop playback, a latching loop playback, or play next or play random. The device can play all your files at 44.1 kHz in 16-bit stereo sound. This is a perfect solution for Halloween props, costumes, or toys. This board packs a lot of functionality without any code required. □



Above ♦
Audio playback on an Arduino made simple

Credit
Adafruit licensed under CC

VERDICT

Perfect when you don't need a microcontroller

10/10

Speaker pHAT

Keeping everything tiny

PIMORONI ◆ £13.50 | shop.pimoroni.com

Even though the audio quality of the Speaker pHAT by Pimoroni isn't up to snuff with the audiophile community, the retro-fantastic look is hard to beat. There are many times, as we say, when the beep, boop, beep of a piezo speaker just isn't good enough. There are also times when you just don't need full HD audio either. That's when the Speaker pHAT really shines.

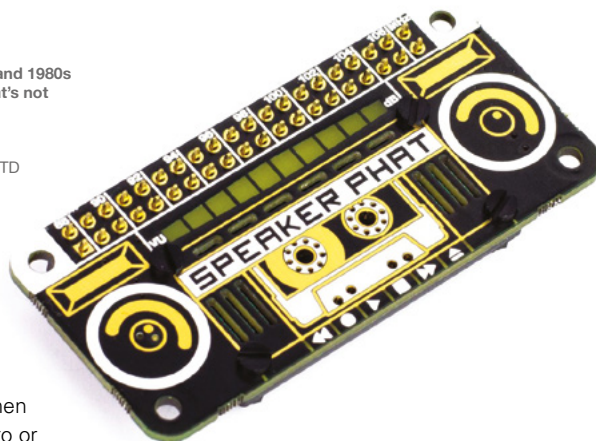
Pimoroni was able to cram a I²S DAC, mono amplifier, 8Ω 2W speaker, and a 10 LED bar graph all onto the incredibly small pHAT form factor. It's a clever system that makes a beautiful little audio sandwich and adds a lot of capabilities to your

Raspberry Pi. And even though its form factor works best when coupled with a Pi Zero or Zero W, it will also work with a Raspberry Pi A+, B+, 3, or 3B+.

Pimoroni has made it really easy to use the Speaker pHAT by developing a simple one-line installer that takes care of everything you need to get it up and running. The installer configures ALSA to output sound through the DAC on the Speaker pHAT, and it installs a custom plug-in that will display the sound level on on-board LED bar graph. You can download all the instructions from the GitHub repo. □

Right ▣
Practical and 1980s retro. What's not to love?

Credit
Pimoroni LTD



VERDICT

All-in-one audio for your Raspberry Pi Zero

8/10

Teensy Prop Shield with motion sensors

Combining light and sound

SPARKFUN ◆ \$19.95 | sparkfun.com

The Teensy Prop Shield is a perfect tool for anyone familiar with the Teensy platform and who wants to create projects with sound effects and interactive light displays. The shield features 10 degrees of freedom sensing by implementing a FXOS8700CQ 6-axis linear accelerometer, a magnetometer, a 3-axis digital angular rate gyroscope, and a precision pressure/altitude and temperature gauge.

Also included in the shield is a 2W audio amplifier capable of driving external speakers, and it's capable

of controlling Dotstar or APA102 type LEDs. In addition to all those features, you also get 8MB of on-board flash memory where you can store sound clips and log data. That is a lot of power packed into such a tiny shield!

The Teensy is a great audio platform as it's got quite a bit more processing grunt than most hobbyist microcontrollers, and this is a particular boon when working with audio signals. If you want to go geeky, this is a great platform to work with, but if you don't then it's also easy enough to use without worrying about the low-level details. □

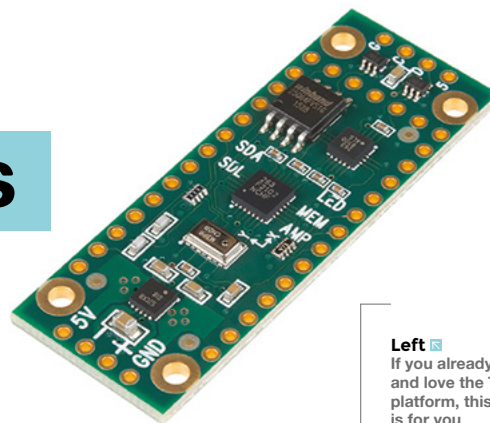
Left ▣
If you already know and love the Teensy platform, this shield is for you

Credit
SparkFun Electronics
CC BY 2.0

VERDICT

A great way to add functionality to your Teensy project

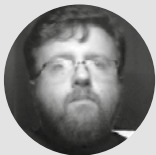
8/10



Can I Hack It?

A mini arcade cabinet?

Can a tiny arcade cabinet be hacked?



Les Pounder

@biglesp

Les Pounder is a maker and author who works with the Raspberry Pi Foundation to deliver Picademy. He also helps teachers/learners to become creative technologists. He blogs at bigl.es

Right

This cabinet is truly tiny, but we have so much scope for hacking and we will not need any specialist equipment!

YOU'LL NEED

Funtime ET7850 Mini Arcade Machine Toy

COST

£19.14

WHERE

hsmag.cc/LTgVEL

We love the arcades of yesteryear, the vibrant colours and sounds that fill the air, designed to entice us to put another 10p into the slot (OK, 50p these days). These arcades

are dwindling, largely due to home consoles being so powerful and fostering large communities of gamers. But for those of us who enjoy the arcades, we dream of owning a cabinet, or even building our own. Thanks to the Raspberry Pi and RetroPie, we can make that dream real; all we need to do is make a cabinet. But what if you haven't got the space? Well,

a tabletop cabinet is possible, but we can go even smaller and, using the 'Mini Arcade Cabinet', we have a 'donor cabinet' that we can use to create our own miniature cabinet. So dear reader, let's take a look at a £20 cabinet that may just help us create our own miniature nostalgia trip!

THE STOCK MACHINE

Powering up the machine and taking it for a spin, there are four categories of games: 'Sports', 'Shooting', 'Puzzle', and 'Arcade'. Inside of these categories we find a mixture of unlicensed games from the early 1980s, Flash games, and some very dubious clones

of popular games. Sure, they are fun and we spent a good hour playing as part of the tear-down, but we can have more fun building our own machine.

GENERAL CONSTRUCTION

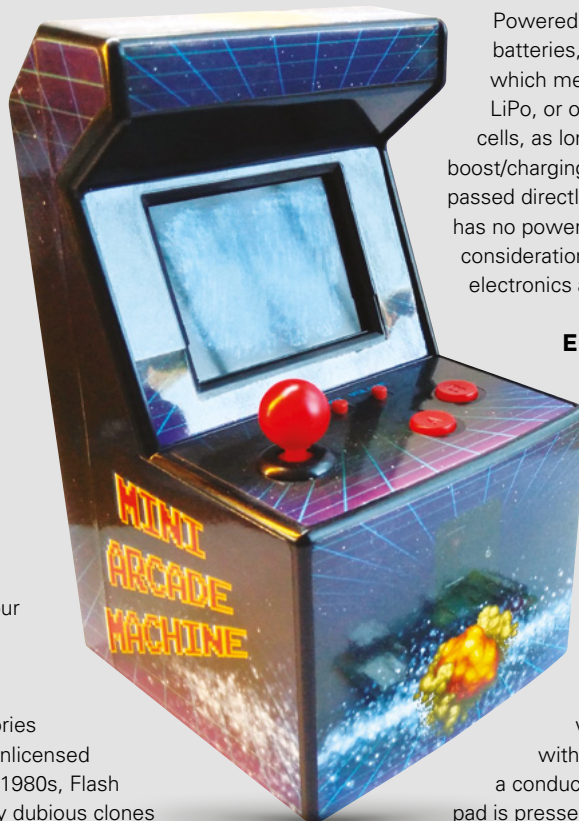
Made from rigid plastic that can be easily worked with hand and power tools, this arcade cabinet is tiny, measuring just 14.5cm tall × 8.3cm wide × 8.6cm depth. Internally there is sufficient space to add your own electronics, including a Pi Zero W, as long as it is mounted to the base of the unit, as this has the greatest surface area.

POWER

Powered by three AA (LR6) batteries, we see 4.5V of voltage, which means we can easily install a LiPo, or other type of rechargeable cells, as long as we use a suitable boost/charging board. Power is then passed directly to the main PCB, which has no power regulation, so careful consideration is needed to ensure the electronics are not damaged.

ELECTRONICS

This is a little bit of a Frankenstein's monster, as the electronics for this cabinet are actually the same as used in the many Chinese clone hand-held consoles that can be picked up for a few dollars online. We can identify this due to six pads on the board, which are normally used with rubber buttons that hold a conductive pad; when a button/pad is pressed, this causes a connection

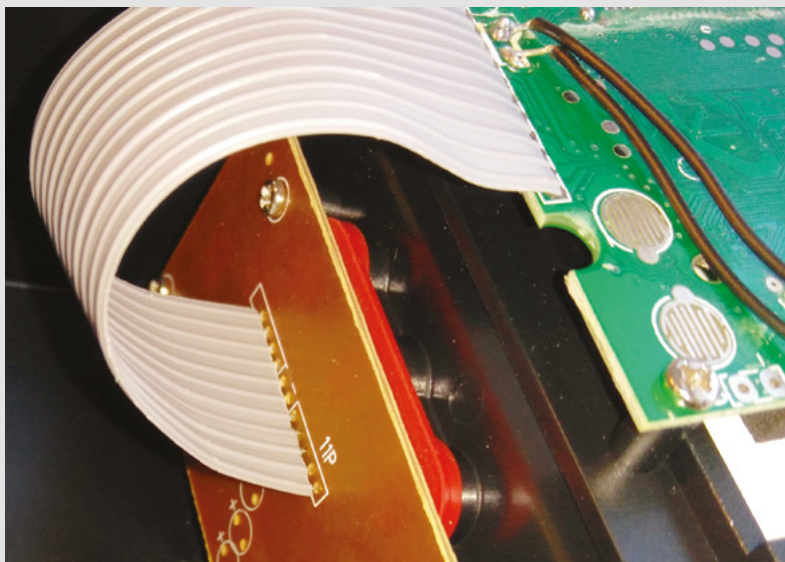


THE DIY APPROACH

Building your own arcade cabinet is not that difficult, seriously! A quick look on eBay and we can find bartop (i.e. to fit on a table) cabinet kits that include all of the electronics and parts needed to build the project. All you need to do is add a screen, Raspberry Pi / Asus Tinkerboard etc., and connect everything up and you have a working cabinet. Using software such as RetroPie retropie.org.uk, we can easily configure the cabinet to run games from the dawn of video gaming using emulators such as MAME (Multi Arcade Machine Emulator) up to the PlayStation era, so around 20–30 years of gaming history. But where do you get the games? Well, this is where it gets a little murky. Game files for emulators are called ROMs, as arcade cabinets originally had their games delivered on large circuit boards filled with ROM (read-only memory) chips. These ROM files are still under copyright, which means distributing or copying them is illegal, but many sites offer them for free download. There's a community of developers creating ROMs that can be shared. There's a selection available at romhacking.net/homebrew that you can use freely.

to GND or VCC and triggers a change in state. This then triggers the game to make our character jump, or spaceship fire its weapons. There are six wires that are directly attached to the electronics. These are for power, an interrupting power switch, and the speaker. Connection between the electronics and the screen is via a custom cable, held in place but rather fragile. The screen is a cheap LCD measuring approximately 5 cm × 3.5 cm, but there is scope to add your own miniature screen, and space to add a slightly larger screen, such as those from Adafruit or Pimoroni. Also present on the board are a number of test points, used in the factory to test the board, and we can use them to bridge connections, fix broken wires, and tap into the power supplied from the batteries.

But the most visible part of the electronics is a big grey ribbon cable at the bottom of the PCB. This links to another PCB hidden under the controls. This PCB is tricky to get out as it is screwed in place and you cannot get a conventional screwdriver in there, so we used a ratcheting offset screwdriver and revealed a simple PCB that uses the same conductive pads to detect input. We could easily reuse this board with an arcade-to-USB interface that can provide more functionality and enable us to use the stock controls with a Pi Zero! Upon closer inspection there are also three unpopulated connections for LEDs, so we could use those to add lights to this tiny cabinet.



CONCLUSION

As a gadget the cabinet is fun, but as a chassis on which we can create our own miniature arcade cabinet, it is sublime. It has the internal volume necessary for a Pi Zero W and a USB breakout board, and we can easily add a new screen to replace the custom display. Being able to reuse the controls via the USB breakout board means the outside appearance of the unit will not change. For £20, this is a bargain and we can have a lot of fun building our own cabinet without the need for a laser cutter or 3D printer! All we need are some screwdrivers and a soldering iron.

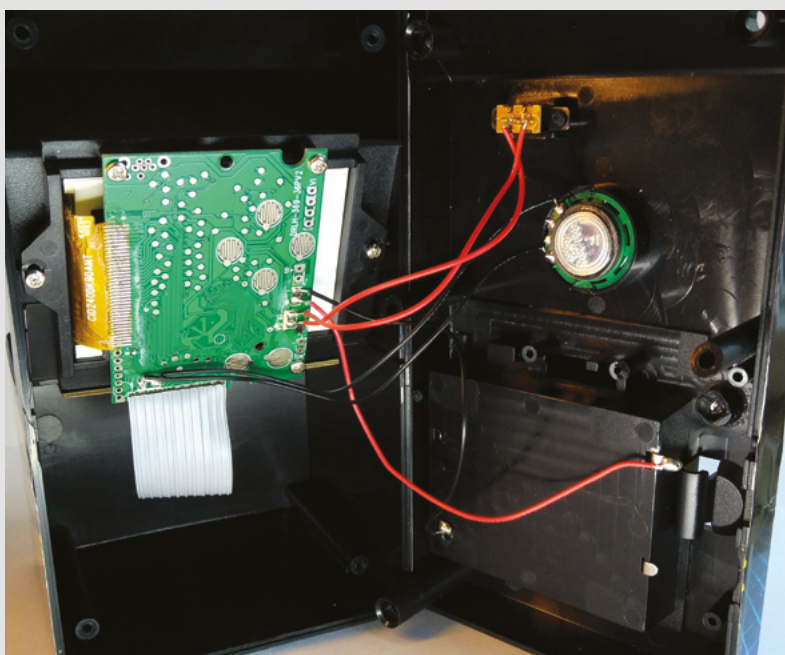
Happy hacking! □

Above

The controls for the cabinet are housed in a separate PCB, which we can reuse with a USB breakout board to control a RetroPie cabinet and retain the original look of the cabinet

Below

The PCB is reused from a handheld console, and we can see the pads that were previously used for the controls




Adafruit Crickit for Circuit Playground Express

A lot of hardware control for the Circuit Playground Express

ADAFRUIT ◆ \$29.95 | [Adafruit.com](https://adafruit.com)

By Ben Everard

 ben_everard

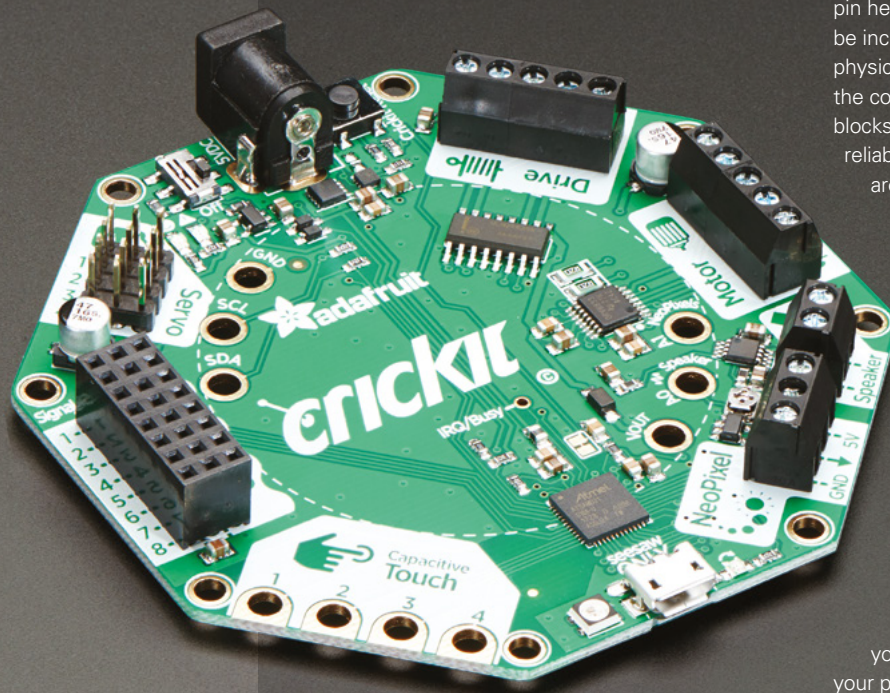
The Creative Robotics & Interactive Construction Kit (Crickit) for Circuit Playground Express (CPX) is an add-on board for the CPX for interfacing with physical hardware. It adds a range of hardware controllers, including two motor controllers (5V, 1A), four servo connections (analogue or digital), four high current (500mA) outputs, a class-D 3W audio output, and four capacitive touch pins. Together, these enable you to control almost any physical hardware you may want (the only

major limitation is for stepper motors – some will work with the Crickit, but not all). All of these items are designed to run at 5V.

As microcontroller boards go, the Circuit Playground Express isn't naturally suited to add-on boards. Almost all other small processing boards have a set of male or female headers that you can easily push an expansion board onto. The CPX, on the other hand, has a circular array of larger holes designed for alligator clips. The Crickit gets around this with six bolts that both hold the CPX in place and create an electrical connection.

Like the CPX, the Crickit eschews the traditional pin headers of many circuit boards. It's designed to be included in things – be they robots or some other physical computing device – and as such, most of the connections are robust. Most are screw terminal blocks, which are both easy to connect/release and reliable for connections on bits that move and bounce around. The servo connections are male headers that most servos can plug straight into, the signal connections are male headers, and the capacitive touch retains the large circular connections of the CPX. There is a micro USB jack, but this is for debugging the SeeSaw controller that handles the input/output on the Crickit. Power comes via a 5V 2.1mm jack connector that can take up to 4A. Be aware that while 9V jacks such as those used to power Arduinos will fit, they will trip the eFuse, which only accepts connections below 5.5V.

There's a mounting hole in each corner of the octagonal PCB, which should give you plenty of options for securely attaching it in your project.



The Crickit can be programmed from MakeCode, Circuit Python, and Arduino. The MakeCode support is still in beta but is available through makecode.adafruit.com/beta, using the Advanced > Extensions block and searching for Crickit. This allows you to drag and drop controls to the motors, servos, signals, and touch. Circuit Python and Arduino support comes via the SeeSaw library, which handles the I²C interface between the CPX and the various peripherals.

When we originally tested the CPX, we were impressed by the wide range of peripherals on board, and the way it ‘just worked’ for a wide range of things. The Crickit brings this same approach to controlling physical hardware. Sure, you probably won’t need all the connections in one project, but you might need most of them at some point in the next few projects, and having them all in one place makes it easier to get started quickly.

The one big caveat with the CPX and Crickit combination is that there’s currently no easy way of communicating with it via WiFi or Bluetooth. Fear not, though. The Crickit is coming soon to the Feather line of boards which includes a wide range of different microcontrollers and numerous connection options (this is expected shortly after this magazine goes to press and might be available

SEESAW

SeeSaw is a framework developed by Adafruit to use a microcontroller as a coprocessor to another microcontroller via I²C to add more peripheral options. This only uses the two pins necessary for the I²C protocol, and you can connect multiple I²C devices to a single I²C bus, so it allows you to add a lot of peripherals without taking many resources from the original microcontroller.

On the Crickit, there’s an ATSAM21 microcontroller that handles all the extra hardware. This means that the processor on the CPX doesn’t get bogged down with the details of sending signals to the servos, motors, etc. It just sends a signal to the Crickit telling it what should happen and the Crickit handles the rest. You can think of it a little like a super-powerful I²C port expander.

The Crickit isn’t the first board to use SeeSaw. The Adafruit SeeSaw board (hsmag.cc/XllrnR) connects via I²C and has analogue, digital, and PWM GPIOs. This board works with Android-compatible boards, Circuit Python, and even Python on the Raspberry Pi.

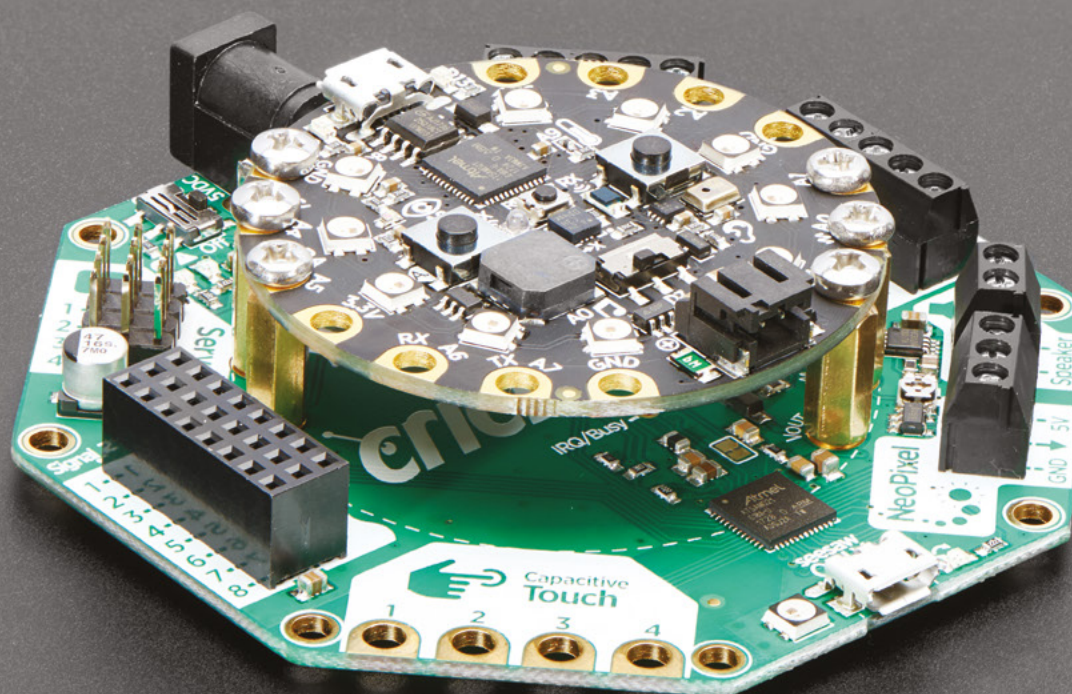
by the time you read this). There’s also a planned version for the Raspberry Pi.

Overall, if you’re looking for a board that works with a wide range of input and output options, it’s going to be hard to beat the Crickit. It’s a generalist board for all us tinkerers who never quite know where the project is going when it starts – with all those output options, you’re bound to be able to add on the extra feature that you didn’t think of when you started, but now absolutely need. ▣

VERDICT

A cracking board for adding a wide range of hardware capabilities to your CPX

9/10



Line-us

Add real-world images and words to your projects

LINE-US ♦ £89 | line-us.com

By Ben Everard

 [ben_everard](#)

The overwhelming majority of graphical output from computers is rasterized, where images are converted to a series of dots that are displayed as light or ink.

With small enough dots, our eyes can be tricked into thinking that these dots represent real curves or other shapes, but it's always a lie. Look closely enough and the dots always expose themselves. There are, however, a few hold-outs – products that refuse to buy into this discrete world of points, and live in the world of lines. Line-us is one such product.

Two servos control arms that move a pen in two dimensions, and a third lifts the pen off the paper when it's not drawing. The simple arm arrangement has quite a limited drawing area that's in the shape of half an ellipse, with a chunk missing. This is due

to the radial arm arrangement, and the body of the machine limiting access to part of the drawing area. It's approximately 16cm at its longest point, and 7cm at the widest. You can access the full area by using the API (see below), but only a rectangular portion of this if you use the app.

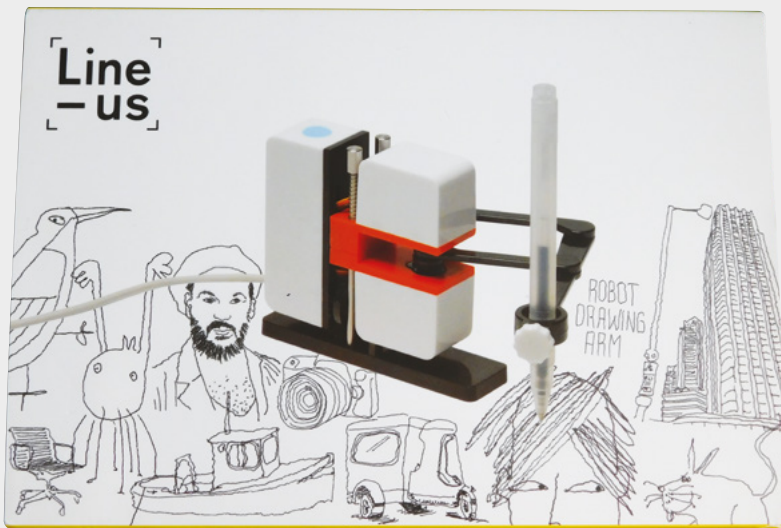
The device comes with a steel backing plate. You place your paper on the plate, then the magnets in the Line-us snap down, holding everything in place while it draws. When you're done, everything comes apart again. With this setup, you can draw on a sheet of paper, a page in a sketchbook, or anything else that the robot can fit on. Although the drawing area is quite small, this setup means you can place it on a much larger sheet of paper (and it can be moved round to the place you want it).

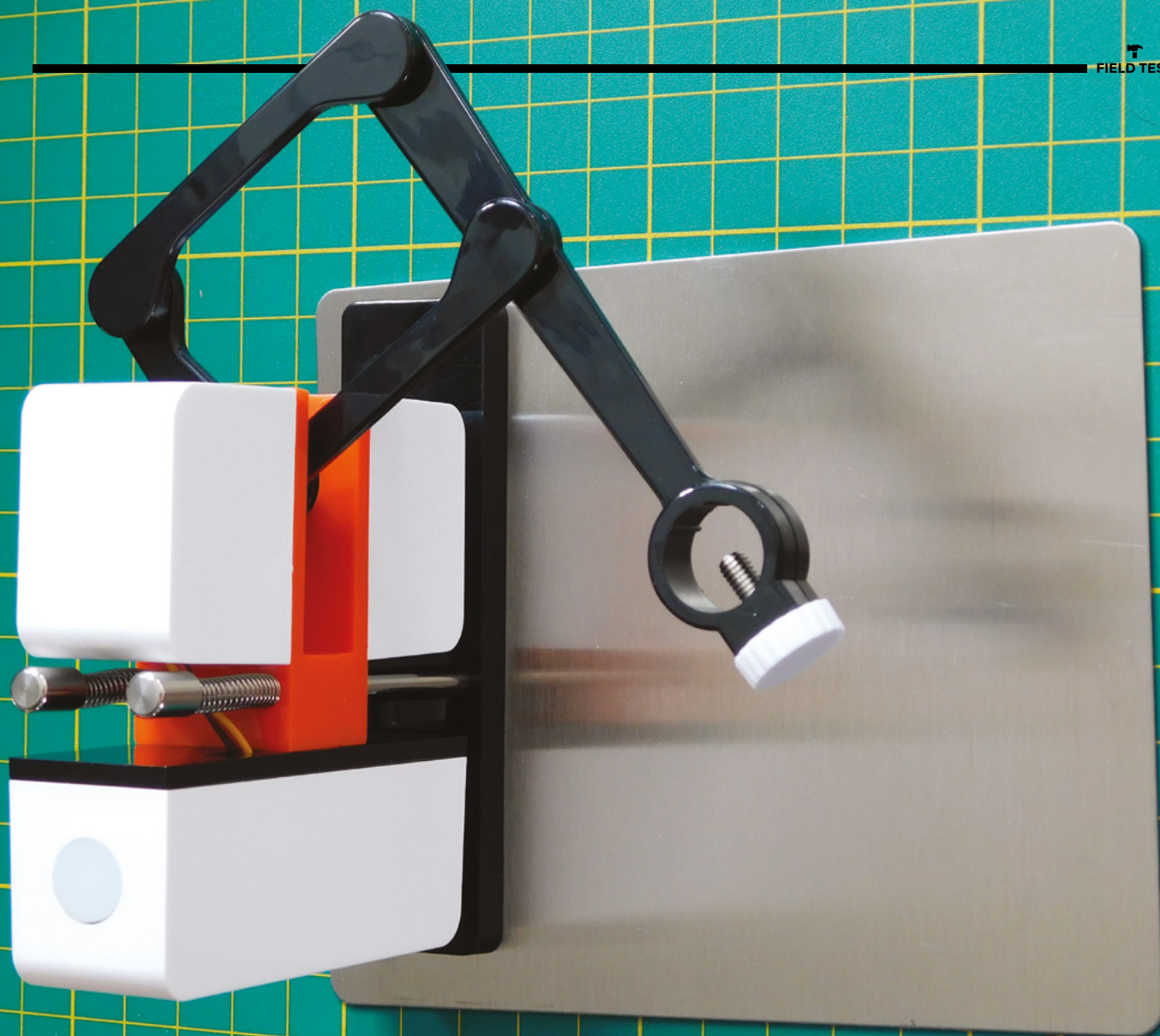
The Line-us can use any pen with a diameter less than 13mm. However, you can't change pens mid-print, at least not automatically – there's nothing to stop you having different sketches representing different colours, but you will need to manually change and calibrate the pen each time.

The easiest way to get started with Line-us is via the app (which is available for Windows, OS X, iOS, and Android). This gives you a sketch area to draw your designs, and access to the cloud, where you can store your designs or share them with others. This connects to the device over WiFi (a Line-us can create its own WiFi network if there's not one already available), so there are no wires to get in the way, or to limit the places you can put the Line-us.

As yet, there are no public sketchbooks (or the ability to share with the world at large), though the FAQ claims that this 'will be added to the software very soon'. These would make it much easier for the more artistically inept of us to get started with the technology, and learn what's possible with it.

Below ♦
The drawing quality is exactly what is shown on the packet – wobbly, but with a certain charm





WIBBLE WOBBLE

The print quality of the Line-us is, frankly, inaccurate. It appears that the resolution on the servos just isn't high enough to accurately reproduce images, and there ends up being a noticeable wobble to the lines. If you're after something for drawing straight lines or detailed images, you're going to need something else. However, the *raison d'être* of the Line-us is as a sketcher. The slight imperfections in the lines can add to the visual effect of this.

What takes the project from toy to hackable tool is that the API to control the device is accessible. Controlling the device is just a case of opening a TCP socket and sending G-code. There's a Python class to help you do this, and examples in C and Processing, but it's fairly straightforward to do in any language.

It's easy to imagine a variety of maker projects using the Line-us. The form factor makes it easy to include anything that needs a graphical output. Perhaps the most obvious is a camera that vectorises images and sketches them out, but it's far more flexible than this. It's not limited to ordinary pens – anything that will fit into the penholder will also work. Dry erase markers are an obvious choice, but more exotic options can also work. The Line-us team have drawn on biscuits using royal icing, for example.

Line-us is a fun tool that's easy to get started with and easy to program. It's priced competitively with other plotters, but comes with some pretty huge caveats – most notably, the small print area, and the accuracy (or lack thereof) in the prints. □

Above □
The Line-us is simple to set up, just plug in the power and insert a pen

VERDICT

An easy and fun way of robotic drawing, but with a little wobble in the lines


8/10

Simba-Pro

A minuscule Bluetooth dev board with a lot of features

SENSIEDGE ◆ \$70 | sensiedge.com

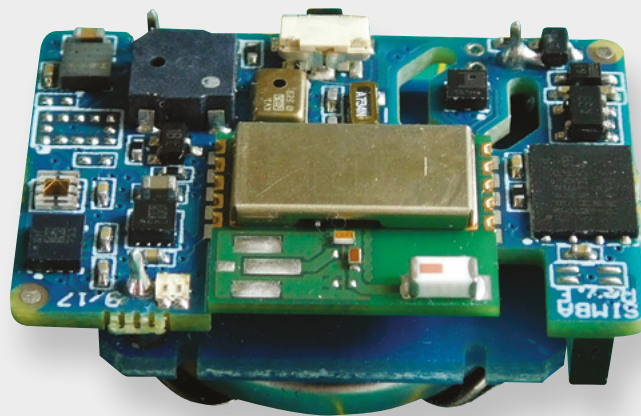
By Ben Everard

 [ben_everard](#)

The Simba-Pro IoT module is a Bluetooth dev board built on an ARM Cortex-M4, with a whole lot of extra features. There are position sensors (accelerometer, gyro, magnetometer), environment sensors (temperature, humidity, pressure, light, microphone), and ways of interacting with the user, including a button and a red-green LED. You can also add onto this with a wide range of expansions via the GPIO pins that support I²C, SPI, CAN, UART, and PWM. This is all packed into a board smaller than an SD card.

There's obviously a lot on this board, and you could do a lot of projects without needing any expansions. In these cases, the tiny size and light weight are a huge advantage. The more you have to add on to this core, the more you lose the key advantage of its size.

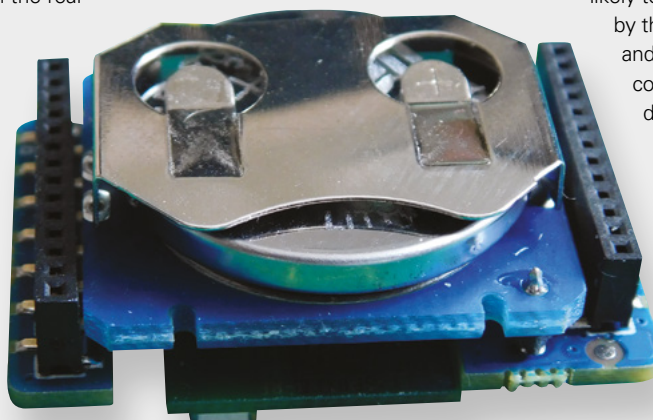
There's a bundle of software for working with it, including the usual Arduino IDE board definition. Particularly intriguingly, there are frameworks for Android and iOS software that can work with it reading the sensor data. For software devs who don't want to get dirty in hardware, this provides an easy introduction to building software which interfaces more with the real world. It would, for example, allow you to build a Wii Nunchuk-style input device without having to get out your soldering iron or breadboard. There's a sample application that you can run without even



touching the hardware – just power it up and you can get the info you need off the device.

While the documentation for the project isn't awful, neither is it great. The information you need is likely to be on the website, but it won't necessarily be easy to find or understand, especially if you're new to embedded hardware development. This is a slightly unusual situation: it's an interesting option for real beginners because you can do so much without having to worry about the hardware, but it's not a great option for intermediate users because once you do have to start worrying about the hardware, it's not as simple as many other options. The more you need to add to it, the more you're

likely to get bogged down by the documentation and relatively small community of hobbyist developers on this platform. For advanced users, the advantages kick in again – provided you don't need to add too much to it, it's still small, low power, and quick to get started with. □



Above □

There's a whole lot of sensors in this tiny PCB

Below □

The CR2032 battery only just fits between the headers

VERDICT

A feature-packed board for Bluetooth development, but with some quirks.

7 /10

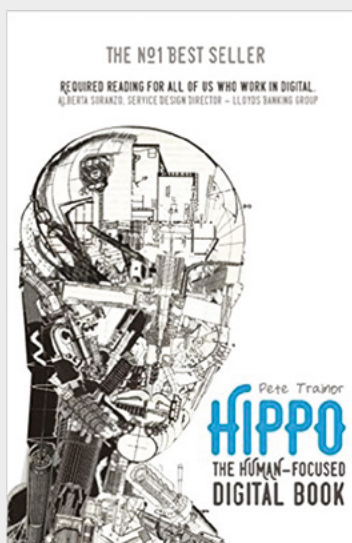
Hippo: The Human-Focused Digital Book

Pete Trainor ♦ £8.99 | nexus.cx

By Richard Smedley

 RichardSmedley

Hippo here is the hippocampus, the part of the brain that – amongst other things – forms contextual associations. In a world where designs can now potentially adapt to each person using them, this book puts in a heartfelt plea for a design based on what it is to be human, not for a 'lost target within the bamboozling and bombarding world of sneakers with shiny lights on.'



Digital design and human-centred design are adaptive methods for producing an output; *Hippo* contrasts 'human-focused digital' as a signature, one that does not change as long as what being human is does not change. Think about questions, not functions, and design for how you want people to feel – not for what you want them to do.

He takes a reductionist approach – 'nature operates in the shortest way possible,' he quotes from Aristotle – in an industry noted for 'complicated approaches to simple challenges.' The underlying aim is always to better enable computers to deal with our mundane chores, so that we can move up Maslow's Hierarchy of Needs towards self-actualisation, and finally make that film or write that book – however terrifying the prospect!

First we must leave Plato's Cave, which, in an update to the analogy for incomplete knowledge of the way things are, is mediated by phones and tablets. What marks out human beings is conversation – chatbots engage, and if they have transparently bot-like personalities, can succeed in fostering dynamic and multidimensional relationships with apps and products.

Along the way, *Hippo* takes in Taoism, considers 'if Socrates had Siri?', and looks at the chemicals produced in our quest for happiness through our screens. Moving past reservations about (a lack of) editing, and a few rather strained metaphors, this is a thought-provoking work which will get anyone involved in designing interactions with computers to step back and ask a few important questions, such as "is this universal or merely novel?"

At heart, this is a plea for companies to do better things, rather than simply do things better, and *Hippo's* contrarian voice is a necessary antidote to the current direction of development. □

VERDICT

This will help your design succeed, by making you look at the 'why' of your ideas and interfaces

7/10

next month

issue
#9

ON SALE
19 JULY



FEATURING TOOLS! TOOLS! TOOLS!

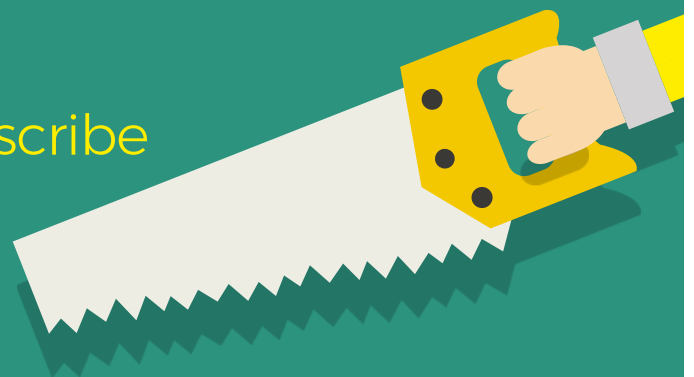
OUR PICK OF THE **BEST**
TOOLS AROUND

ALSO

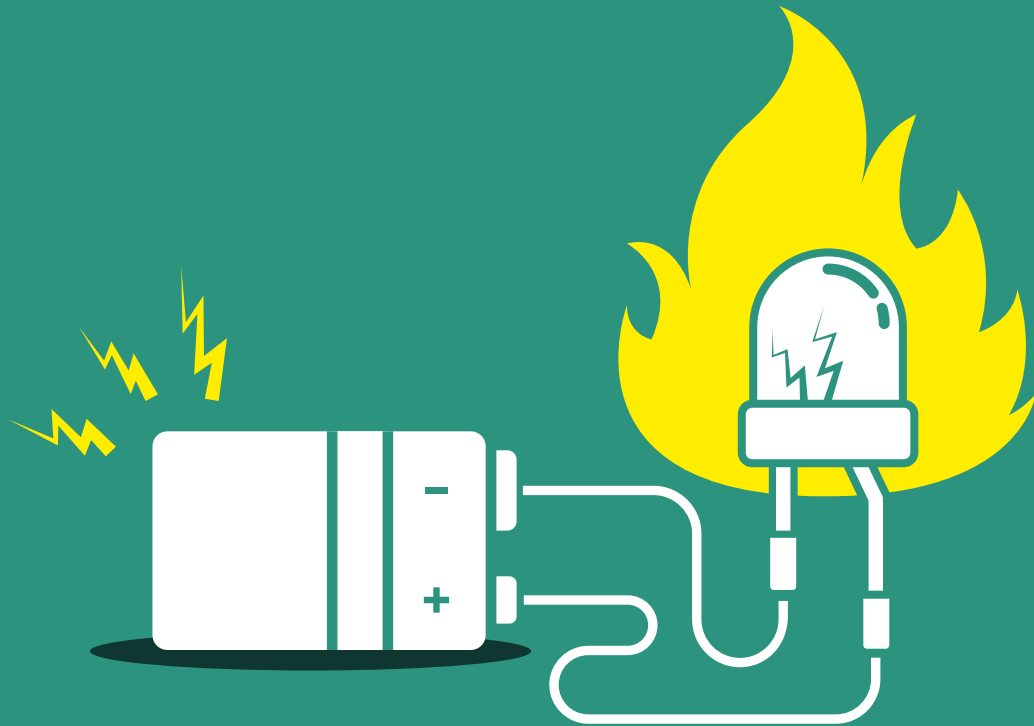
- LEARN ELECTRONICS
- ARTIFICIAL INTELLIGENCE
- USING MIDI
- ARDUINO
- AND MUCH MUCH MORE

DON'T MISS OUT

hsmag.cc/subscribe



HackSpace
TECHNOLOGY IN YOUR HANDS



RESISTANCE ISN'T FUTILE

**LEDS SHOULD GLOW
NOT BLOW**

— hsmag.cc —

Canakit Raspberry Pi 3 Complete Starter Kit



Kit Includes:

- ✓ **Raspberry Pi 3 Model B**
Quad-Core 1.2 GHz 1 GB RAM
- ✓ **On-board WiFi and Bluetooth**
- ✓ **32 GB MicroSD Card (Class 10)**
- ✓ **Canakit 2.5A Power Supply**
- ✓ **High Quality Case**
- ✓ **HDMI Cable with CEC support**
- ✓ **MicroUSB Reader**
- ✓ **Set of Heat Sinks**
- ✓ **GPIO Quick-Reference Card**
- ✓ **Canakit Quick-Start Guide**

Available for worldwide shipping at:

WWW.CANAKIT.COM



\$74.⁹⁹ **£59.⁹⁹** **€64.⁹⁹**
US DOLLARS EXCLUDING VAT EXCLUDING VAT

Raspberry Pi is a registered trademark of the Raspberry Pi Foundation.
Canakit is a registered trademark of Cana Kit Corporation.