

MAKE | BUILD | HACK | CREATE

# HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc

March 2018

Issue #04



## WEARABLE TECH

ELECTRONICS MEETS FASHION

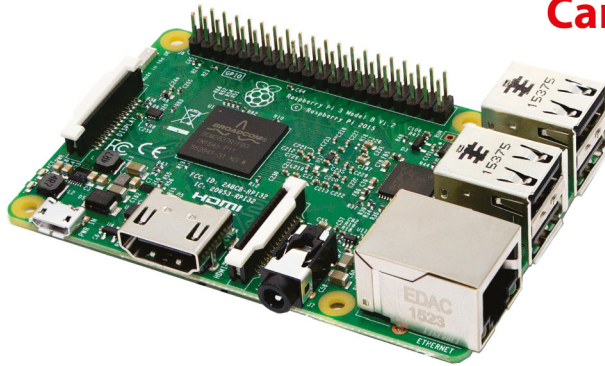
Mar.2018  
Issue #04 £6

LASER CUTTING **HOT WATER** HACKABLE DISPLAYS



## CanaKit Raspberry Pi 3 Ultimate Starter Kit

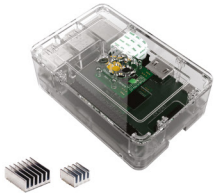
Model B | 1 GB RAM | 1.2 GHz | Quad-Core CPU



- > 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**



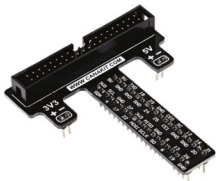
**PREMIUM HDMI CABLE**



**QUICK-START GUIDE**



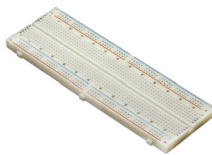
**GPIO TO BREADBOARD INTERFACE BOARD**



**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

Welcome to our wearable technology issue, where we're exploring things that combine electronics with clothing. It can be practical like our LED hat project on page 86 that helps you be seen at night, or it can be purely aesthetic like

**If you want a Circuit Playground Express, get a 12-month print subscription from [hsmag.cc/subscribe](http://hsmag.cc/subscribe) and you'll get one of your own**

the movie-inspired outfit on page 6. The choice, as they say, is yours.

If you're a 12-month print subscriber, you'll get an Adafruit Circuit Playground Express in the post shortly (if you haven't already). This little board packs a big hit with plenty of sensors and output devices (see our review on page 126 for more details). Why not use it as the heart of your next wearables project? If you want a Circuit Playground Express, get a 12-month print subscription from [hsmag.cc/subscribe](http://hsmag.cc/subscribe) and you'll get one of your own.

## BEN EVERARD

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

PAGE **52**

**SUBSCRIBE TODAY**



### GET IN TOUCH

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

### ONLINE

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



### EDITORIAL

#### Editor

Ben Everard

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

#### Features Editor

Andrew Gregory

[andrew.gregory@raspberrypi.org](mailto:andrew.gregory@raspberrypi.org)

#### Sub Editors

Jem Roberts, Nicola King

### DESIGN

#### Critical Media

[criticalmedia.co.uk](http://criticalmedia.co.uk)

#### Head of Design

Dougal Matthews

#### Designer

Lee Allen

#### Photography

Brian O'Halloran,  
Squib Photography

### CONTRIBUTORS

Gareth Halfacree, Lucy Rogers, Andrew Huang, Sophie Wong, Jenny List, Cameron Norris, Peter Kent, Mayank Sharma, John Park, Paul Freeman-Powell, John Wargo, Jenny Fletcher, Archie Roques, Jen Botezat, Bill Grainger, Ricardo Caja, Marc De Vinc, Les Pounder, Liz Clark, Daniel Hollands

### PUBLISHING

#### Publishing Director:

Russell Barnes

[russell@raspberrypi.org](mailto:russell@raspberrypi.org)

### DISTRIBUTION

Seymour Distribution Ltd

2 East Poultry Ave,  
London EC1A 9PT

[+44 \(0\)207 429 4000](tel:+442074294000)

### SUBSCRIPTIONS

Select Publisher Services

Ltd, PO Box 6337, BH1 9EH

[+44 \(0\)1202 586 848](tel:+441202586848)

Mann Enterprises Ltd,  
Unit E, Brocks Business  
Centre, CB9 8QP

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

06

## SPARK

- 06 Top Projects**  
The Prado of DIY projects
- 14 Objet 3d'art**  
Beautiful things made of hot plastic
- 16 C88 and C3232**  
A home-made FPGA-built computer
- 20 Columns**  
How to get started with hardware hacking
- 22 Meet the Maker**  
Jason Hotchkiss of Sixty-Four Pixels
- 26 Letters**  
Write to us!
- 28 Hackspace Unallocated Space**  
Makers in Severn, Maryland, USA

33

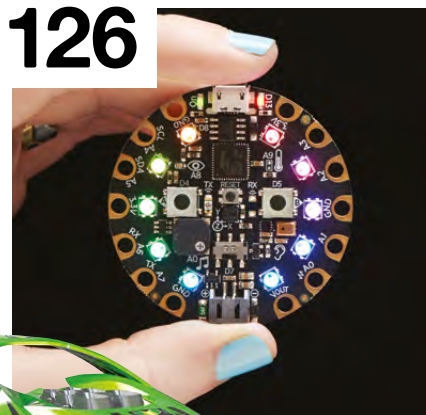
## LENS

- 34 Wearable Tech**  
Add circuits to clothing to produce unique geek garments
- 46 Pick the perfect single board computer**  
Find the best brains for your latest build
- 54 Innovation Camp**  
What came out of a retreat for hardware hackers
- 60 Interview Lucy Rogers**  
The Robot Wars judge tells all about the Guild of Makers
- 66 Improvisor's Toolbox Paper clips**  
Do more with these tiny bits of metal wire
- 70 How I Made: Trump Time to Go Clock**  
Count down to big events with the aid of Nixie tubes

34

## WEARABLE TECH

126



120





60



122



14



28



86

79

## FORGE

- 80 **SoM Workshop basics**  
Make bespoke boxes with your laser cutter
- 82 **SoM Coding For Arduino**  
Deal with data using variables and constants
- 86 **SoM Sew LEDs into a hat**  
Make your brain box more visible to traffic!
- 88 **Tutorial NFC door security**  
Keep intruders out of your hackerspace!
- 92 **Tutorial Build a hot water monitor**  
Piping hot water all the time thanks to a Pi Zero
- 98 **Tutorial Internet of Tea**  
Email your kettle to make you a cuppa. Magic!
- 102 **Tutorial Make a terrarium**  
Grow evergreens in a sealed environment
- 104 **Tutorial ESP8266 Web app**  
Monitor a solar power setup in a browser
- 108 **Tutorial Build a power supply**  
Turn an old laptop PSU into a bench power box

111

## FIELD TEST

- 112 **Best of Breed**  
Find the best display for your current project
- 120 **Can I Hack It?**  
Our latest toy: a bright green remote-controlled car
- 122 **Direct from Shenzhen Tesla Coil Speaker**  
Thunder and lightning in one cheap DIY kit
- 124 **Review LilyPad ProtoSnap Plus**  
One PCB, several gadgets for Arduino-based playfulness
- 125 **Review OKAY Synth DIY Kit**  
A retro blast of fun that you build yourself. And it's pink!
- 126 **Review Adafruit Circuit Playground Express**  
All hail a brilliantly accessible way to create intelligent wearable circuits
- 128 **Review Sonoff Basic WiFi Smart Switch**  
Control your mains electricity points over WiFi with an app
- 129 **Book Review Much Ado About Almost Nothing**  
A history of electronic geekery

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.





# Ghostbusters 2016 Proton Pack

By Sophy Wong

 @sophywong

I made this proton pack to go with my Ghostbusters 2016 costume. Because I used as little metal as possible in the build, the entire pack weighs less than 5 kg, including the proton thrower. To keep it light, I used sheet ABS, 3D printed parts, and even cardboard, wherever I could.

There is even a modified disposable ramen bowl in the centre of the synchrotron, which diffuses the LEDs perfectly.

Two Adafruit Trinkets control the LEDs, and all the numeric displays. Everything is powered by a USB power bank. For lighting in the synchrotron, I used a 60 NeoPixel ring from Adafruit. Next, I'll be adding speakers – when I press the trigger, you'll hear the crackle of an unlicensed nuclear accelerator on my back! □

**Right** □

The lights in the barrel of the proton thrower come from Adafruit NeoPixel sticks

# Rapid Whale Mini Boat

By Josh Tulberg

[Rapidwhale.com](https://rapidwhale.com)

I'm a Bay Area maker – owner and operator of a ridiculously small one-man design shop, Rapid Whale. I have a knack for designing things no one wants (or is willing to pay for), so it's fitting that I designed such a small mini boat.

The Rapid Whale Mini Boat is only 6 foot in length, which I believe makes it one of the smallest mini-boats out there (most are 8'). And as far as I know, it may be the first mini boat ever to be CNC laser-cut. Typically when you think of boat manufacturing, you think of CNC routers, or giant metal moulds for laying up fibreglass. With something as small as a mini boat, it becomes possible (and advantageous) to cut it with a laser. My laser cutter has a cutting zone of just 900mm x 1000mm (just under 3' x 2'), so that partly dictated the size of my boat.

The first mini boat I ever saw was Paul Elkin's 8-foot 'Little Miss Sally'. It was amazing, and it stuck with me for a few years. I became motivated to design and build my own mini boat once I discovered the simplicity of 'stitch and glue' boat building. I

spent countless hours designing and tweaking the boat in CAD before moving onto a scale-model, and then finally three full-scale production models.


My Dad, my good friend Dylan, and I built three of these boats at the same time in a garage. We knocked out most of the work in a weekend and took our time waterproofing and putting on finishing touches. When it came time to launch them for the first voyage, I had some idea of what to expect (knowledge from the CAD model, and the scale model) but was still largely uncertain as to how it would behave in the water. Lucky for us, the boats performed admirably. Primary stability was quite low, which is what allows us to lean into turns. Secondary stability was very good, which is what prevents us from actually flipping. That combination of stability is perfect for having a blast at low speeds, which is great because you aren't going any faster than 4mph with the electric trolling motors.

I'm looking forward to taking our boats out on many more adventures. □







**Left**  Josh's boat will carry a pilot up to 6 foot (183 cm) and 200lbs (91 kg) in weight





# Kerbal Space Program controller

By Hugo Peeters

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

I've been a fan of Kerbal Space Program for quite some time. Because what's better than building rockets and trying to launch them to distant moons and planets? Well, using physical buttons and switches to launch that same rocket of course!

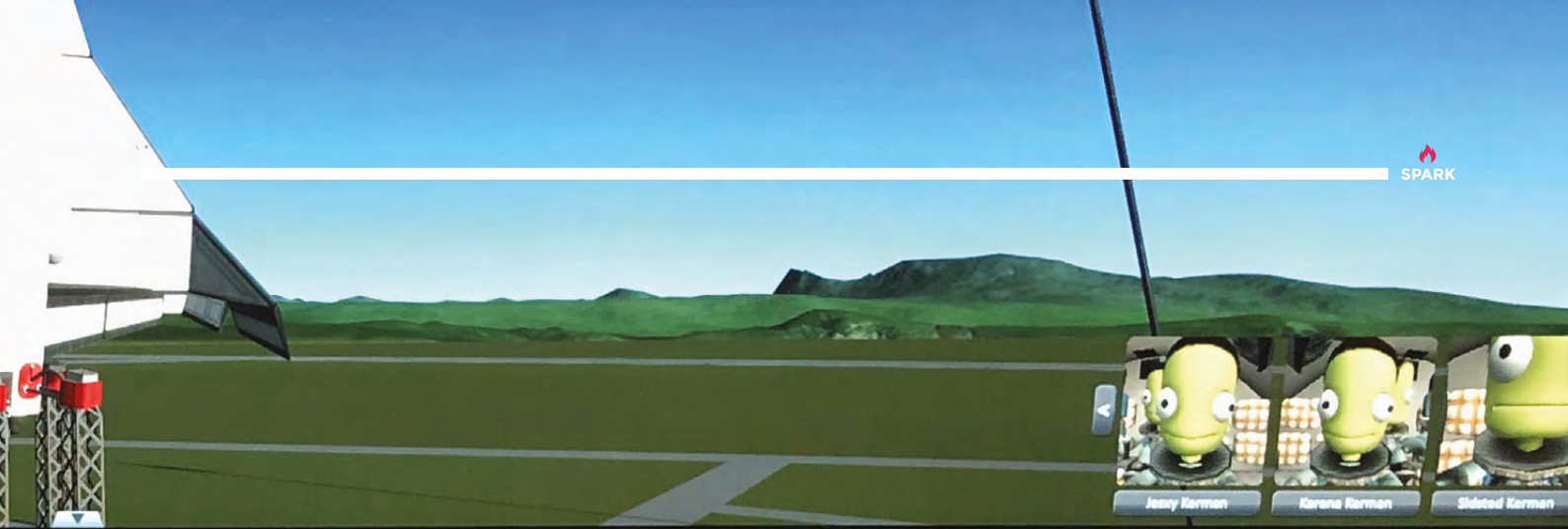
At first it seemed like an impossible task to build such a thing. But if you break it down into small pieces, and by learning from what others have shared online, it isn't that hard at all.

Come find me and other geeks at the KSP forums and the [hsmag.cc/OqfgRj](https://hsmag.cc/OqfgRj) subreddit and be inspired to build your own. □



**Right** □  
There's a thriving Kerbal add-on community making controllers for the space simulator







# NES/SNES Raspberry Pi cases

By Gurinder Kullar

[pigminted.com](http://pigminted.com)

**W**hen I first got into the Raspberry Pi, I couldn't find a case that I really liked. All cases just seemed like, well... cases, and without the Pi they felt out of place. So, I decided that I wanted to make cases that, even without the Pi, had a function.

And that function was art.

After many failed attempts, I designed my custom *Mario Bros 3* NES cartridge. I wanted to emulate the same joy of getting a video game when I was growing up with these custom cartridges. The cool part? You could actually play the game with the Raspberry Pi. I originally made one only for myself, but my co-workers and family saw them and wanted some, and now here we are. ▣







Left There's even more geek woodwork on Gurinder's Etsy shop: [hsmag.cc/eUPYxl](https://www.etsy.com/shop/Gurinder)



# Objet 3d'art

3D printed artwork to bring more beauty into your life

**If your worktop is anything like ours, you'll have loads of batteries rolling around with no order, no idea of how many you have, and no idea which ones are dead, which are almost new, and which you've already tested.**

Enter this neat battery holder from Thingiverse user Adoniram, which has holes for screw-mounting into a wall, or will sit on your desk. It's made up of three parts that are friction fit, and there are also tabs to glue them together. □

➔ [hsmag.cc/JhzkNa](http://hsmag.cc/JhzkNa)



Head to [3dhubs.com/book](http://3dhubs.com/book) to check out the #1 3D printing book on Amazon





**T**he Third Thumb, by London-based product designer Dani Clode, challenges the perception of prosthetics, reframing them as body enhancements rather than medical devices. With sophisticated design sensibilities and a focus on empowering people, Dani's assistive wearable projects are thoughtful, beautiful, and clever.

The device comprises three parts: a motor housing, sensor housing, and the thumb itself, which is printed from 85A shore (very flexible) Ninjaflex filament.

➤ [daniclodedesign.com](http://daniclodedesign.com)



Credit  
Dani Clode

# Homebrew computing with the C88 and C3232

How historical systems inspired a truly clever pair of microcomputers

By Gareth Halfacree

 @ghalfacree

**C**omputers today are designed, by and large, with one purpose in mind: to abstract away the inherent complexity of a black box filled with billions upon billions of electronic switches into something that you can point at, click on, and have your social media feeds pop up, or a game load, or your favourite tunes blare out, and so on.

The earliest computers, by contrast, were built with little to no abstraction: fans of vintage computing will easily recall devices like the MITS Altair 8800, and its multitudinous Altair-bus clones, which dominated early 'personal' computing. These boxy machines were programmed by toggling physical switches on their face plates in order to set individual bits of memory to their zero or one (on or off, true or false) states, introducing the program bit-by-bit and word-by-word until there was enough in the machine's memory to achieve something useful.

Go back still further and the desktop Altair becomes the room-filling Manchester Small Scale Experimental Machine (SSEM), also known as the Manchester Baby. Barely recognisable as a computer to anyone used to modern machines, its 32-bit by 32-word vacuum-tube design is by modern standards laughably limited – but served as inspiration for Daniel Bailey, who decided to create a modern version which would take up less room.

"I wanted to start slow, to make the simplest thing I could, and then continue from there," Daniel explains of his creations, unique desktop devices dominated by two-dimensional LED displays. "I was inspired by the Manchester Baby. I think it was the beautiful simplicity of it, that the entire state of the system is visible on the 'screen.' I just thought 'that makes things so much easier, you can see exactly what's going on.' It's a great way to introduce someone to how a computer works.

"I thought, 'I could do that, but I'm not going to go for a big 32 × 32 grid straight away,' found these little 8 × 8 LED matrix things, thought 'yeah, I could do that, a few switches, should be fairly easy."

That 8 × 8 LED matrix, typically found in hobbyist electronics kits for simple animations or scrolling text, forms the equivalent of the wall-sized 32 × 32 matrix of vacuum tubes that powered the Manchester Baby. Each individual LED corresponds to an individual bit – the smallest unit of computing, equal to a zero or a one – in the computer's memory. Where the Manchester Baby has 32 words of 32 bits each for a total of 1,024 bits of memory – a kilobit – Daniel's first machine, the C88, has just eight words of eight bits each for a total of just 64 bits.

## TAKING THE FIRST BYTE

In modern computing, where a mid-range smartphone family might start at a gigabyte of memory – eight billion bits – a computer with 64 bits is a true marvel, and doubly so for the fact it can achieve functional results. "I made this as a hobby project, mostly just to prove to myself that I knew how to build a computer," Daniel explains. "It's the simplest possible computer that I could come up with that can do something, well... perhaps not useful, but meaningful."

The heart of the C88 is a field-programmable gate array (FPGA), a chip whose function is not set in concrete like its mass-produced equivalent the application-specific integrated circuit (ASIC), but with internals that can be connected together like logical building blocks. Using these, an FPGA can be told to act as almost any other type of chip – including the central processing unit (CPU) needed to drive the C88. Programming an FPGA isn't like programming a normal computer: the chip itself is modified using a hardware description language, and when you're finished there's still the software to write. →





Above  
Even with only eight bytes of memory, the C88 can control external devices



## FEATURE



**Above** Programs are toggled into memory by hand, using tactile switches

Having designed a processor for the C88 complete with its own very simple instruction set – an ambitious project for someone who had never worked with FPGAs before – Daniel was faced with using that processor to achieve something. With 64 bits of memory total that’s a challenge, but one to which Daniel proved the equal: programs available for the C88 include a simple calculator, animations which display on the LED matrix, a dice rolling system, and even programs for interacting with external hardware

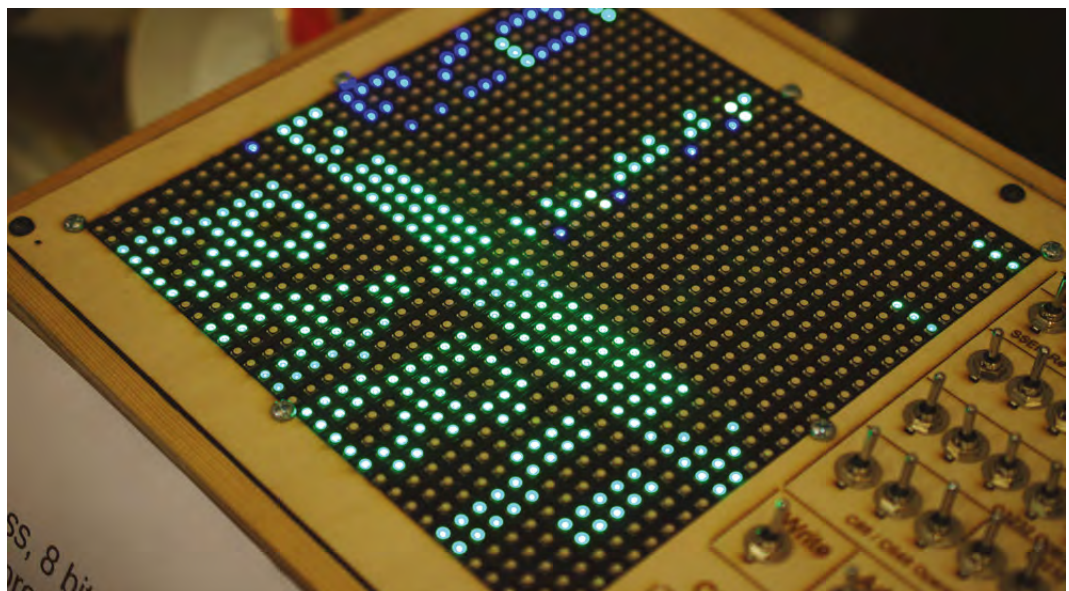
**Like the C88, the C3232 is programmed by hand using toggle switches to alter memory locations one bit at a time**

connected to a general-purpose input-output (GPIO) port. Each of these must be entered into the computer by hand, one at a time, by toggling wonderfully tactile metal switches to alter the contents of individual memory locations.

“Then I decided that’s not big enough, I’m going to do a bigger one, and that’s why I made the 32-bit one,” Daniel laughs, referring to the C88’s bigger brother the C3232 and its hefty laser-cut wooden housing. “It was pretty much just scaling up what I had, but I wanted to try some different things. I thought, if I’ve got a 32 × 32 grid I can definitely emulate the Manchester Baby, but I wanted my own CPU in there, and I thought while I’m at it I can have an emulation of the C88 in there, so the 32-bit one can emulate the C88. I thought I might as well, I’ve got space, chuck ‘em all in!”

Like the C88, the C3232 is programmed by hand using toggle switches to alter memory locations one bit at a time. Unlike the C88, the C3232’s impressive LED matrix shines in rainbow colours – a great crowd puller at the events Daniel tours as a member of the York Hackspace – while it runs a wider range of programs, up to and including anything written for the original Manchester Baby.

“An FPGA is the ultimately flexible piece of hardware,” Daniel explains. “Unfortunately, at the moment, it is a lot harder to get into FPGAs, but I certainly think it’s something hobbyists should be able to do. It means you can do things like this, and do it properly, and you end up with something, usually, that works more efficiently than a microcontroller or other CPU.”



**Right** The colourful display of the C3232 is a large LED matrix



There's another way to build your own CPU, though, and with FPGAs ticked off his to-do list Daniel has created a third machine: the Mini C88. Inspired by a clone of the C88 built onto an Espruino microcontroller for his younger brother to take to his school's show-and-tell event, the Mini C88 swaps the relatively expensive FPGA which powers the original design for an Arduino Zero.

Typically, the way to run software written for one computer on a different computer is to use emulation, a software trick which requires significantly more processing power than the original design and which is often less than perfect. For Daniel, "less than perfect" isn't something worth aiming for, so he took a different approach: Dynamic Binary Translation (DBT), a clever technique for running software from one architecture on hardware using another, to which he was introduced while working at Arm.

### GETTING RIGHT TO THE CORE

"Instead of simulating every state change that a real C88 would encounter as it executes a program, we translate – convert – the program from C88 machine code and execute it directly on the Arduino's Arm Cortex-M0+ processor," he explains of the technical trickery required to build the Mini C88. "This means we can make better use of the hardware. Instead of storing the current C88 register value in memory somewhere, like an emulator would, we just use the Arm R0 register to store the register value. We don't need to store the program counter anywhere either, since the Arm architecture has its own program counter to keep track of where it is in the program.

"We are, in essence, hijacking the hardware built into the M0+ for fetching, decoding, and executing a program by giving it a program in the format that it expects but which happens to be equivalent to the C88 program we want to run." A neat trick, but one with its own particular challenges – including handling C88 instructions for which the Arm instruction set has no equivalent and introducing the ability to alter the speed of execution so you can actually see the program being run step-by-step.

The result is a machine which is as close to the original C88 as possible, yet costs less to build. Built into a smaller, laser-cut acrylic housing – the original C88 being built into the largest project box available at Daniel's nearest high-street hobbyist electronics outlet, having "bought the biggest project box that I could find and thought 'right, I'll make it fit in there,' then just drilled a few holes and thought 'that actually looks pretty good!'" – the major features are still there: the 8 x 8 matrix display is present and correct, as are the




**Above**  The Mini C88 is compatible with its bigger brother, but smaller and cheaper to make

toggle switches which allow the user to program the Mini C88 and switch it between program entry, display, and runtime modes. Only the GPIO port is missing, a sacrifice of the shift to the Arduino microcontroller.

For Daniel, the C88 began as a simple experiment in building a simple CPU on an FPGA. Its concepts, though, could help bridge the gap between modern, abstract, high-level computing and its increasingly opaque low-level underpinnings. While toggling a program in by hand is tiresome work – especially on the C3232, which requires 1,056 individual switch activations for a program which fills its memory – it makes the way in which any binary computer of any age operates at the very lowest levels painfully obvious, in a way simply reading about transistors can never do.

"Not everyone will be interested in the low-level stuff," Daniel admits. "There are plenty of people who just want a Raspberry Pi, but for all the people who want to get right down to the lowest level, as I did when I started learning programming and things, this sort of thing would be great, I think. I wish I'd had something like this when I started learning all this stuff!"

Daniel has considered turning the Mini C88 into a built-it-yourself kit, but at present the easiest way to play with his creation is to either find a Maker Faire or similar event to which the C88 and C3232 are being exhibited or play with the simulated version at [hsmag.cc/QXPCjQ](http://hsmag.cc/QXPCjQ).

Those interested in its inner workings, meanwhile, can find the VHDL code which turns an FPGA into the C88 processor on [hsmag.cc/oTpVBy](http://hsmag.cc/oTpVBy). 

**Below**  "I was inspired by the Manchester Baby," explains creator Daniel Bailey



# Finding ideas

Great projects start as great ideas, where do yours come from?



Lucy Rogers

🐦 @DrLucyRogers

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

## W

here do ideas come from?

My most fun projects have come from other people:

"Can you hack my robot dinosaurs?" "Can you make the end of my concertina fall off?" "Can you make seven inch high dressmakers mannequins?" And so on...

Engineers are not exactly renowned for their social skills. I am not keen on social situations, personally. However, I make a special effort to talk to people who work in completely different areas to ones I have any experience of.

My intro line goes something like:

"Hello, I am Lucy. I make fun, one-off things that solve problems." And then I listen.

I had not been to any kind of theme park for over twenty years when I was asked by the owner of one to hack his robot dinosaurs.

My musical skills stalled, aged ten, when I gave up playing the recorder, but a musician in a comedy band wanted help with a prop. I own a total of two frocks, and a dressmaker asked for help to market her bespoke products.

I would never have guessed that these were problems anyone had. And yet I find there are quirky problems everywhere. Problems that people have been searching for a solution to for years. The sort of problems I love.

**I had not been to a theme park for over 20 years when I was asked by the owner of one to hack his robot dinosaurs**

Ideas bounce off ideas. I look at what others are making. But I also listen to what stories are being told, what is happening in the news, what people are talking about:

"I want my shoes to massage my feet when I have walked a long way" resulted in a sentiment analysis massage pillow.

"I used an e-cig to make smoke for a laser gun" resulted in desktop fire crackers.

Some ideas are happy coincidences – the fartometer I made started out as a carbon dioxide sensor. Until... the cat farted near it.

If I am having trouble finding inspiration, I use tools such as

**inventotron-3000.com**.

I am looking forward to making the time to construct a bicycle that squirts a laminar flow water jet, infused with rainbow LEDs from the front basket to the back pannier.

To help overcome creative blocks, I find 'Oblique Strategies', by Brian Eno and Peter Schmidt, useful. These offer challenging constraints and encourage lateral thinking.

Some projects are worth doing. Others not so much. Tools such as **catwigs.org** can help you decide if your project is as useful as antibiotics, as a garage door remote, or indeed, as a cat wig.

Where's your next idea going to come from? □



# Getting started in hardware hacking

You will break stuff, and that's a good thing



**Bunnie Huang**

[@bunniestudios](https://twitter.com/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 the most difficult aspects of getting started with hardware hacking is overcoming the fear of breaking things. Unlike software, you can't

simply roll back to the last commit that worked; if you let the magic smoke out, it's game over. However, this doesn't mean you have to fly blind or get it perfect the first time. With the right methodology and a bit of planning, there are ample opportunities to practice technique and do comparisons against known good versions.

My first tip is to go dumpster diving. You can learn a lot with little fear of loss

if you're working on stuff that's been acquired for almost nothing. Well-funded organisations tend to throw away gear they think is defective, even if it's for a minor issue. There are also swapfests, Craigslist, and eBay – some of my friends have built impressive hardware labs on a shoestring budget by trawling eBay for great deals and fixer-uppers.

The second tip is when you're preparing to take a deep dive into a single product, try to acquire three units: one to totally tear down and trash; one to work on; and one to keep pristine, so you have a golden reference to check

against. With a little luck, you only have to buy one of these at full price. The tear-down unit can literally come from a trash heap – it doesn't need to be functional. So long as the PCB traces are intact, it serves its purpose. In addition to selectively removing components to assist with tracing out wires, I use tear-down units to dry-run risky soldering procedures. Once the correct temperature, airflow and/or

soldering tip has been determined, I stand a much better chance of first-time success on the 'work' unit. The pristine unit can be borrowed from a friend – after all, the point is to keep it factory-new, so you have a reference

point to sanity-check against. This leaves you with having to purchase just one device – the target you are working on. Of course, time is money. If you're impatient or don't like planning ahead, then perhaps your only recourse could be to buy all the samples at market price.

The final tip is to take apart anything that's destined for the trash heap. Practice makes perfect, and stuff intended for the trash heap is great for perfecting the skill of opening things up with minimal damage, from removing bezels to desoldering RF shields. □

**One of the most difficult aspects of getting started with hardware hacking is overcoming the fear of breaking things**

# Meet The Maker: Jason Hotchkiss

Building your own music

**E**lectronics kits are a staple of the maker community – both for creators and consumers. They're the most accessible route into creating custom electronics and help us share our creations without getting too bogged down in manufacturing. HackSpace magazine sat down with Jason Hotchkiss, the man behind Sixty-Four Pixels, to find out what it's like making electronics kits for musicians. This is the wisdom he had to relate...

## GETTING STARTED

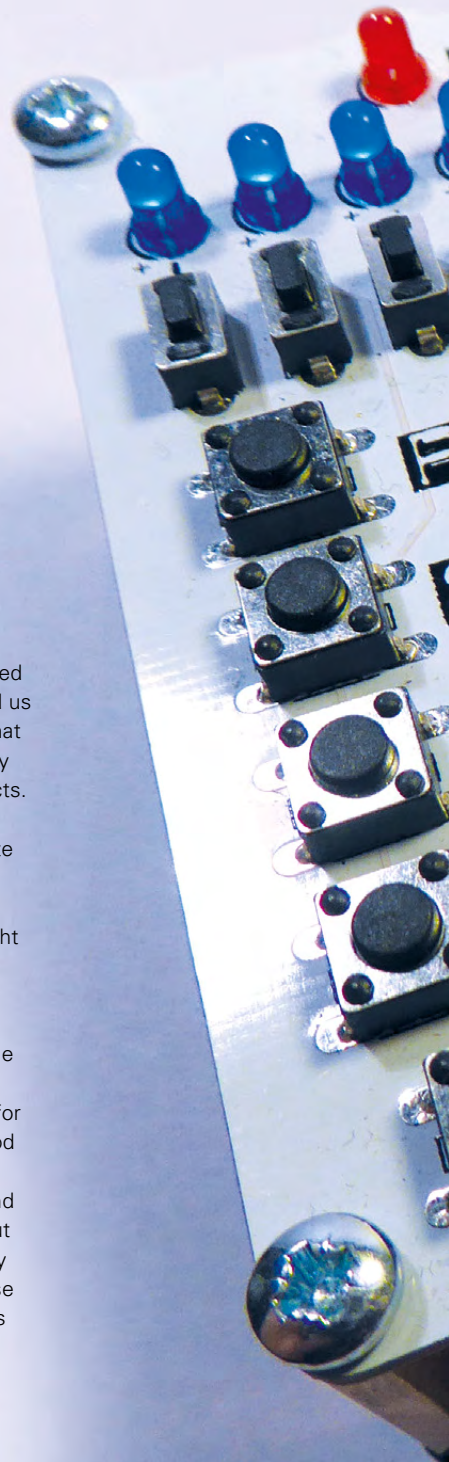
I used to do aimless tinkering with stuff just for fun. I did a bit of MIDI, I did a lot of things with the Novation Launchpad (a grid controller). I was putting the videos up on YouTube and some of them were getting quite a few views and people were asking if I was selling the things I was making, and some of them were... no! One of the things was a clock made out of hard disk drives where the digits were etched through ([hsmag.cc/TJChgv](http://hsmag.cc/TJChgv)). With careful timing and everything, you could make it light up. I wasn't going to sell it because it took me about a week to make, but some of the things like the strum controller and the arpeggiator – which were two of the first things I did – I thought 'yeah, why not?'. I had to go through some things, like how to get PCBs made up in a factory. I was doing stuff on stripboards or etching my own PCBs at home, so I'd got into Eagle, which is a PCB layout program, but I had to go through a bit of a learning process just to get started to know how to get PCBs made. Even just etching your own PCBs – when you first start

out and you think 'what do I even Google?' – you start from nothing. I started going through that process and making my own boards and I had quite a lot of interest on YouTube and I found out how to get boards made up. I got in with Tindie very early on and I stuck with them because they've always been good to work with. Especially at the beginning, it was a nice fairly close-knit thing. Emile Petrone, the guy who started it, used to have Google Hangouts every week and us Tindie sellers would all get together and have a chat – it was a really small thing then. I guess now they have hundreds of sellers and thousands of products. Certainly at the beginning it was like a little club.

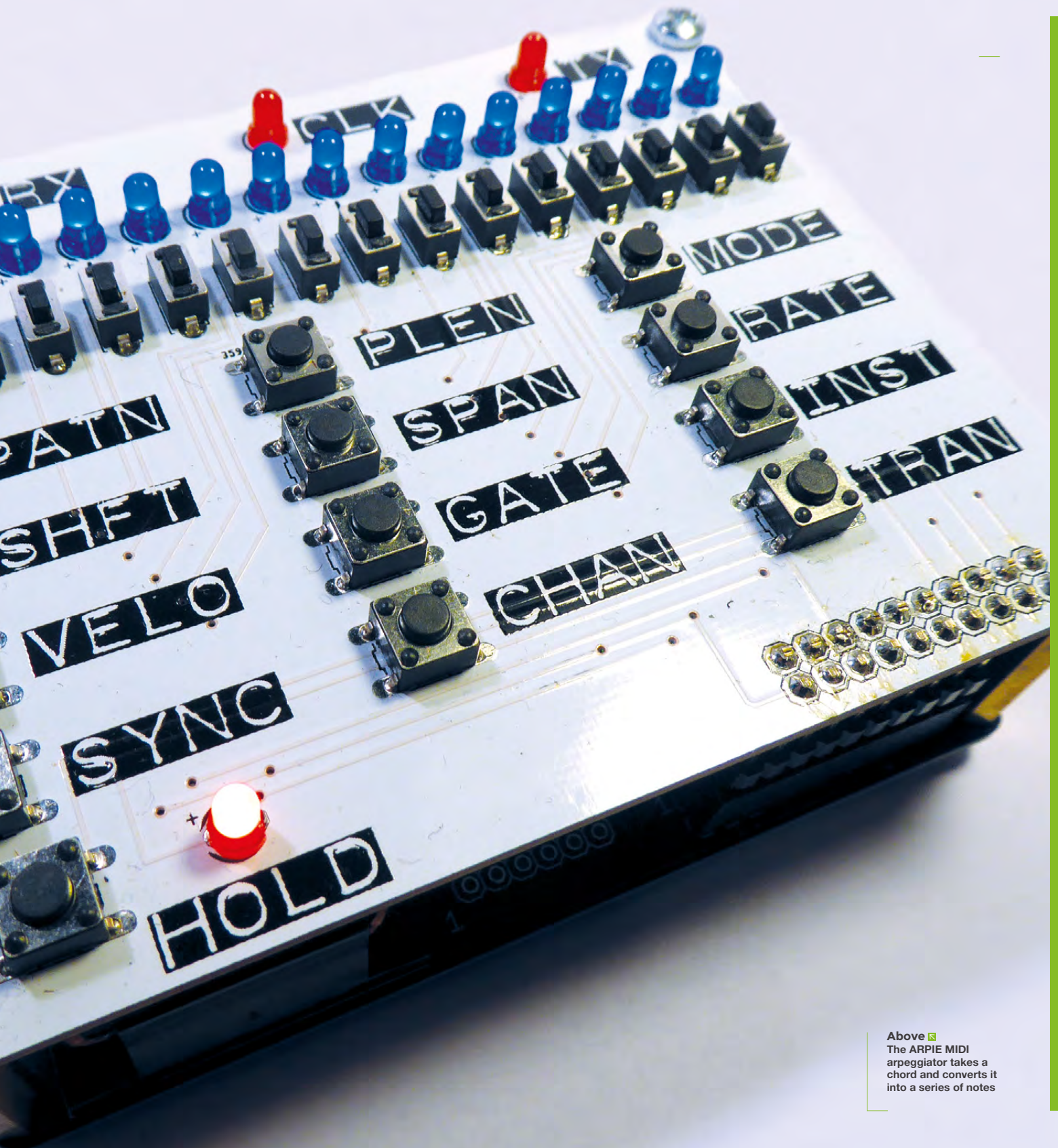
Things have just gone from there. It's been quite slow growth ... Having said that, a couple of the products, when I first put them out, sold quite quickly – it's surprising how many they sold straight away and I was almost not ready for the volume. We're talking like 50 or 100, which for me is still quite a lot.


I started that way around – I didn't start from the aspect of wanting to sell things, more I was just tinkering and I was just making my own projects for fun and sticking them on YouTube. It's always good to have a bit of attention, isn't it? – people giving you feedback. And people wanted to buy them and that's where it started. I never really thought about it being a business, but now I'm trying to work my way out of the day job and do it full-time – because I enjoy it more than wanting to be a millionaire. As long as it can pay the mortgage, I'll be happy.

I started with electronics projects about five years before that, about 2008. I'm a computer →







Above  The ARPIE MIDI arpeggiator takes a chord and converts it into a series of notes

programmer by profession – I knew a lot about coding, so getting in an Arduino and code was never a problem, but the electronics I never really knew much about. I think the digital side of it was kind of easy because it's really just an extension of programming, so working with shift registers, multiplexers... they're just an extension of code. I've been trying to understand a bit more analogue stuff, which is a bit of a black art, but it keeps life interesting trying to do that.

### MAKING MUSIC

I do love the fact that MIDI is so well crafted. To do so much so efficiently and yet be so simple and it's stood the test of time. It's, what, nearly 40 years old, and it's still going strong.

There have been all these other things that have come out and have been supposed to replace it but it's still the de facto standard.

I like making music and always have done, so I'm a bit of a synthesizer junky and play guitar and bass in bands, so music's always been my main passion really. It's not like I got into music because I thought I'd sell lots of kits; more that I started doing music kits because that's what I learned first – my first Arduino projects were MIDI.

But I think the reason that some of the things have sold well is: the feedback I get is that the functionality is good – I think that's because I come up with lots of things because I am (or try to be) a musician. I have ideas that seem to translate quite well into features on things like that. I think that's why that's eventually become the focus – it's something that I


can think of good functions and features for. I've used lots of similar products myself and I think 'oh, wouldn't it be good if it had this feature or that? I'll make one!'

### FACING CHALLENGES


The biggest problem for me is always time. Everything takes longer, by probably a factor of ten, than you think it's going to. Especially because, at the moment, I'm working around the day job. I only work three days a week, but it still gets in the way quite a lot. Apart from that, there haven't really been a lot of big problems. I think that because I've not tried to do things too quickly, the money side of things has never really been an issue. I can invest the money I'm making off things into new parts or whatever. The outlay is never very high. At most, you're talking a few hundred pounds to start something.

Space is getting a bit limited because as it's grown, certainly my wife thinks that I'm starting to fill the house up. I'm thinking of getting just a small business space because I've started to employ a couple of friends as well, with some of the work for manufacturing.

Everything takes up a bit more space – finding space for someone to work, finding space for somewhere to keep things.

**Below**  Hot-swap up to six MIDI outputs with near zero latency on the Banana Split



**Right**  The first step to selling is getting noticed, and YouTube is a great place to start showing off your makes



### ADVICE FOR BUDDING SELLERS

First of all, just go for it! Don't think it's going to be really hard. Things are easier to find now than they were five years ago in terms of printed circuit board making. You've got very good things like OSH Park that are quite accessible now. When I was doing it, you had to find someone in China and deal with them directly. There were a couple of sites that made it a bit easier, but quite often it really was a little bit difficult.

I guess my point is that all these things that look quite complicated at the beginning, you get past them all. There's so much stuff online – especially now – there are so many forums on that stuff, so the first thing is don't be scared that it's going to be really hard. The second thing is to have an idea that you're really passionate about. You can't start and think 'right, I want to make a product, what shall I do?' Well, I guess you can – that's what companies do all the time – but I think that doing it on your own, it's really important to have the idea for the product before you have the idea to start a business – at least from my side. Maybe other people can do it the other way around, but for me you have to have the passion for a product – and it helps to have one product, one idea to begin with – and start small, then build on it as you go. Be patient: things do take a while to get going.

I've always done things open-source. My family think I'm mad and a lot of people just don't get it – I never used to. I was always wondering why wouldn't you just patent everything – what if somebody nicks your idea? I really benefited when I was learning from other people's open-source stuff, particularly the Arduino project – it really got me started with embedded coding and the fact that all the people very freely give code examples online and stuff, so I think just keep that spirit of sharing going. I guess you do take the risk that someone's going to take your idea and just run with it, but for me, I've not really had that happen. I've seen projects come along that were very similar to what I've done, but I've never regretted sticking everything up online. I do actually use a non-commercial open-source licence ... I'm not quite prepared to let go. [But] keep things open-source; I think it generates goodwill. More often than not, somebody starting out is going to be benefiting from other people's goodwill to get started. □



**Top** □  
Meet the maker, Jason Hotchkiss of Sixty-Four Pixels

**Middle** □  
Le Strum allows you to play your synthesizer as though it's a guitar, by strumming the stylus across the notes

**Above** ◆  
Connect your MIDI signals to synth control voltage with CV.OCD

# Letters

## EARTH SONG

Thanks for the recycling focus of last issue. There's something about the disposable nature of a lot of making, 3D printing in particular, that makes me feel uncomfortable. Maybe it was seeing David Attenborough's *Blue Planet 2* and watching all the plastic being thrown in the oceans, but I don't want to contribute to that by printing extra plastic objects when I can use scrap wood or recycled metal. There are plenty of projects to keep me busy so I don't need to clog up the world with more 3D printing. I hope to see more environmental subjects in HackSpace mag in the future.

**David**

Wharfedale

**Ben says** We know how you feel, when you're throwing away your third failed print at the weekend it can feel like you're wasting plastic, but in reality 3D printing is going to save a lot more waste than it creates. Think of all those appliances that get thrown away because one fitting has snapped – just one little bit of printing could save a whole load of waste materials, not to mention the energy that goes into manufacturing and shipping the thing.



## MAKE THAT CHANGE

Your Humanitarian Makers piece in issue 3 showed what making is all about: the DIY ethos that means you can see a problem and fix it for yourself without waiting for someone else to come along and tell you what to do. There's something about the independence of it that I really love. It's not about saving money on parts or whatever, but about the feeling of accomplishment you get from taking ownership of something that's broken and turning it into something that works. Humans are amazing – the more tools we put in the hands of humans to fix their own problems, the better off we'll all be.

**Helen**

South Shields



## ATTENTION ALL MAKERS!

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



## DEAD TREES

Is it just me who pays for a print subscription and then ends up reading it on my iPad? The printed magazines look too good to mess up by turning the pages, and by the time the magazines have arrived round here, I'm already halfway through the PDF version. I do the same thing with vinyl too – buy the album, then download the FLAC and listen to that.

**Gary**

London



**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.

# CROWDFUNDING NOW

## OpenDSKY

Recreation of Apollo guidance computer

From \$350 | [kickstarter.com](http://kickstarter.com) | Delivery: June 2018

**T**he Apollo series of missions put a man on the moon almost fifty years ago. This was two years before Intel made the first microprocessor, when computers were cumbersome things. The Apollo Guidance Computer was the only computing power on the rocket, and astronauts interacted with it via the Display&Keyboard (DSKY) interface.

The Open DSKY is a reimplemention of the original system on the Arduino platform and packaged in a replica of the DSKY which is available in 3D printed plastic, steel, or (like the original) aluminium. The metal ones don't come cheap though, with the aluminium option selling for \$8000. As well as the original functions, the Open DSKY comes with the ability to play sounds – something which may come in handy for users who don't happen to need the landing calculations for a lunar module.

The project has already reached its target, but they are only making 100 of the limited edition computers, so you'll have to be fast if you want to own one (they also have non-limited edition kits available). □



## Mycroft Mark II

Privacy-focused home assistant

From \$99 | [kickstarter.com](http://kickstarter.com) | Delivery: December 2018

**V**oice-controlled AI home assistants are everyday now, but most of the popular commercial options are run by companies that are interested in collecting all the data they can about you to power their advertising and sales machines. Mycroft, however, is an open source, privacy-orientated assistant that you can be confident isn't amassing your personal data for some shadowy company's commercial gain.

Mycroft is powered by skills, with each skill relating to a particular 'thing' that it can do. For example, read a Wikipedia page or set a countdown timer. Many of these skills have been developed by the community of developers contributing to the open source project. It's worth taking a look at the available skills before committing to the project to make sure it supports the things you want to work with. If you can't wait until December, download the current version of the software for a Raspberry Pi or Linux device. □



# Hackspace of the month: Unallocated Space



## Unallocated Space

➔ [unallocatedspace.org](http://unallocatedspace.org)

🐦 [Unallocated](#)

✉ [info@unallocatedspace.org](mailto:info@unallocatedspace.org)



**Corey Koval wrote to let us know about Unallocated Space, a makerspace in Severn, Maryland, currently celebrating seven years of making.**

### WHEN DID YOU START?

Unallocated Space started in the summer of 2010 with two bored hackers driving home from work looking for something to do in their free time. The thought “let’s start a hackerspace” got floated around and the wheels were put into motion. From there they reached out to the community to find other interested parties and quickly built up the group that became the founding key holders of the space. The group was able to raise funds and move into a building later that year where the space still resides. Over the past seven years the space has grown and formed a thriving community with a wide variety of expertise and interests.

### WHO IS IT FOR?

One of the unique things about UAS is that we’re free to the public. We provide classes, tools, and equipment to the wider community. Members of course get perks, but really UAS is a labour of love by people who enjoy learning and helping others to learn. As for our membership base, we’ve got a diverse group, but the area we’re in is the most cybersecurity-focused region in the US. So, naturally we have a lot of cybersecurity, electronics, and RF people. But that’s of course not exclusive and really we accept anyone who wants to be a part of the space. There is something magical about watching a senior FPGA engineer work on stained glass with our crafts instructor, or showing a grandmother how to program Arduino because she wants to fabricate her own cat toys. We’re a social group of wildly different people, each of us bringing our own speciality.

### WHAT EQUIPMENT DO YOU HAVE?

The usual suspects: servers, networking equipment, bins of Bluetooth / RF hardware, three 3D printers, power tools, a plotter, as well as radio equipment.

We’ve got a decent amount of VR hardware as well. Most recently, we just picked up an entire pallet of lasers. We also have shelves full of electronic components, computer parts, and a lot of other odd stuff. Oh, and we have a tornado made of fire. (Complete with licensed pyro-technicians, of course.)

### HOW DO YOU SHARE YOUR IDEAS?

Our most active communications platforms are Slack and Google Groups. But we also frequent (and participate in) the security conference scene. We always have a good showing at DEFCON, BSidesDC, BSidesCharm, and ShmooCon. We also like partnering with other hackerspaces in the region like our neighbours to the south, HacDC, or our friends up north at the Baltimore Node.

Additionally, we use other social platforms such as Facebook, Twitter, and Meetup.com to let people know what all events are happening at the space. We also do a lot of outreach into the community by participating in local STEM events and information security conferences.

### WHAT TRAINING DO YOU DO?

Unallocated Space offers a wide variety of classes to the public. Some of our continuously running classes include Information Security, Arduino and 3D Printing. We have also hosted many other classes in the past including Software Defined Radio, Web Development, Electronics Fundamentals, Linux, and a very wide variety of others. All instructors teach on a volunteer basis, so all classes depend on instructor availability.

To find out more about our classes, or to host one yourself, check out [hsmag.cc/hiTvUn](http://hsmag.cc/hiTvUn).

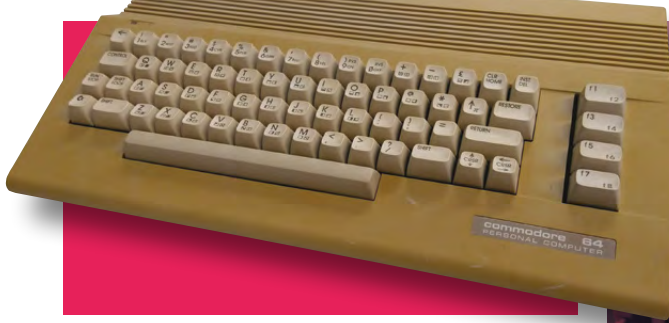
### WHAT HAVE YOU MADE RECENTLY?

Our members are always busy working on projects, whether it’s something for themselves or for the space. Some of our projects are hard to capture in a picture, such as our complex network infrastructure or many of our members’ software projects, but there are plenty of hardware projects to check out in the pictures on the opposite page. ➔





**Above** ♦  
 As well as the very neatly arranged set of spanners, Unallocated Space has a fire extinguisher, in case anyone forgets rule 0 (do not be on fire)



**Above** ♦  
 Nuclear goop or glow-in-the-dark printing filament?





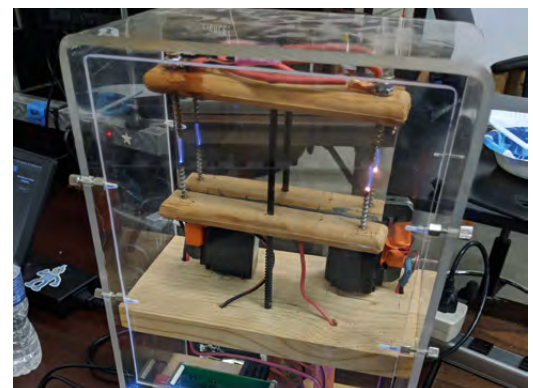


**Infinity Clock (Max) ♦**

The clock uses a GPS receiver to provide accurate time with a display based on edge-lit layered panels. The effect of the display is similar to an infinity mirror until one realises the colours at each level are different. The project is a combination of hi-tech (Arduino Mega 2560 with a custom PCB for the audio and GPS) as well as lo-tech (handmade wood case with stained glass panels and 3D printed parts).

**Fire Tornado (Multiple Members) ♦**

The fire tornado is often the main attraction at our anniversary parties. The elaborate setup of box fans aimed at a bowl in the centre of the circle create a tall, beautiful, flaming vortex. It has been featured on the Science Channel and in other publications.



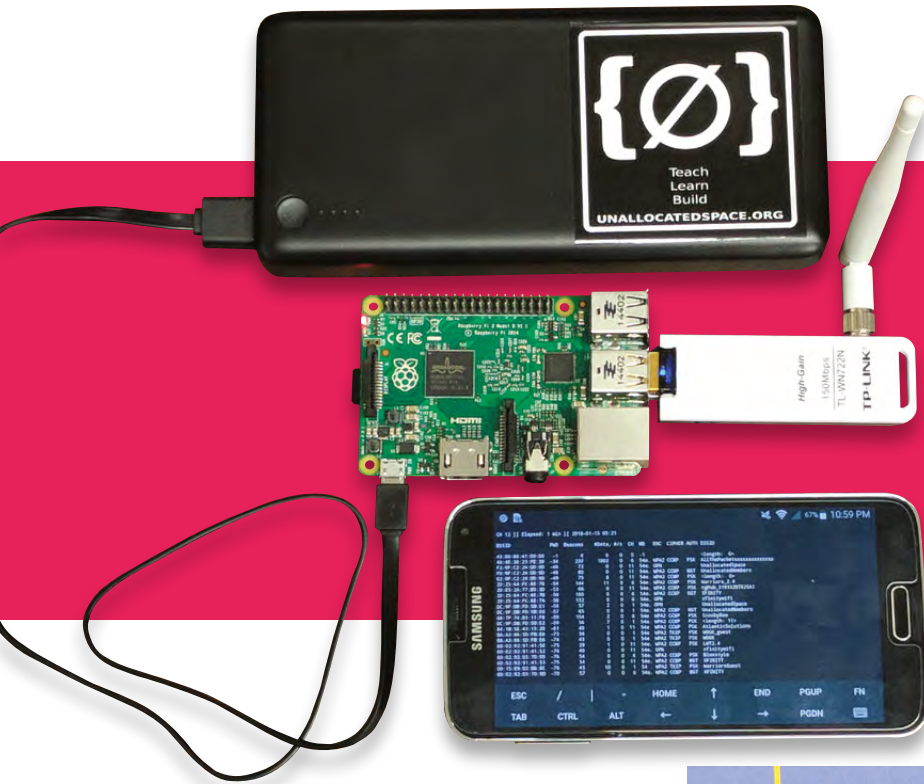
**Plasma Speaker (Corey, Bowie, and Roger) ♦**

The plasma speaker is a device that uses electrical arcs to produce sound by heating the air. It follows us to many events and draws quite a bit of attention. The design of the plasma speaker has been continuously improved since it was introduced to the space in 2013.

**HOW TO SUPPORT US?**

As a non-profit organization, UAS relies on the generosity of our community through donations and accepts many other forms of contributions. You can learn more about how to support our hackerspace at [hsmag.cc/edgBoL](http://hsmag.cc/edgBoL).





## 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](mailto:hackspace@raspberrypi.org) with an outline of what makes your hackerspace special and we'll take it from there.

## Black Widow and X-Men Belt Buckles

3D printed models, created for cosplay representations of movie/comic costuming accessories.

### Hackberry Pi (Usako)

A headless Raspberry Pi running Kali Linux, designed to be used for mobile wireless attacks and fox hunting (direction finding).

### Lightsabers (Bowie)

According to legend, Jedi make their lightsabers on board an ancient starship in deep space, guided by eccentric droids. Our set up is kind of like that. It's in space: Unallocated Space.





# Build a Makerspace for Young People

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

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



# LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG  
46



## PICK THE PERFECT MICROCONTROLLER

Choose the brains to power  
your next world-beater

PG  
54

## INNOVATION CAMP

One French chateau + many  
great minds = problems solved

PG  
60



## INTERVIEW LUCY ROGERS

Robots, avoiding hard maths,  
the spirit of sharing, and the  
Guild of Makers

PG  
66

## PAPER CLIPS

Turn everyday office detritus  
into a superior siege weapon  
(and other uses)

PG  
70

## HOW I MADE TRUMP CLOCK

Glowing Nixie tubes to  
measure the term of US  
President number 45

PG  
34

## WEARABLE TECH

Pick the right circuits, materials,  
and power source for you – then go  
out and make something brilliant



# The **WEARABLES** ISSUE

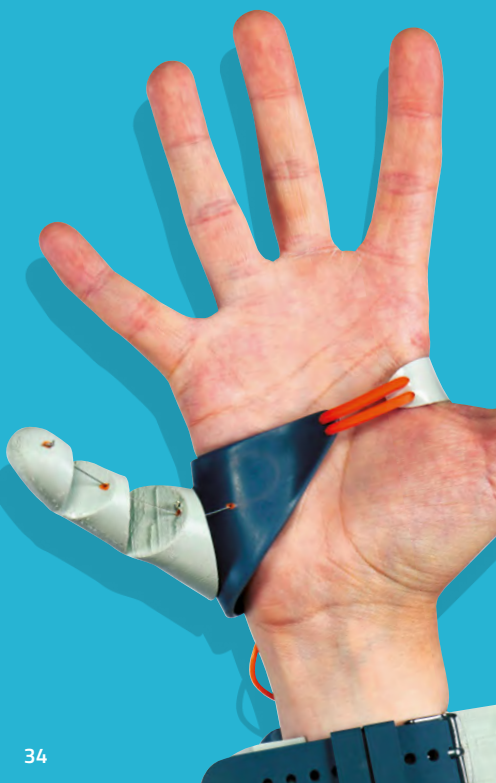
## CONTENTS

→ **06**  
TOP PROJECTS

→ **16**  
OBJET 3D'ART

→ **36**  
COVER FEATURE

→ **86**  
LED HAT



Wearable Tech

FEATURE

# WEARABLE TECH

DIVE INTO THE WORLD OF WEARABLES

**W**hat do you think of when you hear the words “wearable electronics”? Maybe you think of Snapchat’s Spectacles, the pop-coloured sunglasses with a built-in camera. If you’re a cosplayer, you might think of a glowing, chest-mounted ‘arc reactor’ à la Tony Stark. Wearables can be fitness trackers, virtual reality headsets, spacesuits, cosplay, and more. If it needs a battery and you can wear it, it’s a piece of wearable electronics!

As we speed toward ever tinier technology, it is now possible to build the kind of wearable devices that we used to only dream about in comic books. With access to components like sewable microcontrollers and flexible LED strips, tinkering with wearables has never been easier. A little foundational knowledge will set you up for success with your first projects, so let’s dive into the concepts, tools, and best practices for wearables. →





**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](http://sophywong.com), chronicling her adventures in making.






### LED HAT

The best way to learn any subject is to start small. Turn to page 86 for a quick and easy introduction to sewing circuits.



Above  This cute and cuddly Jawa costume uses a

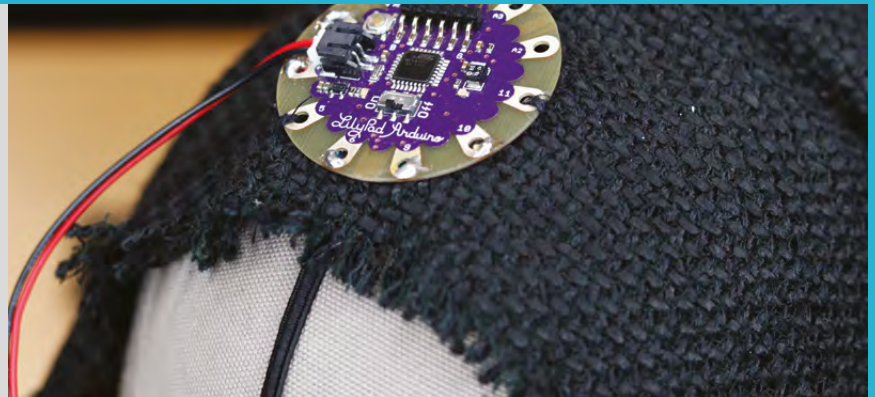


# BUILDING FOR THE HUMAN BODY


**Y**our idea might be as complex as a holographic computer on your head, or as simple as an LED sewn into a wristband. But because they all go on the human body, all wearables face a shared set of challenges. Designers often solve these challenges in similar ways, and you may notice underlying similarities between wearables that seem very different at first glance.

One of the biggest challenges for wearable designers is how to apply the rigid, flat materials of electronic components to the fleshy, round surfaces of the human body. Wearable designs generally avoid sharp edges and pointy corners, for the comfort and safety of the wearer. You can see this idea at work in both the rounded corners of Microsoft's HoloLens and the circular form factor of the LilyPad Arduino. Rounded corners also protect soft materials like fabric from snagging during movement. Keep this in mind when selecting materials and components for your wearable projects.

Wearables are often circular or tubular in form. This is because the human body is basically made up of cylindrical shapes – you can think of your arms, legs,



torso, and even your fingers, as cylinders. Fitness trackers, twinkle LED skirts, and VR headsets all loop around the body in a circle, tube, or arc. In this sense, the human body is a perfect fit for electronic circuits, which are also loops. The challenge is how to make an electrical loop that can open and close, stretch, or bend around the human body. Think about this when looking at consumer wearable gadgets – how do they solve this challenge? →

**Above**  This Jawa costume uses a LilyPad Arduino to power the glowing eyes.

## FIVE TIPS FOR EVERY WEARABLE PROJECT

**There's a lot to consider when designing your wearable project. Whatever form your project takes, these five tips will help you make something you'll enjoy wearing when you've finished building it:**

- Don't forget the on/off switch. If at all possible, include an on/off switch in your design, and make it accessible during wear. Resist the temptation to rely on removing and inserting batteries for power control. Ideally, you can switch your project off quicker than someone else can say, "Ouch, those lights are bright!"
- Put the battery where you can reach it. Like the power switch, keep your battery accessible. Often your battery will be the heaviest, bulkiest part of your project, and you'll want to hide it, but don't bury it too deeply. If you're sporting your wearable for more than a few hours, chances are you'll need to change the battery at some point. Bonus points if you can change it in public!
- Figure out how to clean it. Protect your hard work by ensuring that your project is cleanable in a practical manner. Some wearable components are actually hand washable, but wash carefully and follow manufacturer's instructions. Another option is to make your electronic circuit removable using hook and loop fasteners or snaps, and launder the garment as you normally would. You can also wear a 'laundry layer', something that can be easily washed, underneath your project. For rigid accessories like headsets and helmets, spot clean the areas that touch skin with alcohol-free baby wipes.
- Solder big projects. Sewn circuits are a great way to get started with wearable electronics, but be prepared for your imagination to outgrow the limitations of conductive thread. The resistance of conductive thread makes it unsuitable for long circuits, and complexity is easier to manage with insulated wire in different colours.
- Think about the wearer. When working with wearables, keep the wearer in mind. Make comfort your goal, and don't build anything that puts their safety at risk. Keep circuits and batteries away from skin, and don't impede vision or movement. Your wearable will look much better if the person wearing it is smiling!

# BUILDING YOUR CIRCUIT

**W**earable circuits don't necessarily have to be complex. Keep your project streamlined and use the simplest circuit

**you can.** A simple circuit, implemented well, can have a big impact. Most tutorials for wearables will use one of these three common techniques. Learn them all, and you'll be able to choose the best option for your own wearable designs.

## PREMADE CIRCUITS

A quick and easy way to get started is to use a premade circuit. Instead of building the circuit from scratch, you can skip ahead to figuring out how to put it into your garment. Fairy lights are great premade circuits, as most come with a simple battery pack and a built-in power switch. Another type of premade circuit is an all-in-one board like Adafruit's Circuit Playground Express, a microcontroller with built-in lights and sensors.

When using premade circuits, your main challenge will be how to embed them into your project. For soft garments, get creative with snaps or hook and loop fasteners so that the electronics can be removed for washing. If your sewing machine has a buttonhole function, adding buttonholes to a garment is an easy way to make passthroughs for wires. For rigid accessories like helmets and headsets, use hot glue or strong adhesive tape to attach your circuit.

## SEWN CIRCUITS

Sewn circuits, also known as e-textiles, are also a great place to start. Sewn circuits use conductive thread for electrical connections, no soldering necessary! To build your circuit, you simply sew from one component to the next. There are many types of sewable components available, including microcontrollers, sensors, LEDs, and more. Sewable components have large, open holes for making electrical connections with conductive thread, and some are even washable.

You may already have the hand sewing tools you need for sewn circuits: large-eye needles, scissors, and a thimble. You'll also need conductive thread

and sewable electronic components for your circuit. Conductive thread is thicker than regular thread, and a dab of Fray Check or superglue will help keep knots tied.


Because conductive thread is not insulated, it can be difficult to avoid short circuits, and you will need to insulate your sewn circuit when it's complete. Cover the thread lines with puff paint or strips of fusible interfacing. This usually works well on the inside of a soft project, but think about the outside too – a long run of conductive thread can fold back on itself and cause a short circuit. Because of the resistance factor of conductive thread, insulated wire is a better choice for long circuits.

Despite the challenges of building large projects with sewn circuits, great complexity can be achieved with clever engineering and imagination. Masters of sewn circuits embrace the constraints of conductive thread and turn circuits themselves into works of art. However, if sewn circuits are your entry point to electronics, it's likely that the limits of conductive thread will drive you to learn to solder. We love that idea!

## SOLDERED CIRCUITS

If you're new to it, learning how to solder may seem daunting, but it's fun and easy once you learn the basics. A quick lesson from a makerspace or an experienced friend can get you started. Then it's a matter of practice, and investing in the tools: a soldering iron, metal solder, wire strippers, and wire for your project. Silicone-coated stranded wire is great for wearables; it's soft, flexible, and can be

Choosing a battery for your project is a balance between size and power

**Above**  There are lots of power options for choose from



incredibly thin. You can still use sewable components designed for conductive thread, just solder your wires into the holes. Soldering opens the door to the wider world of electronics, and you'll be able to make use of components not specifically designed for wearable applications.

Even if you already know how to solder, there are a few things to keep in mind when soldering for wearables. Soldered joints aren't flexible, and will break if flexed repeatedly during wear. A dab of hot glue can reinforce a soldered connection and make it last longer. Fabrics stretch, but wires don't – make sure there is enough length in your wires to allow for movement. And while your wires may be insulated, you'll still need to keep soldered connections and components away from moisture and skin. Coat them with a clear acrylic spray or clear nail polish when your build is complete.

## BATTERIES

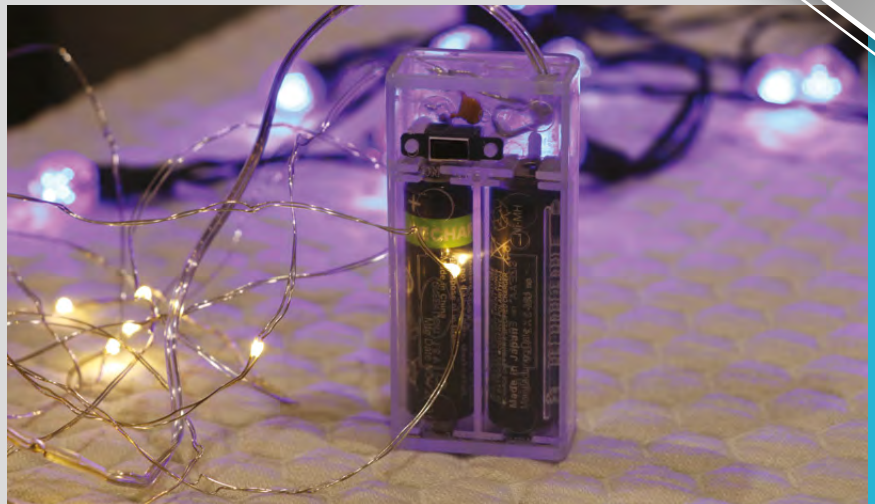
Choosing a battery for your project is a balance between size and power. For wearables, you'll want the smallest battery you can get away with, given the power requirements of your project. To estimate your power needs, check the datasheets for each component in your circuit. Find the maximum current draw for each, and add them together. It's better to provide more amps than not enough, so choose a battery that meets or exceeds this total. If you're just starting out, stick with the batteries recommended in tutorials and example projects. As you build more and become familiar with components, you'll get a better sense of what batteries to choose.

## COIN CELL BATTERIES

These are your tiniest option for battery power. Coin cells are great for low-power projects like powering a few LEDs, and maybe even a tiny microcontroller. Battery holders often come with a built-in on/off switch, and come in sewable form. Coin cell batteries are low capacity and don't last long, but they're small enough to carry a spare set.

## ALKALINE AND NI-MH RECHARGEABLE BATTERIES

Like coin cells, these are easy to find and you already know how to use them. But unlike coin cells, rechargeable versions are available, which is great for the environment and your wallet. Three or four AAs should work for most wearable microcontroller projects. On the down side, AA and AAA batteries are heavy in multiples, and hiding a bulky battery pack is tricky in a small project.



## LIPO BATTERIES

Because they are small, rechargeable, and powerful enough to drive complex wearable projects, LiPo batteries are the battery of choice for many wearables. However, they are also more delicate than other types of batteries and require careful handling for safe use. Never puncture, compress, or expose them to heat. Don't place them directly against skin, and don't store them in your project. If your project is made of fabric, remove the LiPo battery before recharging it.

## USB POWER BANKS

For projects that need a hell of a lot of juice, a USB power bank is a great choice. They are high-capacity, safer, and more durable than LiPo batteries, and most come with some kind of on/off switch. Often, the power bank's specifications are printed right on the case, check them against the power needs of your project. USB power banks come in many shapes and sizes, but tend to be heavy. >

**Above** ♦ Wearables projects don't have to start with specialised kit

**Below** ♣ Conductive thread blends invisibly with fabric but is difficult to work into complex circuits



# MICROCONTROLLERS FOR WEARABLES

For wearable tutorials, coding lessons, and more, check out:

[learn.adafruit.com](http://learn.adafruit.com)

[learn.sparkfun.com](http://learn.sparkfun.com)

[wearabletutorials.com](http://wearabletutorials.com)

[kobakant.at/DIY](http://kobakant.at/DIY)

[instructables.com/howto/wearables](http://instructables.com/howto/wearables)

This is the board that ignited the DIY wearable movement when it was released in 2007

**A** rduino and other microcontrollers are basically tiny computers that let you add programmability to your project. With a microcontroller, you can incorporate light animations, sensors, motors, sounds, and more,

all driven by code. Wearable microcontrollers are often circular in form, and can be both sewn and soldered to. All the boards listed below are washable.

There are many wearable microcontrollers to choose from, and scores of wearable components that you can connect to. Let's take a look at some great microcontrollers specifically designed for wearables:

## 1 LILYPAD ARDUINO

LilyPad is the original sewable microcontroller, invented by Leah Buechley and produced by SparkFun. This is the board that ignited the DIY wearable movement when it was released in 2007. Today, there are other boards based on Buechley's iconic design, and even the LilyPad itself comes in several different flavours. The LilyPad line also includes sewable LEDs, sensors, buttons, switches, battery holders, and more.

## 2 ADAFRUIT FLORA

Inspired by the original LilyPad Arduino, the FLORA wearable platform by Adafruit is powerful, easy to use, and supported by a massive library of tutorials created by Adafruit and its community. Work through a few Adafruit tutorials, and you'll be designing your own wearable projects in no time. The FLORA line includes sewable versions of powerful components like GPS, a Bluetooth module, and NeoPixels, Adafruit's highly addictive individually-addressable LEDs.

## 3 ADAFRUIT GEMMA

GEMMA is a 1 inch diameter version of FLORA that's perfect for smaller projects that only require a few inputs and outputs. It's great for beginners, and can

feel less intimidating than a bigger board with lots of bells and whistles. Despite its small size, GEMMA still has convenient features like a built-in on/off switch, a JST battery connector, and micro USB for programming.

## 4 TINYLILY MINI

The TinyLily Mini by TinyCircuits is about the size of a thumbnail. It's unbelievably tiny! To achieve this tiny size, the board lacks some conveniences built into bigger boards, and requires a separate USB converter for programming. But with the same processor as a full-sized LilyPad Arduino, it's still powerful enough to drive some truly awesome wearable projects.

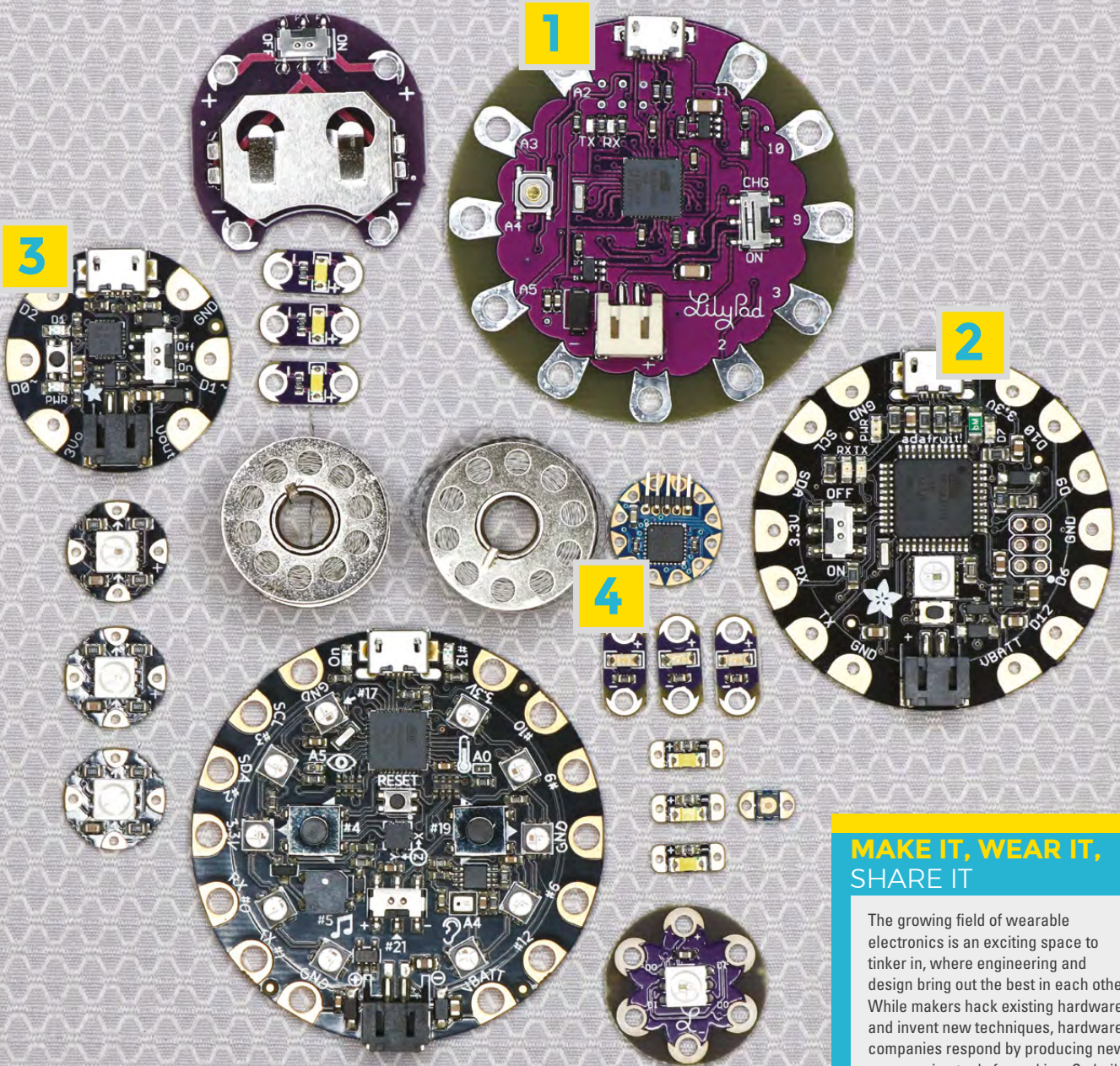
## ALSO CONSIDER: STITCHKIT

StitchKit is a new wearable microcontroller designed specifically for fashion tech and wearables by MakeFashion. Based on their experience running fashion shows, the StitchKit is designed to combine durability with ease of use for fashion designers who are new to working with hardware. The StitchKit Kickstarter campaign has been fully funded, and MakeFashion plans to start shipping boards in April of this very year. →

## CODING HELP

If you're new to code, fear not – tutorials and example projects abound on the internet and it takes minutes to get up and running with example sketches in the Arduino software. Start with simple code provided in tutorials, and you'll learn to modify it to suit your needs and do more. There are also visual programming aides that make coding simpler: try MakeCode by Microsoft, or Xod. When you're ready to write your own code, check out CircuitPython, a derivative of the programming language MicroPython, released and supported by Adafruit. CircuitPython aims to make it easy for complete beginners to write code for their microcontrollers.





### MAKE IT, WEAR IT, SHARE IT

The growing field of wearable electronics is an exciting space to tinker in, where engineering and design bring out the best in each other. While makers hack existing hardware and invent new techniques, hardware companies respond by producing new, empowering tools for making. So build your project, enjoy wearing it, and then share it with the rest of the world. You'll spark ideas for other makers, and help drive innovation in the world of wearables!



# GET INSPIRED

You've seen the basics: now get inspired with our pick of the best wearables projects around

Alina Granville's Torbjorn cosplay is a masterpiece of wearable electronics

## ALINA GRANVILLE

After more than 1 500 hours of work, Alina Granville's Torbjorn cosplay is a masterpiece of wearable electronics. Every piece is 3D printed (boots included!) and houses individually addressable LEDs driven by 5V Arduino Pro Minis. To make it, Alina designed and built her own 3D printer, learned to airbrush, and tackled accelerometers for the first time.

 @spoon\_makes

Credit  
John Jiao





## LEAH BUECHLEY

Leah Buechley's ground-breaking work includes more than just inventing the seminal LilyPad Arduino and founding the High-Low Tech Group at the MIT Media Lab. As a maker, she explores the intersection of engineering, art, and design. She currently runs a design firm and is experimenting with generative, code-based forms in fabric and wearables.

[leahbuechley.com](http://leahbuechley.com)

Credit  
Leah Buechle



## MAKEFASHION

MakeFashion is a Calgary-based initiative that seeks to merge fashion with cutting-edge electronics. Through hands-on workshops and international runway shows, MakeFashion brings designers and engineers together to create high-tech fashion.

 [makefashion.ca](http://makefashion.ca)

**MakeFashion**  
brings designers  
and engineers  
together to  
create high-tech  
fashion

Now that you're informed and inspired, you're ready to build amazing wearables! What will you make? Whether it's your first project or your 100th, we want to see it! Show us your builds via email at [hackspace@raspberrypi.org](mailto:hackspace@raspberrypi.org)

**Credit**  
Pretty Flowers by  
Maria Orduz Pinto  
for MakeFashion


**Photo**  
Kelly Hofer





## KOBAKANT

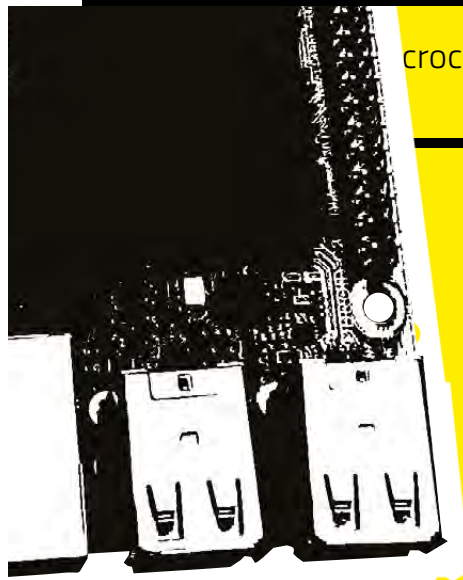
Hannah Perner-Wilson and Mika Satomi have been pioneering e-textiles together since 2006. Their projects include a crying dress, and a beautiful collection of sensor-laden conductive textiles. They've now launched KOBAKANT, a true tailoring shop for e-textiles in Berlin. For one year, they'll build bespoke wearables and share every build as an open source tutorial online.

 [kobakant.at](http://kobakant.at)

Credit  
KOBAKANT



microcontroller



# PICKING THE Perfect MICROCONTROLLER



Jenny List

[@Jenny\\_Alto](#)

Jenny is the creator of the [@LanguageSpy](#) electronics kits for Raspberry Pi and ham radio. She's also a key member of [Oxford Hackspace](#).

Some less familiar boards can provide the power you need for your next project

**T**here's a whole world of microcontroller boards available to hobbyists, yet the hobbyist scene is dominated by just a few. Join us now as we delve deeply into the world of microcontrollers to find some lesser-known gems you may not have noticed.

Before we start, what exactly are these single-board computers, or SBCs, that we're looking at? A very simplistic answer might be that it is a computer which combines processor, peripherals, and storage on a single PCB, and can be run without additional hardware. However, for most of the purposes you might think of for single-board computers, it's an extremely broad definition. There are PC motherboards with on-board flash storage that fit it, for example, but they are light years away from an Arduino.

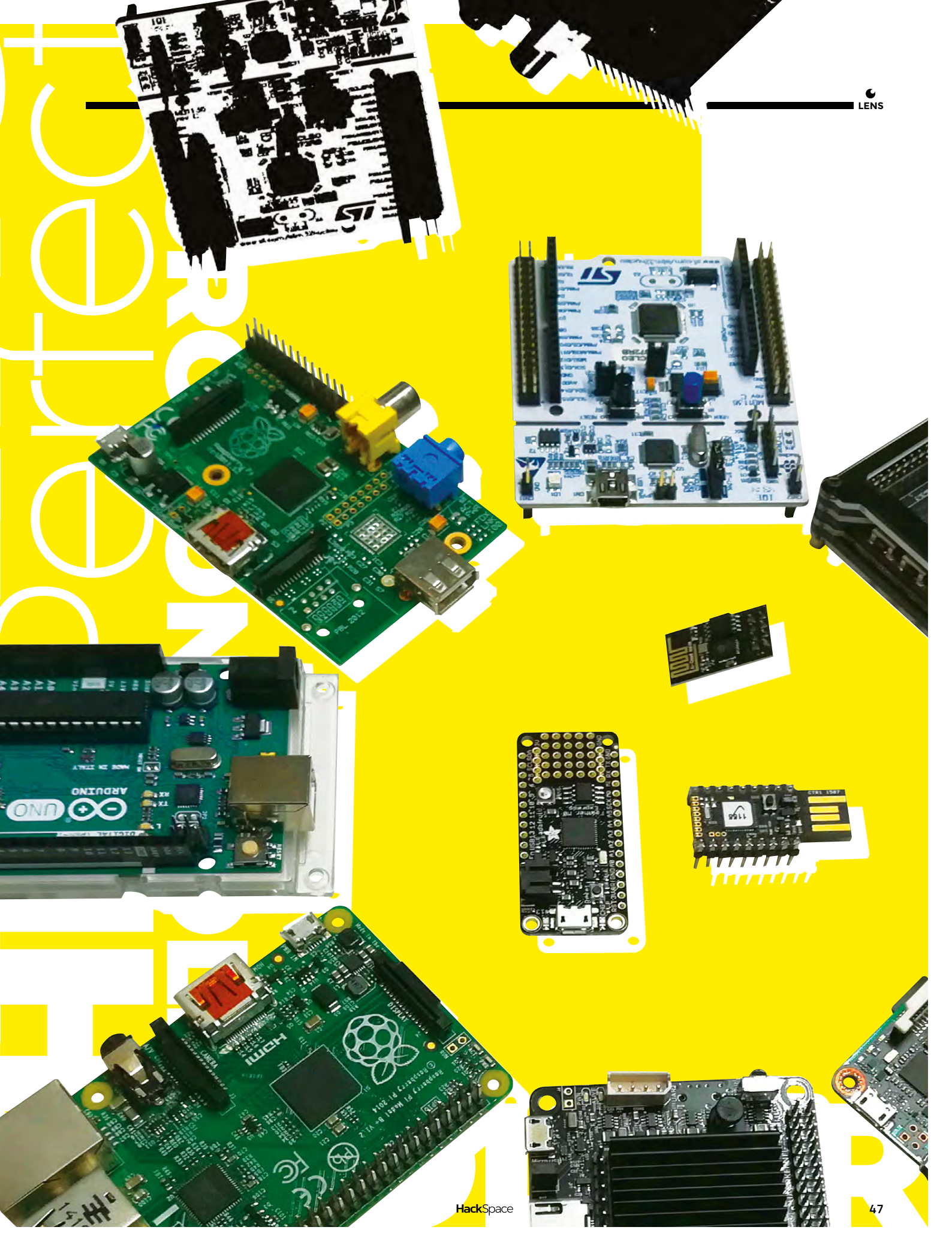
Generally when we think of an SBC, we're imagining a small and inexpensive computer board, usually one with plenty of access to input/output pins for hooking

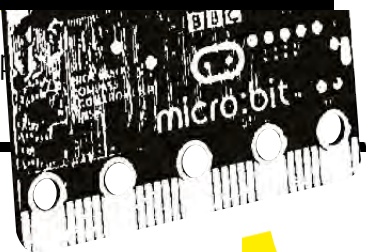
up electronics and peripherals. But even then it's a wide field, so to narrow our focus a little it makes sense to divide it up. On one side are SBCs that usually run a fully fledged operating system such as Linux or Android; on the other are ones that run bare-metal code directly on a processor, such as the Arduino Uno. Examples of the former are the Raspberry Pi, the Beagle boards, or the Arduino Yún – and while they are all amazing devices, they are a world away from the microcontroller boards competing with the Arduino Uno. Instead, the boards and platforms we're bringing you here are all microcontroller-based and run code in a much more direct fashion, rather than through an OS.

There are a variety of processor families you will commonly find in SBCs at our level, and they have achieved their prominence over their competitors either due to familiarity or because their manufacturers have gone the extra mile to ensure that the support they deliver makes their platform attractive. When you pick a board for your project, it is as well to take a →



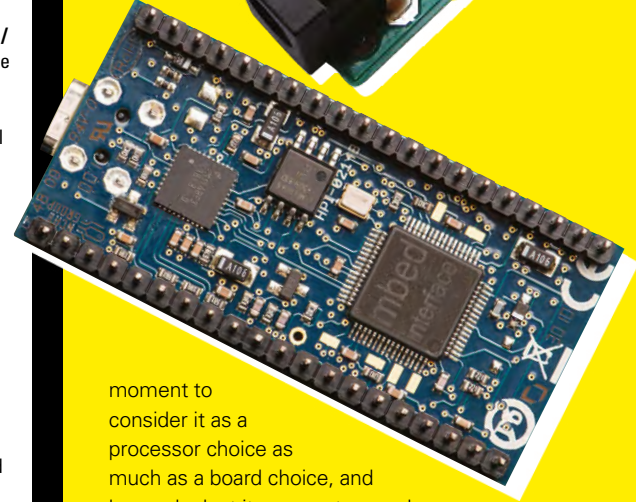






## ATMEL AVR

We haven't looked at the main Arduino products, or Atmel's AVR chips, here because we're exploring the lesser-known options (for a detailed look at the former, read our Arduino feature from issue 1, available online at [hsmag.cc/myfvuw](http://hsmag.cc/myfvuw)). While the ubiquitous Arduino Uno makes a great first board, it's useful to look beyond this comfort zone to see what other options are available. Even if you decide that the Arduino boards are right for you, by checking out the other options, you'll learn more about the board you are using. You'll also find out what the other possibilities are, and your next project might just need an extra feature that you can't get with your first choice.



moment to consider it as a processor choice as much as a board choice, and have a look at its ecosystem and the record of its manufacturer in supporting it. It's unpleasant to spend time and effort learning to use a platform only for it to suddenly disappear. Intel for example put a lot of resources into promoting its x86 microcontrollers at this level, only to pull the Galileo, Joule, and Edison products from the market and leave developers working on these platforms without any hardware to use.

So among the multiple CPU families to be found, we'll look at a few of the common ones in the hacking and making community. We'll start with Microchip's PIC series of processors, as they are probably one of the oldest families still in the game. Their origins lie in an early 1970s I/O peripheral for a 16-bit microcomputer, but they became popular at the hardware hacker level over 20 years ago, as single-chip versions with on-board flash appeared that could be easily rewritten with user

code. The PIC is an astonishingly versatile family, with everything from tiny 6-pin SOT23 to very powerful 32-bit variants, and you will find them in many of your electronic devices. They suffer from the disadvantage though that writing PIC software is not easy to get started in, and that their IDE and toolchain for developers is proprietary and quirky. While it does have some great uses for professional engineers, this difficulty in getting started means that it's not widely used by hobbyists.

### FULLY ARMED

The ARM series of processors have their origins in the 1980s, as the innovative RISC device that powered the Acorn Archimedes series of computers. Since then they have evolved to a series of related processor cores designed for everything from low-power microcontrollers to high-power general-purpose computers, and through a system of licensing can be found in products from multiple different semiconductor manufacturers. The multicore powerhouses that lie behind your tablet, smartphone, or even boards like the Beagle or the Pi, are not the ARMs that interest us here – instead it will be its Cortex M series of microcontroller cores that you will find in products at this level.

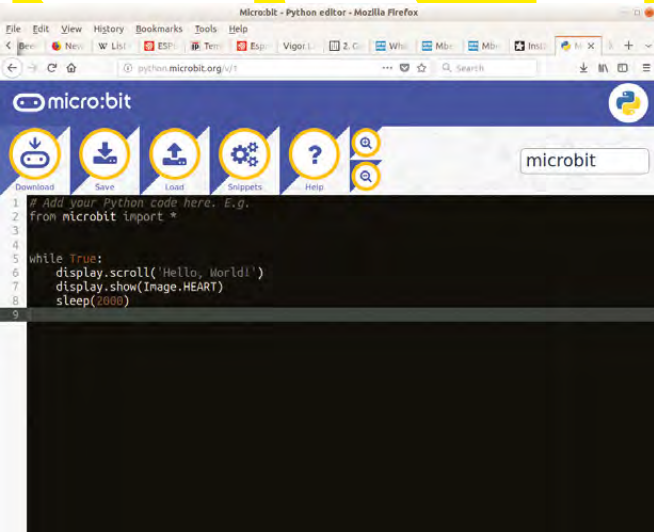
THE ARM SERIES OF PROCESSORS HAVE  
THEIR ORIGINS IN THE 1980S, AS THE  
INNOVATIVE RISC DEVICE THAT POWERED THE  
ACORN ARCHIMEDES SERIES OF COMPUTERS

**Top** The Arduino Uno is still widely used, but lacks some of the more advanced features of more powerful options

**Above** The Mbed LPC1768 provides an ARM Cortex M3 in breadboard-friendly packaging

The final processor family we'll mention here is Espressif's ESP8266. This differs from the others in that it comes from a Chinese company and is a microcontroller designed for the specific application of wirelessly connected IoT devices. It contains a 32-bit





**Above** ♦ The in-browser Python editor for the BBC micro:bit is a fully functional development environment that does not require any other software beyond a web browser on school computers

Tensilica core and full WiFi connectivity, and comes with a TCP/IP stack allowing it to be very easily connected to the internet. What brought it to the attention of our community when it arrived in 2014 was that it is extremely cheap, costing only in the region of a few dollars each in single numbers to

hobbyists. A huge effort was undertaken to create open-source tools for it, despite its only having limited documentation at the time. The ESP8266 is usually to be found on a series of standardised modules intended to be fitted by the million into Chinese-made IoT devices, and many consumer products such as ESP8266 IoT light switches have been repurposed as development systems with ready-attached power control hardware. More recently, though, these modules have appeared on more conventional development boards. It's fair to say that the world of ESP8266 devices is something more of a Wild West than the others we've mentioned, but the low price – especially when you consider their WiFi connectivity – makes them an extremely interesting option.

Having considered microprocessor architectures, it's worth taking a moment to look at how microcontroller boards are programmed. It's easy enough to say that the majority of them use a serial link while others use USB and a few of them appear in your operating system as a disk drive, but that conceals an important point →

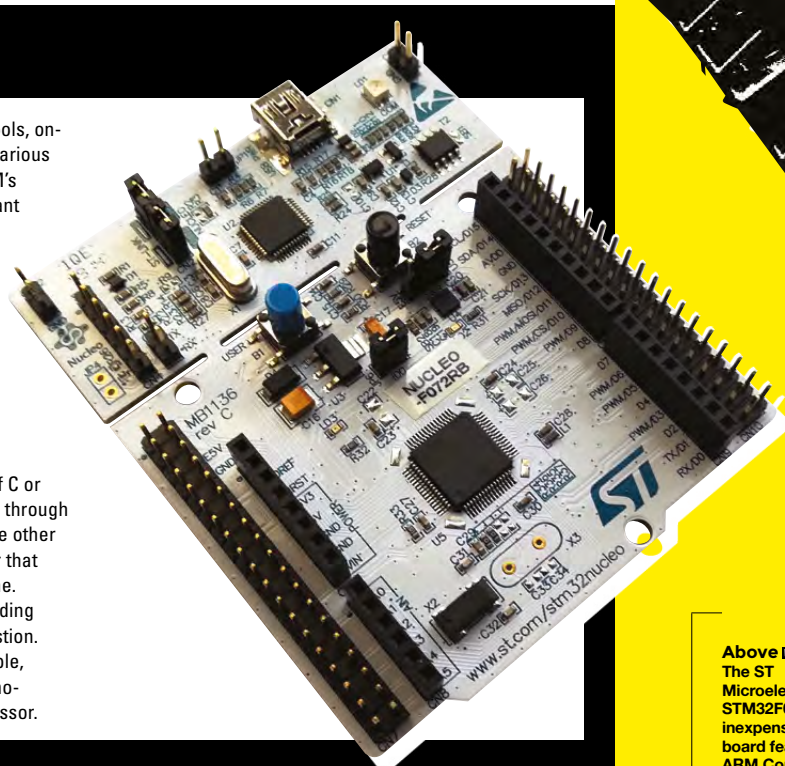


## MBED, AN ARM FOR ALL SEASONS

Mbed is a combination of software development tools, on-chip OS, and reference hardware designs for the various ARM microcontrollers. It can be thought of as ARM's answer to the Arduino ecosystem, with the important distinction that ARM itself does not produce any Mbed boards. Instead, just as it licenses the processor cores to chip manufacturers, each of those manufacturers produces its own Mbed-compatible platform for its chips. This provides Mbed with the extremely useful feature of being available on a huge variety of boards, each with its own features depending on what the manufacturer has included on the silicon alongside the ARM.

Development for Mbed is through the medium of C or C++, and there are two development routes. One is through an extremely easy-to-use web-based IDE, while the other is a more traditional command line-based compiler that allows you to use your IDE of choice if you have one.

Mbed boards are available at all budgets depending upon the capabilities of the microcontroller in question. The ST Microelectronics board pictured, for example, cost well under ten pounds and provides an Arduino-compatible shield footprint for its Cortex-M0 processor.



**Above** ▣ The ST Microelectronics STM32F072, a typical inexpensive Mbed board featuring an ARM Cortex-M0

## Picking the perfect microcontroller


### FEATURE

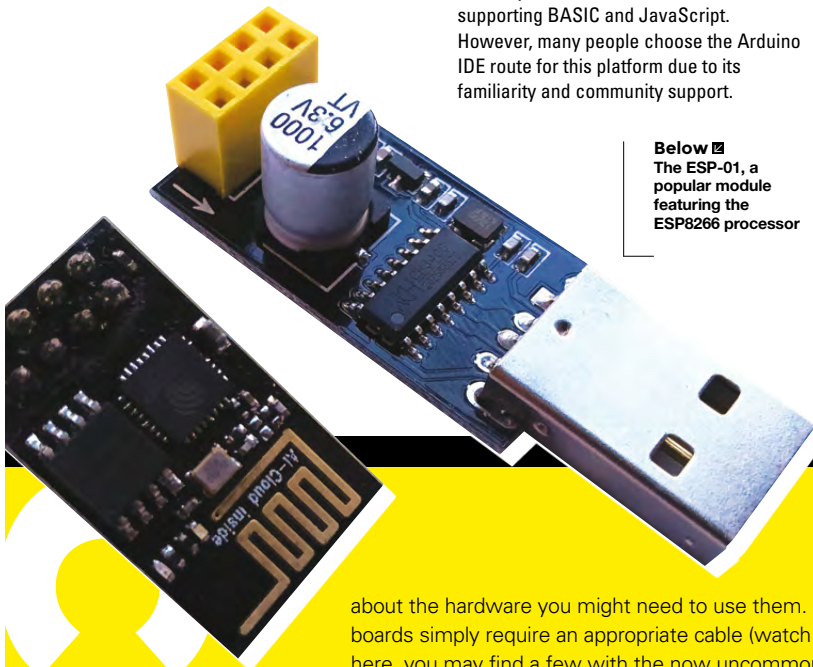
## CHEAP AND CHEERFUL: THE CAPABLE ESP

The ESP8266 made a significant splash when it arrived without warning and with very little documentation in 2014. It offers a powerful processor and WiFi connectivity for a fraction of the cost of other options, but it is available in such a wide variety of form factors that choosing the one for you can be challenging. Thankfully, help is at hand to ease the confusion, as a variety of manufacturers have integrated it for more friendly development boards with all

associated components. Adafruit's Huzzah and SparkFun's ESP8266 Thing are both very similar modules for example, designed to bring the ESP's connections out to a familiar dual in-line footprint.

On the development front, the ESP is an extremely versatile device, having a variety of language, development environment, and firmware choices for the would-be coder. As well as C++ through an Arduino IDE add-on and firmware, there is MicroPython, as well as other firmwares supporting BASIC and JavaScript. However, many people choose the Arduino IDE route for this platform due to its familiarity and community support.

Below  The ESP-01, a popular module featuring the ESP8266 processor



about the hardware you might need to use them. USB boards simply require an appropriate cable (watch out here, you may find a few with the now uncommon mini-USB), but serial boards will require some form of serial interface. This is most often a USB-to-serial board or cable that can be had for a few pounds, but in some cases it can be a proprietary interface that may carry a more significant cost. It is worth investigating this before making your choice.

So, you've got your eye on a processor and you know what hardware you'll need to program it. Now it's worth turning to their software environments. This may influence your choice of platform more than the processor architecture or the

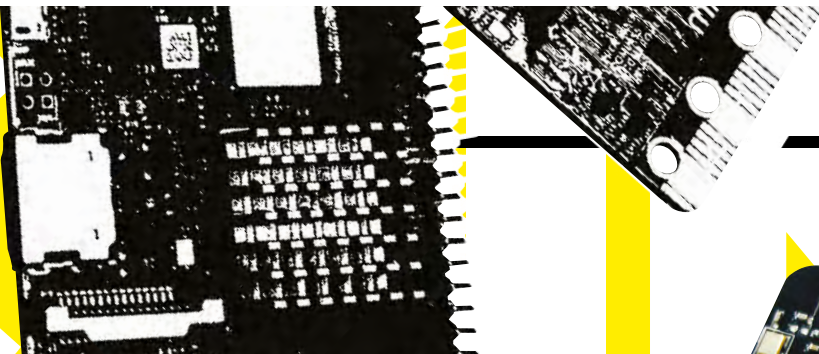
manufacturer, because you may have a familiarity with a particular language or IDE.

Traditionally a microcontroller will come with an associated software toolchain from its manufacturer, with an IDE in which you write your code, and a compiler which turns it into a binary file and sends it to your board. These are typically proprietary packages, and the language in question will often be some dialect of C or C++. Using these environments gives you the maximum access to the capabilities of the chip, but will often require a detailed knowledge of the chip architecture for tasks such as configuring the functions of each pin.

The Arduino IDE and bootloader combo is another C-based environment, but one that simplifies this by abstracting many of the difficult tasks associated with microcontroller configuration. Instead of having to consult the data sheet and write code to set a particular pin as a PWM output or an analogue input, a standard pin configuration is set up and provided through an easy-to-use software library. The Arduino IDE is now supported by a vast number of boards, including many that aren't made by Arduino. Because it is an open-source platform, there are many plug-ins to allow it to be used with other non-Arduino boards. It is often a sensible choice to use the Arduino IDE, even if you don't have prior knowledge of it, because there is a huge body of online code as well as help and advice about it, and it is a skill transferable to other platforms.

THE ARDUINO IDE IS NOW SUPPORTED BY A  
VAST NUMBER OF BOARDS, INCLUDING  
MANY THAT AREN'T MADE BY ARDUINO...  
IT IS OFTEN A SENSIBLE CHOICE TO USE IT





## NOT JUST A TOY: THE BBC MICRO:BIT

The BBC micro:bit is a single-board computer with integrated Bluetooth Low Energy, some buttons, and a simple LED matrix display. It is well known because of its creation as a teaching aid and its having been given to schoolchildren. Its low price and ready availability make it an attractive proposition.

The micro:bit is designed as a self-contained piece of hardware with an attached battery pack containing a pair of AAA cells, but it also features a selection of interface lines on an edge connector. One disadvantage, though, is that the connector used is a slightly unusual one.

Under the hood, it is an ARM board (in fact, at heart another Mbed), though we are treating it as a separate platform here because of its unique software development environment. Because it is designed to be used by children, it has a selection of web-based IDEs (from a simple block-based drag-and-drop one to a Python environment), compilation happens in the cloud, and transfer to the device is extremely simple.



**Above** ■ The BBC micro:bit is a specialised Mbed board aimed at schoolchildren

**Credit**  
Ravi Kotecha

**Below** ◆ The Circuit Playground Express packs a lot of hardware onto a small board

## EXPANDING HORIZONS

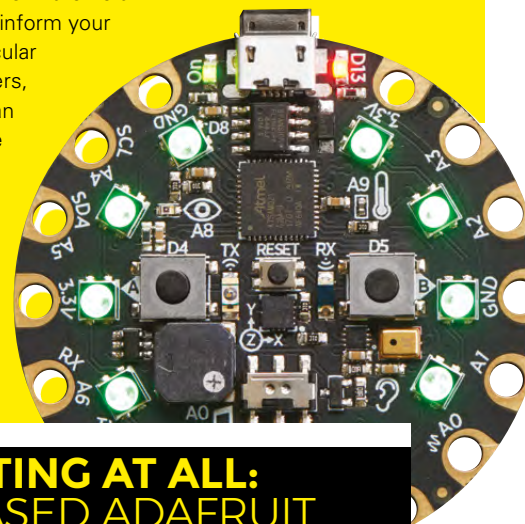
C and C++ are by no means the only languages that can be used with a microcontroller; a popular choice, for example, is Python. There is a version of Python called MicroPython (you may also encounter Adafruit's CircuitPython variant) that is tailored for microcontrollers, and particularly if you have encountered Python before on a platform such as the Raspberry Pi, it has a fairly minimal learning curve. Instead of being a compiler that places a binary file on the microcontroller in the manner of the Arduino IDE, MicroPython is an interpreter that directly runs raw Python code on the microcontroller itself. It provides access to the microcontroller features as well as the familiar Python prompt via a serial link. On devices such as the ESP8266 with a network connection, it even provides access to the prompt via a web browser.

Finally, there is a class of development environment that operates entirely within a browser. All the coding is done in a browser IDE, the compiler is situated on a web server, and the resulting binary file is presented as a download that can be uploaded to the microcontroller board. In some cases, such as the BBC micro:bit, the board appears as a USB disk drive, so this is as simple a process as copying the binary file into the drive for it to run. These environments can sometimes have fewer features than their more traditional counterparts, but time will inevitably deliver upgrades to the software, and you can't beat them for simplicity.

Whether you are emboldened to try pastures new or you are happy on familiar ground with your

Arduino Uno, it's never a bad thing to know your way round a few of the other contenders in the field. Whichever you select, you can further inform your choice by looking at how much a particular board has been adopted by other makers, and by looking at their projects to get an idea of how straightforward it might be to work with.

When people try new platforms, they do so by creating new and exciting projects. We look forward to what you will create with your new boards, and whatever they are we hope to see them within these pages. □



## NOT CONSTRICTING AT ALL: THE PYTHON-BASED ADAFRUIT CIRCUIT PLAYGROUND EXPRESS

Adafruit produces a range of boards designed to be programmed using Circuit Python – a variant of the popular Python language that's designed for microcontrollers. Development can either be through an editor such as Mu, which compiles code that can be transferred to the board over USB, or via the serial console

that allows you to run interactive code. It's a simple device to use and comes fully loaded with a range of hardware sensors. Take a look at our review on page 126.

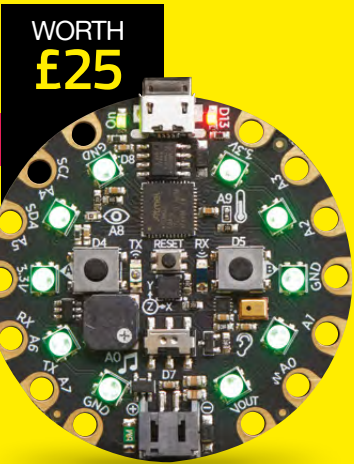
Anyone with an annual print subscription to HackSpace magazine will receive a free Circuit Playground Express in the post. See page 54 for more details. Have fun with it!



**FREE**

# CIRCUIT PLAYGROUND EXPRESS

**WITH 12-MONTH PRINT SUBSCRIPTION**



FROM JUST **£55**

**Already a 12-month print subscriber?**  
Your Circuit Playground Express is already in the post!

- 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](http://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](https://hsmag.cc/subscribe)

# Innovation Camp



Cameron Norris

@cameronsnorris

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

Ideas need a little coaxing sometimes....

**A**t Chateau Millemont, a 16th-century castle near Paris, over 100 makers, engineers, and designers, gathered to tackle 'destructive consumer culture' by making open source, sustainable products the

**new norm.** They aimed to produce a proof concept highlighting that citizen pioneers can build a fossil-free, resource-efficient society. As a result, POC21 emerged as a five-week innovator's residency that blended strategic design, prototyping, co-making, and co-living. The event was organised by Open State and Oui Share, two design collectives that joined forces to support and raise awareness of open source sustainability solutions during the 21st U.N. Climate

Change Summit, COP21 in 2015. POC21 participants developed 12 open source sustainable projects, paving the way towards a fossil-free, zero waste society.

Camp participant, Tristan Copley Smith, described his surprise when he arrived to find Chateau Millemont filled with beds, sofas, bean bags, 3D printers, and CNC-fabricated plywood tables. The castle gardens, surrounded by a vast 100-hectare forest, were covered with 30 large canvas tents, solar panels, and fairy lights to guide the way of exhausted participants at night. Even the old stables were fitted with a woodshop, metal welders, and a treasure trove of high-tech digital fabrication tools to realise almost any project.

Without assigned 'leaders' and structures of accountability, it was not uncommon for people to defy their perceived roles. For example, a photographer emptying a compost toilet or a computer hacker pitching a tent was commonplace. New arrivals often required a few days to adjust to this informal structure and governance.

Perhaps most importantly, all POC21 participants were united in the belief that open source, sustainable products have the potential to scale into the mainstream and become the new normal, as traditional barriers to building your products continue to dissolve. Dominik Wind, Open State co-founder and POC21 organiser, described how solutions to the ongoing climate and resource catastrophe must include changing the behaviour of hundreds of millions of people, and how what we consume is produced. He believes that changing behaviour is easier when communities band together to take action.

Let's take a look at three of the most successful projects that came out of POC21.

**Below**





# Faircap

Providing clean, safe, drinkable water to those in need

**A** ccording to Faircap, over 1.8 million people die from drinking contaminated water every year, while waterborne diseases infect over 4 billion people. In response to this, Faircap has developed “an open source antibacterial water filter, intended to provide clean drinking water for everyone”. This pocket-sized filter can be screwed into a plastic bottle, enabling the safe consumption of water from almost any source. As a home filter, a single Faircap can purify enough water to provide for a family of four.

## CLEAN WATER

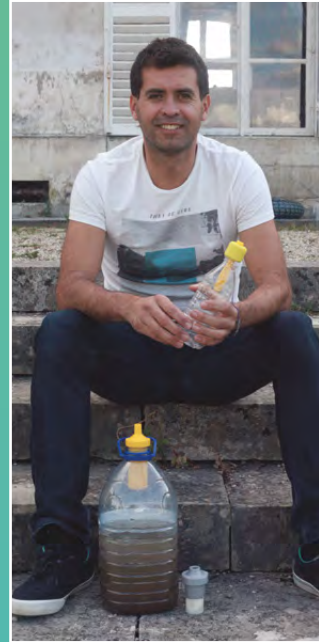
Led by Mauricio Cordova, The Faircap Project began after a trip to the Amazon Rainforest, “I realised that no matter where you are on the planet, we all will be suffering from more contamination from human and industrial activities in rivers, lakes, and natural water reserves,” Mauricio explained.

Following POC21, Mauricio was selected by the Humanitarian Innovation Fund (HIF) to receive a grant supporting the research, design, production, testing, and implementation of the Faircap filter project. And as part of the grant, Mauricio was partnered with Oxfam International to prototype multiple versions of the Faircap filter for various uses cases, including a mini version for personal use and a more robust filter for small communities.

## THE MAKER MOVEMENT IS KEY...

Oxfam aims to take Mauricio’s design to a final product that can be produced in large quantities and at a low cost, so that any humanitarian organisation can make use of the Faircap filters. Four months after first receiving HIF support, Mauricio made significant progress in defining the final 3D designs while working with two labs to prepare the first batch for production.

Mauricio explains that one of the most important reasons the maker movement is key to humanitarian



**Left** ◀ Faircap’s initial 3D printed prototype filter

**Above** ▶ Mauricio Cordova proudly takes a swig of freshly-filtered water

innovation is that “it opens up many more opportunities for finding technical solutions by many more actors”. Previously, only well-funded research labs or established companies could reasonably offer a product or service for emergency relief or economic development. Now, however, there is a growing army of makers who are immensely passionate about digital fabrication.

Mauricio’s approach of developing a product with an open approach from the very beginning meant that it was much easier to leverage a large community of contributors to make improvements on Faircap designs and even propose new ideas and solutions for future water filtration systems.

Mauricio explains that the main difference between traditional research and development, compared with open source and open innovation, is that access to online information means that there are no restrictions on who can become an inventor. Open innovation and design can help transfer more power to responsible and creative end-users seeking to drive change through decentralised problem-solving. →

# Showerloop

Minimising water and energy usage through real-time shower 'looping'

**T**ypical 'linear' shower systems are wasteful; taking clean water heated to a comfortable temperature, the resulting grey water and heat energy goes down the drain after a moment's use. The longer the shower, the bigger the associated energy and water footprint becomes. Jason Selvarajan, an environmental engineer from Finland, started the Showerloop project in an attempt to create a water and energy-saving shower solution to reduce this footprint, while bringing water and basic sanitation to people and

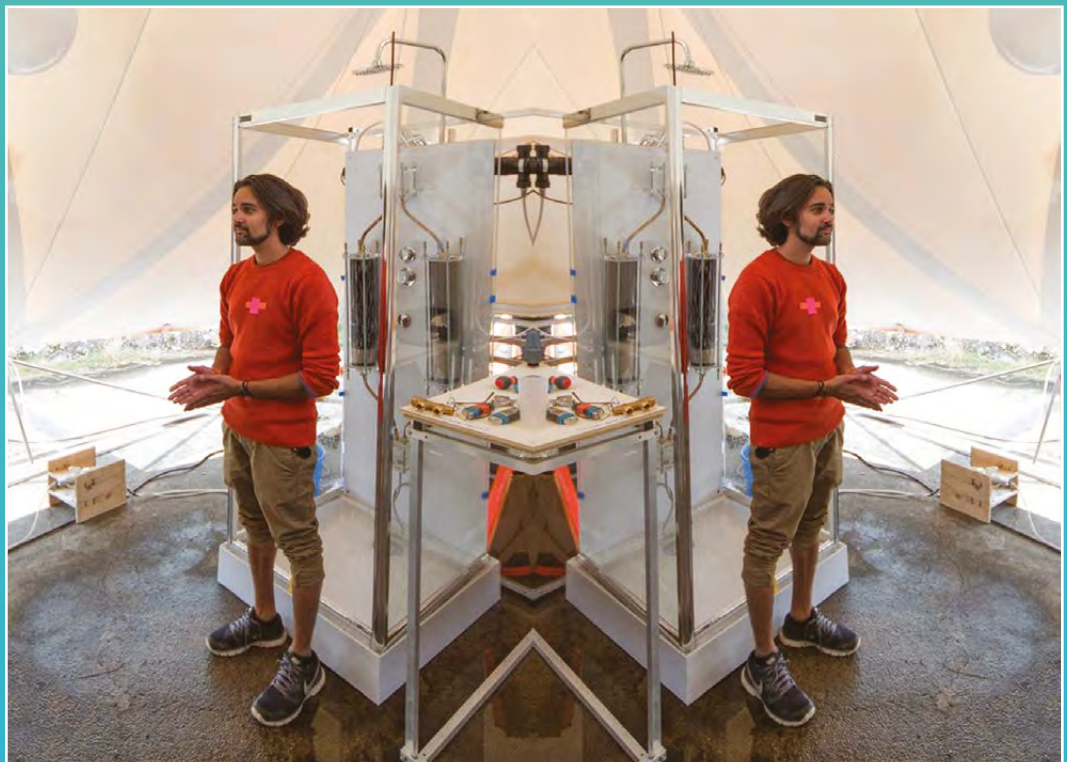
places that otherwise wouldn't be able to access it. Jason's primary objective during POC21 was to improve the appearance of his system and to introduce his work to a broader audience.

## THE MAN WHO LOVES TO SHOWER...

A standard shower uses approximately 10 litres of water per minute and a substantial amount of energy to heat the water, depending on the temperature. Showerloop's filtration system cleans shower water in real-time, reducing the amount of water and energy consumed by up to ten times. According to Jason,

**Below** ♦ Jason Selvarajan and Eduard Kobak discuss Showerloop's design at POC21's public conference

**Right** ▣ Save 650 kWh of electricity per year with an Arduino and some hardware







**Left** ◆  
As well as saving the environment, Showerloop adds a certain steam punk aesthetic to your bathroom

A year after POC21, Jason and collaborator Eduard Kobak won the Galileo Knowledge Prize at the GreenTec Awards 2017 for developing “a truly sustainable innovation that we urgently need in times of global warming and climate change”, according to GreenTec Awards judge, Stefan Gödde. Jason and Eduard have continued to support the Showerloop project, attracting the attention of Autodesk University in Las Vegas, where they built and displayed their latest prototype “sink loop” before launching the Showerloop concept as a fully fledged business. While DIY kits of the Showerloop are periodically available for purchase, Jason and Eduard have also made the designs open source, enabling anyone to replicate the Showerloop technology independently. The only condition is that you share what you learn with the Showerloop team.

“ I think we’ve already got all the technology that we need so that everyone on the planet can live sustainably and with an even higher quality of life than before. Open Source is the fastest and fairest way to do that ”

Jason Selvarajan

using Showerloop every day “would save an estimated 33 000 litres of water and 650 kWh of energy per person and year” compared to a normal shower. The system works by capturing hot water from the drain and filtering it through a microfibre screen to remove hair and other large contaminants. Sand is then used to filter out smaller particles, while homemade activated carbon mixed with salt removes chemicals and foul odours from the water. Finally, ultraviolet light is used to kill any bacteria in the water, before pumping it back to the showerhead for another ‘loop’. The resulting recycled water is so pure that it exceeds both US and EU standards for drinking water. In the future, Jason intends to establish collaborations with more international partners that can help him build a clearer understanding of how the project can be applied to a humanitarian context.

Jason designed the Showerloop system in an attempt to balance filtering capacity with aesthetics and deployability. The filters had to be large enough to adequately cleanse grey water, while remaining small enough to fit into a typical bathroom without causing complete disruption. Showerloop’s latest design uses two slim-profile ‘flow regulators’ to split the flow of grey water between two filters before recombining the filtered water into a single stream ready for UV sterilisation. This design enables two thinner filter tubes to do the same job as a single large-diameter filter.

**FINNISH DESIGN**

Jason, who lives near Helsinki, first had the idea for Showerloop in 2009, while working on a class project as part of a course in thermodynamics. His background as an environmental engineer honed his technical skills, but he found trying to position the concept of Showerloop as a product significantly challenging. “It’s one thing to design something, but another to figure out if it will work in the real world,” he mused.

Currently, Jason and Eduard have ongoing collaborations with several organisations in Finland and across Europe, including Aalto FABLAB, Metropolia University of Applied Science, and Turbiini Startup Accelerator. The current focus is on exhibiting their work at the Finnish Design Museum. They have designed a flat-pack shower stall that can be produced and assembled using a laser cutter and basic hand tools to accompany the Showerloop system.

Jason’s advice to others who want to develop sustainable technology? “You should be ambitious to get it out there and make it big.” →

**Below** ◆  
Finnish environmental engineer Jason Selvarajan



# AKER

A modular growing system for producing food in cities

**B**ased in Denver, Colorado, Aaron Mararuk and Tristan Copley Smith are co-founders of AKER. AKER evolved from a previous citizen science project the duo launched in 2014, called Open Source Beehives. Both Aaron and Tristan

felt that digital fabrication could be used to create innovative, snap-fit solutions for the urban farming movement, and joined POC21 to make that a reality.

The purpose of AKER is to provide people in urban areas with the opportunity to grow their food while creating habitats for wildlife, even in small spaces like balconies and rooftops. Envisioning a city where lifeless concrete structures are transformed into living ecosystems, the current range of kits includes a chicken coop, vermicomposter, three plant beds, and two sorts of beehives.

**Below** ♦  
Flatpacked kits are easy to manufacture and transport

## DIGITAL FABRICATORS BECOME URBAN FARMERS

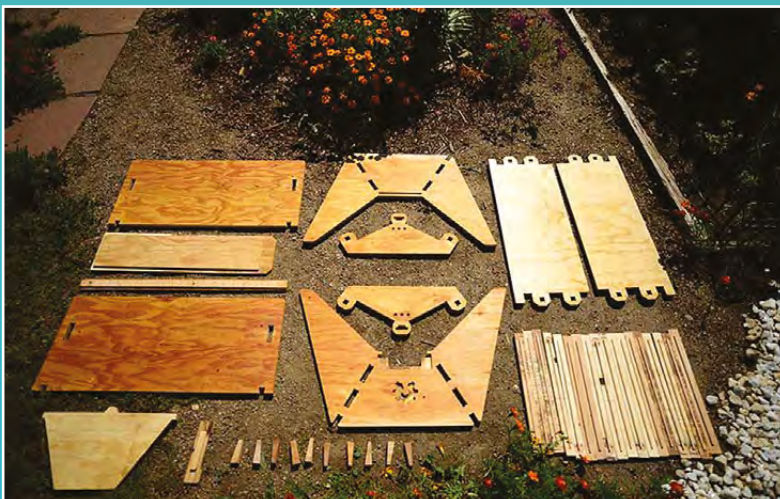
Aaron and Tristan's first project, Open Source Beehives, is a network of citizen scientists that use sensor-enhanced beehives to help identify the causes of declining bee populations throughout the world. They believe that open source collaboration and open data are vital tools for directly addressing such a pressing issue.

By monitoring hive audio signals, the group is gathering data that enables them to correlate the sound of the bees with hive state and hive health. For example, bee colonies that are preparing to swarm will emit an identifiable audio signature. Using Open Source Beehives hardware, it's possible to monitor the activity of multiple hives remotely.

The primary goal of Open Source Beehives is to identify the unique sounds of hives that are in a near-collapsed state so that researchers can correlate those signals with likely causes. They also aim to provide users with an advanced warning if their bees are in danger, enabling bee-keepers to intervene quickly, hopefully saving the colony.

During POC21, Aaron and Tristan had produced six new open source designs for urban agriculture products. These new plans included a modular, multi-level, raised planter bed, a raised bed, a top bar beehive, a wall-mounted planter, a two-hen chicken coop, and a worm composting bin. According to Tristan, all of these items can be used to build "a complete backyard farm." Just like Open Source Beehives, the AKER kits can be assembled without the need for power tools or even screws or glue and can be easily flat-packed for storage or transportation.

Each kit is routed using high-end CNC machinery from plywood that meets Carb II standards for "no added





urea formaldehyde,” ensuring that soil, chickens, bees, or urban farmers using the kits, are not exposed to harmful chemical substances.

Furthermore, the meticulous layout of the design files ensures that as much of the plywood sheet is used as possible, with only a small percentage of the material ending up as compostable waste.

### GROW YOUR OWN FOOD

Tristan’s personal aim is to spread access to homesteading equipment so that more people can produce their own food and live more healthily. “There’s a growing interest out there with people wanting to get back to the land,” he says.

Since POC21, AKER co-founder Aaron has been awarded a fellowship grant from the Shuttleworth Foundation, which provides funding and support for individuals seeking to implement innovative ideas for social change. Rather than a traditional academic fellowship aimed at research or study, the Shuttleworth fellowship is an opportunity for those interested in pushing technological boundaries and challenging accepted norms.

Aaron’s fellowship focus was to continue developing the methodologies, tools, and kits used in the development of AKER to restore balance to food production and consumption cycles. Much like Open State and Oui Share, The Shuttleworth Foundation believes that openness is vital for ideas to become stronger and spread through replication.

Initially only available to order in the US and Canada, AKER kits are now available in Europe and Australia, with other regions in the works. Since first bootstrapping the company, AKER is now a profitable business that



continues to expand their kit line, which currently includes a catalogue of eleven open source designs and accompanying online training materials to foster an international network of collaborators.

Another area the team is exploring is to distribute the kits to refugee camps to create pop-up gardens. Aaron believes that if food can be produced within the refugee camps themselves, not only will more healthy food become readily available, it may even result in the formation of a micro-economy to help support that community. “Locations where people sometimes have to stay for ten or 20 years could become living regions rather than more of just a place where you’re stuck,” he tells us. □

**Above** □  
AKER’s six new open source designs on display beside the POC21 exhibition tent

**Below** □  
Aaron Mararuk and Tristan Copley Smith in the old stables workshop



“ **My secret motive is to spread the merciless idealism that will likely be emanating from the camp, along with the idea that the human ability to dream, create and repair is alive and well amongst your fellow human beings — despite what you may have heard** ”

Aaron Mararuk

HackSpace magazine meets...

# LUCY ROGERS

Columnist of this parish and general polymaker

**“I’ve only just realised that I’m not actually an engineer: I’m a maker.”** So says Dr Lucy Rogers, fellow of the Institution of Mechanical Engineers, the Royal

Astronomical Society, and the British Interplanetary Society. She’s also got a City and Guilds in wood turning, a PhD in blowing bubbles, and is an alumnus of a NASA problem-solving academy that’s finding ways to save us from fiery doom brought about by space junk.

In an age when specialisms are rewarded and people increasingly know more and more about less and less, Dr Rogers knows lots of things about lots of things. That makes her a good person to learn from and an even better person to sit down to have a cup of tea with, which we did on a cold frosty morning in February. →





Above 🚩  
Dr Rogers has just realised she's left the soldering iron on



**HS Morning Lucy! There's so much we can ask you about: the Guild of Makers, the Internet of Things, space, robots, dinosaurs... How did you get into making in the first place?**

**LUCY ROGERS** It's really cool, isn't it? I've got every six-year-old's dream job. My first year at university they sold all the lathes because they were going more academic rather than practical. So I didn't learn that much hands-on stuff at university. I had a car that I did up and got through its MOT; I think I learned more with my little Renault 5 than I did with the academic stuff. But every project I did at university, where I got the choice, I did the ones that made things. When I got my first job I was sponsored by Rolls-Royce. I loved the manufacturing part much more than the maths. The hardest maths that I use is trigonometry. I don't use much more than that.  $V=IR$ , a bit of algebra.

My PhD was using Bernoulli's equation, but it's not the maths that excites me. For my PhD I was looking at how bubbles are made in firefighting equipment. So I had all the maths of how much air's going to get retained, how much surfactant do you use, how much soap solution do you use, how much water do you use. The bit that excited me was making a nozzle out of Perspex and getting a high-speed video camera and watching bubbles being made. That was my PhD – making bubbles. And every project since, if I've been able to make something, I have. My academic background is all engineering, but I'm a maker and that's what I want to do, that's what I love.

**HS Do you think your engineering degree helped you to become a maker? I know quite a few people who don't have degrees who don't realise that what they missed out on by not going to university wasn't that much at all. A PhD might be a little bit different, mind.**

**LR** A PhD is mostly about tenacity. Knowing what I know now, I would probably have done an apprenticeship. If I was recommending to a 16- or 18-year-old nowadays who wanted to go into engineering, I'd say do an apprenticeship (which may also lead to a degree), but that's just another way of doing it.

I wanted the hands-on factor, and back then you were academic or you were practical: you couldn't do both. Whereas my grandfather probably left school at 14, didn't have an education, but could make clocks or model cars or steam boats or spinning wheels and could work it all out. He went to the Greenwich Museum where John Harrison's chronometer was, and would go there with a ruler when my mum was small. My mum was left to play on the docks and my granddad would go in with his ruler, take a measurement, go home,

When I got my first job I was sponsored by Rolls-Royce. I loved the manufacturing part much more than the maths

get a bit of brass and make that piece. He'd come back the next week, measure another thing, come home... he made the first quarter-scale model of Harrison's chronometer number 1, and it's now in the Science Museum.

He was making without the education. [But] I wouldn't be where I am without having 'Doctor' in front of my name. Because it gives that credibility.

No-one cares what the doctorate's in: I've got a PhD in bubbles. I got chartered with the Institution of Mechanical Engineers when I was 25, and became a fellow at 35-ish, because I knew that I needed that piece of paper in order to be taken credibly in the industry, as a freelance and as a woman as well. That was a big push in why I did that, and it's worked. So I have the credibility in having a degree, but what I learnt during the degree is not so relevant.

You learn more from your mistakes. At university I really didn't get on with either the computing or the electronics, and for my final degree I just selected the mechanical and industrial manufacturing segments instead. My electronics was  $V=IR$  and that was about all I could do. I knew where to put a resistor.

So in 2011 I started to get into Arduino and I remember getting a music shield on the Arduino, and the ground on [it] wasn't zero volts. And I blew up two of these music shields at £30 a go, which at the time I didn't have, I couldn't afford it.

I had no idea what I'd done. I didn't know that ground wasn't always 0 volts, that ground is relative.

Where do you learn that sort of stuff?

**HS You learn it by blowing stuff up.**

**LR** And by someone saying when they make the same mistake. Fortunately we've now got Twitter, and people can share their mishaps. "I've done this – where have I gone wrong?" There's almost always someone out there who'll respond with "ha ha, I did that – this is what you've done".

**HS This seems like the perfect time to talk about the Guild of Makers, which has already been helping people out on the internet before it's even launched. Tell us about it: what are you planning, and why does the world need it?**

**LR** I've been round Maker Faires and you can't get to have a go at things because there are too many kids in the way. It's not politically correct to kick a child out of the way. I wanted to have a conference for makers who are making professionally. So not: "You've never touched a soldering iron before; this is how we solder", but "this is how you set up a business", or "this is how you go into mass production" and all those sorts of things.

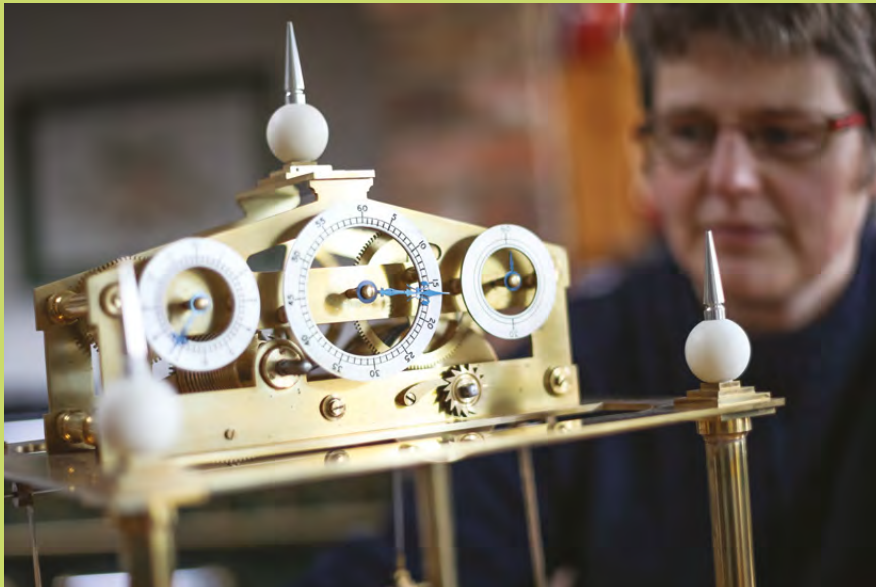
If you wanted to set up a company to make stuff 20 years ago you'd →





Above  
As well as having a PhD in bubbles, Lucy has a City and Guilds in wood turning – she made this chair





**Left** ♦  
The clock is made from brass, by Lucy's grandfather – a self-taught maker

probably need an advertising person and a marketing person and a logistics person before you even started. Now you can order stuff online, you can sell your stuff on Amazon – most of the difficulty has gone. You can remove a lot of the dull, dangerous, and dirty bits of your production. You can either outsource it to a factory that has that kind of equipment, or you just don't have to do it any more.

So nowadays you don't have to do every part of the process. But you can still be in control of every part of the process.

I think cottage industries are coming back with more people not having a job for life, and more wanting to be creative and actually realising that they have a route to do that.

So the Guild of Makers is for professional makers and those who want to be professional makers. That's how Makers' Hour started [follow the hashtag #MakersHour on Twitter every Wednesday 8–9pm UK time]. Before I go and set up the Guild of Makers as a limited company, let's try Makers' Hour, because setting up @GuildofMakers was free. And it's taken off. We've got a queue of people wanting to host it, we've got probably 30 or 40 regulars joining in, there are probably 200 people who've joined in at various times.

**HS** You're planning the launch at the beginning of March, so anyone reading this before then should check back on the website after 1 March. How are things going behind the scenes?

**LR:** You'll be able to join as a member, which will be relatively inexpensive. It doesn't matter who you are, what you're doing, anyone can join. You will get

“Nowadays you don't have to do every part of the process. **But you can still be in control of every part of the process**”

access to a load of other makers, you'll get discounts at our conferences, workshops, you'll get discounts from (these need to be confirmed) RS Components, Adafruit, Autodesk training...

For the purposes of the Guild, a maker is a practical person who takes pride in creating physical items using their imagination and skills. I know a photographer could be a maker – artists are makers – but the main focus of this guild is for those who make practical items, physical items. Computer programmers, computer gamers, won't fall into that definition. If they want to

join, they're welcome to join, it's just that I'm not focusing on their specific need. It's all practical stuff.

So that's normal membership; you can also become accredited, where you'll be peer-reviewed, so not only do you make a lovely widget – that is, if you're a wood turner, not only are you an excellent wood turner – but you can also make it to time, to budget, make it so that you can make a living out of it, so you're not just making pompoms and selling them on Etsy for the cost of the wool.

The accreditation will be like a kite mark, a way of saying “yes, we have seen that this person can make a good product”. If it's an electronics thing it's not just a breadboard; they can make it on a serviceable PCB, install it in your factory and it actually works, and it will work three months, six months later.

Then if a big company approached and said, “We need someone who can do this,” instead of us saying, “Get Jane Smith, she's really good at that,” the Guild of Makers can act as a brokerage. Jane Smith can invoice Guild of Makers and the Guild of Makers invoices the large company, so you take out the thing that happens when

small people deal with big companies where it takes six months to get set up on their accounts system. That's going to be in the future. It's not what I'm launching with.

By 1 March I'll have memberships open and there'll be a founder member perk if you join before 1 April. So that's for the individual maker. Companies can get involved too. If you're a company member you don't get the discounts that individual members get, but you do get access to the makers, the directory of makers, first dibs on sponsoring things at the events.





**Left** ♦  
As seen on TV:  
you may recognise  
this dinosaur from  
such TV shows as  
*Robot Wars*

Scratching each other's backs works really well. A lot of this stuff isn't commercially sensitive. Information about the difference between a sole trader and a partnership is stuff that you might as well share. I'm also hoping to partner with a legal company, so if you ever do need that kind of help, you'll have access to it.

In this village there are probably about 50 makers, but I don't know them. I've gone round people's houses and seen what they do as part of the Open Studios days. Some of it is for fun. A lot of it is for a hobby and they sell things for the price of the raw materials. Which is great, but it's not a profession.

This started with me wanting to know more makers. I want to know more people who make professionally.

I've now got people around the world wanting to join, and so it's not only going to be in the UK – it's going to be franchised or licensed somehow internationally. I've got people in New Zealand, in Sweden, The Netherlands, Greece, all wanting to do the Guild of Makers in their own country.

**HS** Do you think open source has helped in creating the conditions where you can do this? You spoke earlier about the spread of cottage industries. Is open source and sharing a big part of that?

**LR** Yes, most definitely. From sharing, from people running workshops – even paid-for workshops – I could run a workshop on how to start with Raspberry Pi; someone else could run one on making chairs. We've got not just Makers' Hour, we've got #makershelp, and if you've got a problem we've got quite a few people watching that hashtag who'll direct you to someone who can help.

A lot of the stuff that I make personally, I have used other people's open-source software. And I refer back to it when I write blogs and how-tos. And I don't like saying, "Yay it's mine now!" Because it's not. I don't want to patent something or make a profit on someone else's work.

But I can write the blog, the step-by-step guide. And now some of those people who had been helping me can now refer to my blog. "I helped Lucy do that thing. How did it work again? Oh, she's written it up! That's how we did it."

It seems to be something that people have been waiting to crystallise around. And this is it.

Whether I want to or not, this is going to happen.

**HS** Do you think that you're trying to fill a niche that the hackspace movement is already filling?

**LR** I think the hackspace movement is growing up. The maker movement is growing up, and those who are doing it as a hobby want to do it as an industry. So the makerspaces, the hackspaces are wonderful, but only if you've got one locally and you've got the right people in it, because they're all run by volunteers.

There aren't many makerspaces that have been successfully run as a business. I'm a member of the Society of Authors, and when I see that the Society of Authors is offering me workshops on how to give a talk, on how to do your tax return, on how to protect your intellectual property... I wanted that for makers, and it wasn't there... so, that's ultimately what the Guild of Makers is for. □

**Below** ♦  
"Of course, the best page in HackSpace magazine – or any other publication – is page 20"



# PAPER CLIPS



Hold, build, and break out things with this inexpensive ubiquitous tool



Mayank Sharma

[@geekybodhi](#)

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

**E**verything in the 21st century office would seem alien to H. G. Wells's Victorian-era time traveller – except for the paper clip.

Invented in the 1870s, the loop within a loop design of this handy little office item hasn't been improved since. William Middlebrook patented the design for the machinery to create paper clips in 1899 and sold it to Cushman & Denison. The American office supply manufacturer registered a trademark for the Gem name in connection with paper clips in 1904. Paper clips are still sometimes referred to as Gem clips, and the Swedish word for a paper clip is gem. While Norwegian Johan Vaaler is often credited as the inventor of the paper clip, his design was different and never mass-produced.

Before the paper clip came along, the straight pin was the paper fastener of choice. While it was cheap and easy to use, it left rust stains and holes in

the paper. In the mid-1800s, the mass production of low-cost steel that had the right balance of strength and malleability helped dislodge the straight pin in favour of more flexible alternatives like the looped paper clip. There were several other clip shapes that were developed close to the beginning of the 19th century. The Fay clip is often credited as the earliest patented design in 1867, followed by the Wright clip patented in 1877, and the Niagara clip a couple of decades later. Some of these clips used less wire, while others could secure larger stacks of paper. However the Gem clip won, not only because of its elegant design, but also because its production was easy to automate. All it took was three bends and a snip. There were no sharp edges and the paper clip was supple enough to snug papers between the loops and then hold them together.

Over the years the paper clip has been twisted, pulled apart, and used as a tool for everything from ejecting optical drives to inserting SIM cards and even to pick locks. Kyle MacDonald famously traded a red one for a house. The humble paper clip was even used as a symbol of resistance by the Norwegians during the Second World War against the Nazi occupation that forbade people from wearing badges or pins depicting national symbols. In a spiritual continuation of that tradition, the paper clip has perhaps been immortalised as a symbol for the digital era in the form of the universal attachment icon.



# PAPER CLIP

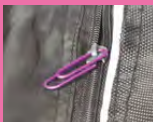
## PAPER CLIP HACKS



**MOBILE PHONE STAND**  
Use pliers to slightly bend upwards one end of the paper clips. Now stretch the other loop in the shape of an inverted V and rest your mobile phone between the two.



**BOOKMARKS**  
Slightly pull one end of the paper clip, wrap a piece of paper around it, and cut it in the shape of a flag. Then press the pulled end back into place.



**EMERGENCY ZIPPER**  
If the pull-tab of a zipper is broken, you can replace it by looping a paper clip through the slider mechanism.



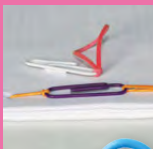
**FIND THE SEAM ON A ROLL OF TAPE**  
Stick a paper clip to the end of a tape roll to easily locate its seam. To use the tape, first pull it with the paper clip and then reattach it to the end before you tear away the tape.



**SECURE UNRULY WIRES**  
Loop a rubber band through one side of a paper clip. Now wrap the band around a coiled wire and hook it on the paper clip to prevent the wire from unravelling.



**EMERGENCY HOLDER**  
Straighten a paper clip and then bend it in the middle and use pliers to twist the ends into hooks. Use one end to attach to a drawer and the other to hang trinkets.



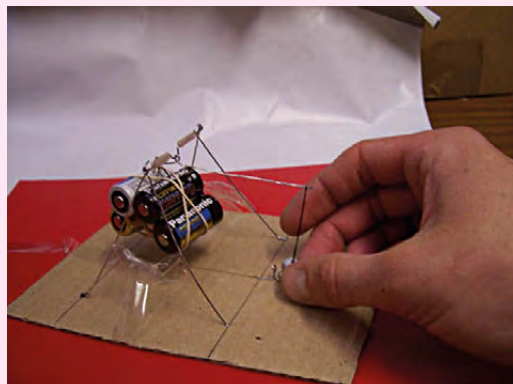
**BIND STACK OF PAPER**  
Knot a rubber band to one end of a paper clip. Then run it through the punched holes in a stack of papers and hook it to the loop at the other end.

# TREBUCHET

**A**lex Palfreman-Brown has the perfect cure for cubicle boredom. All you need is a handful of paper clips and a pair of pliers. You can use them, together with some other pieces of office stationery, to build a trebuchet that's powerful enough to hurl balls of Blu-Tack across the cubicles and wage a war on your colleagues. Alex shows you how to first straighten the paper clips and then intricately shape them into the swinging arm, the axle, the trigger, and other components of the trebuchet. You then assemble all of them on a piece of corrugated card, together with ballast (Alex uses a bunch of batteries), roll pea-sized drops of Blu-Tack into balls along with some string, and fire away. The whole contraption takes about an hour to put together. The build is so popular that it has earned Alex several Instructables.com Pro memberships, which he gives away as competition prizes in the hackspace he helps run. The evil mastermind is fully aware of the sinister implications of his war machine: "If I'm ever feeling glum, I just consider how many man-hours have been lost internationally to bored office workers building my trebuchet. That always puts a smile on my face." □

**Project Maker**  
**ALEX PALFREMAN-BROWN**

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



**Left** Alex's trebuchet has been featured in two Instructables books and in Bre Pettis's Weekend project videos

FEATURE

# PAPER CLIP SCULPTURES

**Project Maker**  
**THOMASIN DURGIN**

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

**Right** ♦  
Thomasin started creating this structure while stuck in a wrong job using the materials around her cubicle

**T**homasin Durgin is a teaching artist in Memphis, Tennessee and fabricates jewellery using all kinds of metals.

Almost a decade ago she found herself in a cubicle in a job that was "not a good fit" for her. So she started creating art from the materials readily available in her workspace – paper clips. To make these sculptures, Thomasin adopted the traditional basket weaving techniques to work with paper clips. She outlined the shape and then drilled holes into a wooden base and glued several 12-inch-high heavy-gauge wires into them. She then threaded two of these wires into each paper clip, sliding them down and staggering them into rows. This particular sculpture took over 2000 paper clips. Commenting on a photo of her sculpture, Thomasin writes that the process is rather



limiting: "I'd like to adapt true basketry techniques to be able to ditch the wood and create more rounded vessels, spheres, etc. Then I can combine both methods to make more complex structures." □

# MINIATURE WEAPONS

**Project Maker**  
**BRETT**

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

**Right** ♦  
Brett has patiently shaped paper clips into bows and arrows, different types of swords, a gauntlet, as well as a fishing rod

**P**aper clips are a wonderful medium to express yourself creatively. Their malleable nature allows them to be bent into all kinds of shapes that can be held over a period of time thanks to their sturdiness. Armed with a pair of needle-nose pliers and some glue, Brett took some paper clips and transformed them into beautiful miniature weapons. Brett hasn't published the procedure for sculpting the paper clip armoury, but you can reverse-engineer his process thanks to the excellent macro photographs of the creations by Brett's friend, Dan Nicholas. Dan's images of Brett's awe-inspiring work are detailed enough to help you make out each and every bend, turn, and twist of the paper clips. If you're like us, all it'll take is one look at Brett's rudimentary weapons and you'll be instantly compelled to try your hand at creating them. □





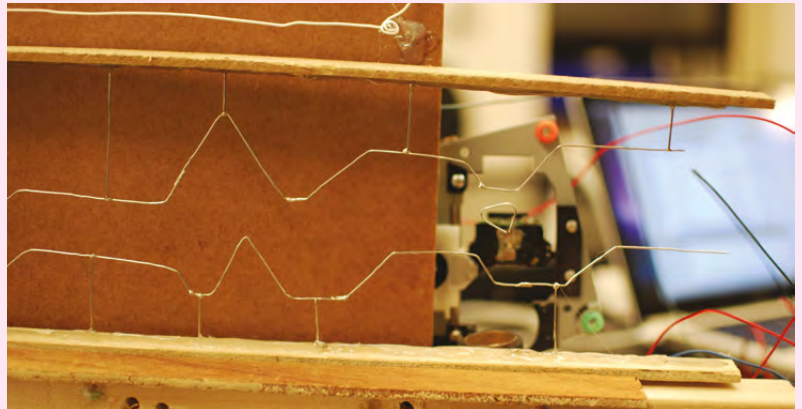
# METAL RACE GAME

Project Maker  
**GREG BORENSTEIN**

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

**A** talk on the rise of mobile gaming back in 2009 inspired game designer Greg Borenstein to build a physical one-dimensional scrolling game based on a racing game he played on the TI-83 series of graphing calculators. In his version, the player's car essentially stays in place while the track scrolls from right to left at a constant rate. The player can move the car up and down to navigate the twists and turns of the circuit. Greg used a couple of motors to move the car and the track that was made out of paper clips. A triangular piece of metal serves as the car and when it comes into contact with the paper clip circuit, they close a switch that triggers a buzzer indicating a crash and the end of the game. Greg has detailed the mechanical and electrical parts of the build in a couple of blog posts and also published a video of the whole contraption in action. □

Below ▣  
Greg soldered several paper clips into a maze track that stays within the attached motor's vertical travel range



# PAPER CLIP JEWELLERY

Project Maker  
**LINA DARNELL**

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

**L**ina is a master crafter and a mum of two young kids. Inspired by a blog post on another website, Lina engages her kids creatively by using a bunch of paper clips and some duct tape to design some simple pieces of jewellery. She hooks paper clips together to the desired length of the necklace or bracelet and then wraps about an inch of tape around the middle of each clip. While it sounds simple, the end result is an attractive trinket. Lina's post is dotted with images and she also discusses ideas to extend the simple designs by adding more paper clips and dangling beads and small pendants to the end. "This is a great craft for summer camps, scouts, or simply an afternoon at home," writes Lina, whose kids love the paper clip jewellery. □

Left ▣  
Bump the designs a notch or two by using a pair of pliers to bend the paper clips and create artsy-looking pendants



How I Made

# TIME-TO-GO-CLOCK

Count down to future events in style

By Peter Kent

**O**n 20 January 2017, Donald J Trump was inaugurated as the 45th President of the United States of America. I

remember the date well,

because that was the day I stumbled across Nixie tubes for the first time. Out of these seemingly unrelated events emerged the Time-To-Go Clock, Trump Edition.

Like countless others, I felt a compelling need to say something about the new Commander-in-Chief. Spilling out more polarising words seemed pointless, however. Instead, I'd make something that would, literally and figuratively, speak for itself.

The Clock's defining feature, broadly hinted at in the name, is that it can display the time to any future event. In the case of the Trump Presidency, the US Constitution helpfully provides the exact time and date when his term of office will end. Of course, it is possible President Trump could be re-elected, in which case the clock can be easily set to the end of his second term (or any other date) without further programming.

## FEATURES

Somewhat conceitedly, the clock has its own dedicated website, which lists its major features as follows:


- Optional cycling through time, date, Trump administration days-to-go, and Trump administration hours, minutes, and seconds-to-go
- Simple, menu-driven setting using an LCD and website provided by the clock's own server
- Celebrates President Trump's affinity for Twitter by tweeting the time left for his Administration at a random time each day
- Uses Russian IN-14 Nixie tubes and Soviet-era military-grade toggle switches
- Uses an IR motion detector to turn off the tubes if no one is around to see them, and logs activity on the clock's website
- Full operating instructions are also on the website



**Above** ♦ Toggle switches add a more tactile experience than push-buttons





Above  Nixie tubes have ten elements, one for each digit, but only light one at a time

- Time and date obtained from the internet
- Colour and brightness of LED backlights can be easily configured by the user
- 12/24-hour display choice

## DESIGN

My first thought was to go with what I will politely refer to as the Trump Aesthetic – ‘Versailles-built-in-Blackpool’ perhaps sums it up best.

The big drawback to this approach was that I intended to display the clock in my own house. I have nothing against Blackpool – I am from there – or even Versailles, which is quite nice too. It’s simply that combining two equally wonderful things does not necessarily square the output.

So in the end I elected to give the clock a more Cold War beaten-up military vibe with a repurposed B&K A/V systems controller as an enclosure.

## THE BUILD

### Components – driver board and tubes

As a newbie to Nixie tubes, I wanted to cut down the risk of a total project failure. I also wanted to complete the project fairly rapidly.

For these reasons, I decided to base the clock on a pre-built driver board which was designed as an Arduino shield. I bought mine from GRA & AFCH in Ukraine. The board came pre-populated with IN-14 Nixies and was under \$100, shipping included.

The enduring appeal of using Nixie tubes in clock displays is not difficult to fathom. Despite their ineffable beauty, they are an obsolete technology, a

physical embodiment of the passage of time, and a reminder of all things lost and irretrievable.

Using the B&K as an enclosure was meant to amplify this theme. Solid and well-made, it might have provided many more years of service, but the lack of internet connectivity and HDMI connections simply overwhelmed it.

The one thing that didn’t quite fit in was the faceplate of the B&K. It was far too nice. Fortunately, removing this exquisitely machined chunk of aluminium revealed a utilitarian steel panel ideal for my purposes. In all likelihood, you do not have a spare B&K Reference 30 A/V System Controller lying around. Do not despair. Just about any new or repurposed metal box will do – I’d recommend something at least 250 mm × 150 mm × 75 mm though, especially if you intend to include a four-line LCD and the other features I ended up with. →

## FEATURE



Left ◊ Stripping back the front panel revealed just what I had to work with

### Components – the rest

- **Arduino Mega or clone equivalent** – although GRA & AFCH say the Clock shield would work with an Arduino Uno, in my implementation I needed the pins and extra processing grunt of the Mega.
- **Arduino Ethernet shield or clone** – if I were to do this project again I'd probably try to use an ESP8266-based board for connectivity, but at the time I was more comfortable with the Ethernet shield.
- **I<sup>2</sup>C 20 × 4 LCD module.** The display serves several useful purposes. It indicates what mode the Nixie tube display is in, greatly simplifies setting the future time being monitored, and indicates the time the clock has (or will) tweet that day. Another reason I included it though was as an example of a display technology that spelled the end for the Nixie. More practical, much cheaper, but hardly a thing of beauty. Take your pick.
- **Rotary encoder.** The Clock shield comes with some momentary button switches which can be used for setting the clock. However, the additional functions of the clock would make button control alone very cumbersome and in any event they are not accessible in my enclosure design. The B&K actually uses a very nice rotary encoder as part of its own control scheme which I could have reused. However, the B&K's encoder did not incorporate a momentary switch, which meant you could not easily select an option dialled in using the encoder.
- **Switches.** The rotary encoder working with the LCD display allows many setting and control functions to

be combined in a user-friendly way. However, for some functions you just cannot beat the convenience and satisfaction of flipping a sturdy toggle switch.

- **Potentiometer.** Used to control dimming of the backlight on the LCD display.
- **IR motion sensor.** Some Nixie clock implementations use a timer feature to turn the tubes off at night or other set intervals. This doesn't seem entirely sensible to me – much better to use this \$5 motion sensor, which works incredibly well. The clock also logs the last detected motion on its website, a feature which I find very useful for reasons that need not detain us here.
- **12V 1A power supply.** The driver board shield is designed to take 12V power from the Arduino Vin pin, so a 12V supply is a must.
- **Miscellaneous consumables** such as jumper wire, crimp connectors, grommets, heat shrink tubing, etc.

That might sound like a lot, but with a bit of careful shopping the whole lot (minus the enclosure) can be had for not much more than \$150.

### TOOLED UP

All you need to complete this project is a drill (ideally with a step bit), a soldering iron, a multimeter, and various hand tools. As with most projects, if you have more stuff, you will probably find a use for it. I drew upon many different resources for the coding side of things, primarily Google. Other search engines are available.

### Preparing the enclosure

I removed some of the B&K's electronics and power supply to give me plenty of room to work. The hardest part by far (and even this wasn't too difficult) was drilling the six holes for the Nixie tubes in the top of the case. Two additional, smaller, holes are also required for the digit-separating neons. Alignment of all these holes is obviously very important and this is the one time when not having the board pre-populated would help. I considered various ways to get the alignment right but in the end used the simplest method. After drawing a line across the case



Left ◻ The clock cycles through different ways of displaying the time left



where I wanted the tubes, I turned the board and tube assembly upside down and marked where the tip of each tube met the line. I then centre-punched these marks, drilled pilot holes, and got to work with the step bit.

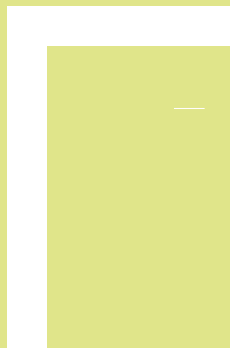
This is not the first project where I wished I had a drill press, but it is certainly doable without.


The great advantage of the step bit approach is that it allows you to dispense with all that measuring malarkey and just keep going until the holes are the right size. That said, using digital calipers is a more convenient way to check the hole size as you go along.

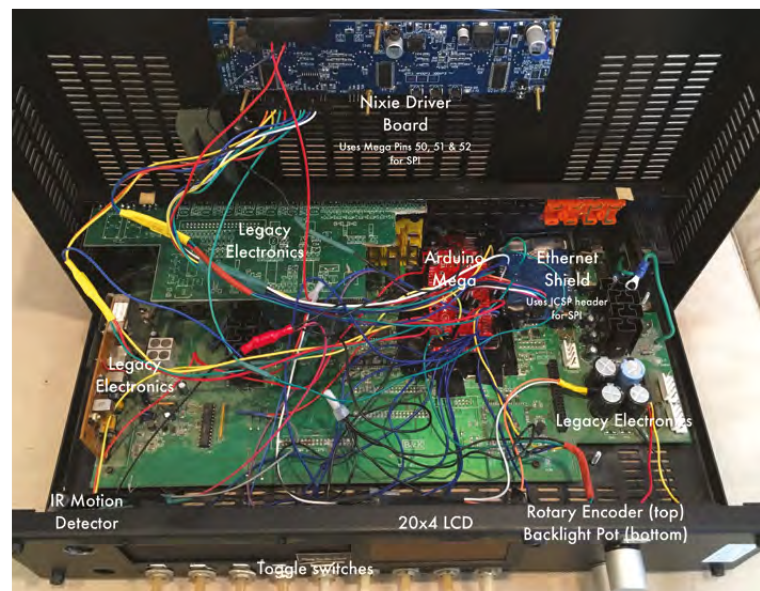
Unless you have a machine shop, or are a lot handier than me, you will still end up with slightly misaligned holes. This is not really an issue since the holes need to be little bit bigger than the tubes (we will be applying around 180 volts of direct current to the tube pins, so contact with a metal case would not be good).

Oversizing the holes also provides space for including an insulating grommet where the tube pins pass through the case. Conveniently, the grommets will also hide minor misalignment issues.

Although the driver board was designed to be an Arduino shield, I couldn't install it like that for this project. For one thing,



**Right**  I removed just enough electronics to fit in the new boards



it couldn't be stacked with the Ethernet shield. For another, I figured I might as well mount the Arduino and Ethernet shield so their external connections (power, USB, and Ethernet port) would be accessible from the rear of the case. This actually greatly simplified mounting of the driver board and tube assembly, which could be secured to the underside of the case with small nuts and bolts.

Connections between the driver board and the Arduino (and the Arduino and Ethernet shield) would now be all by jumper

wires, with the Arduino and Ethernet shield mounted separately to the bottom of the case.

With that taken care of, I needed to figure out where to put the LCD display and switches. The B&K enclosure had a rectangular cut-out in the front panel almost exactly the right height for the display. The cut-out was too long, but I solved that problem with a piece of black metal mesh removed from an old speaker.

The front panel also had nine square holes where various buttons had been located. Nine holes meant nine switches, to which I assigned the following names and functions:

- **DST** Quick way to switch between DST (BST) and Standard Time (GMT). Depending on the time server being used, this may not be strictly necessary
- **12/24** Selects 12-hour or 24-hour display formats on the Nixie and LCD displays
- **Tweet** If the clock has not tweeted that day, it will tweet immediately
- **Web** Starts the clock's web server (mainly for configuration and instructions)
- **Tubes Off** Turns off Nixie tubes →



## FEATURE

- **Red** Turns off red LEDs
- **Green** Turns off green LEDs
- **Blue** Turns off blue LEDs
- **Edit Mode** Puts clock in edit mode (effectively, this changes the operation of the rotary encoder so it can be used to change the future event being monitored)

Frankly, if there were fewer cut-outs, I could have dispensed with some of these or assigned them to the encoder. But their convenience means they do get used more, especially changing the LED backlight colours.

Finally, I installed the rotary encoder, LCD dimming potentiometer, and motion detector in various other existing or newly created holes.

### STAYING INFORMED

You can see the code at [hsmag.cc/nSTnRR](http://hsmag.cc/nSTnRR). A not inconsiderable advantage of buying the GRA and AFCH driver board is that

they offer some sample code to run a clock using the board. That said, since I needed to make some fairly significant modifications to this code, it was important to start with an understanding of how their code works.

First, some basics. Nixie tubes work in a similar way to an array of ten LEDs with a common anode (at least, for a tube that can display the digits 0–9). Each LED represents one digit, so you can light any LED/digit by

*The clock's defining feature, broadly hinted at in the name, is that it can display the time to any future event*

applying the requisite voltage to the anode, then grounding the cathode of the LED you want to display. In this arrangement, you can just connect each LED cathode to an Arduino pin (through a resistor, of course) and drive the appropriate pin low to turn on that LED. Since the Arduino keeps pretty good track of elapsed time, switching on each LED sequentially in one-second intervals is not much more complicated. Already we are well

on the way to making a clock! To make a working Nixie-based clock, however, we still need to overcome three obstacles:

- Nixie tubes require around 180V DC to light up; an Arduino cannot source or switch anything like that.
- A clock displaying hours, minutes, and seconds requires control of six tubes. If we persist with the approach above, we'll need 60 pins just for switching the Nixies; even an Arduino Mega has only 54 I/O pins.
- While the Arduino keeps pretty good track of elapsed time, it doesn't actually know what time it is.

Since we've already moved on to discussing the code, let's just note that the driver board solves obstacle one by stepping up the 12V from the Arduino's Vin pin to the required voltage.


The second obstacle is overcome by a mixture of hardware and software. On the hardware side, the driver board has three shift registers, each with 20 high-voltage outputs. Connected in series, these registers can therefore do the necessary switching of all six tubes without multiplexing.

On the software side, we need to tell the registers which of their pins to switch. This is done by converting the digits to display on the clock (held in the `stringToDisplay` variable) to a 64-bit variable, then sending that data via SPI in eight byte-sized chunks to the registers. This is handled by some clever bit manipulation in the `doIndication` function (courtesy of GRA and AFCH).

Last, we can crack the third problem by using a library of time-related functions (TimeLib) and getting the current time from an internet time server.

Computers, microcontrollers, and the like establish what time it is by counting the number of seconds from a reference point, for example 1 January 1970. The TimeLib library helps deal with the tedious mental arithmetic needed to figure out what the time and date is if, for example, it is 1516365708 seconds



Above  The LCD screen can display any messages you want



since the beginning of 1970. The code also uses the TimeLib library to calculate the time to the future event being monitored.

And that's about it for the central task of displaying current time and time to the end of the Trump administration.

## TWITTERING

In addition to establishing a dedicated Twitter account, setting up the Twitter feature requires including an authorisation token in the code, which can be obtained from [hsmag.cc/HrvMId](https://hsmag.cc/HrvMId). While it would be easy to have the clock tweet at a fixed time each day, that would be very boring – the hours, minutes, and seconds to go would always be the same. So the code uses the `randomSeed` function to pick a particular second on each day to fire off the tweet. A bit of care is needed to make sure the tweet is grammatically correct whether the time units are singular, plural, or zero.

## SERVER AND WEBSITE

The clock has an 'external' site created on and hosted by Squarespace. As part of the whole project, I wanted some decent-looking internet presence and this was a quick way to realise that. Of more interest to readers here, however, the clock also has its own internally hosted site. This provides a convenient place to include an instruction manual, a place to configure the future event name to be sent to the LCD, and a log of the motion detection. It also has a free widget that counts down the time to the end of the Trump administration from [timeanddate.com](https://timeanddate.com). On one level, it's a bit depressing that this information is so easily available after the not inconsiderable effort invested in building the clock. On the other hand, the widget is not a Nixie clock, and it does provide an independent time check. Unlike the Squarespace site, there's nothing fancy about the 'internal' site, though I did use some Bootstrap templates to make it look halfway decent. The internal site (which can be made available over the internet using a free hostname and dynamic DNS address from [no.ip.com](https://no.ip.com)) is only available when selected by the user.

Quite a bit of code is used to manage the display and rotary encoder. This is fairly straightforward – take a look at the

previous code link if you want to dive into the nitty-gritty.

With so many switches and other controls, some labelling was required. I've never figured out how to do this even halfway satisfactorily. For this project I ordered a couple of metal dog tags customised with the words I needed. Cut up and appropriately 'distressed', these went at least some ways to complementing the overall design.

While this build took quite a bit of time and effort, each part of it is fairly straightforward, and you could start with a simpler display. It can also be reused many times – there will always be some future joyous event to which you'll want to count down. □

**Below** ♦  
Sure, LEDs could display the data just as accurately, but nothing appeals to us on a primal level quite like Nixies



# pi-top

Inspiring inventors and creators to seek the skills of tomorrow and create their future, today.

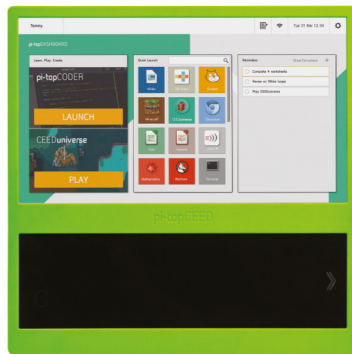


OCR  
Oxford Cambridge and RSA



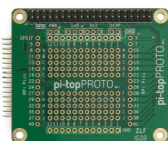
The modular laptop with sliding keyboard

- 8HR BATTERY LIFE
- 14" FULL HD 1080P SCREEN
- 180° HINGE
- CUSTOM PASSIVE COOLING BRIDGE
- MODULAR RAIL



The modular desktop

- 14" FULL HD 1080P SCREEN
- MODULAR RAIL
- ADJUSTABLE VIEWING ANGLES



pi-topPROTO



pi-topSPEAKER



pi-topPULSE

## pi-top

Colors Raspberry Pi 3 optional

AWESOME INVENTOR'S KIT INCLUDED

20+ projects to explore

Explore beyond the screen and keyboard by creating with the all-new **pi-top** modular laptop.

Get started with 20+ inventions in the inventor's guide booklet. There are 3 inventor's journeys - Smart Robot, Music Maker and Space Race.

## pi-topCEED

Colors Raspberry Pi 3 optional

**pi-topCEED** is the plug & play modular desktop. It's the easiest way to use your Raspberry Pi. We've put what you love about our flagship laptop in a slimmer form factor. Join hundreds of code clubs and classrooms using **pi-topCEED** as their solution to Computer Science and STEAM-based learning.

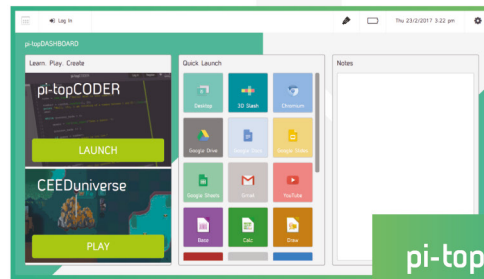
## Modular Accessories



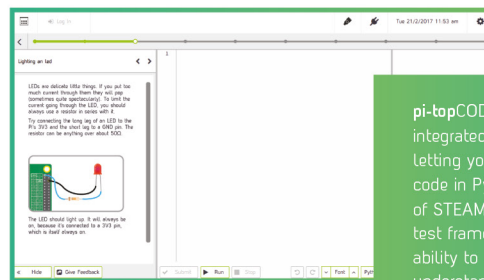


**pi-top** is an award-winning ecosystem designed to make experimenting, coding and building electronics, simple, affordable and fun. **pi-topOS** is here to guide you through the world of making!

The OCR\* endorsed **pi-topOS** (Operating system) platform comes pre-installed on the SD card shipped with every unit. **pi-topOS** software suite lets you - browse the web, - check emails, - create and edit Microsoft Office compatible files. Gain access to dozens of hands-on learning lesson plans with **pi-topCODER** and have fun learning to code with **CEEDuniverse**!



**pi-topOS**



**pi-topCODER** has a fully integrated coding environment letting you program hardware, code in Python and learn lots of STEAM skills! Our integrated test framework gives you the ability to assess your own understanding as you learn.



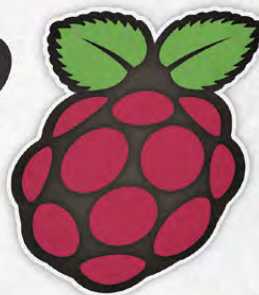
**CEEDuniverse**  
Learn programming concepts through our minigames, for example, learn problem decomposition by solving visual programming puzzles.



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

YOUR OFFICIAL **RASPBERRY PI** MAGAZINE

# The *MagPi*



The official Raspberry Pi magazine

Issue 67

March 2018

[rasberrypi.org/magpi](http://rasberrypi.org/magpi)

# RETRO COMPUTING

HARDWARE EMULATION & CLASSIC CODING WITH RASPBERRY PI

PUT YOURS  
INSIDE A  
Spectrum

**SMART HEARING AID**

Building better audio aids with Raspberry Pi

**MAKE A GEIGER COUNTER**

Prep for fallout with a smart radiation tool

**WATCHDOG ROBOT**

The security guard bot on patrol

# FREE PI ZERO W

With your 12-month subscription to the print magazine

# [magpi.cc/Subs1](http://magpi.cc/Subs1)

PLUS! FREE  
CASE, THREE  
COVERS &  
CABLES

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

## NFC DOOR

Control who comes in and out of your fortress of solitude/shed

PG  
92

## HOT WATER MONITOR

Keep an eye on how much time you can have in the shower

PG  
98

## IOTEA

Get the internet to make you a cup of tea. Britons, rejoice!

PG  
102

## MAKE YOUR OWN LIVING TERRARIUM

Create Victorian-style living art with simple materials



PG  
104

## ESP8266

Set up a web app to monitor solar energy production

PG  
108

## POWER SUPPLY

Build a bench power supply out of an old laptop PSU

PG  
80

## SCHOOL OF MAKING

Start your journey to craftsmanship with these essential skills

80 Laser cut enclosures

82 Arduino programming: Variables, constants, and more

86 Sew lighting into a hat





## WORKSHOP BASICS

# Laser cut enclosures

Making the case for cases



John Park

➔ @johndgarpark

John Park is a maker who builds creative technology project videos and tutorials for Adafruit Industries. He's the former host of the Make: TV show on American Public Television, and a 20-year veteran of CG animation at Disney, Sony, and other animation studios.

**M**akers and hackers love to build a wide variety of projects. These creations may range from the fun, interesting, and weird, to the serious, practical, and useful. They may be intended for music creation, gaming, education, assistive technology, communications, holiday decoration, media streaming, puzzles, theatre props, and – well, the sky's the limit! These wonderful projects are often made of electronic components, mechanical parts, motors, microcontrollers, buttons, switches, dials, speakers, LEDs, keypads, batteries, displays, and even small embedded computers.

Among this infinite, delightful variety, nearly all have one thing in common – once the prototyping is done, and the project is ready to be deployed and used, they need a case or enclosure to bring all of the elements together, to provide some protection, and even to give form and layout to their inputs and outputs.

Given the variety and creativity of your projects, why settle for an off-the-shelf, generic project box, with holes drilled and cut into it, when you can build your own case that's customised for your exact needs?

With access to a laser cutter at your local hackerspace, or by using an online vendor such as **Ponoko.com** (which has local hubs in a number of regions), you can build exactly what you need to suit your project enclosure needs.

The high precision and minimal kerf (the gap of material lost along a cut) of laser cutting means that you can build joints and features to accommodate your components with very tight tolerances. You can design your enclosure using free, open source vector drawing tools, such as Inkscape. There are also online, browser-based box generators such as **makerbase.com** to get you started.

Including components is easy. Add a proper circle to your design and you've got a perfect, press-fit opening for an arcade button!

### THREE METHODS

There are a number of ways to design and construct a laser cut enclosure. Many of these methods are adaptations of traditional joinery techniques used in woodworking or other types of flat stock fabrication. With certain adaptations, these techniques work well as laser cut designs, and can be made more quickly and easily than their wood shop counterparts, which require many repetitions of highly accurate cuts.

### FINGER JOINTS

The finger joint (a.k.a. box joint) is possibly the most popular technique in laser cut joinery. It is perfect for joining two pieces at 90 degree angles, is strong, and creates clean, flush joints.

The key to making finger joints is creating interlocking tabs with a cut depth equal to the thickness of the material stock you are joining. This is because the pieces you'll join will be at right angles, so the tab of part A will fit into the gap in part B without falling short or protruding beyond the face of part B.



**Above** ♦ The different joint types lead to different looks in the finished product





After cut and assembly, finger joints are typically glued together with an appropriate adhesive for your material type, such as wood glue for wood, or acrylic cement for acrylic.

Note, you can place the tabs wherever you like on the edges, so long as you match them to the gaps of the joining face. It's not uncommon to see finger joints run across the entire length of each edge, which is what the parametric online software usually does, but that's not usually necessary — you can get away with many fewer tabs, which makes for a quicker cut on the laser.

### MORTISE & TENON JOINTS

Another popular technique is the mortise and tenon, which is the woodworking name for a fitted slot and peg joint. These are great when you want overhanging faces for some parts, such as a

rounded lid that sits on top of the side walls. This face will have no tabs, only mortises that fit over tenons protruding from the side walls. By limiting your tenon height to the material depth, you'll create joints flush to the surface.

**Left** The finger joint is usually designed by automatic software, but can be customised for your needs

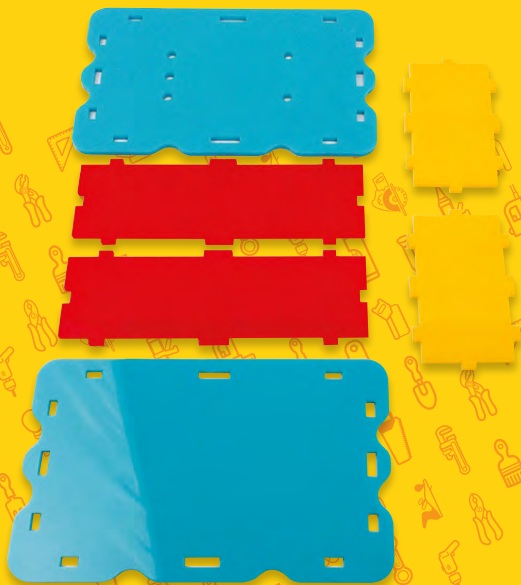
### CAPTIVE NUT AND SCREW T-SLOT JOINTS

Finger joints and mortise and tenon joints are usually glued together. If you need to open and close your case, consider using screws and nuts with the clever t-slot design. You'll be able to join panels at right angles as before, but this time by holding a square nut (a hex will do, just not quite as neatly) in one panel, and sending a screw through from the face of the second. No glue required!

Don't forget, you can mix these methods as needed, such as creating an enclosure with five box jointed/glued sides, plus a t-slotted back panel for easy removal using screws and nuts.

By designing your enclosure to fit your parts, you can make the exact fit and spacing you need. Use data sheets, dimensional drawing, and a set of digital callipers to fit things such as cut-outs for buttons and jacks, mounting holes for parts, and windows and bezels for displays.

There are, as you may imagine, many other ways to join together laser cut parts, but these fundamental methods will serve you well, and get you building great enclosures quickly.



**Left** Mortise refers to the hole (from the old Arabic word meaning to 'cut a recess') and tenon is the finger (from the Latin word meaning 'to hold')

**Right** You can combine techniques, as we've done here with a t-slot to give a removable panel to a finger-joint box



# Arduino Programming: Variables, constants, and more

Dig deeper into the data capabilities of the Arduino language and make your sketches easier to read and maintain



John Wargo

@johnwargo

John is a professional software developer, writer, presenter, father, husband, and geek. He is currently a Program Manager at Microsoft, working on Visual Studio Mobile Center. You can find him at [johnwargo.com](http://johnwargo.com)

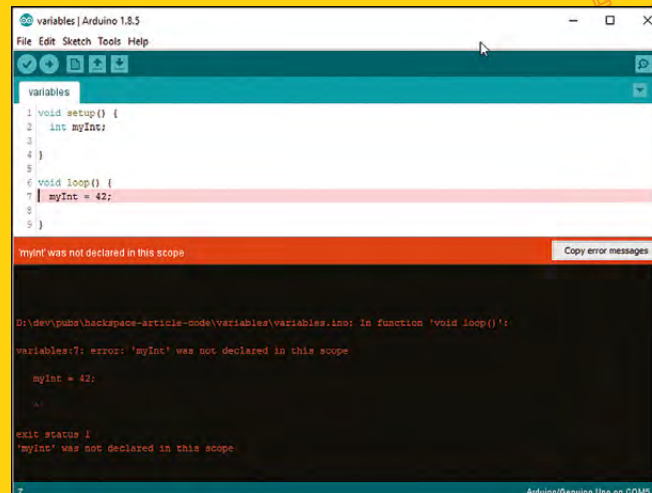
**D**o you want your Arduino sketches to look like they were written by a professional? Use variables and constants to self-document your code.

In this series so far, you've learned how to wire up some simple circuits and write some Arduino code to interact with the circuit. In this article, you'll put aside the hardware and learn about some fundamentals of the Arduino's language: variables, constants, and more. The goal is to make you a better programmer and help you deliver code that's easier to read and for others to understand.

Development languages such as Arduino's language provide mechanisms for managing data: the information (numbers, Boolean values, strings) that your sketches use to do their work. These are implemented through Variable and Constant language elements you've seen used in our previous articles. Arduino devices have a limited amount of memory and processing capabilities, and you'll want to make the best use of both in your projects. As you code your Arduino sketches, there are things you, as a developer, can do to manage the memory profile for a sketch, and make the sketch more readable at the same time.

## VARIABLES

In the Arduino language, a variable is a pointer to a memory location that stores a particular piece of data. Variables are used to store data used by the sketch, but also to increase code readability and maintainability. Variables enable you to refer to a data value by name rather than its value. As your sketch runs, your code will store and retrieve data by referring to the variable in expressions. Unlike some other languages, in the Arduino language you can't use a variable until you've defined it. So to define a variable, use the following expression:



**Figure 1** The Arduino compiler will block some types of error before you can upload your code to the board

```
DATA_TYPE VARIABLE_NAME;
```

Notice the semicolon at the end of the expression – it's required.

In this example, **DATA\_TYPE** defines the type of data that will be stored in the variable (supported types are array, bool, boolean, byte, char, double, float, int, long, short, string, unsigned char, unsigned int, unsigned long, word). The compiler allocates memory for the variable based on the data type you specify when you create the variable. The **VARIABLE\_NAME** element of the expression defines the name you'll use to refer to the variable in your code. A variable's name can only have upper-case and lower-case letters, numbers, and the underscore character; using anything else will cause a compiler error.

To create a variable called **myInt** that stores an integer value, use the following variable declaration:



## VARIABLE TYPES

Each variable can only hold one type of data, which you have to define when you create the variable. The most popular options are:

- **int** A whole number between -32 768 and 32 767 on 16-bit boards such as the Arduino Uno and -2 147 483 648 to 2 147 483 647 on 32-bit boards like the Zero or MKR1000.
- **long** A whole number between -2 147 483 648 and 2 147 483 647.
- **float** A number that can include decimal places between  $3.4028235 \times 10^{+38}$  and as low as  $-3.4028235 \times 10^{-38}$ . However, only the first six or seven significant figures are preserved.
- **string** Text (which needs to be quoted for it to be assigned to the variable).
- **char** A single byte which can represent an ASCII character.

```
int myInt;
```

Typically, you'll name your variables so the variable name says something about the value stored in the variable; this improves code readability. For example, if you were writing a sketch that recorded temperature values (like the examples in the Arduino tutorial in issue 3), you'd create a variable to store the current temperature (and self-describe itself) using the following:

```
float currentTemp;
```

In this example, I created a float variable since I know temperature will be a decimal value.

To assign a value to a variable, use an equals sign:

```
myInt = 42;
```

You can also assign an initial value to a variable as you're defining the variable:

```
int myInt = 42;
```

To use a variable in an expression, simply refer to the variable by name in your code. When the compiler reaches the variable reference, it will generate the code to retrieve the data from the variable's memory location and pass it into the expression. For example, to convert a temperature value from Celsius to Fahrenheit, you would write the following code:

// Development languages such as Arduino's language provide mechanisms for managing data //

```
float tempC = 17.5;
float tempF;
tempF = (tempC * 1.8) + 32;
```

The value assigned to **tempF** is calculated using the value stored in **tempC**.

Variables can be defined at any point in the sketch, but they behave a little differently depending on where they're defined. If you define them at the start of the sketch (before the **setup()** function), you can access the variable from anywhere. This is known as a global variable. However, if you define a variable within a function, you can only access the variable from within the function you define it in. This is known as the scope of the variable.

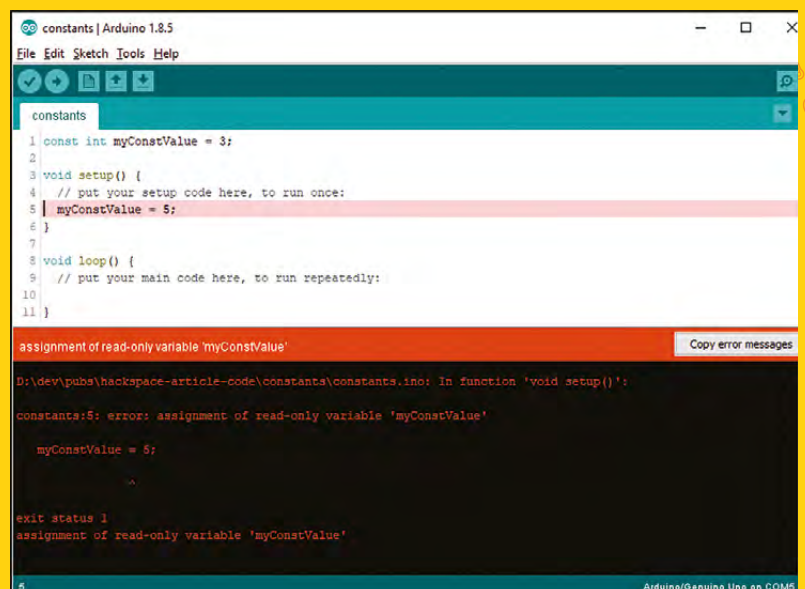
Variable names must be unique – you can't create two variables with the same name in the same scope. You can't have two global variables called **mySpecialVariable**, but you could have two local variables called **myInt**, if each variable is defined ➤

Figure 2

The Arduino compiler will try to highlight the line that causes the error, but this is sometimes misleading

### QUICK TIP

Assigning a value to a variable uses a single equals sign (=); comparing two values uses a double equals sign (==).



### QUICK TIP

Good variable names are essential for making your code easy to read. Think about them at the start as they're awkward to change later

in separate functions; however, doing this is likely to lead to an intense headache if you have to debug this code. **Figure 1** shows the compiler error you'll see when you violate scope.

### CONSTANTS

In C, constants define storage locations for data that cannot be modified; the data is referenced by a sketch, but never changed. They're typically used to define values used in calculations or string values used repeatedly by a sketch. Constant values are stored in memory and can be easily accessed by any part of your sketch.

Constants are defined the same way you define variables, except the constant definition always starts with the **const** keyword and you must assign a value to the constant. To define a constant in a sketch, use the following expression:

```
const DATA_TYPE CONSTANT_NAME = VALUE;
```

Constants are referenced by name, so **CONSTANT\_NAME** defines the name associated with the constant. **VALUE** defines the data value stored in the constant. Here are some example constant definitions, creating (in order) an integer constant, floating-point constant (decimal value), and a string constant (a character array) :

```
const int myConstValue = 3;  
const float pi = 3.14;  
const char myString = "My string array";
```

**Figure 3** Keep an eye on the size of the sketch as you're developing it to make sure you don't suddenly run out of space on your chosen hardware

Constants have the same scope options as variables. Constants defined at the beginning of a sketch, before the sketch's **setup()** and **loop()** functions, have a global scope which means they can be accessed from anywhere within the

sketch. Constants defined within any function are local constants and can only be accessed within the function.

As with variables, to use a constant in a sketch, simply refer to the constant any place where you want to access the constant's value. For example, to use the pi constant in a circumference calculation, you would use the following:

```
const float pi = 3.14;  
float circumference;  
float radius;  
  
radius = 2.52;  
circumference = 2 * pi * radius;
```

Constants are used primarily to increase code readability and maintainability. If your sketch

measures the radius of a circle, then calculates the circle's area and circumference, you could insert the value of 3.14 into each of the formulas. What happens then if we want to use a different value

for pi (perhaps including more decimal places)? You'd have to change the value for pi in each of the expressions that reference it. By storing the value in one place, you'd only have to change its value in one place to effect all of the expressions that use it. Additionally, from a readability standpoint, seeing pi in your code is much clearer in meaning to the reader than 3.14.

When you try to change the value assigned to a constant in a sketch, the Arduino compiler will complain, displaying the error message shown in **Figure 2**. You will not be able to deploy this sketch to the Arduino device until you've resolved this error.

### MEMORY MANAGEMENT

Arduino devices are inexpensive programmable microcontrollers, and in order to meet its imposed cost limitation, the devices have limited memory

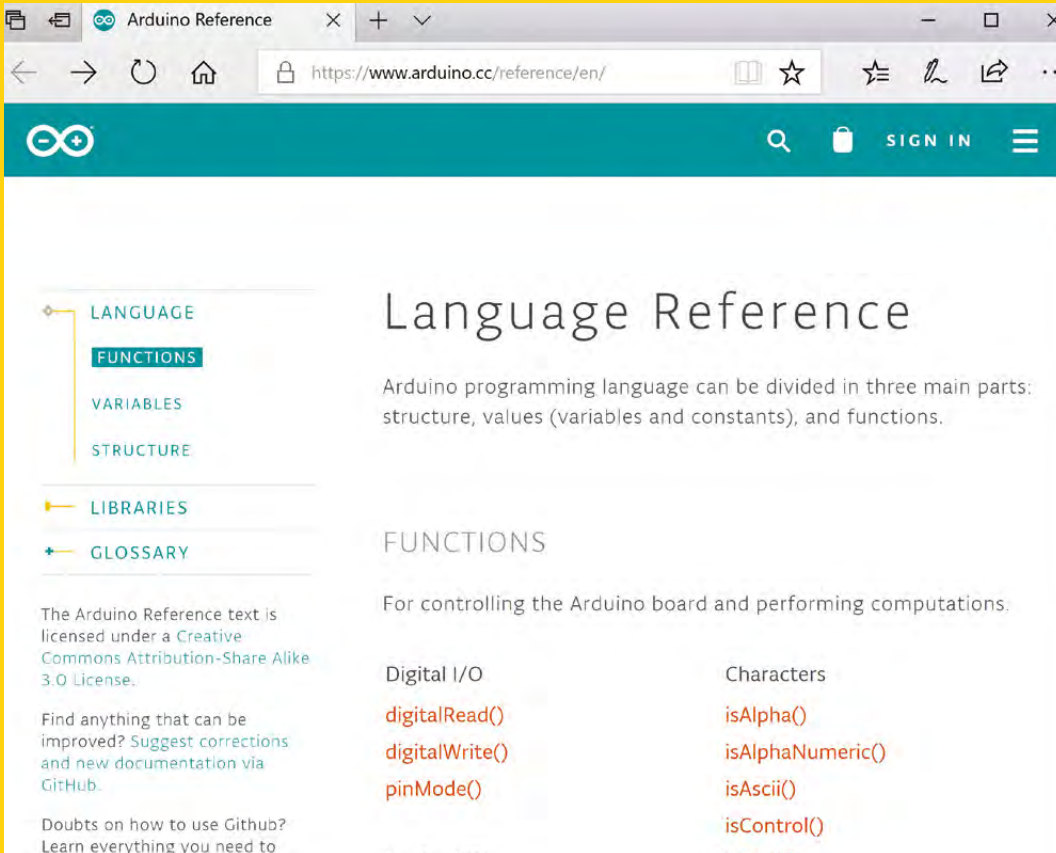
Constant values are stored in memory and can be easily accessed by any part of your sketch

```
Done compiling  
  
Archiving built core (caching) in: C:\Users\john\AppData\Local\Temp\arduino_cache_522048\core\core_arduino_avr_uno_c3bfe3f78ffbeab93536a1a484b588d5.a  
Sketch uses 4780 bytes (14%) of program storage space. Maximum is 32256 bytes.  
Global variables use 352 bytes (17%) of dynamic memory, leaving 1696 bytes for local variables. Maximum is 2048 bytes.
```

### QUICK TIP

If you learn to program on devices with limited RAM, it will set up up with good habits for the rest of your coding career





**Left** If you're ever unsure about exactly what a variable type means, you can check the Arduino language reference at [hsmag.cc/VYrTRY](https://www.arduino.cc/reference/en/)

and processing capabilities. As a developer, you must always pay attention to how you use the device's memory and processor.

**An Arduino device has a limited amount of three types of memory:**

- Flash Memory: Stores the Arduino bootloader and your compiled sketch.
- Static RAM: Used for variable storage.
- EEPROM (electrically erasable programmable read-only memory): Non-volatile memory you can use to store data while the Arduino is turned off.

**The Arduino Uno, for example, has the following memory footprint:**

- 256kB of flash memory
- 2kB of static RAM
- 1kB of EEPROM memory

When you compile your sketch, the compiler will inform you of how much memory your sketch consumes, as shown in **Figure 3**. As you code larger and more complicated sketches, or use more

**//** **Arduino devices are inexpensive programmable microcontrollers, and in order to meet its imposed cost limitation, the devices have limited memory and processing capabilities** **//**

libraries in your sketches, you must pay attention to this data in order to ensure that your sketch will run as coded on your selected Arduino device. In the example shown, we're only using 14% of flash memory (program storage space) and 17% of static RAM (dynamic memory) for my sketch, so we're in good shape (for now).

**Sketch uses 4780 bytes (14%) of program storage space. Maximum is 32256 bytes.**

**Global variables use 352 bytes (17%) of dynamic memory, leaving 1696 bytes for local variables. Maximum is 2048 bytes.**

If your sketch exceeds the available memory on the device, you must either re-factor the application to make better use of the memory available to you, or switch to another device that provides more memory. **□**

# Sew lighting into a hat

Keep yourself warm and visible on dark nights with a simple circuit



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](http://sophywong.com), chronicling her adventures in making.



**T**his is a simple and easy sewn circuit project, and it's great for getting started with wearables. We'll sew three LEDs onto a knit hat for more visibility at night. Once you've gotten the basics down with this project, you can use this circuit to sew LEDs into other garments too. Let's get started!

## CIRCUIT OVERVIEW

We'll be sewing the LEDs in parallel, so that each LED gets the same amount of voltage from the battery. All the LEDs will be in the same orientation in our circuit. Look for the + and - on each LED and follow the circuit diagram (shown right) as you build. Leave the battery out of your battery holder until you've finished building the circuit.

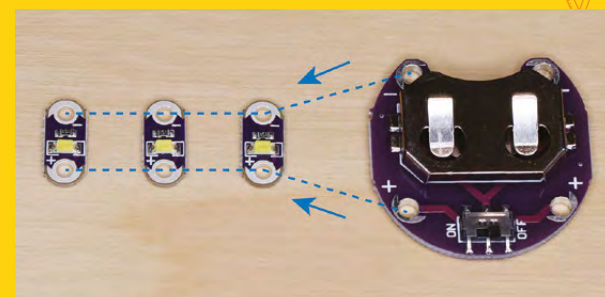
## PREPARE THE HAT

The LEDs will be attached to the outside of the hat, and the battery holder will be hidden inside the hat's folded cuff. You'll either need to use a hat with a cuff or fold back the edge of a longer hat. If your cuff

doesn't want to stay folded, tack it up at the sides with a few stitches of regular sewing thread. The cuff should be at least 4 cm wide to accommodate the battery holder.

## ADD THE LEDs

If your sewable LEDs came in a strip, gently snap them apart with small pliers. Find the front of the hat. Place the three LEDs on the front of the hat fold with the negative (-) side up and the positive (+) side down. Space them about 1.5 cm apart. Tack each LED in place with a small amount of hot glue.



**Above** Be seen in the dark with a light-up hat

**Right** Connect the LEDs to the battery holder





**Above** ♦ The battery holder lives on the inside of the folded-up part of the hat, its unsightliness hidden

### ADD THE BATTERY HOLDER

Since the battery holder is a rigid, flat piece, we'll place it on the side of the hat so it will be against the flat part of your head. Fold the edge down at one side of your hat. Place the battery holder so that its opening points toward the edge of the hat and the negative sew tab is on top. Use a dab of hot glue to tack it in place for now.

### SEW THE CIRCUIT

Now that we have our components tacked in place, we can sew the circuit to connect them. Thread your needle with about 50cm of conductive thread and tie a knot at the end. This may feel like a long piece of thread, but we want to sew each leg of the circuit in one continuous run.

Start at the battery holder: begin your run by stitching four or five times around the positive sew tab of the battery holder. Pull the thread snug against the sew tab for a good connection.

Use a simple running stitch to sew to the positive sew tab of the first LED. Loop around the positive sew tab four or five times, pulling snug each time.

**Below** ♦ The battery holder should go along the side of the hat, where it will fit along the side of your head.



Continue sewing this leg of your circuit, and connect to the positive sew tabs of the other two LEDs in the same manner.

When you've connected the last positive sew tab, tie off your thread with a tight knot. Trim the thread close to the knot to prevent short circuits.

Rethread your needle with another 50cm length of conductive thread. Go back to the battery holder and repeat this process to sew the negative leg of your circuit. Keep this second thread run at least 1cm away from the first.

### LIGHT IT UP

Now, slide a coin cell battery into the battery holder.

The LilyPad sewable battery holder has a handy on/off switch, so turn your project on and try it out. The battery holder should sit comfortably on the side of your head, and no conductive thread or component should be touching your skin.

If your LEDs don't light up properly, check

to make sure they're all aligned correctly with each other, and that the battery holder is in the same orientation: negative sew tab on top. Also make sure your conductive thread tails are trimmed to prevent shorts, especially at the battery holder.

Your author doubled the LED magic by repeating this process on the back of her hat with red LEDs. How will you customize this project and make it your own? Show us your build at [@HackSpaceMag!](https://twitter.com/HackSpaceMag) □

### YOU'LL NEED

- ♦ Knit Hat (find one with a folded cuff)
- ♦ 3 Sewable LEDs
- ♦ 1 sewable coin cell battery holder
- ♦ 3 V coin cell battery
- ♦ Conductive thread
- ♦ Regular sewing thread
- ♦ Wire cutters
- ♦ Needle, thimble, scissors
- ♦ Hot glue gun and glue sticks

Once you've gotten the basics down with this project, you can use this circuit to sew LEDs into other garments too

**Below** ♦ White LEDs at the front, red at the back



# Build an NFC-powered door lock

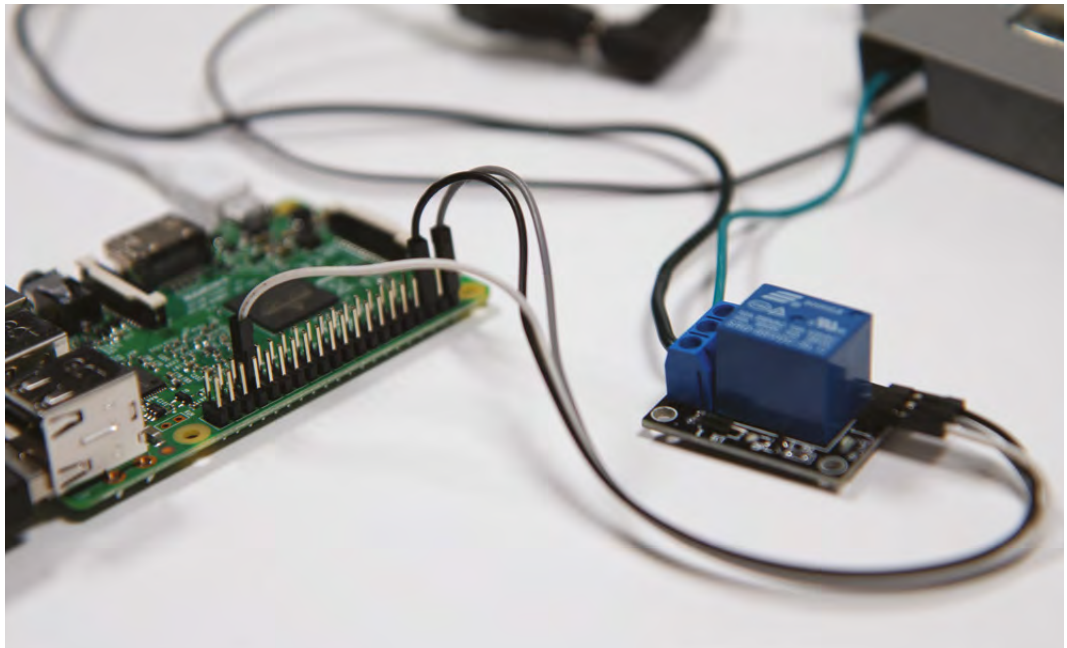
Forget keys and codes; secure your private hackspace with fob access!



Paul Freeman-Powell

@paulfp

Paul is a technology enthusiast, geek, writer, presenter, video producer, YouTuber, drummer, husband, and father to three children, including twins. You can find his website at [switchedonnetwork.com](http://switchedonnetwork.com)



**E**ven if it's completely unnecessary, there's always something quite exciting about adding extra layers of security to your workspace entrance, harking back to childhood days of asking "what's the secret password?"

before letting your friends into your den. If you're anything like us, your space will have some pricey toys and expensive equipment in it, so it can certainly give you some extra piece of mind to have it protected by an additional barrier from any would-be intruders!

This is a great first project if you're just getting started in the world of hacking and Raspberry Pi, as we'll learn not only about accepting input to the system from something other than a standard keyboard and mouse, but also about operating a relay switch to control an entirely separate (and higher voltage) electrical circuit.

Before we get started we're going to assume that you already have your Raspberry Pi all set up with the latest version of Raspbian installed and up-to-date. With that all ready, start by connecting up all the hardware components.

First, plug in the RFID reader to any of the USB ports. This type of reader is especially easy to work with as it acts just like a keyboard, in that the operating system sees the data contained in any fob presented to it getting typed in, followed by a new line. To see this for yourself, open up a text editor and then hold each fob to the reader in turn; you'll see a string of numbers getting quickly typed in followed by a newline character each time.

## GPIO PINS

Next, connect your relay to the GPIO pins. It's a 5V relay, so the positive pin (which will be labelled either

### Above

The finished build with all components connected. The Electronic Door Strike is powered by a separate 12V power supply and switched by the relay switch



```

pi@raspberrypi ~
login as: pi
pi@192.168.0.123's password:
Linux raspberrypi 4.9.59-v7+ #1047 SMP Sun Oct 29 12:19:23 GMT 2017 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*-copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jan 16 16:07:59 2018

pi@raspberrypi:~$ ls -lah /dev/input/by-id
total 0
drwxr-xr-x 2 root root 60 Jan 16 16:07 .
drwxr-xr-x 4 root root 120 Jan 16 16:07 ..
lrwxrwxrwx 1 root root 9 Jan 16 16:07 usb-Sycreader_RFID_Technology_Co... Lcd S
VC_ID_IC_USB_Reader_08FF20140315-event-kbd -> ../event0
pi@raspberrypi:~$
    
```

**Above** Each USB input device (e.g. mouse, keyboard) is assigned its own ID by the Raspbian operating system. You'll need to know the ID your RFID reader has been assigned so that you can listen to its input

'+' or '5V') needs connecting with a jumper wire to a 5V pin on your Pi, and the negative (labelled either 'GND' or '-') can go to any ground pin. The signal pin (labelled 'S' or 'in') needs to go to any numbered GPIO pin. You can actually use any you like, but the code examples later use the GPIO 13 pin, so you might as well do the same, really.

The electronic door strike works with a little electromagnet inside which pulls a small lever out of the way when activated by a 12V supply. As that is more than the Raspberry Pi can supply, you need a separate 12V power supply for this. A little DC power jack allows you to easily connect to the positive and negative from this supply's cable. If you want, you can wire the striker plate directly into the 12V supply and test it out by flicking the power switch and seeing how it allows you to open it only when the 12V is being supplied.

### OPEN SESAME

To incorporate this into your circuit, wire the positive supply from the 12V socket into the middle terminal (COMMON) of your relay switch. Then, connect the final loose wire from the door strike into the Normally Open terminal (usually labelled NO). Your striker plate is now switched on/off by the relay switch connected to your Raspberry Pi.

Now it's time to get serious and write some simple Python code to act as the brains between all of the

hardware components, switching the relay only when a valid key fob is presented to the RFID reader.

The first thing to tackle is reading the data contained in a key fob. Since we know that the newline character is 'typed' when the fob's code has finished being entered, you can use that newline character's appearance as an event which will trigger some decision-making in your code along the lines of: "is this fob allowed to unlock the door?"

In order to get to that stage, you first need to write some code to listen to the Input Device and capture what's being entered. This will involve using the evdev module, so before you do anything else, type the following into a Terminal window to ensure it's installed and available to use on your Raspberry Pi:

```
sudo pip install evdev
```

With that installed, create a new Python script called **lock.py** in your favourite text editor. The first thing to add to your script is the following lines: →

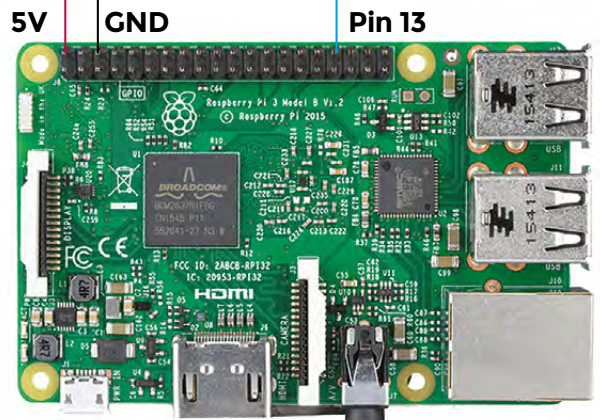
### YOU'LL NEED

- ◆ Raspberry Pi
- ◆ USB NFC/RFID reader + some coded fobs
- ◆ Electronic door striker plate
- ◆ 12V 1.5A power supply
- ◆ Female DC power jack
- ◆ 5V relay switch (single-channel)
- ◆ 3 × DuPont jumper cables (female/female)
- ◆ 1 × extra cable (male/male)

12V



**Left** A typical Raspberry Pi setup with relay switch powered by pins 2 (5V) and 6 (GND) and switched by the GPIO 13 pin



### POWER SOURCE

To avoid having to run two separate power supplies, consider using a dual-voltage power supply. If you have an old computer PSU lying around, you can easily make one yourself: simply use your multimeter to find the 12V and 5V supply cables normally delivered to a SATA hard drive. To make the PSU power-up without needing an 'On' switch on the front of a computer case, short out the green (signal) cable to an unused black ground cable.

### Right

The RFID reader inputs data it reads from the NFC tags (key fobs) into the operating system like a keyboard inputting text

```
#!/usr/bin/env python3

from evdev import InputDevice
from select import select
```

The first line specifies the environment in which you'll be working and that you want to use Python 3. Then, we need to import **InputDevice** and **select**, both of which will allow us to read the data from the RFID fobs.

Next, add the following three lines of code:

```
rfid_presented = ""
keys = "X^1234567890XXXXqwertyuiopXXXXasdfghjkl
XXXXyxcvbnmXXXXXXXXXXXXXXXXXXXX"
dev = InputDevice('/dev/input/event0')
```

The first line creates and initialises an empty variable in which we'll store the characters that make up the data string from the key fob, one by one. The second line defines a string of characters which later code will refer to in order to establish which character has been typed. The third line defines your input device so the code knows which to listen to. You'll need to check what ID your input device has been given by the Raspbian operating system and modify your code accordingly. To do this, type the following into a Terminal window:

```
ls -lah /dev/input/by-id
```

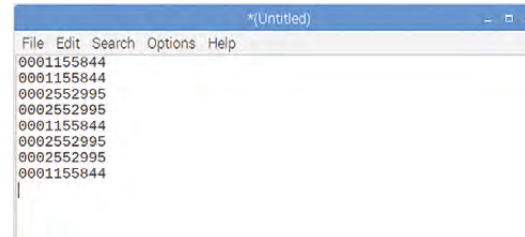
You should see a list of all your Raspberry Pi's input devices, with a symlink for each one giving it a friendlier name. Look for the long name which is clearly your RFID reader – in this case it's 'event0' (because I'm shelled into the Pi via SSH and there are no other USB devices attached, such as a keyboard and mouse).

### ALWAYS LISTENING...

The system needs to continuously listen for input from the reader, so we achieve this by defining an infinite loop like this:

```
while True:
```

Everything else in the program takes place within that loop, so it's now constantly listening for an event.



To handle the event, add these lines to the while loop:

```
while True:
    r,w,x = select([dev], [], [])
    for event in dev.read():
        if event.type==1 and event.value==1:
            if event.code==28:
                if rfid_presented=="0001155844":
                    # Unlock Door
                    print("Unlocking Door.")
                    GPIO.output(13,GPIO.HIGH)

                    time.sleep(5)

                    # Lock Door again
                    print("Locking Door Again.")
                    GPIO.output(13,GPIO.LOW)
                else:
                    print("Access Denied.")

                rfid_presented = ""
            else:
                rfid_presented += keys[ event.code ]
```

Inside this loop, we're listening for events of type 1 which means a (virtual) key has been pressed and a character entered. Each time that happens, we look at the event code to see if it's 28, which corresponds to the newline character, which we know from before means the end of the data being read from the fob. If the data that has been entered isn't a newline, we take that data and add it to the **rfid\_presented** variable using the '+' syntax, gradually building it up with a character at a time to create the string of text. Once we get a character which is 28, i.e. the newline, we stop building the text string up and see if it's the data that we're after.

For the sake of clarity in this example, I've hard-coded the data we're looking for into the if/else statements (see the 'Building in flexibility' box for a better real-world alternative). You'll obviously need to substitute that with the data from your own key fob. If the string read from the key fob is the one we're looking for, then we jump into action and do five things:

1. Print a line of text to the Terminal window saying what we're doing (which is very useful for debugging)

## BUILDING IN FLEXIBILITY

Instead of hard-coding the data from the RFID fob into your code, consider storing a list of authorised codes in a MySQL database instead, and modify the Python code with an SQL query to look up the presented code in the database to check validity. This opens up a world of possibilities; you could create a web interface to manage access remotely and assign owners' names to fobs, or you could add extra columns to your database table to restrict access to certain days or times... ideal for cleaners or dog walkers, etc.



2. Unlock the door by switching the GPIO pin to HIGH, which triggers the relay
3. Wait for five seconds to allow time for the door to be opened
4. Print another line of text saying the door is now being locked again
5. Lock the door again by switching the GPIO pin back to LOW

If the data read from the fob isn't the data we're looking for, then we do nothing apart from print a line of text saying 'Access Denied' to the Terminal. After either outcome, it's extremely important to remember to clear out the `rfid_presented` variable so that we're starting afresh next time ready for the next unlock attempt.

Before those lines of code which control the GPIO pin will work, you first need to import the `RPi.GPIO` module for use in your script, as well as the `time` module which will enable the `sleep` function to work. To do this, go back to the top of your script and underneath the two imports you already have, add these lines:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(13,GPIO.OUT)
```

As well as importing the two extra Python modules you need, this will also configure and initialise the GPIO pins ready for use.

With your Python script now complete, save and exit by pressing **CTRL+X** to exit and then pressing **Y** to save changes. Then to load your program, type:

```
python lock.py
```

### THE MOMENT OF TRUTH - DOES IT WORK?


If you hold the authorised key fob to your reader, you should immediately hear the relay 'click' and if you try the striker plate, it will allow itself to be opened. After five seconds, you'll hear another click and the striker plate will no longer open. Then try another fob and you should be met with a blank wall of silence, and an Access Denied message printed to the Terminal.


That's all there is to it! You now have a sophisticated security system for whichever room (or cupboard!) that you're securing. What's more,



```
File Edit Tabs Help
pi@raspberrypi:~ $ python lock.py
Unlocking Door.
Locking Door Again.
Unlocking Door.
Locking Door Again.
Access Denied.
Access Denied.
Access Denied.
Unlocking Door.
Locking Door Again.
Unlocking Door.
Locking Door Again.
```

**Left**  The Python script prints text to the Terminal, showing when the door is unlocked and locked again, as well as when access has been denied

since it uses a Raspberry Pi which can be connected to the internet, it's full of possibilities for expansion. You could write an app to unlock the door remotely by simply toggling the GPIO pin between high and low, or you could add a keypad to require a PIN to accompany the NFC tag presented. If you build one yourself, be sure to share it with us so we can see what you've come up with! 

**Below**  Once you're done, you can install the electronic striker plate into the doorway that you want to secure with NFC fob access!



# Smarter hot water

Use a Raspberry Pi Zero to guarantee you'll never run out of hot water again



Jenny Fletch

Jenny is a historian, administrator and organiser who likes it when life is made simple.



**W**e've all been there – halfway through a relaxing shower and suddenly the hot water runs out! If you're not first into the bathroom, how do you know if the water will last?

Using a Raspberry Pi Zero and a series of digital temperature sensors, this project will show you exactly how much hot water remains in the tank using a series of coloured LEDs – as the water runs out, they will gradually change from red to blue. It will even upload the data to Google Sheets so you can check the status remotely.

The same technique could be used to monitor the temperature of almost anything, such as a greenhouse, a fridge, or a fish tank.

## GETTING STARTED

This project is suitable for domestic hot water systems that use a vented hot water cylinder, usually installed in an airing cupboard. It will not work with an unvented pressurised hot water system. You can tell if the system is vented as there will also be cold water tank in the loft.

The hot water cylinder is typically made of copper, and covered in a thick layer of foam to insulate it and prevent the water cooling down too quickly.

Hot water is held in the cylinder and heated either from the central heating boiler or by an electric element. When you use hot water in the house, it is taken out from the top of the cylinder and it is replaced with cold water flowing into the bottom.

**Above** ■ Best of friends – when all the LEDs turn blue there's no more hot water, but with a smart thermostat you can reheat it from wherever you are



As the hot water is used up, the level of cold water rises up the tank. When the cold water reaches the top, that's it: cold showers all round.

This project works by having a series of temperature sensors fitted to the side of the tank at regular intervals. By connecting these to a Raspberry Pi, the rising level of cold water can be measured – so you can keep an eye on your hot water consumption and put an end to unnecessary cold showers.

## MEASURING UP

The Blinkt! LED module has eight LEDs, so this project uses eight sensors – each sensor controls the colour of one LED, changing it from red when the water is hot, to blue when it is cold.

The first step is to determine how far apart the sensors need to be. The top sensor should be close to the outlet in the middle of the cylinder. The bottom sensor should be about 5 cm above the point where cold water enters. Mark these points on the outside of the insulation with a marker pen.

Measure the vertical height from the top sensor location to the bottom one, divide by eight, and



### Above

Eight sensors are fitted at equal intervals from the top to the bottom of the hot water tank

mark off the intermediate positions. Use this information to calculate how much cable is required between each sensor. Also work out where you will mount the Raspberry Pi (somewhere easily visible and near a power supply), and how much cable is needed between the Pi and the first sensor.

## PREPARING THE HARDWARE

The next step is to solder the header pins to the Raspberry Pi. The Blinkt! LED module is ideal as it's nice and easy to connect to the Pi, and the existing Python library makes setting the colour and intensity of each individual LED really simple. The downside is that it covers all 40 GPIO pins, making it difficult to connect anything else.

Even though it covers all the pins, it actually only uses four of them – numbers 2 and 6 for 5V power and ground, and 16 and 18 for LED control.

The DS18B20 digital temperature sensors run off the 1-Wire protocol. This means all data communication is transmitted on one wire, no matter how many sensors are connected. The Raspberry Pi can read these signals on pin number 7 (GPIO 4). The sensors also need a 3.3V and ground connection, which are taken from pins 1 and 9 respectively.

By pushing pins 1, 7, and 9 of the header through the plastic strip before it's soldered, the three pins required for the temperature sensors are left sticking out of the back of the Pi, while the other 37 stick forwards in the normal way. The Blinkt!



Using a Raspberry Pi Zero and a series of digital temperature sensors, this project will show you **exactly how much hot water remains in the tank**



module is connected to the front, leaving the three connections for temperature sensors easily accessible behind.

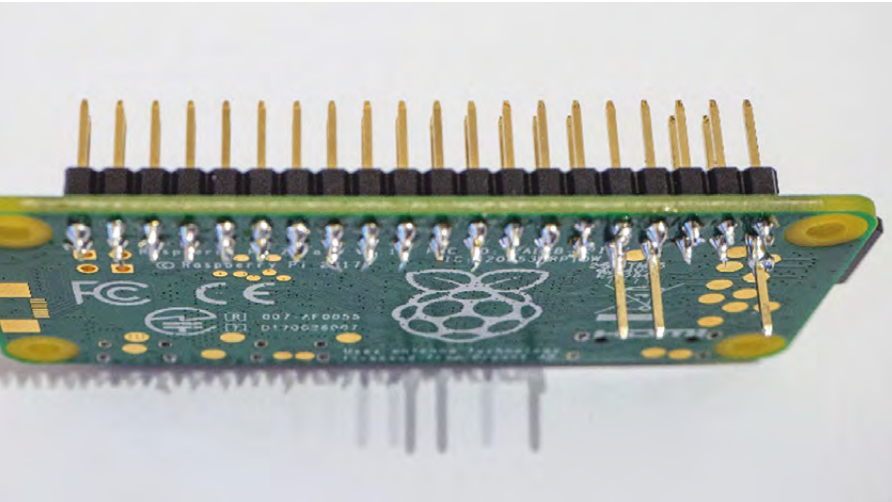
## WIRING THE SENSORS

The sensors used here are pre-wired, with red (+3.3V), black (ground), and yellow (data) cables attached.

A 4.7 k $\Omega$  pull-up resistor is required between the data wire and +3.3V. After this, it's a simple matter of daisy-chaining the sensors together, connecting red to red, black to black and yellow to yellow, each time extending the chain by the length required to place the next sensor further down the tank. →

## YOU'LL NEED

- ◆ **Raspberry Pi Zero W** (but not the latest version with pre-soldered header)
- ◆ **Case**
- ◆ **Power supply**
- ◆ **40-pin 2 × 20 male header**
- ◆ **Blinkt! 8 LED array**
- ◆ **8 × DS18B20 temperature sensors** (it's easiest to buy these pre-wired in a waterproof casing)
- ◆ **4.7 k $\Omega$  resistor**
- ◆ **30 g tube of thermal compound paste**



Above Pins 1, 7, and 9 protruding from the back of the board

### PREPARING A GOOGLE SPREADSHEET AND SETTING PERMISSIONS

The Python script is designed to upload data to a Google Sheet called 'Hot Water'. This can be changed to anything you like, but it is important that the spreadsheet is pre-existing in Google Drive and that the correct access permissions have been granted to allow data to be added.

Creating and granting permissions to the spreadsheet is best done on a full-size computer, not the Pi Zero.

Start by going to Google Drive on the web and create your spreadsheet. As said before, the Python script expects this to be called 'Hot Water', but this can be changed to anything you like, as long as the code in the Python script is changed to match.

Creating and granting permissions to the spreadsheet is best done on a full-size computer, not the Pi Zero

### UNLOCKING SECURITY

Now the security access permissions need to be granted for this spreadsheet:

- Go to the Google Developers Console (<https://console.developers.google.com>) and create a new project. Call it anything you like (e.g. hotwater).
- From the list of options on the left-hand side, select 'Library' and then search for the Google Drive and Google Sheets APIs. Enable them both.

- Now, from the list of options on the left-hand side, select 'Credentials'.
- Click the blue 'Create credentials' button and select 'Service account key' from the list of options.
- In the new screen, 'New service account' will be preselected as the only option.
- Enter a name in 'Service account name' – this can be anything you like (e.g. data\_upload), and choose Project / Owner as the role.
- An email address will be displayed in the window, ending in 'gserviceaccount.com' – make a note of this email address as you will need it later.
- Selected 'JSON' as the key type and then click 'create'.

A file will be downloaded to your PC. Rename this to **client\_secret.json** and save it somewhere convenient on your computer. If you open the file in a text editor, you will see the email address from earlier embedded within the code.

This file needs to be copied onto the Raspberry Pi and saved in the same folder as the Python

script, preferably in a folder called **/home/pi/python**. If you create the Python file in a different folder, the code should be updated to reflect the different folder location. This JSON

file becomes the digital key to unlock the spreadsheet. When Python attempts to write data to the Google Sheet, it sends a copy of the JSON file to the server. Google will confirm that the Sheet has been shared with the embedded email address. It's therefore important not to share this file with anybody else as it allows access to your files.

Return to Google Drive and open your blank 'Hot Water' spreadsheet.

In the top-right corner, click 'Share' and then add the email address from earlier. You may get an 'address not found' email from Google a few minutes later, but this can be safely ignored.

### QUICK TIP

A threshold temperature is calculated, which is halfway between the temp\_max and temp\_min. If the temperature is above this threshold, it is counted as 'hot' when calculating the percentage of water remaining.



## PREPARING THE SOFTWARE

Download and install Raspbian to an SD card – Raspbian Lite is perfect for this project. During setup you will need a monitor and keyboard attached, but these can be removed once the system is installed in place.

Once the Raspberry Pi has started, run the raspi-config utility:

```
sudo raspi-config
```

Under 'Interfacing Options', enable the 1-Wire interface.

It is also useful to enable SSH at the same time to allow remote access to the Raspberry Pi so you don't need to dismantle everything if the code needs to be tweaked. Make sure you change the default password when you do this.

It's also at this stage you need to ensure the Pi can connect to your WiFi network so that data can be uploaded.

## INSTALLING THE REQUIRED PYTHON LIBRARIES

Additional Python libraries are required for the Blink! LEDs and for pygsheets, which will be used to upload the data to Google Sheets.

Installation of the Blink! packages is done by entering this Terminal command:

```
curl https://get.pimoroni.com/blink | bash
```

Installation of pygsheets is achieved by entering this command:

```
pip install pygsheets
```

## THE PYTHON SCRIPT

The full code can be downloaded from [hsmag.cc/issue4](https://hsmag.cc/issue4).

## EXPLAINING THE CODE

The script is made up of 7 key sections:

### PART 1

#### Import libraries

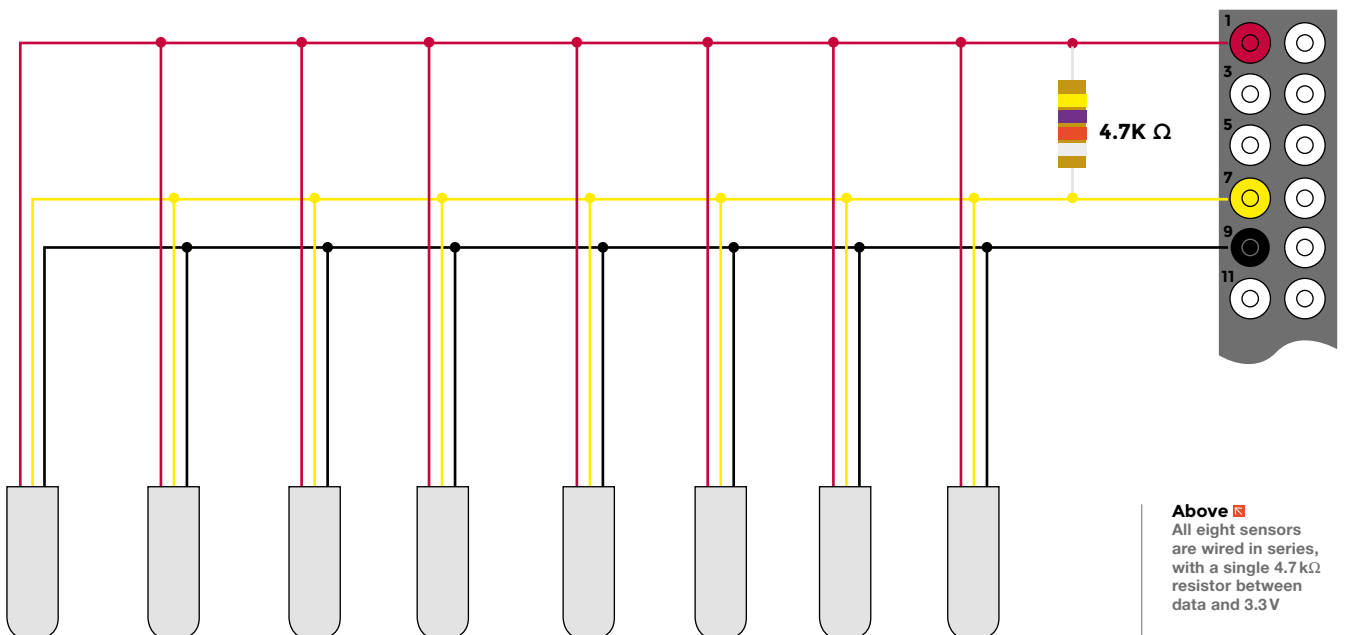
The non-standard functions are loaded.

### PART 2

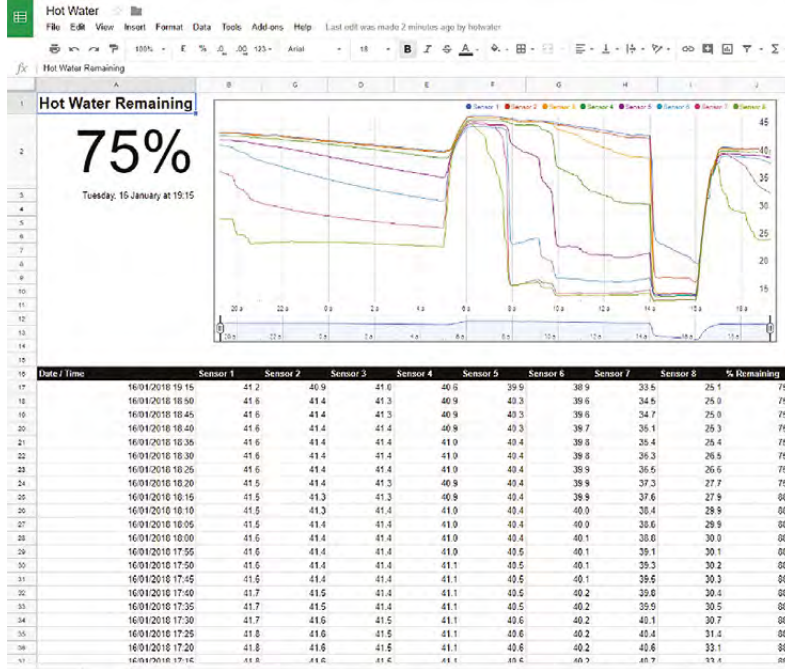
#### Key variables are defined

There are two areas that need to be manually customised for each project:

1. **temp\_max** is the temperature you consider to be fully hot, and **temp\_min** is the temperature where it's no longer hot enough – it may take a few days of experimenting to find the ideal settings for these.
2. Then there is a list of eight sensor addresses. These will be different for every project, so you must find the correct 64-bit serial numbers for your sensors and change them →



TUTORIAL



**Above**   
 With the data uploaded to a spreadsheet, it's possible to make charts of the temperature over time

as required. Sensor 1 should be the one at the top of the tank, and sensor 8 at the bottom.

The box 'Addressing the Temperature Sensors' will show how you can find the code number for each sensor.

**PART 3**  
Opening the sensor drivers

Starts the 1-Wire interface.

**PART 4**  
A subroutine to upload the data to Google

This is the code that will write the data to your Google Sheet. This is just a case of using the `update_sheet` method from the `pygsheets` module.

**ADDRESSING THE TEMPERATURE SENSORS**

Every sensor has a 64-bit serial number that identifies it. The digital data can be accessed through the Linux file system at `/sys/bus/w1/devices/<sensor address>`. In this folder is a text file called "w1\_slave". The temperature is the last few digits, in 1000ths of a degree C.

The data is added into a new row near the top of the sheet and the last row of data is deleted. Based on repeating the script every five minutes, this will capture 24 hours of data before being overwritten.

The first few rows are deliberately left blank to leave space for a chart.

**PART 5**  
A loop that runs eight times - once per sensor

This looping code is repeated eight times, once for each sensor. On each loop, it does the following:

- Reads the sensor text file. This is made up of two lines of text.
- Checks if the first line ends in 'YES' - if it does, then it knows the file has valid data.
- In the second line, it looks for an equals sign and then saves the numbers that come after. This is the temperature data in 1000ths of a degree C.
- The data is converted to degrees C.
- If the temperature is above the threshold value, then it's considered 'hot', and the variable `hw_left` is incremented by one. At the end of all eight loops, the total number of sensors above the hot water threshold value is converted to a percentage.
- Based on the temperature, the colour of the corresponding LED is set: to fully red if above `temp_max`, fully blue if below `temp_min`, or an intermediate shade of purple for temperatures in-between.
- Finally, the sensor address, temperature, and LED colours are printed to the screen. This is useful for troubleshooting.

**PART 6**  
Formatting the data into a tidy state

The percentage of hot water remaining is calculated and the time is converted to a useful format.

**PART 7**  
Calling the data upload subroutine

The final step is a call-back to the subroutine defined in Part 4. This writes the data to the Google Sheet.



## CHECKING IT WORKS

Once everything has been built, powered up and programmed, it's time to check it works. This is best done before installing in the airing cupboard and with the Pi Zero attached to a monitor and keyboard.

**Run the script using this command:**

```
python /home/pi/python/hotwater.py
```

You should see the information from each sensor being printed to the screen, followed by messages about the data being uploaded to Google.

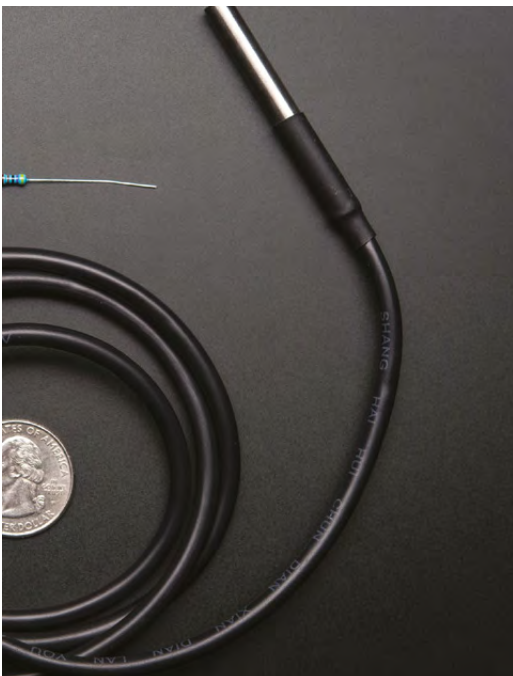
Once everything is working when you run the script manually, the final step is to have it run automatically at regular intervals. The nice thing is that the Raspbian operating system already has a built-in utility to do just this: crontab.

## CRONTAB IS YOUR FRIEND

Crontab is a Linux utility for automatically performing tasks – either at fixed times of the day, or at boot – but most importantly here, at regular intervals.

**Crontab is opened using the following command:**

```
crontab -e
```



**Above** ♦  
DS18B20 temperature sensors are available in waterproof packages like this or three pin packages for more confined spaces

## TROUBLE SHOOTING WHAT TO CHECK

**If you are experiencing issues, check the following:**

- The Google Sheet has the correct name and has been shared with the email address generated by the Google Developers Console
- The client\_secret.json file is in the correct place
- The codes for each of the eight sensors are correctly identified and entered into the script

You will see a text file in the standard Linux text editor. Scroll to the end, and on a new line add the following:

```
*/5 * * * * python /home/pi/python/hotwater.py &
```

This will automatically run the script every five minutes. If you want a different frequency, then change the number at the beginning.

You should now be able to open the Google Sheet on another computer and watch the data come in live every five minutes!

## THE FINAL STEP

When everything is working correctly on the bench, it's time to fit the system into the airing cupboard and the sensors to the hot water tank.

Starting at the mark made for the first sensor location, use a blunt tool (a wooden skewer is ideal) to poke a hole in the hot water cylinder insulation. Wiggle it slightly to enlarge the hole until it is a little smaller than the diameter of one of the temperature sensors. Make sure the hole has passed all the way to the copper wall of the internal cylinder and that the insulation is scraped away from the metal surface.

Put a blob of thermal paste about the size of a small pea on the tip of the sensor and push it into the hole until it makes contact with the cylinder wall. The insulation layer should hold it in place.

Once everything is in, make sure all sensors are reading approximately the same value when you know the water is fully hot (within around 1°C). If one is noticeably different, ensure there is no remaining insulation between the sensor and tank wall and make sure you have used enough thermal compound to fill any air gaps.

That's all there is to it. You can now monitor your hot water system from wherever you are and you'll never have to endure a cold shower again. □

## QUICK TIP

Only poke holes in the insulation with something soft that will not damage the internal cylinder (avoid anything made of metal).

# IoT: An internet-connected tea machine

Build a beverage-making robot that'll make you a cuppa on command

## WARNING!

This project involves mains electricity and boiling water.

Make sure you know how to handle these safely before starting.



Archie Roques

@archieroques

By day a humble A level student, by night a hardware engineer, Norwich Hackspacer, and general projects man. Also blogs at roques.xyz.



**T**ea is a favourite drink among those who lovingly put together HackSpace magazine – but these life-saving cuppas don't make themselves.

We have to literally use our hands and arms to make ourselves a nice cup of tea manually after a hard day's work – a Luddite injustice that feels as though it should be solved, in this new millennium of self-driving cars, AI-driven heating systems, and flashing computerised bobble hats. This project uses a Raspberry Pi Zero W as the brains behind a simple system to make the perfect cuppa just as soon as you say 'OK, Google'.

The system we've put together here uses a normal electric kettle, which is filled by a small peristaltic liquid pump, turned on for just the right amount of

time to pump a cupful of water. A relay then controls the kettle (the kettle switch must be already pressed down so that when the relay fires, the kettle turns on). Lastly, a small solenoid valve fitted to the side of the kettle opens to dispense the water into a waiting cup, with teabag. You have to add your own milk and sugar, but it's only 2018.

Once you've got all your components, the first step is to test your valve, and work out which way round it should go – solenoid valves typically work in one direction only. The easiest way to test this is by blowing through the valve and seeing if air comes out the other end, or alternatively pouring a little water in it. Hook it up to a (usually) 12V power supply to open the valve and try blowing/pouring again.

### Above

The finished product looks like something Professor Branestawm might've come up with, but it makes a tasty cuppa!

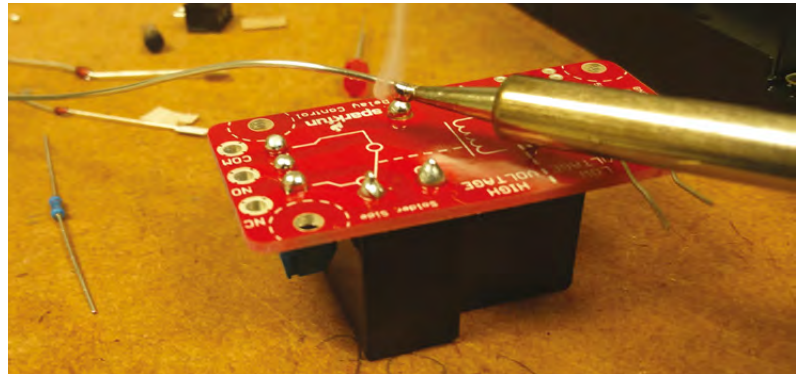
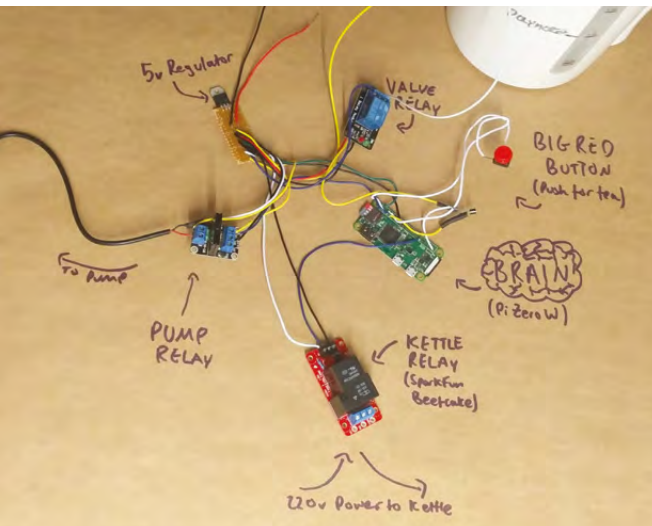


## TAKING THE (BEEF)CAKE

Once you've worked the valve out, it's time to do the same for the relays. You'll need to solder the Beefcake relay kit using the instructions on SparkFun's website at [hsmag.cc/RKjTYI](http://hsmag.cc/RKjTYI). It might seem a little counter-intuitive to use two different types of relay, but relays which can handle the full 13A a kettle draws whilst boiling aren't too common, and it's one place you really don't want to be cutting corners. You'll need to work out if the relays are active-high (if they turn on when their control pin is connected to 3.3V), or active-low (if they turn on when their control pin is connected to ground). Keep a note of which each one is – you'll need to use it when programming later.

The next step is to connect everything up according to the diagram on the following spread. You'll want to solder wires directly to the GPIO of the Pi Zero W (no header attached). Mount the 5V regulator on a piece of stripboard, and mark on which pins are for input voltage, output voltage, and ground (you can find this info by reading the data sheet for your regulator – search for the name of the part). You should also use a heatsink, and add capacitors to either side of your regulator to smooth the voltage in case of any spikes; the data sheet should have information on what's required. You'll need to solder the connections to the valve and pump, and use the screw terminals on the relay for the rest. Leave out the kettle wire to begin with, as it's easier to test it without the relay.

**Below** ♦ Colour-coding wires is a great way to simplify things when working on complex projects like this



**Above** ♦ The Beefcake Relay kit is good fun to assemble, but you'll need a meaty soldering iron to make sure there's a good connection on the relay terminals

Once you've connected everything up, the next step is to attach the valve to the kettle. The valve you use has a big impact on what bits you need to connect it – the best thing to do is go to a physical shop and try different bits until you have a setup you're happy with. We ended up using two 15mm nuts with rubber seals to connect the valve to the kettle, and a right-angle 15mm tap connector to add a small spout to the 'out' side of the valve, which directs the water into a cup – but it depends on the valve you get, and how big a hole you want to make in the kettle. It's a good idea to use metal fittings so you can be sure they won't melt.

Next, drill a hole in the side of the kettle to accommodate the fittings for the valve. A stepped-cone drill bit is the best way to do this. If your kettle is plastic, it's important to take things slowly and be careful because the plastic cracks incredibly easily, and if that happens it'll ruin the whole kettle. Once you've drilled the hole, fit the valve. Use a mole wrench or spanner to tighten up the nuts really well – but not too far or you'll risk cracking the plastic.

Once everything's connected up, it's time to program. Flash an SD card with Raspbian OS (there's a tutorial for this on the Raspberry Pi website at [hsmag.cc/UPSBIK](http://hsmag.cc/UPSBIK)), connect the Pi to a screen, power supply, mouse and keyboard (you'll need a USB hub and a few adapters), and boot up. Note that you absolutely mustn't connect the 12V power supply whilst you've got your Raspberry Pi connected to another power supply, or bad things will happen!

## DON'T BAKE THE PI

The first step is going to be testing all the relays and valves thoroughly. Fire up a Python IDE window (under Raspberry Menu > Programming), and then open a new window (File > New window).

Type the following code – if any of your relays were active low, you'll need to add the code `active_high=False` in the brackets after the pin →

## YOU'LL NEED

- ♦ Raspberry Pi Zero W
- ♦ A pair of low-voltage relays  
[boardsebay.co.uk](http://boardsebay.co.uk)
- ♦ A SparkFun Beefcake relay  
[sparkfun.com](http://sparkfun.com)
- ♦ A kettle (the simpler the better)
- ♦ A water container
- ♦ 2 V Solenoid valve suitable for 100°C use
- ♦ Small peristaltic water pump  
[hsmag.cc/xmHijt](http://hsmag.cc/xmHijt)
- ♦ Small push-button
- ♦ 5 V voltage regulator
- ♦ 12 V power supply (at least 3 A)
- ♦ A small piece of stripboard
- ♦ Plenty of wire
- ♦ The necessary plumbing fittings to connect your valve to a kettle and have it pour into a cup

### TAKING THINGS FURTHER OPTIONS FOR ENHANCEMENT

There are lots of ways this project could be improved. For example, if the water tank is empty it could lead to the kettle boiling dry which isn't a good idea – so perhaps a water sensor could be added to alert you when it's running out.

Another potential issue is that the system relies on the kettle being full to the bottom of the valve before operation (any liquid under the valve level won't be dispensed into the cup), so maybe a liquid level sensor could be used to ensure the kettle is full enough.

The system as it stands will make a cup of tea whenever it receives an email (which is open to hacks!), so it could be advantageous to refine the code to only make tea when emails from a specific address are received.

You could also add a system to remove the teabag from the cup after a period of time. There's only one thing worse than no cup of tea, and that's an over-brewed cup of tea!

It should go without saying that it's important to make sure all electronics are suitable for use in 100°C operation – if in doubt, don't use it!

number, as per the first example. Run the code. It won't do anything to begin with, but you can test each item by typing `pump.on()`, `pump.off()` etc.

```
import gpiozero
```

```
pump = gpiozero.LED(3, active_high=False)
```

```
valve = gpiozero.LED(26)
```

```
kettle = gpiozero.LED(5)
```

```
button = gpiozero.button(2)
```

All this code does is set up each relay using the `gpiozero` library, which makes it really easy to control physical hardware with a Raspberry Pi. We're using the `LED` class for the relays because although they aren't LEDs, it allows us to turn them on and off, which is all they really need to do. It also sets up the button. Make sure everything works before moving on.

Once you've got the relays worked out, the next step is to write an algorithm for making a cup of tea. It basically consists of turning on each component of the system for a length of time, one after the other.

```
def makeACuppa():  
    pump.on()  
    time.sleep(pumpdelay)  
    pump.off()  
  
    kettle.on()  
    time.sleep(kettledelay)  
    kettle.off()  
  
    valve.on()  
    time.sleep(valvedelay)  
    valve.off()
```

Variables are used here for the amounts of time, because it makes it easier to change the delay times later (there's a fair bit of trial and error involved in that stage!). When that code's done, add the line `button when_pressed = makeACuppa` to trigger the algorithm when the button's pressed. Run the code again and try it (without any water in the system). This code will work fine for simply making tea at the press of a button, but if we want our tea to be cloud-connected we need to add another block of code.

If This Then That (IFTTT) is a great service for IoT projects – it ties together all sorts of online services. However, it's not too easy to integrate the output of an IFTTT applet with a Python script without hosting a full-blown web server, and that's another level of complication that this project could do without – so the following steps use email as a way to get the internet to talk to the Raspberry Pi. Set up a Gmail account for your project with a unique email address (unless you want a cup of tea every time you receive an email). The code for this part is mostly the same as Adafruit's LED Email notifier project ([hsmag.cc/jGEDcx](https://hsmag.cc/jGEDcx)), so that's a good place to go if you're looking at making modifications to this code.


You'll need to install the `IMAPClient` library – use `sudo pip install imapclient` in the Terminal to do this. Then you'll need to modify Adafruit's code a bit: remove all the references to `RPI.GPIO` and `LEDs`. Replace the code in the `if newmails > NEWMAIL_OFFSET`: section with `makeACuppa()` and delete the `else` statement so it only makes you a cup of tea if a new email's detected.

You also need to replace the `NEWMAIL_OFFSET` variable with one called `prevMails`, and assign that to the number of new mails found every time the loop runs. This is because you want it to run whenever there's a new email since the loop last ran, not whenever you have any number of unread emails (a subtle but important distinction!).

Finally, copy and paste the contents of the loop above the code, so that when you run it, it checks the number of emails to begin with and assigns that.

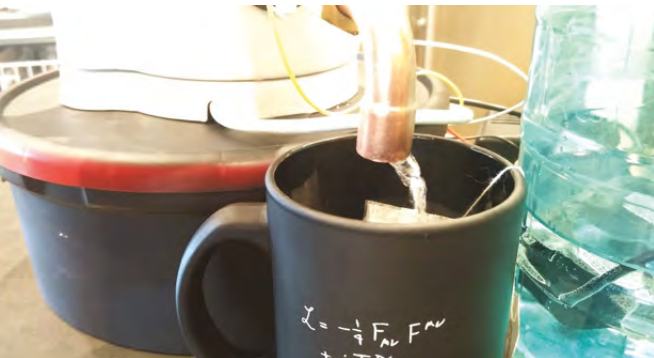
### QUICK TIP

The `rc.local` file works for auto-running scripts on the Raspberry Pi and is a useful thing to keep handy. There are also all sorts of other ways to do this including using `.bashrc`, `crontab`, and `init.d`.

**Below**  The kettle must be kept filled to the level just below the bottom of the valve







**Above** ♦ And there you have it: an IoT connected, Raspberry Pi Powered tea machine, complete with steampunk spout!

### YOU'VE GOT MAIL

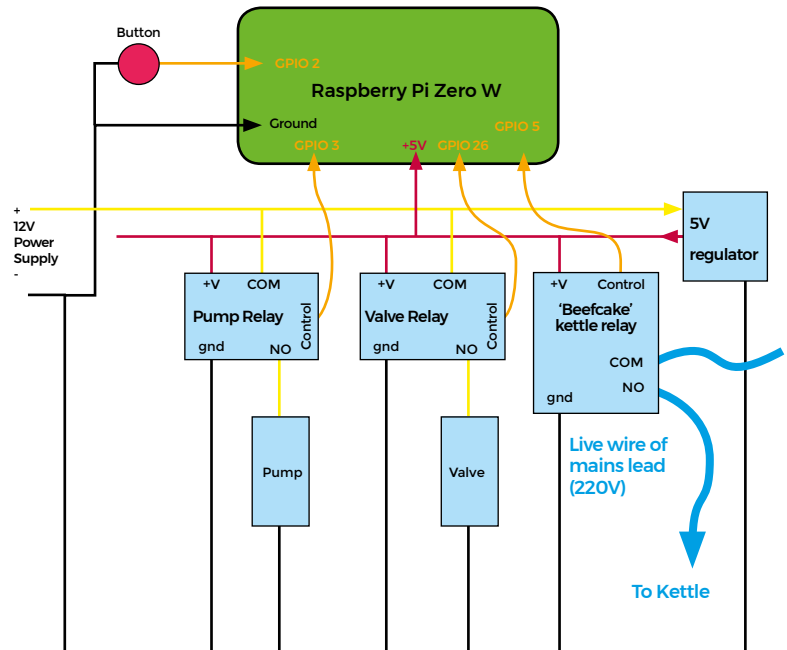
The full code for this project is available at [hsmag.cc/CZSPYm](https://hsmag.cc/CZSPYm), so cross-check your code with that when you're done editing. To make the code run as the Raspberry Pi starts up, use the Terminal to edit the file `rc.local`. Enter `sudo nano /etc/rc.local`, and add the line `sudo python /home/pi/IoTea.py &` at the end of the file, but before the `exit 0` line. If you saved your file somewhere else or under a different name, modify that line to point to your file.

The last code step is to refine the variables. This will likely get messy – make sure your electronics are well away from the setup and in a water-resistant place when you do this! There's a lot of trial and error involved. You need the value for the filling to be conservative, so there's no risk of the cup overflowing (but not too conservative or you'll have a glass half empty). If you can do this near your screen it saves a lot of time. Test it by putting the input tube in your water tank and the output in a mug. Then test the kettle. You'll need to fill it to the bottom of the valve, and then empty the mug of water in. Plug it into the mains, and time how long it takes to boil. Add a few seconds (we aren't bypassing the kettle's in-built switch so there's no need to be exact here – the kettle will turn off when it's boiled). The final step is to see how long it takes to empty the kettle. Again, you can add a few seconds on here. Change the variable values within the code, save it, and then shut down the Raspberry Pi.

### ALL ABOUT TIMING

Now you just need to wire up the kettle. Unplug it from the mains, and re-route the live (brown in the UK) wire to pass through the NO contacts of the Beefcake relay. Do not undertake this if you don't know what you're doing – in doubt, consult someone qualified! You'll also need to connect your 12V power supply to the regulator board if you haven't already done that.

Now it's time to test! Make sure the switch on the kettle is pressed down so that when the relay



activates, the kettle will actually turn on. Place a cup containing the teabag of your choice underneath the kettle's spout, and unplug the Raspberry Pi from your programming setup.

You'll need to make sure there's enough water in the kettle so that it just reaches the bottom of the valve. Plug the 12V power supply in, wait a couple of minutes for the Pi to boot up, and press the button. You should get a freshly made cup of tea!

Now the moment of truth – try firing off an email to the Gmail address you created earlier, and you should get another piping hot cuppa!

The final part of the puzzle is going to be to add the I in IoT. It's all very well being able to send off an email to make yourself a cuppa, but even that is a bit too much of a faff when you're desperate to quench your thirst after an evening's hacking. Set up an IFTTT account using the Gmail address you created earlier (it's important to do this with the specific email address – even if you've already got another account). Click the 'create a new applet' button, and pick 'Google Assistant' as the IF skill. Set it up with the Google account you use every day, and enter values so that you can ask it to make a cuppa. Then, for the THAT skill, select 'Email'. Enter whatever subject and content you like, as that's invisible to the code. Save the applet.

The final step is to package everything up into a case. We used an old chocolate tin, spray-painted black. This serves two purposes: keeping the electronics dry, and lifting the kettle above the cup so it pours out properly into the mug. Now you can make a cup of tea by simply talking to your phone! Welcome to the future – two lumps for us, thanks. □

**Above** ♦ The system is a complicated one, but well within the realm of the average maker

### OTHER BRAINS

Whilst this project uses a Raspberry Pi as the brain, there are lots of other boards you can use for IoT projects like this. An Arduino would work well and ESP8266 boards would be a good choice if you want to connect it to a network.

# Make your own living terrarium

Create a circular ecosystem in a jar and watch it grow



Jen Botezat

@neurogenerator

I am a DIY enthusiast with a passion for making. You can often find me in the garden, whether my own or a community one, as I especially love making them look beautiful. I volunteer at the Alpine & Rock Garden at Kew Botanical Gardens.

**T**errariums create a closed environment for plants, free from dust and pollution, with a constant level of humidity that allows tropical plants to thrive. Perhaps our fascination with terrariums is part of the larger movement to bring plants into our homes, especially when we have little space. In this tutorial, we will be making a tropical terrarium, inspired by the first one ever invented. You rarely need to water it, and you can watch it grow over many years as a brilliant living addition to your home.

## PEBBLE AWAY

Start with a glass container which you can seal. Here, we are using a demijohn (traditionally used as the vessel of choice by merchants storing wine and spirits). Carboys, boiling flasks, jam jars, vases, fish bowls, and even the odd light bulb make excellent options for a tropical terrarium.

Fill the container with a layer of pebbles for drainage. Make sure to wash the pebbles beforehand. A layer about 10cm-high works well for the demijohn. You want to leave space in the container for the compost and plants to grow. To make adding pebbles easier, make a funnel out of an old plastic pot (make a cut along one side of the pot, wrap it inwards, and insert it into the top of the demijohn).

## THROW IN CHARCOAL!

This is an important step that's often forgotten when making a tropical terrarium. Sprinkle a teaspoon

of activated charcoal powder over the top of the pebbles. Try not to get the powder on the glass walls. The charcoal helps purify the water as it circulates through the terrarium in a closed loop, and prevents the build up of unwanted moss and algae.

Fill the terrarium with a layer of regular compost or potting mix, about 20cm for the demijohn. A funnel is really useful for this step. Don't worry if you leave soil residue on the sides of the container – we will clean this at the end.

## WHAT TO GROW

For this tropical terrarium, you will want plants that are native to the tropics and thrive under canopy. Ferns (such as Button ferns and Boston ferns), ficus pumila, calathea, ivy, red or green fittonias, and the friendship plant are all excellent choices. Likewise, cushion moss will look chunky and lush in your terrarium. Select plants with different colours, textures, and heights to give your terrarium that extra wow factor. If the plants come too big, divide them by gently pulling apart and teasing out the roots.

Get your creative juices flowing as it's time to design. How you arrange the plants inside is what makes your terrarium unique. For this step, you will need to make some tools specifically for a terrarium. You can make most of these yourself. Make a cork tool and a sponge tool by attaching a cork and a small sponge onto sticks.

Now use the long stick to make a small hole in the compost. Poke the hole until you feel the pebbles below. This is where your first plant will go. Drop the plant down into the terrarium and carefully wriggle into place with the long tweezers or stick. Use the cork tool to tamp down the compost around the plant. Don't plant more than one plant in a single hole, as this can get fiddly. Rather, plant each piece one by one, making holes as you go along. Drop in the cushion moss and tamp down with the cork tool to position it.

## THE FIRST TERRARIUM

In 1829, entomologist Dr Nathaniel Ward put a Sphinx moth chrysalis and some leaf mould into a sealed glass container to watch it develop. Instead, a miniature fern grew inside. Ward concluded that the container maintained a constant humid environment, perfect for tropical plants, and so the 'Wardian Case' was born. Wardian cases allowed tropical plants to survive long sea voyages to the UK and spurred a fern craze in Victorian times. The fern motif began to appear everywhere, even on Custard Cream biscuits!





**YOU'LL NEED**

- ◆ Sealable glass jar
- ◆ Pebbles
- ◆ Activated charcoal powder
- ◆ Regular potting mix
- ◆ Small tropical plants and moss
- ◆ Spray bottle with water

**TOOLS**

- ◆ Make your own terrarium tools with a cork, sponge, and several long sticks
- ◆ Long tweezers and radiator brush (if you have them)
- ◆ One plastic plant pot (to create a funnel)

**DESERT TERRARIUMS**

You may have seen succulent and cactus terrariums. Such terrariums require an open container and dry, arid conditions that mimic their native desert habitat. Effectively, they require the exact opposite to the humid environment of a tropical terrarium. If you see a cactus or succulent planted together with a fern, it won't last long!

Once you have finished planting, use the sponge tool or radiator brush to wipe off any soil residue from the walls of your terrarium. A dry sponge or paper towel works well.

Terrariums don't do well when water is poured directly over them. Instead, use a spray bottle and spray inside five to seven times, going around in a circle. This should be enough to start off your terrarium. And voilà. Your terrarium is ready!

**KEEP AN EYE ON IT**

For the next two weeks, your terrarium will go through a hardening period. During this time,

alternately seal it with a cork for a few days and leave it with no cork for a few days. There is no strict rule for this, but doing so will help your plants adapt to their new environment and regulate the humidity levels inside the terrarium. Monitor for any signs of dying plants during this time and pluck them out or replace. If any are to die, you will see it in these two weeks. Water one more time after the hardening period and feel free to leave the terrarium alone, sealed, for a long while.

Now, you can sit back and watch the plants grow and evolve. A terrarium is never static, and you'll be able to watch it grow and change over time – the perfect green feature for your home.

Where you put your terrarium matters. Direct sunlight will cook the plants and too little sunlight will kill them too. Place the terrarium in a bright place, but out of direct sunlight. A windowsill is perfect for this. Also, keep the terrarium away from radiators and avoid overwatering. If the plants get too large and leggy, trim them back. Avoid letting them touch the glass. To clean the terrarium inside and out, use a dry cloth, and never use harsh chemicals. □

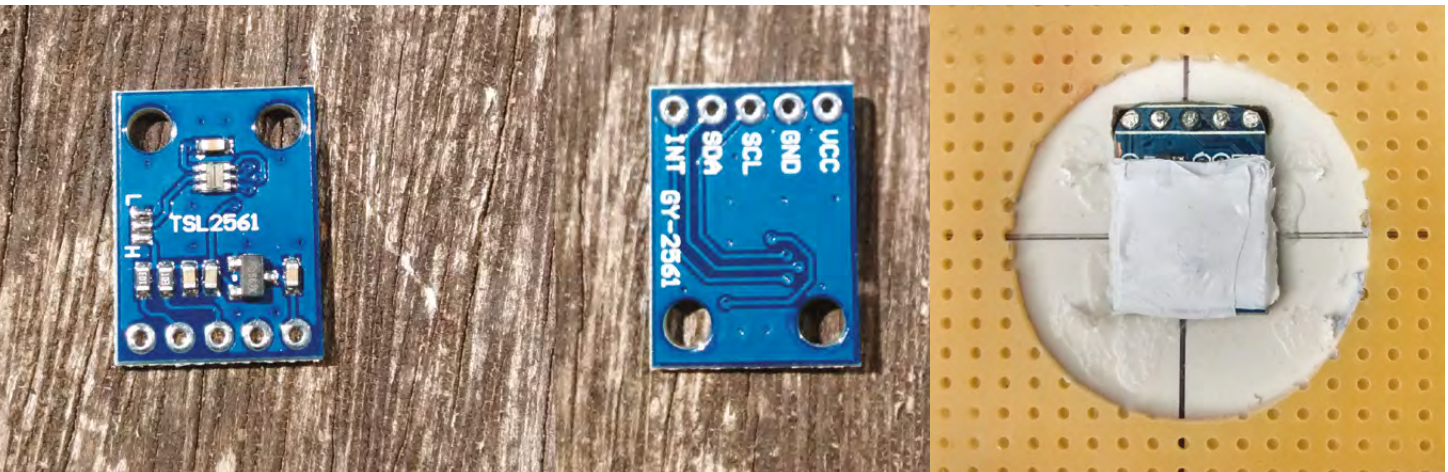
**Above** □ Think about the height of your plants for a 3D look


**Below** ◆ The tools you need can easily be made or improvised



# Monitor your solar setup in your browser

Use an ESP8266 to let you keep an eye on energy generation from any device



**Above**  Light sensor TSL2561 – back, front, and with PTFE filter in place



**Bill Grainger**

Bill Grainger is a physicist and design engineer who loves coding and having computers interact with the real world. He works with wind energy on all scales and enjoys running, sea kayaking, and mountaineering.

**T**he amazing ESP8266-based boards include all the hardware to connect to a WiFi network and it can be programmed to act as a web server.

Here we set one up to host a website, which displays the sunshine measurement on a simple dashboard. Any device, computer or smartphone, on the same WiFi network can view the data.

Light measurement is useful for assessing when to switch on extra light or monitoring the performance of a solar system. The TSL2561 light-integrated circuit is a sophisticated device designed for making indoor light measurements in lux units. Sunshine, or solar irradiance is usually measured in  $W/m^2$ . Proper solar sensors can be expensive, but this little unit can be modified to give reasonable performance at a low price.

The integrated circuit contains two sensors: one which responds to both visible and infrared light, and another which responds to infrared light only. For solar measurement, only the first sensor is needed, especially if the measurements are used to evaluate a PV system, as it responds to sunlight in a similar manner to a PV panel.

The TSL2561 communicates using the I<sup>2</sup>C interface, making it easy to connect to many small computer boards such as the those using the ESP8266, and of course the Arduino and Raspberry Pi. Modules containing this chip and the necessary ancillary components are widely available.

For this project, a NodeMCU V1.0, is used as the host computer. The NodeMCU is easy to use as it plugs into a USB port on the computer running the Arduino IDE.

## HARDWARE

I<sup>2</sup>C is a way of communicating over two data lines known as SDA and SCL. The sketch assumes that GPIO0 (pin D3 on the NodeMCU) is the SDA line, and GPIO2 (pin D4) is used for the SCL line. The TSL2561 module also needs GND (0V) and VCC (+3.3V) connections to operate.

To start with, the sensor and the NodeMCU are mounted on a breadboard and the two connected with jumper wires.

The maximum length depends on the connecting cable capacitance, the clock speed, and the pull-up resistors. In most cases a separation between the



computer board and the sensor of a couple of metres works fine.

## SOFTWARE

The simplest way to program ESP8266 boards like the NodeMCU is via the Arduino IDE. The core Arduino IDE package must be extended with a board manager to work with the ESP8266. Many sketches and libraries developed for the Arduino series of computers can be used on the ESP8266 – a huge advantage, given the wealth of Arduino applications. The sketch (Arduino program) relies on a library written to access the TSL2561 sensor by the people at SparkFun. Setting up the Arduino IDE and loading the SparkFun library is described in the box on page 106.

## HANDLING THE SOLAR SENSOR

First attach the libraries needed for the WiFi, the solar sensor, and the I<sup>2</sup>C interface.

```
#include <SparkFunTSL2561.h>
#include <Wire.h>
```

Next, enter the calibration for the sensor. The number, 3000, in the code is for indoor light for initial investigation. Once you have calibrated your sensor, change this to suit.

```
float solcalib=1000.0/3000;
```

Tell the ESP8266 how the I<sup>2</sup>C hardware is connected.

```
#define SDA 0 // GPIO0 / D3
#define SCL 2 // GPIO2 / D4
```

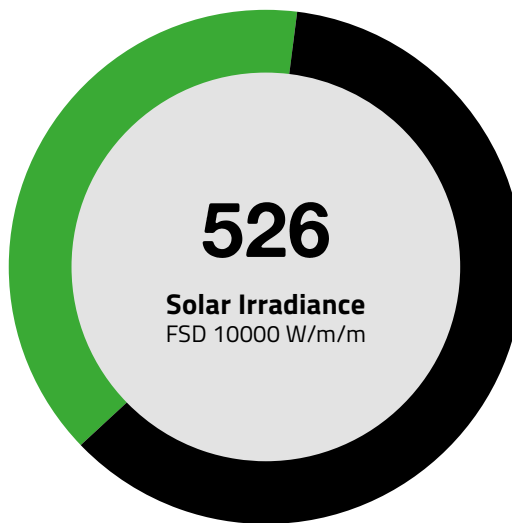
Create an instance of the light sensor and set the highest reading that will be likely.

```
SFE_TSL2561 light;
const float solarMax = 1000.0;
```

This next section is run once from the main `setup` function and collects together all the instructions required to set up the solar sensor.

```
void setup_solarirradiance() {

    boolean gain = 0;
    unsigned int sitime;
    unsigned char time = 0;    // time 13.7
    milliseconds
    light.begin();           // Initialize the
    TSL2561
    delay(1000);
    light.setPowerUp();
}
```



**Figure 1**  ESP8266 website for solar data

The following function is used to take readings from the sensor.

```
float reading_totalsolar() {
    unsigned int sdata0, sdata1;
    const float error =9999.0;
    if (light.getData(sdata0,sdata1))
    {
        return solcalib*sdata0;
    }
    else
    {
        return error;
    }
}
```

## NODEMCU PHONE HOME

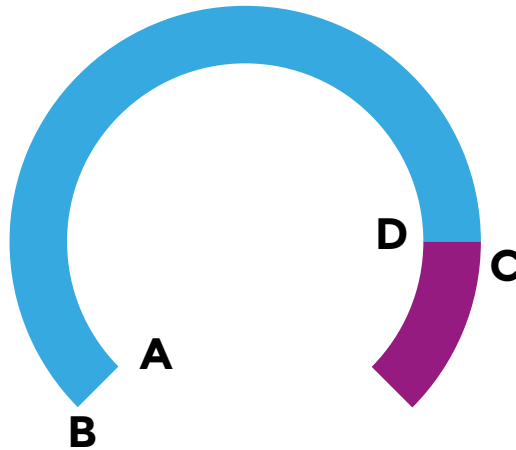
Setting up the WiFi on an ESP8266 device is straightforward, but you will have to enter details of your WiFi network to make this section work. In particular, you will need to enter the network name in the `ssid` variable and the network password in `password`. The other settings may should suit most networks, but if your router has a different IP address or the address 192.168.0.186 is already assigned, you will need to change the other settings.

(If you are unsure what IP address your router has, try using the incredibly useful, free smartphone app, Fing, to see all the devices on your WiFi network and their IP addresses.)

```
#include <ESP8266WiFi.h>

const char* ssid = "*****"; // Wi-Fi network
const char* password = "*****"; // Wi-Fi password
IPAddress ip(192,168, 0, 186); // static IP
IPAddress gateway(192,168,0,1); // router IP
IPAddress subnet(255,255,255,0);
```

Figure 2  
Dashboard  
arc labelling



The WiFi connection is started, and a static IP address assigned in the following section.

```
void startWiFi(void) {  
  Serial.print("Connecting to ");  
  Serial.println(ssid);  
  WiFi.begin(ssid, password);  
  while (WiFi.status() != WL_CONNECTED) {  
    delay(200);  
  }  
  WiFi.config(ip, gateway, subnet);  
}
```

An instance of a WiFi server is created to receive requests to access the website from other devices, clients.

```
WiFiServer myserver(80);  
String req;
```

### FEELING RESPONSIVE

The sketch waits for web requests to be received over the WiFi and then returns the webpage with the latest data from the sensor (Figure 1).

The dashboard is drawn by drawing a black background circle, an arc with angle related to the value to be displayed, a silver circle on top with the value displayed as a number, and finally a text title. The complicated bit is drawing the correct arc. This involves calculating the screen co-ordinates of the four vertices of the light blue arc, shown in the diagram (Figure 2).

A full arc will have an angle of 270 degrees.

The webpage is created by printing lines of HTML and SVG code to the client. HTML instructions mainly cover setting up a webpage, layout, and text. To produce interesting graphics, an extension like SVG is required.

Each time the solar web server receives a valid request from a client, it calculates the co-ordinates of the arc on the dashboard and produces the image. SVG files are a text-based XML format, so we can generate them easily in the code.

The main SVG instructions used are circle, path with the arc(A) setting, and text.

In the sketch, when a double quote (") character has to be sent to a client using the `client.println()` instruction, it must be preceded by `\`, or it will be interpreted as the end of a string. This makes the code more complicated to read. For example, the first line in the SVG creation is sent with:

```
client.println("<svg version='1.1' xmlns=\n\n    \"http://www.w3.org/2000/svg\"\n    height='\"\"' +\n\nString(3*centreOffsetY)\n\n    + \"\\\" width='\"\"")
```

To allow it to be easily adapted for other applications, the webpage drawing code is split into two sections: the first is application-specific, and the second a generic part which draws the dashboard.

In the application-specific part, the message that a client will use to request information, the units of the variable being displayed, the title for the display, and the full-scale value are specified. In addition, it needs a function called `dashCollectValue()` which will return the value to be displayed. This value should be between zero and the full-scale value.

The dashboard is drawn in the `watchForMessage()` function. It starts by responding to contact from the client over the WiFi. The dashboard creation section takes the value to be displayed, calculates the co-ordinates of the arc, and then 'prints' the appropriate HTML and SVG code to the client. This section is not reproduced in full here, but detailed

## INSTALLING THE ARDUINO IDE FOR ESP8266-BASED BOARDS

The Arduino IDE runs on PCs using the Linux, Windows, and Mac OSX operating systems. Go to the Arduino site [arduino.cc](http://arduino.cc) and download the appropriate version for your PC.

To set up the Arduino IDE for ESP8266 boards, follow the straightforward instructions at [hsmag.cc/hHqHDU](http://hsmag.cc/hHqHDU).

Once this is complete, select the correct board in the IDE by using Tools > Board and choosing NodeMCU 1.0.

The TSL2561 library can be directly loaded in the Arduino IDE:

1. In the Arduino IDE, select Sketch > Include Library > Manage Libraries.
2. This will display a new window. The IDE will contact a server to update a reasonably comprehensive list of libraries available. Once this has finished, type TSL2561 in the filter box to find the library for this project; the TSL2561 one from SparkFun is the required one.
3. Click More Info, select the latest version, and then Install.



explanation is included with the complete sketch at [hsmag.cc/IDAiDE](http://hsmag.cc/IDAiDE).

```
void watchForMessage() {
  WiFiClient client = myserver.available();
  if (client) {
    while (client.connected()) {
      if (client.available()) {
        String req = client.readStringUntil('\r');
        client.flush()
        if (req.indexOf(messageTag) != -1) {
          value = 100.0/dashFSD *
dashCollectValue();
          .....
          // using the coordinates create the web
page
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          .....
        }
      }
    }
  }
}
```

The standard `setup` function calls the specific setup functions for the serial monitor, the I<sup>2</sup>C interface, the WiFi connection, the web server, and finally the solar sensor.

```
void setup()
{
  Serial.begin(115200); // used for testing
  delay(10);
  Wire.begin(SDA,SCL); // set up I2
  startWiFi();
  setupWebServer(); // start the web server
  setup_solarirradiance(); //start solar sensor
}
```

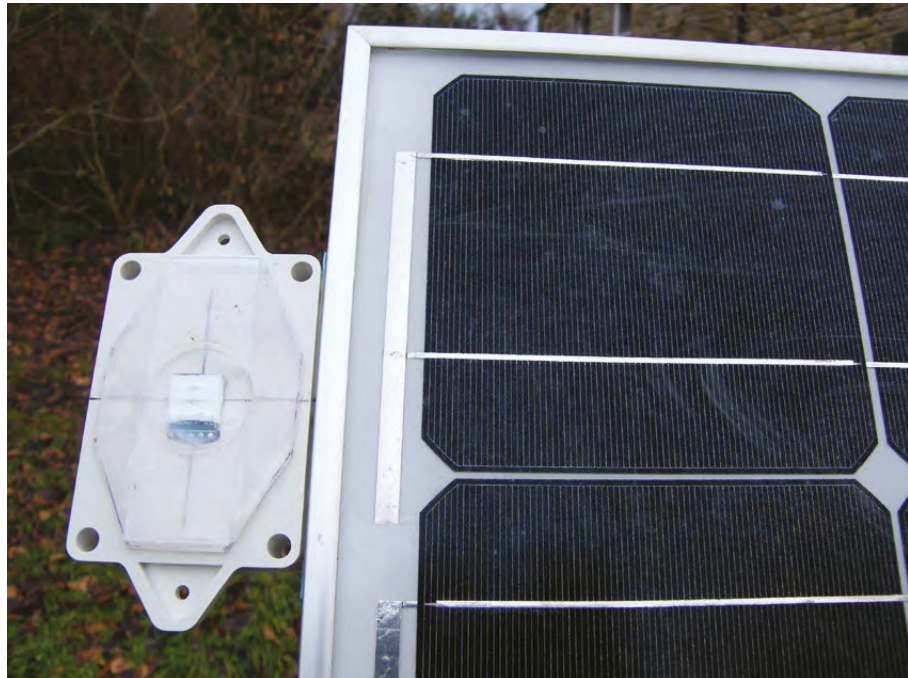
The main control loop is very simple, just continually calling the function watching for a request from a client over the WiFi.

```
void loop()
{
  watchForMessage();
}
```

The sketch, `IOT-solarWeb.ino`, contains additional comments to aid understanding, and can be downloaded from [hsmag.cc/VmLxHn](http://hsmag.cc/VmLxHn).

### PUT ON THE SUNGLASSES

As the TLS2561 is designed for indoor use, it can be overloaded in bright sunshine and a filter is required to



reduce the light falling on the sensor. PTFE is a good filter material, as it reduces the light across all relevant wavelengths uniformly. A cheap source of PTFE is the tape used for sealing threaded joints by plumbers. This tape is non-conducting, so it can be wrapped around the module. The thickness of the tape is not always the same, so it is difficult to be precise about how many layers are required.

Set up the device and run the `IOT-solarWeb.ino` sketch detailed above, as you add layers of tape. The Serial Monitor in the Arduino IDE will display the raw counts, while a client accesses the solar web server. The maximum count returned is 65535 and is returned when the device is overloaded. Wrap layers round the sensor to reduce the count in bright sunshine to less than 32000.

With the sensor mounted in its final configuration, note the maximum count,  $C_{max}$ , when it points directly at the sun in a clear sky, around midday. The solar irradiance under these conditions will be approximately  $1000 \text{ W/m}^2$ ,  $I_{max}$ . This value and the maximum reading provide the calibration:

$$I_{max} / C_{max}$$

This is used in the sketch so that output will be in  $\text{W/m}^2$  and should be the value assigned to the constant `solcalib`.

To see the solar dashboard, on a device connected to the same WiFi network, type:

```
192.168.0.186/ESPolar
```

...into the address box of a web browser. You should now see the dashboard. The display will update every second. □

**Above** ♦  
A sensor mounted outside, protected by a glass cover

# Build a variable power supply

Get electrons into your circuits with the correct energy.



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.



Figure 1

All the elements are packed in a practical case that can be easily carried around

**If you are tinkering with electronics, soon or later you will need to power your projects with different DC voltage levels.** We show you how to easily build your own variable power supply.

When starting to play around with electronics, many people power their prototypes directly from controller boards, such as an Arduino. However, as the prototypes become more complex (e.g. with the addition of servos or motors), they soon require more power than these controllers can safely deliver. At this point, you're going to need a variable power supply that can provide different voltage levels. Of course you can buy such a power supply, but it's a piece of cake to build one yourself.

A good starting point to build a variable power supply is a laptop power adapter. These adapters normally provide relatively high output voltage and

ampage. So just rescue an old laptop adapter from gathering dust at home, and you're ready to go. The adapter used in this project provides 19 VDC and almost 2 A, more than enough to power most of the prototypes you'll ever work with.

## ONE STEP BEYOND

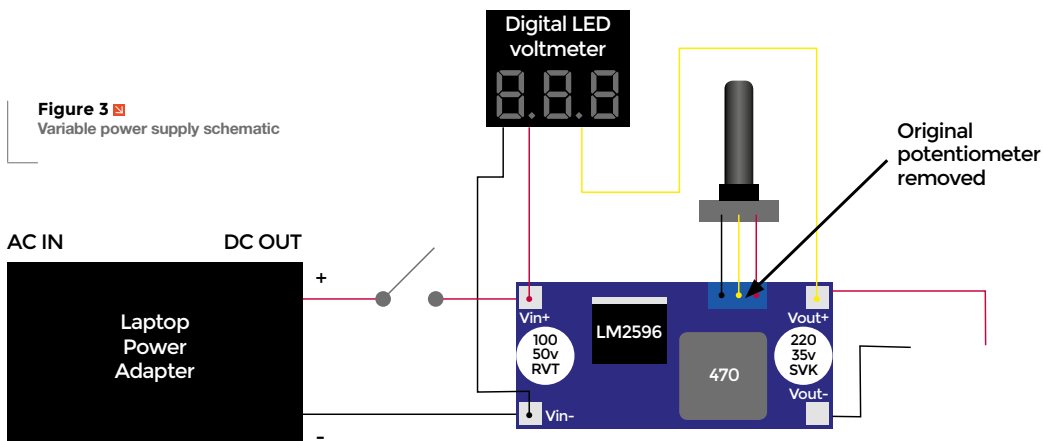
As the laptop adapter provides a fixed amount of output voltage (19 VDC in this case), you will need some device that "does the magic" of reducing the DC voltage to lower levels. The best option for this purpose is a DC-to-DC voltage reduction module that includes the LM2596 DC-to-DC step-down converter. It costs about £2 and supports an input voltage of up to 35 VDC and 3 A. This module comes with a tiny 10 kΩ potentiometer for regulating the output voltage, which can vary between 1.25 VDC and 35 VDC (19 VDC in this case, due to the laptop adapter

## QUICK TIP

Depending on the specifications of your laptop power adapter and the DC-to-DC module, the minimum amount of output voltage will vary. This means that you won't be able to obtain voltages below ~1.25 VDC.



**Figure 3** Variable power supply schematic



**YOU'LL NEED**

- ◆ Plastic enclosure (or your own hand made case)
- ◆ Laptop power adapter
- ◆ DC-to-DC voltage reduction module
- ◆ Digital LED voltmeter
- ◆ 10 kΩ potentiometer
- ◆ Potentiometer knob cap
- ◆ Switch
- ◆ Banana plugs
- ◆ Banana cables
- ◆ Wires
- ◆ Heat shrinkable cable sleeves

limitation). You should first carefully desolder the tiny potentiometer and then solder back a regular 10 kΩ potentiometer, which can be handled easily from the exterior of the power supply enclosure.

After removing the plastic case of the laptop power adapter, solder its output wires to the input of the DC-to-DC voltage reduction module through a switch, so that the power supply can be turned on and off.

A digital LED voltmeter (again something that you can get for £1–£2) is connected to the DC-to-DC voltage reduction module and shows the selected output voltage to the user. The voltmeter used in this project is able to provide a readout of voltages from 2.5 to 30 VDC with 1% accuracy.

Now it's necessary to fit everything into an appropriate enclosure, to have all the components safely connected and be able to transport the power supply wherever you fancy. The DC-to-DC module can just hang from the wires to which it's soldered, but be careful to isolate it properly if you use a metallic enclosure. To fix the laptop adapter to the enclosure,

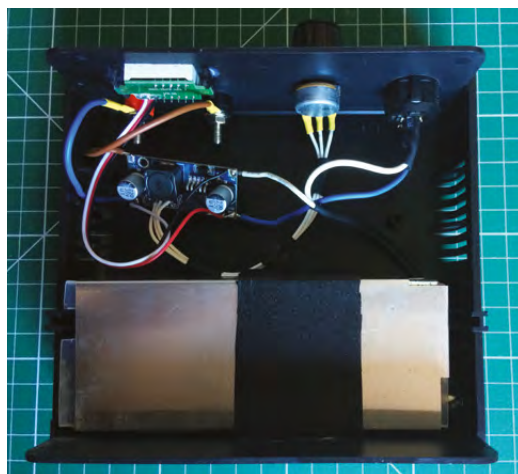
**CONTENTS MAY BE HOT**

Safety advice: be careful handling the DC-to-DC voltage reduction module while it's functioning, as some of its components (especially the DC-to-DC step-down converter) may become very hot. Make sure that it says cool enough not to damage your enclosure, and that there's nothing that could catch fire.

a good idea is to cut a piece of its original plastic case (they will have the same size and thus fit tightly) and use it as a bracket, glueing it to the enclosure.

The final step is to install all the external elements on the front panel: switch, potentiometer, digital LED voltmeter and banana plugs (to provide the output voltage to the banana cables), as well as the laptop adapter 220 VAC input on the back or on one side of the case. As a finishing touch you can add a potentiometer knob cap, so it's easier to turn it to select the desired output voltage and, why not, it looks much nicer! □

**Figure 2** Heat shrinkable cable sleeves are highly recommended to avoid undesirable electrical contacts



**Below** Everything wired up and ready to power your next project



# 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





# FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG  
120

## CAN I HACK IT?



Tear-down a radio-controlled tracked vehicle for fun and japes

PG  
122

## DIRECT FROM SHENZHEN: TESLA COIL SPEAKER

This awesome (though fragile) electronics kit makes sound from lighting

## REVIEWS

124 LilyPad kit

125 OKAY Synth DIY Kit

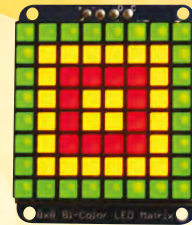
126 Adafruit Circuit Playground Express



PG  
112

## BEST OF BREED

Turn digital data into something nice to look at with our pick of the best display units



128 Sonoff Basic WiFi Smart Switch

129 Theo Gray's Mad Science



ONLY THE BEST

# The ever-evolving world of displays

Visualising data in your next project

By Marc de Vinck

@devinck


**Y**ou might think that all displays are the same. And you also might be thinking of those typical 16x2 character displays with either a blue or green tinted screen that so many of us got our feet wet with early on in our electronics building adventures. And although those types of displays are very useful, and easy to find for a minimal cost, there are dozens of other display technologies that are available to creators and tinkerers that are a lot more interesting.

So how many varieties could there really be for you to choose from? It seems like every week we see a new technology or form factor, many of which inspire us to think about how we could incorporate them into the next project.

Before we look at some of our favourite types of displays, and why we like them, let's look at what

we mean by 'display'. Simply defined, when talking about your electronics project, a display is a visual representation of data. It could be something as simple as a one-colour LED or mechanical device that notifies the user of an action or alert. In many cases this might be a good solution. In fact, many times a project can be trimmed down and only use a simple RGB LED as a very functional display.

In this roundup we'll be sharing some of the best in class graphical and character displays for those projects that need more than just a simple blinking LED. We've selected a cross-section of both old and new technologies from a variety of companies.

**Below**   
A classic 16x2 character display that most of us have used at some point

**Credit**  
Adafruit licensed under CC





# Adafruit 32×32 RGB LED Matrix Panel

ADAFRUIT ◆ \$39.95 | [adafruit.com](https://adafruit.com)

**S**ometimes bigger is better, or rather, sometimes you just need a larger display for your project. Think really large, like Times Square-TV large. Well, that's exactly what these panels could be used to build. You can find them all over the place in urban areas in the form of large-scale video advertising displays. They can be chained together to create really large animated, and full colour, displays. That being said, there are a few considerations when it comes to creating large video screens.

As displays get bigger, especially when thinking about full colour display, the prices can skyrocket. However, the Adafruit 32×32 RGB LED Matrix Panel still comes in at a respectable \$39.95. That's not too bad for 1024 RGB LEDs in a 7.5" × 7.5" form factor. Yes, it gets expensive if you need

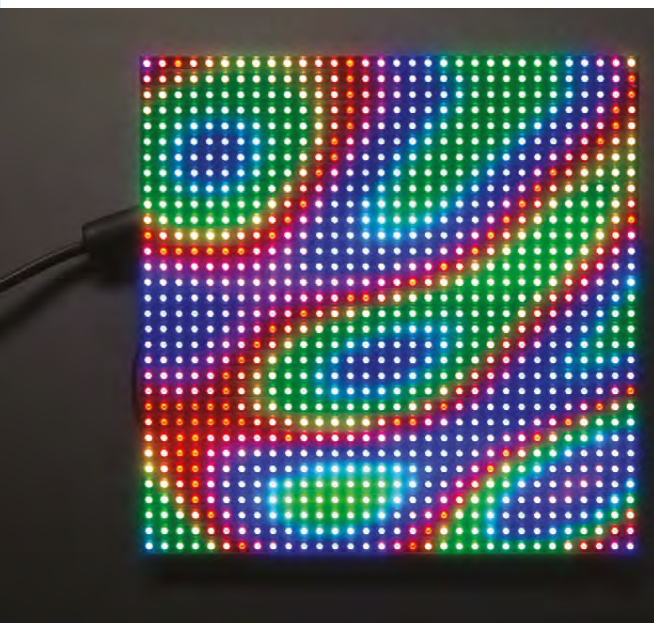
dozens of them, but in many instances one will work just fine.

Something else to consider when using large displays is the horsepower required to drive them – or, in our case, the speed that is needed to control them effectively. These particular panels were designed for high-speed FPGAs or similar processors. They rely on the image being drawn over and over at a high speed, something that a typical microcontroller can't do very well since a lot of data needs to be processed quickly. Your best bet is to use a Raspberry Pi with an RGB Matrix HAT or Teensy microcontroller, rather than the less powerful Arduino microcontroller.

Another thing you'll need to consider is the power required for the panels. Fortunately, the drivers are built into the back of the panel, and the power is low voltage, requiring only 5 volts and 4 amps when all the LEDs are on full power. There are plenty of guides on the Adafruit website to help you get started quickly, no matter what controller you choose.

One of my favourite projects that uses this display is SmartMatrix Animated GIF Player by Louis Beaudoin. It uses a microSD card and a Teensy to play back animated GIFs on the 32×32 RGB display. With the simple addition of a translucent covering, the pixels are blended and smoothed out, which looks great with the looping animated GIF. You can check out the complete tutorial at [hsmag.cc/sBBbNS](https://hsmag.cc/sBBbNS).

Another cool project that uses this display is the Raspberry Pi LED Matrix Display by Tony DiCola; [hsmag.cc/crohbf](https://hsmag.cc/crohbf). In this build, Tony goes over how to use the `rpi-rgb-led-matrix` library, a Raspberry Pi 2, and a few other components to play low-resolution games, movies, and images. And although this won't compete with your new 4K TV, it still has a redeeming retro charm. □



**Left** ◆ If the 1024 RGB LEDs still aren't enough for your project, you can easily add more panels to create larger displays

**Credit**  
Adafruit licensed under CC

## VERDICT

When you need a large display, this is the board to consider.

9/10

# Micro Dot pHAT vs Adafruit Bicolor LED Square Pixel Matrix with I2C Backpack

PIMORONI ◆ \$25 | [shop.pimoroni.com](http://shop.pimoroni.com)

ADAFRUIT ◆ \$15.95 | [adafruit.com](http://adafruit.com)

**W**hen it comes to displays, it's hard to beat the look of something retro. Yes, modern LCD touchscreens are super-cool, but they tend to be a little boring, at least visually.

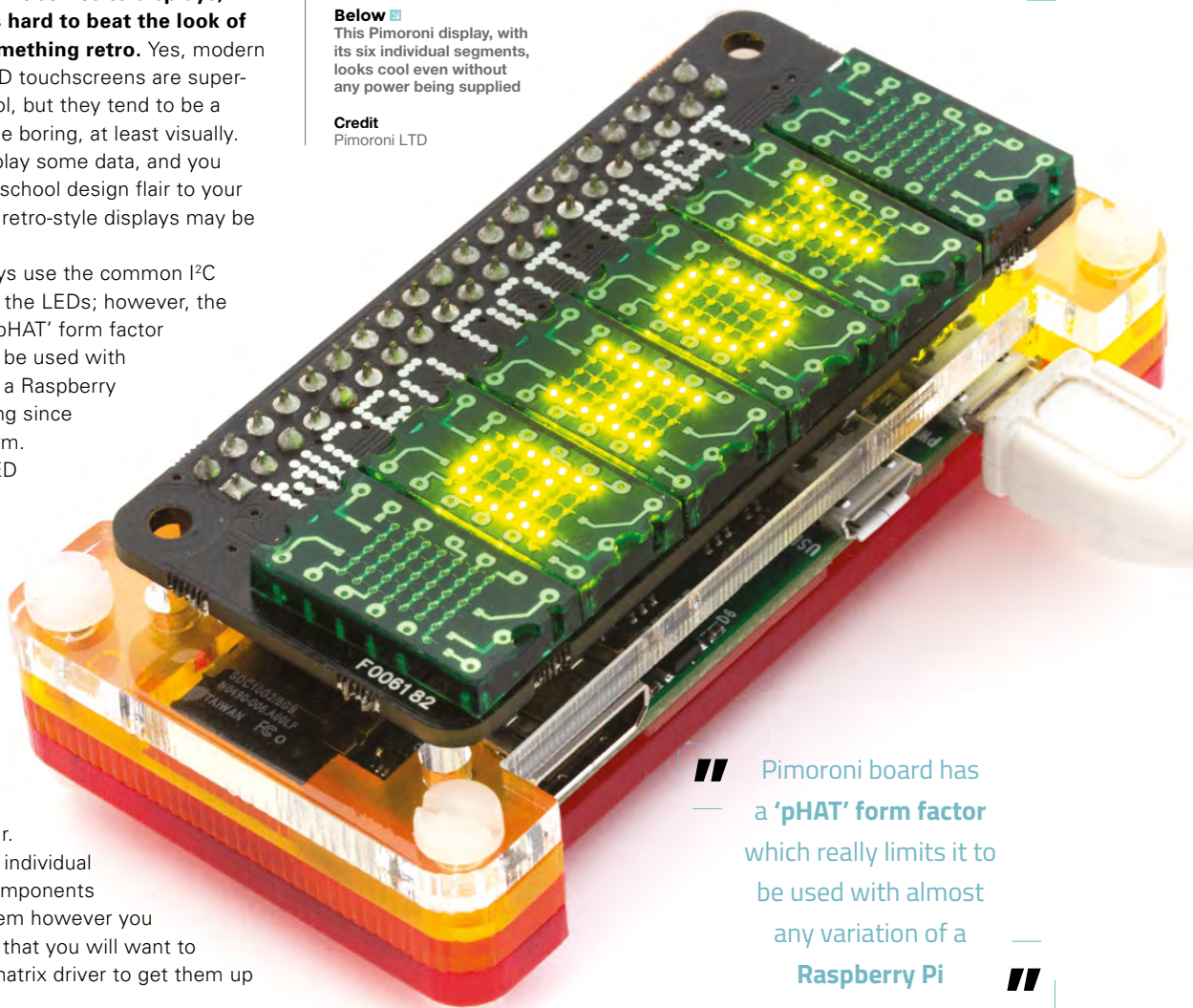
When you need to display some data, and you want a little added old-school design flair to your finished project, these retro-style displays may be your answer.

Both of these displays use the common I<sup>2</sup>C protocol for controlling the LEDs; however, the Pimoroni board has a 'pHAT' form factor which really limits it to be used with almost any variation of a Raspberry Pi – not such a bad thing since it's such a great platform. The Adafruit Bicolor LED Square Pixel Matrix is platform agnostic, which allows you to easily use it with whatever controller your project requires, including Arduino. If you aren't using a Raspberry Pi and you really love the look of the Micro Dot pHAT, don't despair.

Pimoroni also sells the individual LTP-305 LED matrix components so that you can use them however you like. Just keep in mind that you will want to implement a suitable matrix driver to get them up and running.

**Below** ▢  
This Pimoroni display, with its six individual segments, looks cool even without any power being supplied

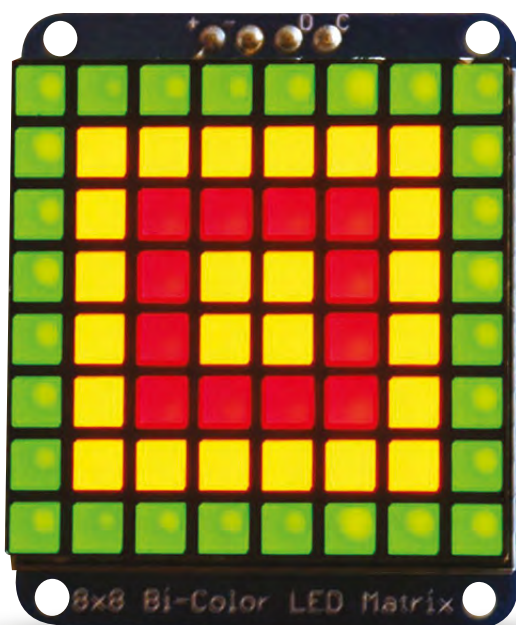
**Credit**  
Pimoroni LTD



**||** Pimoroni board has a 'pHAT' form factor which really limits it to be used with almost any variation of a Raspberry Pi **||**



// The Adafruit Bicolor LED Square Pixel Matrix also includes a library to make it easy to use with an Arduino



Both displays have great tutorials and source code for using them with your Raspberry Pi. The Adafruit Bicolor LED Square Pixel Matrix also includes a library to make it easy to use with an Arduino. And both displays have plenty of well-documented projects that you can find online from a simple search.

One other thing you should be aware of with these boards is that when scrolling text or a symbol, the spacing of the pixels on the Micro Dot pHAT can make it a little more difficult to read compared to the Adafruit Pixel Matrix. If you aren't planning on doing any scrolling, then it's really not an issue. And if you still want the look of the Micro Dot pHAT, we found that you just need to scroll the letters a little slower compared to other LED matrices and it does just fine. You could also diffuse the pixels a bit by adding a translucent piece of plastic [gasp!], but that would completely kill the beautiful aesthetics of the Micro Dot's segments.

## NIXIE TUBES AFTER ALL THESE YEARS

The 1950s were an amazing time when it came to innovation, from the invention of the first solar cell and hovercraft, to the modern plastic hula hoop and integrated circuit. It was also in the 1950s that the Nixie tubes were manufactured by the Haydu Brothers Laboratories. And although you may have thought this technology would have disappeared with all the modern displays available, you'd be wrong – at least it's still going strong within the electronic hobbyist community.

Nixie tubes typically consist of ten cathodes in the shapes of numbers 0-9 that emit a beautiful glow when about 150–200 volts are applied. It's a fairly simple technology to understand, but take one look at the workings and you will quickly see that they are a small marvel of manufacturing. If you want the ultimate in retro-chic for your next project, look into Nixie tubes – just be careful when using such high voltages!



At the end of the day, both of these displays are really nice, but we give the edge to the Adafruit Bicolor LED Square Pixel Matrix since it is compatible with more microcontrollers and has the capability to display three colours. Yes three, not two as you might think, since you can display both red and green LEDs at the same time, giving you an orange glow. It was a really close call since they both function so well and are so easy to implement.

Which one you choose really boils down to what you need for your project. If you really want a beautiful-looking display, even when it's powered down, choose the Micro Dot pHAT. The ability to see the connections of each segment is really beautiful. If you need multiple colours in one display, or if you want to animate or scroll letters or simple graphics, pick up the Adafruit Bi-Color Matrix. Or better yet, neither one is too expensive, so do yourself a favour and add both of them to your lab, you won't regret it. □

**Far Left** ♦  
Adafruit's bicolour display is ideal for creating a retro look for your project

**Credit**  
Adafruit licensed under CC

**Left** ♦  
A bit more challenging to work with, but the look of a Nixie tube is hard to match

**Credit**  
Hiroyuki Takeda licensed under CC BY-ND 2.0

## VERDICT

Micro Dot pHAT

Amazingly beautiful retro fun from all angles.

8/10

Bicolor Backpack

When you need a retro-looking display with more than one colour, this bicolour matrix is perfect.

9/10

# Spark X Flexible Grayscale Display

SPARKFUN ◆ \$49.95 | sparkfun.com

**T**here are times when you are working on a project and you need a display that not only works well, but it needs to be flexible. If you find yourself in this situation, the Spark X Flexible Grayscale Display from

SparkFun could be a perfect fit. Most displays are mounted directly to a ridged PCB, or the screens are mounted to glass, which makes flexing them impossible, never mind wrapping them around a tight radius. This OLED doesn't have either of those constraints, so it can be bent in a 40 mm radius without any damage and will still look great.


The Spark X Flexible Grayscale Display still has a ridged PCB carrier board, which is where the communication and power circuitry can be found, but the OLED screen is attached to it via a small flexible PCB which allows it to be mounted fairly far away from the ridged part of this display. It would be great if the entire component was flexible, but this is a good solution until the control technology catches up to the advancing display technology.

What we really like about this particular OLED display is the ability to show off 4-bit grayscale images on its 160x32-pixel display using a standard

“ The Spark X Flexible Grayscale Display still has a ridged PCB carrier board which is where the communication and power circuitry can be found ”

three-wire SPI interface. SparkFun has created a library making it very easy to get up and running with an Arduino. It has also released the schematic and Eagle files, so you can modify them as needed to work in your next project. The images displayed are clear and seem to magically appear on this impossibly thin 0.5mm display. And if your project requires just standard text to be displayed, this screen will render it fairly well and with the great contrast that most OLED screens are capable of generating.

Even if you don't need the flexibility this screen has to offer, it may still be worth looking into since it features a nice-looking OLED screen, a well-documented library, and plenty of examples to get you going quickly. And, if you do need your display to flex around a tight radius, it's really hard to beat the Spark X Flexible Grayscale Display from SparkFun. □

**Below**  This display can handle more than just plain text: it features the ability to display graphics in 4-bit grayscale at a resolution of 160x32

**Credit**  
Sparkfun Electronics  
licensed under CC  
BY 2.0



## VERDICT

How many graphical displays can be wrapped around your coffee mug?!

8/10



# Adafruit 2.8-inch TFT LCD with Touchscreen Breakout Board

ADAFRUIT ◆ \$29.95 | [adafruit.com](http://adafruit.com)

## Below

A beautiful TFT display from Adafruit with built-in touchscreen and SD card slot

## Credit

Marc de Vinck

**M**ost displays are good at one thing, and one thing only: showing some kind of data to the user. However, some displays can add additional features like user input through either capacitive or resistive touch. This kind of technology can get complicated to implement, until you discover the 2.8" TFT LCD with Touchscreen Breakout Board w/MicroSD Socket by Adafruit.

This full-colour TFT screen allows the user to enter data, press virtual buttons, or even draw on it, similar to a screen found on a smartphone – although the resolution of this display is a bit lower than most smartphones these days, coming in at a mere 240×320 pixels. But even so, the images look beautiful and the user input via resistive touch is extremely useful. Think about adding virtual buttons to your next project. You can go back and rename, or rearrange them as needed, and all of those changes occur in software. That's a lot faster than designing physical button layouts in hardware. We also like the idea of being able to show the end-user detailed visual feedback of systems or processes that are occurring in real time. Or you could easily add photos via the SD card, or draw simple shapes using code.

Communicating with the display is made easy though either 8-bit mode or SPI, and it doesn't require a powerful microprocessor to send the data since the board features a built-in controller with RAM buffering. That makes it perfect for microcontrollers like the Arduino that lack some of the advanced features and speed of computers like Raspberry Pi. There are libraries and example code that make implementing its features really easy with an Arduino, and you can find code examples for running it on other platforms too. □



//

Communicating with the display is made easy though either 8-bit mode or SPI and it doesn't require a powerful microprocessor

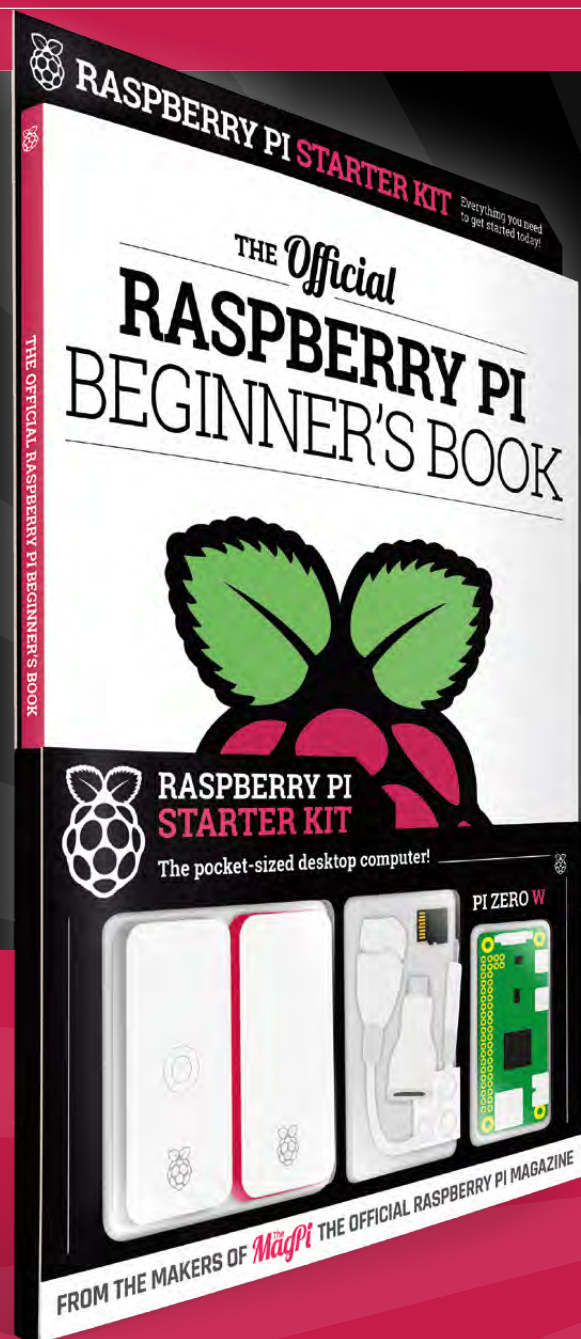
//

## VERDICT

A great choice when you need the added functionality of touch input.

8/10

# 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 microSD card
- 116-page beginner's book

*Available  
now*



Buy online: [store.rpiexpress.cc](https://store.rpiexpress.cc)



**£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:**

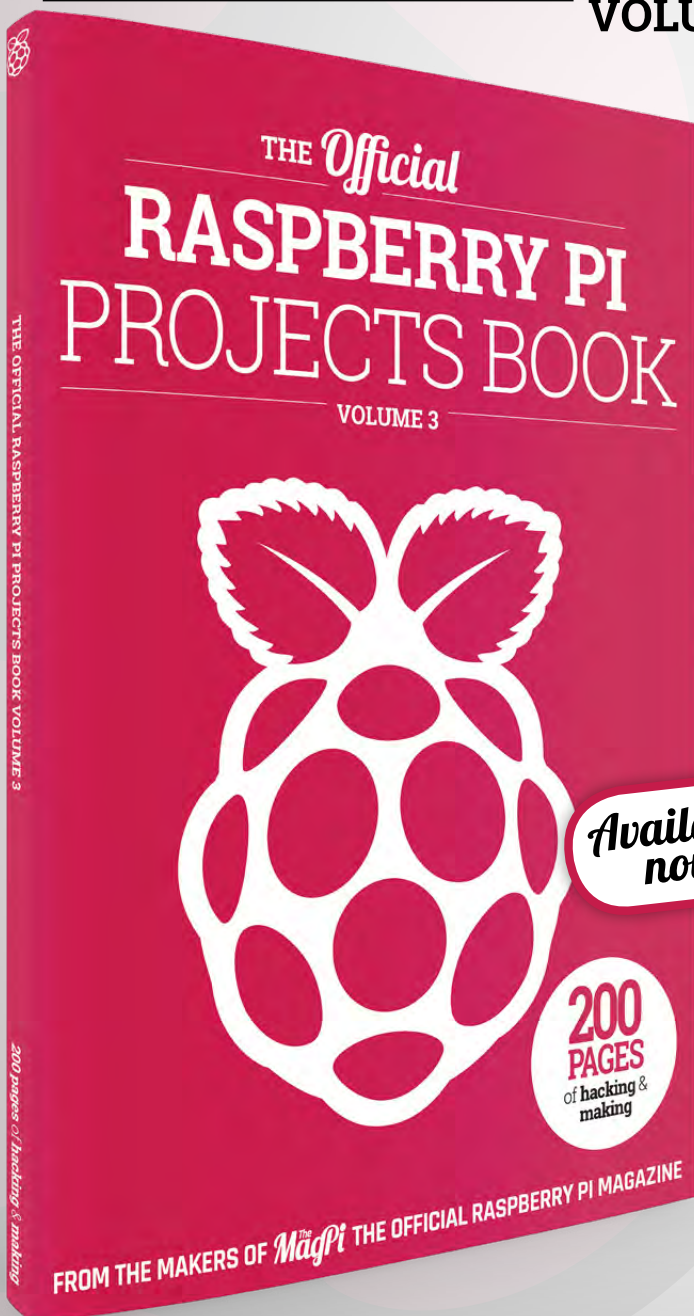
- How to get started coding on Raspberry Pi
- The most inspirational community projects
- Essential tutorials, guides, and ideas
- Expert reviews and buying advice

**Available  
now**

**[store.rpiexpress.cc](http://store.rpiexpress.cc)**

plus all good newsagents and:

WHSmith **BARNES & NOBLE**



## Can I Hack It?

# A radio-controlled car?

Can a radio-controlled car be hacked to work with a microcontroller?



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](http://bigl.es)

### YOU'LL NEED

**Nikko 9018 Velocitrax radio-controlled car**

### COST

£29.99

### WHERE

[amazon.co.uk](http://amazon.co.uk)

### Right

Lean, mean, and a shocking shade of green. Sure, it might not be the prettiest colour, but we have a great all-terrain vehicle in a cost-effective package, so we can always spray-paint it!

**R**adio-controlled cars, tanks, and ponies have been a staple of children's play for decades. They enable children to control scale models of real-world vehicles and create motorised play sessions.

For parents, radio-controlled toys typically meant batteries... lots of batteries! But in recent years we have seen better-quality radio-controlled vehicles at much cheaper prices. So what can you get for around £30 for consumer-level tech, and how can we hack it? Let's find out.

### GENERAL CONSTRUCTION

The Nikko 9018 Velocitrax car is made from a generally firm plastic, with some softer plastic

sections typically used to absorb impact. The plastic used is easy to work with, so hacks involving rotary tools, drills, and hand tools are possible.

Measuring approximately 29cm by 16cm and 12cm tall, the car is big enough to easily work with and inside it has plenty of space to add further hacks.

A tank-track-style tread enables the car to cover most terrains. Power comes from the rear wheels, which means that the front wheels are merely there to complete the tracks and provide adequate tension.

Standard crosshead screws hold the shell to the chassis of the car, and only six screws need to be removed in order to gain access to the insides.

### BATTERY POWER

Radio-controlled vehicles have come a long way since those early battery-eating monsters! These days we find a 7.2V Nickel-metal hydride (NiMH) battery rated at 700mAh. In tests we found that the battery was reporting as 8.1V after a fresh charge. The battery connects to the car using a bespoke connector that physically prevents reverse polarity connection.

The battery is charged by removing it from the car and connecting to the supplied charger unit; charging times are around one hour.

### MOTORS

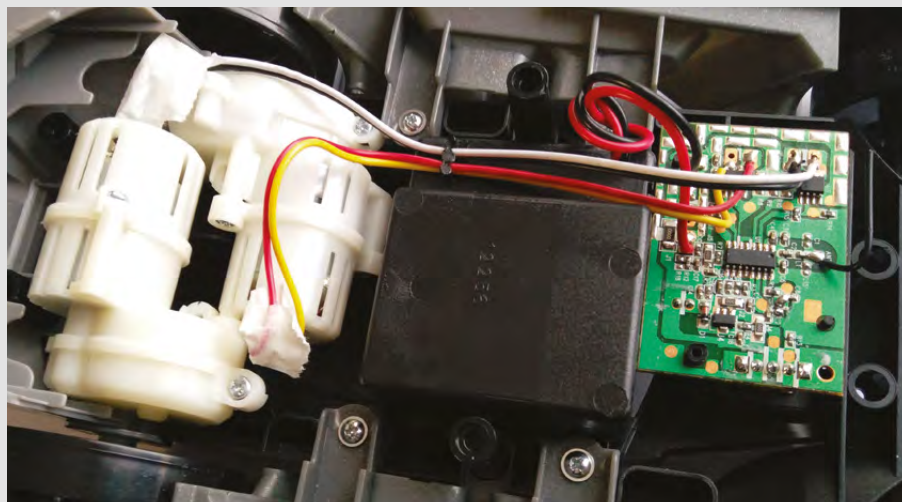
There are two motors at the rear of the car; these run at 7.9V and provide both forwards and backwards movement. Each motor comes with three 100nF ceramic capacitors (104) used to smooth the current flowing to the motors. The motors provide plenty of power for the car; in fact, they are a little too powerful for the size and weight of the car and it is liable to perform wheelies when starting. When first started, the motors will pull a quick 2.2A, before settling down





## UNUSED PARTS

So you've replaced the existing electronics with something else, but what can you do with the old X2445 and the CP2119 on the old board? Well, now you have a simple 2.4GHz radio receiver which can be used to build your own radio-controlled vehicle or project that can be remotely triggered using the electronics found in the controller. Based on the 4.5V power source required for the controller, an Arduino or other 5V logic board could be used to simulate the button push and be used as a remote trigger. Waste not, want not!



to consume around 720mA, of which the left motor (when looking face to face with the car) pulls 445mA, whereas the right motor pulls only 285mA, so this may cause a bias in the steering.

## ELECTRONICS

Firstly, the soldering for the circuit board is very poor: for the motor connections the solder was barely on the correct part of the board, and there was evidence of the wiring being deformed due to excess heat, so this would be a great candidate for resoldering. That aside, the circuit board is packed with components, but the most important is the X2455 IC that acts as a 2.4GHz radio receiver and decoder. The X2455 is connected to an antenna and receives radio signals from the controller, which it interprets and then sends the correct response to the two CP2119 H-bridge motor controllers that then rotate the motors in the required direction. Despite being a common board found in Nikko products, not much is known about the CP2119 motor controller; in fact, the nearest data sheet we could find was for an RZ7889 and this is for a motor controller that works between 3V and 15V with a peak output current of 5A. Should we wish to replace the electronics, then a suitable motor controller would be the L298 series as this can handle up to 4A of current, but don't be tempted to use the on-board 5V regulator to power a Raspberry Pi / Arduino as this is not suitable and could cause issues. If you haven't got an L298, then at a push a DRV8833 could be used but this has a peak output of 2A, so the startup current of the motors means that you are pushing it past its limits. No matter what choice of controller, the existing 7.2V battery can be reused. Removing the circuit board is simple: it's held in place by a friction fit and the wires soldered to it. There is plenty of space for an Arduino / Raspberry Pi Zero or other small controller boards and this would make an instant deadly robot!

## RADIO CONTROLLER

Held together with simple clips, the controller is a cheap plastic shell offering two inputs, each with only a forward and backward control option. They are simply covering momentary switches, those that are used on breadboards, and the user merely mashes the plastic of the controller to the button. We see a 2.4GHz radio transmitter that relays the controls to the X2455 in the car. The controller is powered by three AAA batteries and is a generic unit used across the Nikko range.

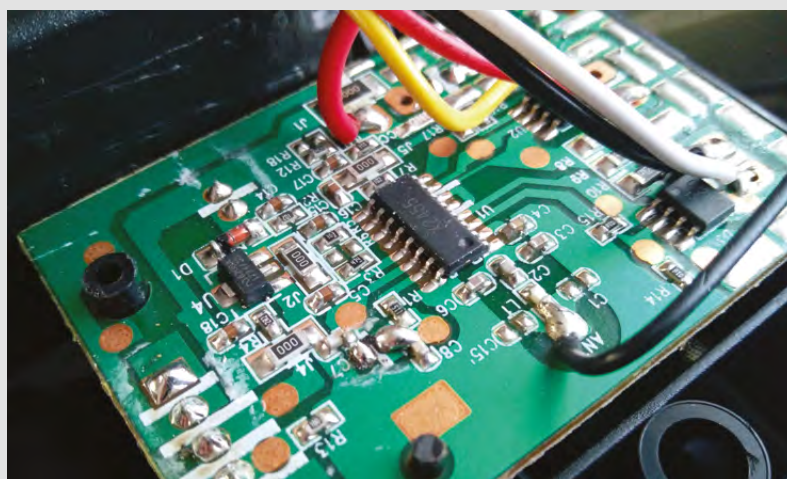
## CONCLUSION

The car offers plenty of scope for hacks. It can be quite useful as a donor machine for a Robot Wars / Pi Wars competition thanks to a great battery and space to contain your own electronics. Build quality is decent and there is easy access thanks to using a common type of machine screw. The tank tracks offer stability over many different terrains.

This is a fun platform to hack around with and quite cost-effective when compared to other robot donor candidates. □

**Above** ♦ Removing the circuit board is simple, and it leaves us enough space to insert a small Arduino, Pi Zero or other control board along with a motor controller

**Below** ▣ A great board that can be repurposed for other projects. It combines a radio receiver and motor controller to enable the user to control the car



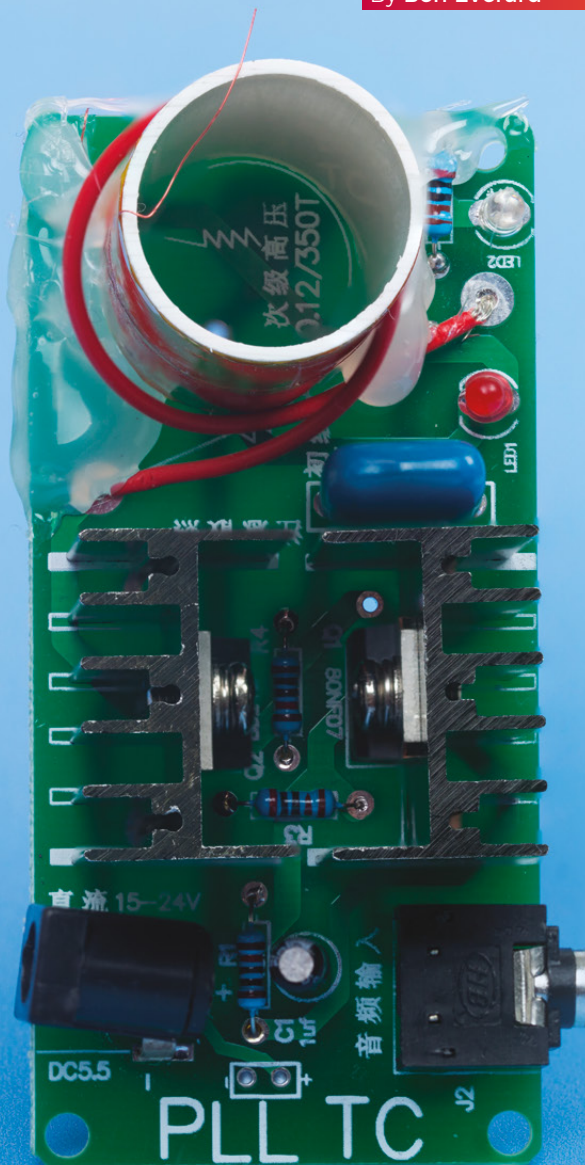
DIRECT FROM  
**SHENZHEN**


# Tesla coil speaker

Make music with lightning

By Ben Everard

[ben\\_everard](#)



**Right**  We only managed to secure the coil by taking the glue to the edge of the board

**T**he basic principle is very simple: an audio signal goes in, it's amplified, and then sent to a wire that's wrapped a single time around a large coil. This single wrap is a rudimentary primary coil and the varying electrical flow in this (being created by the music signal) creates a much larger voltage in the secondary coil. This secondary coil is finished with a loose wire that sticks into the air, and a sufficiently large voltage is created in the secondary to produce a spark. This spark heats the air around it, so as the spark comes and goes with the pulsing electrical current, this heating causes a vibration in the air. This vibration is the same frequency as the audio signal going in. In other words, it's an elaborate speaker system.

The tesla coil speaker – also known as a Zeusophone or a Thoramin, after the Greek god of lightning and the Norse god of thunder – is widely available in kit form from direct-from-China electronics sites. We got a kit from Yi Ma Trading Company Ltd on Ali Express, though identical kits are for sale elsewhere. Our test kit cost just £4.96, including postage.

Our kit came as a PCB and a handful of through-hole parts. The instructions were in Cantonese, but it was easy enough to follow along as all you really need to know is which part goes in which holes in the PCB, and they were labelled in English on the board. There are heat sinks for two power transistors, but again, these were easy to attach. We've seen some similar kits advertised as coming with thermal paste, though ours didn't, and it doesn't get very hot under moderate use.

The only thing that the kit didn't come with was a power supply. It takes a 15-20V barrel adaptor (the same shape as an Arduino Uno). This can also be



supplied through headers soldered into the PCB if you've got a bench power supply.

Connecting a music player to a device capable of producing enough voltage for this spark is a little nerve-racking – especially since we'd soldered it ourselves with the instructions written in a language we don't read. We didn't want to risk frying our phone, so we found an old MP3 player in the back of a drawer and fired this up. A small (approximately 5mm) spark danced from the end of the loose wire on the coil. You'll need to dim the lights to fully appreciate the majesty of the raw electricity flicking to the tune of your favourite song. It's not compulsory that you play Electric Six's 'Danger! High Voltage', but it is strongly recommended.

### QUALITY CONTROL

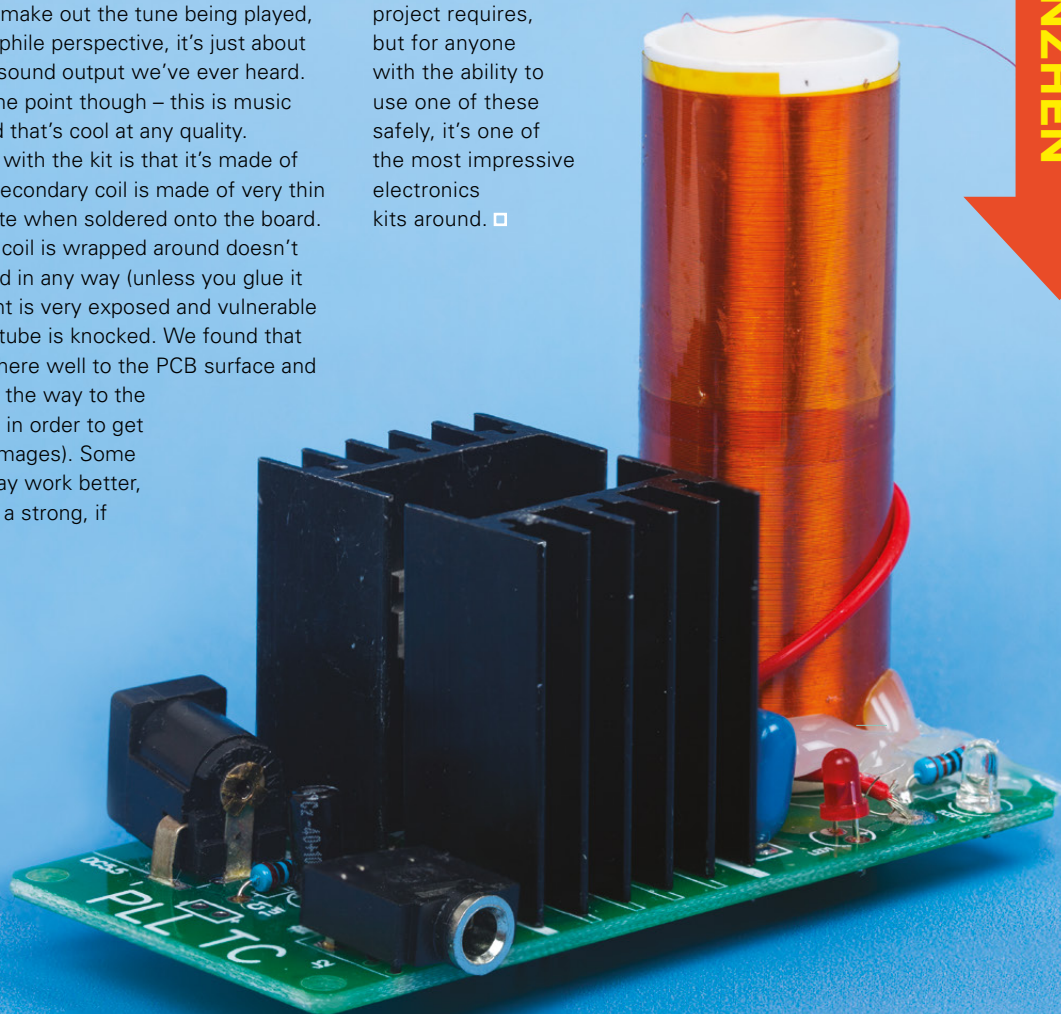
The sound produced is audible but quiet. It's just about possible to make out the tune being played, but from an audiophile perspective, it's just about the worst quality sound output we've ever heard. That's not really the point though – this is music from lightning and that's cool at any quality.


The main issue with the kit is that it's made of weak parts. The secondary coil is made of very thin wire that is delicate when soldered onto the board. The tube that the coil is wrapped around doesn't attach to the board in any way (unless you glue it down), so this joint is very exposed and vulnerable to breaking if the tube is knocked. We found that hot glue didn't adhere well to the PCB surface and we had to glue all the way to the edge of the board in order to get a solid joint (see images). Some other adhesive may work better, but this produced a strong, if unsightly, joint.

The power socket failed after several uses. The flimsy metal connector that had been slightly bent in transit snapped in two, despite gentle handling. This is fairly easy to replace (and there are additional points on the PCB to add wires to supply power), but it's another source of annoyance. For the price we paid, we can live with the fact that it's prone to falling apart as it was a fun project to make and even with a few uses we feel we got our money's worth. However, it does make it a little hard to recommend this to hackspaces and clubs unless you're willing to take some measures to toughen it up.

Obviously a certain amount of caution is necessary with a kit like this. It will output plenty of electromagnetic interference, could cause a fire, and generally needs treating with the respect any high voltage project requires, but for anyone with the ability to use one of these safely, it's one of the most impressive electronics kits around. □

DIRECT FROM SHENZHEN




**Right**  With two heat sinks and a coil, it's quite an imposing project even when it's not spitting lightning

# LilyPad ProtoSnap Plus

\$49.95 | sparkfun.com

By Ben Everard

 @ben\_everard

**T**he LilyPad is based on the Arduino platform, but in a form that makes it easier to incorporate into wearable projects. As well as the main microcontroller, there's a broad range of input and output add-ons under the LilyPad brand that are all designed to look good and to work with conductive thread.

The ProtoSnap Plus kit includes the LilyPad USB Plus microcontroller board (which includes six white LEDs in a bar graph, and one RGB LED), a light sensor, a button, a slide switch, eight sewable LEDs and a buzzer. All these parts come on a single PCB where they're connected via traces and can be used with no wiring or soldering. However, the individual parts can be snapped out so they can be rearranged before being sewn into a circuit.

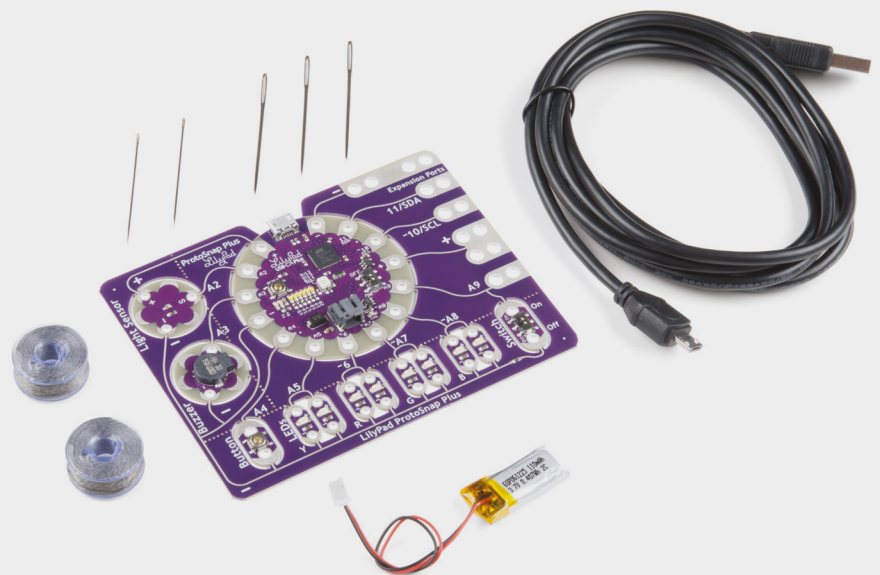
Alongside this PCB, you get a micro USB cable, a 110mAh LiPo battery, two bobbins of conductive thread, and a selection of needles. With this kit you have everything you need to create your own wearable electronics. The LilyPad can even function as a battery charger (through the USB power supply), so it does have everything you need in this kit.

By shipping the various components on a single PCB, it makes it easy to get started – you don't have to fiddle around with connectors before testing out your code. Once you've got your code working, you can snap the components out and sew them into your wearable project.

The integrated on/off switch might sound like an unusual thing to praise on a bit of electronics, but it's useful for ensuring that you don't burn through the battery on your wearables before getting to the party (or wherever you're going).

The microcontroller gives you ten GPIOs, with seven of these taking analogue input and four with PWM output. This gives you enough pins to drive all the included hardware with an additional two available for other bits and pieces.

The LilyPad can be made to work with most 3.3 volt electronics, but there's a series of peripheral



“

By shipping the various components on a single PCB, it makes it easy to get started – you don't have to fiddle around with connectors before testing out your code

”

boards that are designed to work well with sewable circuits including accelerometers, temperature sensors, an Xbee module, an MP3 player, a BlueTooth board, a reed switch, protoboard, and plenty of flashing lights. They all come on the iconic purple PCBs and look great when sewn into fabric.

The LilyPad ProtoSnap Plus doesn't have an overwhelming number of features, but what it does have is well thought-out and works well. It makes a fantastic introduction to sewable circuits. □

**Above** ♦  
The 6 ft (183 cm) USB cable makes it easy to plug in, even when sewn into an outfit

## VERDICT

A great introduction to wearable electronics with everything you need to get started, but more forms of input would make more complex projects possible

9/10



# OKAY Synth DIY Kit

\$40 | oskitone.com

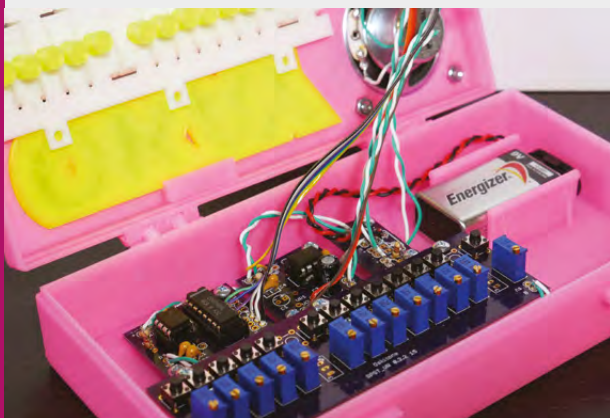
By Liz Clark

@BlitzCityDIY

**T**he OKAY Synth Kit is a **monophonic analogue small-scale synthesizer**. Produced by Oskitone, a one-man operation out of San Francisco, it comes in two flavours: the electronics kit with an included 3D printed enclosure, or a 'bring your own printer (BYOP)' edition that includes the electronics, but assumes that you will print the enclosure yourself using the files and settings available on Thingiverse. For the full experience, the BYOP edition was built for this review.

Upon receiving your parcel from Oskitone, the first thing that you'll undoubtedly notice is the level of detail and care in packaging of the components for the electronics portion of the kit. The kit is made up of four custom PCBs and each board is in a separate bag with its corresponding components. Aiding in that experience are the instructions that also arrive inside the package in a lovely printed booklet; a rarity in this increasingly digital world. The instructions are also available online as a PDF on the Oskitone website.

The printing process for the enclosure went smoothly following the provided settings. As long as you've taken care in ensuring that your printer is calibrated properly, then you should not have any



issues with the prints. One unique aspect of the printing process is that many of the parts print with two colours utilising a colour changeover at some point in the print. Oskitone has taken the time to put in the layer height for each part where this occurs so that you can easily achieve this effect. The design of the parts is another place where the kit shines. You can tell that many hours of CAD work are behind the final files that you slice and load into your printer.

After your printer has cooled, it's time to heat up your soldering iron for the electronics portion of the kit. It's here that the assembly process becomes a bit aloof. The instructions for each PCB assembly are quite sparse and assume an intermediate level of electronics knowledge. For most kits, this wouldn't be an issue but after coming from the detailed instructions for the 3D printing process it feels incomplete. Despite the murky instructions, the design is compact, and the components and PCBs are a nice quality.

The instructions come back up to the level seen during the 3D printing stage for the final assembly. The 3D printed parts go together well, and the PCBs are spaced comfortably in the case. It's after this that the real fun begins. As a synthesizer, it's a delight to play around with. The tone is punchy, and the built-in speaker offers a surprising amount of volume for its dinky size. □

**Above** □ The tiny, yet mighty, OKAY synth. It can of course be printed in tamer colours if 80's fluorescents aren't your taste

**Left** □ A look inside the OKAY synth. As you can see, a lot of thought has been put into the design of the case and PCBs

## VERDICT


The OKAY synth kit offers a unique build experience with the added bonus of being a fully-functional analogue synth for all of your 8-bit and retro music dreams.

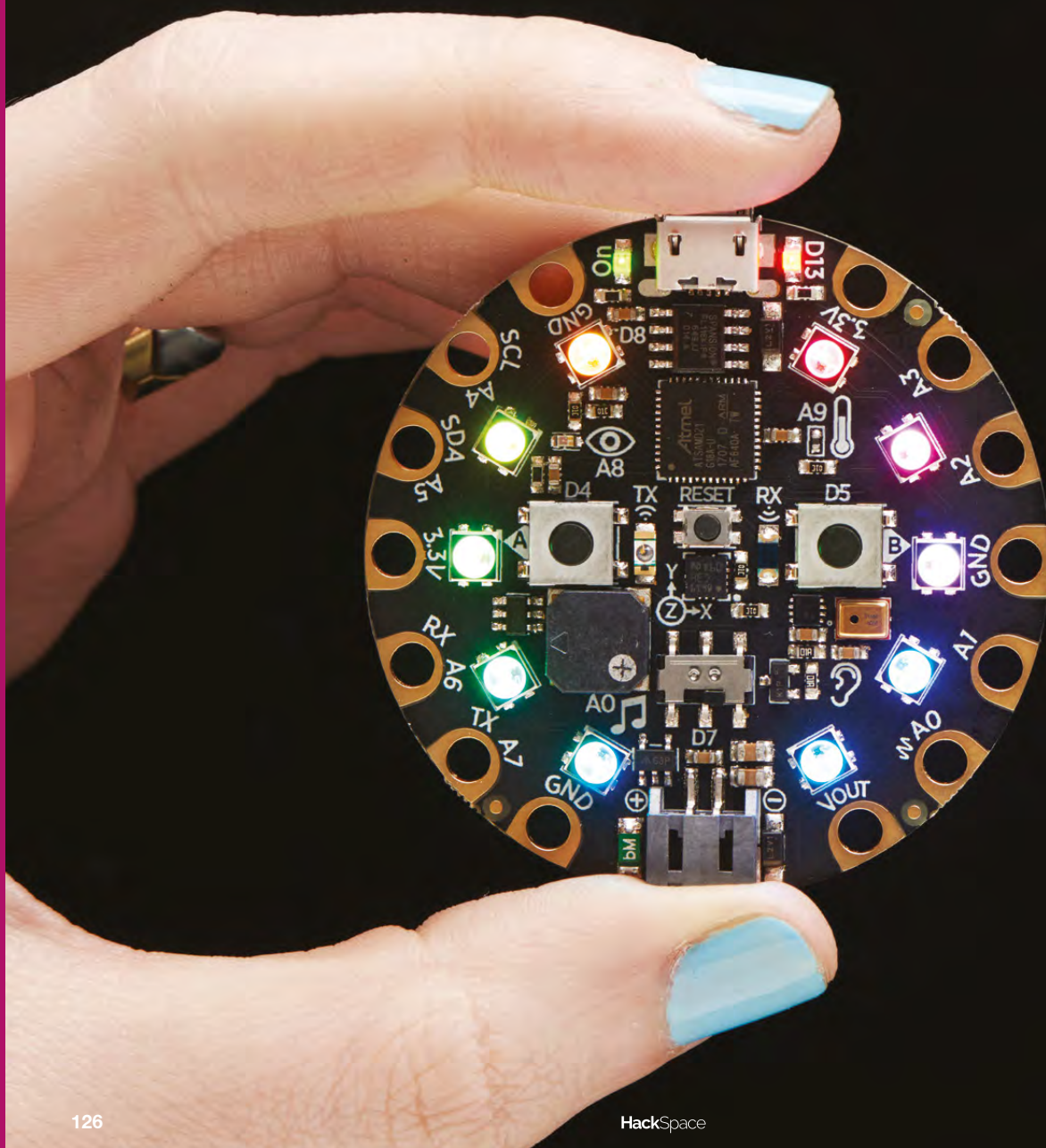
8/10

# Circuit Playground Express

\$24.95 / £25 | [adafruit.com](https://adafruit.com) (US) / [pimoroni.com](https://pimoroni.com) (UK)

By Ben Everard

 [ben\\_everard](https://twitter.com/ben_everard)





**T**he Circuit Playground Express (CPX) is a programmable microcontroller board that makes it really easy to get started.

Plug it into a computer with a micro USB cable and you're ready to start coding.

The UF2 firmware takes software into

the board in two ways: either you can press the reset button to enter programming mode and copy UF2 files into a new drive that will appear on your machine, or you can upload directly from the Arduino IDE. For beginners the first method will be easier, as you can generate these UF2 files from either the web-based MakeCode block-based editor or from a Circuit Python development environment.

On some versions of Windows, you'll need to install drivers, but on Mac OS X, Linux and Windows 10, you don't need to install anything if you use MakeCode – just plug your board into a USB port, then point your browser to [makecode.adafruit.com](https://makecode.adafruit.com) to start your first project.

### BUILT-IN FEATURES

From here you can take advantage of the wide range of hardware that's packed onto the board. For output, there are ten NeoPixels and a speaker (not just a buzzer).

For input, there are two buttons, a slide switch, an accelerometer, a temperature sensor, a microphone, and a light sensor. With all this, you can put together some pretty fancy projects straight away. For our first project, we made an indicator for cyclists. It uses the large holes on the CPX to attach it to the back of a cycling glove, then uses the accelerometer to recognise when the hand is held out indicating a turn and flashes the NeoPixels orange. This took 18 lines of dragged-and-dropped code in MakeCode and no additional hardware (most of these were to ensure that there was a stable reading from the accelerometer). While it's hardly a complex project, it shows that you can build useful projects quickly with no additional hardware or software.

There's nothing on the Circuit Playground Express that's fundamentally unique – you can get the same sensors and outputs to attach to virtually any microcontroller. What makes this board special is the way it's brought together into a single package. It's hard to think of any project where you'll need all the features available, but given the £25 price tag, it's easy to justify the cost even if you only need one or two of the extra input or output options. By bringing them all onto the main board, there's no extra setup or wiring, and it's all supported by the software without having to add any libraries, which again makes it thankfully easy to get started.

The one obvious thing missing from the device is any form of networking. It does have a built-in infra-red

receiver and transmitter, I2C and UART, but no WiFi or Bluetooth. The other major limitation of the board is that it's not breadboard-friendly – it's far more suited to crocodile clips or banana plugs.

### MANY, MANY POSSIBILITIES

Whether or not these are really limitations depends a lot on the sort of projects you're working on. The CPX isn't going to work well for Internet of Things-type applications. It's also not going to work well as a controller for building complex circuits – having just eight GPIOs limits the amount of hardware you can connect. Anyway, there are lots of microcontroller boards far more suited to these uses. However, the CPX makes it fantastically easy to get started with embedded and physical computing projects. You can build on the integrated hardware with the eight GPIOs. Seven of these can detect capacitive touch input, so to add more user input to your project, you just need a few crocodile clips and leads. As all the GPIOs can read analogue input, it's also trivial to add input from and device that gives a varying voltage as it's output. There are also five PWM output pins for driving LEDs

“

The CPX is well suited to people getting their first microcontroller, either buying one themselves for fun or as part of a taught course

”

at different brightnesses. The MakeCode platform is a code repository as well as an IDE, and there are a set of tutorials from Adafruit to help you get started with the platform and the hardware.

The CPX is well suited to people getting their first microcontroller, either buying one themselves for fun or as part of a taught course. For this purpose, it's genuinely hard to fault the CPX. It's easy to learn with no (or minimal) software to install, yet at the same time allows you to use more advanced languages if you've got the knowledge and experience. It packs a fantastic range of input and output options onto the board, which means that you can dive right into some more interesting projects without getting immediately bogged down in attaching extra hardware. It's also easy to start to build simple circuits off using crocodile clips.

Perhaps the ultimate test of any bit of hobbyist kit is whether or not it sparks excitement. For us, the CPX gave us a childlike sense of glee, because it makes so much so easy. This is great for beginners and any hobbyists who like to quickly prototype projects. □

### VERDICT

The best microcontroller board for introducing people to physical computing projects.

10/10

# Sonoff Basic WiFi Smart Switch

\$6 | itead.cc

By Daniel Hollands

[maker.limeblast.co.uk](http://maker.limeblast.co.uk)

**W**ith more and more smart appliances entering the market, allowing control via a phone and/or voice commands, a lot of people are looking at their perfectly good 'dumb' appliances and wishing they could be switched on and off in the same way. This is the problem the Sonoff Basic Smart Switch aims to fix.

Unlike WiFi socket adaptors, the Sonoff Smart Switch is designed to be patched into the power cord of your appliance. After disconnecting from the mains, you snip the cord in half, strip the insulation from each end of the exposed live and neutral cords, and attach the wires to the clearly labelled connectors on each side of the module. You then reattach each side of the earth cord and ensure everything is secured safely without any exposed wires.

All this faff means more effort is required before you're able to use it but, as a result, they're far cheaper to buy than socket adaptors. Of course, this means if you're not confident with a pair of wire cutters and mains voltage, this product isn't for you.

Once installed, head to the app store and download the eWeLink app. Following the instructions contained within, you'll need to create an account

in the cloud, then link the device allowing it to connect to your WiFi. We had some trouble using a OnePlus 3T phone for this, but a Nexus 9 tablet worked just fine, and once it was set-up we simply needed to log into the app on our phone for full control.

Using eWeLink, you power devices on and off and set schedules. You're also able to share access to your devices with other people, and connect with the Amazon Echo, Google Home, and Nest services, allowing control via each. As if all that wasn't enough, there's also support for IFTTT, giving you a lot of options for automation.

The reliance on a cloud-based service means you can control your devices remotely, but also means you'll need an active internet connection. I should also note that in the four months we've been testing it, the service went down twice for a couple of hours each time. When this happened, we could still use the button on the module itself to control the power.

Under the hood is an ESP8266 microcontroller which is connected to a relay. The significance of this is the ability to flash your own firmware onto the module, removing your reliance on eWeLink, and letting you add support for protocols such as MQTT. There are two such community-supported projects that offer hackers more options for controlling their switches.

While the Sonoff Smart Switch isn't as plug-and-play as some solutions – requiring a bit of snipping with wire cutters – it should work with anything with a power cable and has plenty of options for control. □



**Above** □ Clearly labelled connectors make this easy to wire up

**Left** ♦ Providing smart power to the author's homemade 'mad scientist' table lamp

**VERDICT**  
A cheap smart switch solution for people willing to roll up their sleeves

8/10



# Theo Gray's Mad Science: Experiments You Can do at Home – But Probably Shouldn't

Theo Gray ♦ £12.95 | graysci.com

By Richard Smedley

@RichardSmedley

“For better or worse,” writes Gray, “the fire, smoke, smells, and bangs of chemistry are what inspired many scientists to become scientists in the first place.” Mad Science exults not just the spirit of amateur scientists, but their substantial achievements. But mostly, the citizen scientist is someone who gets to have fun.

Starting with experimental cuisine, the first course is making salt – with a warning that this experiment is the most dangerous in the book, and should only

be tried by an experienced chemist. The health and safety warning at the beginning of this book is realistic, and more serious than the ones found in most modern books.

“This book does not tell you enough to do all of the experiments safely,” and that’s deliberate – some experiments are not spelt out in enough detail for you to do because you shouldn’t be doing them until you’ve built up the right experience and knowledge. For the rest, the real dangers are highlighted, but there are plenty of experiments that you can try with children – and some that kids can do on their own.

Simple tricks are a fun way to amuse your friends – metal spoons that melt in hot water, ice cubes that sink, and rocket fuel from snack food. Along the way the science will get absorbed far more readily than in school chemistry lessons. Gray’s choice phrasing – “construct a light bulb the modern way with some helium and an old welder” – will keep you diving into each new project.

Learning how to make everything from matches to nylon is an inspiring thing for your children to experience. It’ll also make you feel less like you’d revert to a Neanderthal without all of your solid state technology around you. Of course, not all of the materials and equipment will be easy to come by – best hang on to that internet connection for a while.

Regardless of any atavistic urges – and the fire chapter will also appeal to those – the fun comes with plenty of potential for education, and you’ll be wishing that they could still do this sort of experiment in school. Inspiration for the well-insured hackspace perhaps? At the very least, a hugely enjoyable coffee table book of vicarious geekery. □



## VERDICT

So many great experiments – a cornucopia of fun, imbued with a love of science. Inspirational.

9/10

next month

issue  
#5

ON SALE  
22 MARCH

FEATURING  
**LIMOR FRIED**

ALSO

- THE BEST PROJECTS FROM AROUND THE WORLD
- ARDUINO SKILLS
- THE ONLY KNOT YOU'LL EVER NEED
- ENABLE MAKEATHON
- ULTIMATE SMD SOLDERING CHALLENGE
- AND MUCH MUCH MORE

DON'T MISS OUT

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



HackSpace  
TECHNOLOGY IN YOUR HANDS



DO **NOT** SET  
YOURSELF ON **FIRE**

It's really inconvenient.

— [hsmag.cc](http://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<sup>.99</sup>** **£59<sup>.99</sup>** **€64<sup>.99</sup>**  
USD 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.