HS

Issue #09 **August 2018**

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

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc | **August** 2018 | **Issue** #09

ULTIMATE LED CUBE

The challenges of building a giant illuminated box

BUILD A DRUM MACHINE

Kraft homemade electronic beats

AUTOMATIC WATER CANNON

Blast your way to garden supremacy

YOUTUBE

Use a machine to count your online followers

50 TOOLS FOR BETTER MAKING

Everything you need to build (almost) anything

+

› **Remap project**
Use your maker skills to help others

› **Electronics for beginners**
Discover the might of the electron!

› **Pi Wars**
Inside the ultimate Pi challenge

RACHEL WONG

Meet the woman mixing fashion and LEDs

50 Tools For Better Making

Aug.2018
Issue #09 £6

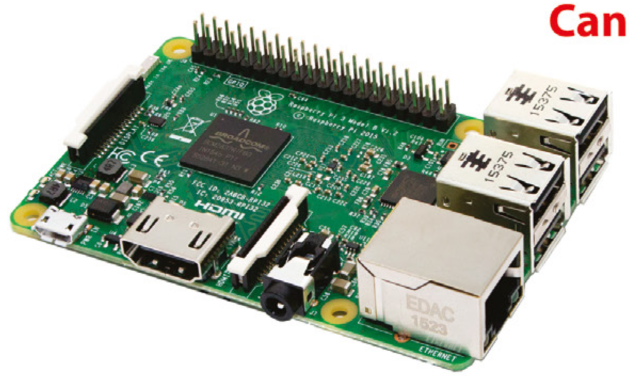
9 772515 514006 09

Raspberry Pi PRESS

HYDROPONICS **MASTER MIDI** NAIL CLIPPERS **MEASURE INERTIA**

hsmag.cc

CanaKit™



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

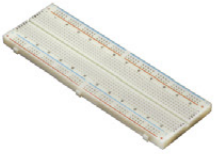
PREMIUM HDMI CABLE

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 APPROVED RESELLER

Raspberry Pi Zero W

Available at CanaKit!

WiFi™ Bluetooth®

ELECTRONIC KITS • ELECTRONIC PARTS • RASPBERRY PI • ARDUINO

Welcome to HackSpace magazine

When you get used to a tool, it almost seems like it becomes an extension of yourself. It can even affect the way you think about the world. As the saying goes, when you have a hammer, everything looks like a nail. Likewise, when you have a soldering iron, you start seeing the potential for incorporating electronics in everything, and when you get a welding kit, everything looks like it would be better if it were made out of scrap metal stuck together, and you start dreaming of great finds in scrap heaps (or is that one just me?).

Getting the right tools isn't just an economical decision, **it's a philosophical one as well**

Getting the right tools, then, isn't just an economical decision, it's a philosophical one as well. It's not just about the things you want to make, but the way you want to approach life. Take a look at our cover feature on page 32 for inspiration on what tools you need to go in the direction you want with your making.

Whatever tools you have, the most important thing is that you actually use them. There are few things sadder than an unused tool gathering dust, so get out there and get making!

BEN EVERARD
Editor @ ben.everard@raspberrypi.org

MAKE BUILD HACK CREATE
HackSpace
TECHNOLOGY IN YOUR HANDS
50 TOOLS FOR BETTER MAKING
Everything you need to build (almost) anything

GET IN TOUCH

- @ hackspace@raspberrypi.org
- f hackspacemag
- hackspacemag

ONLINE

- hsmag.cc

EDITORIAL

Editor
Ben Everard
@ ben.everard@raspberrypi.org

Features Editor
Andrew Gregory
@ andrew.gregory@raspberrypi.org

Sub Editors
Nicola King, Jem Roberts

DESIGN

Critical Media
criticalmedia.co.uk

Head of Design
Dougal Matthews

Designer
Lee Allen

Photography
Brian O'Halloran,
Connor Ballard-Pateman

CONTRIBUTORS

Lucy Rogers, Andrew Huang, Jen Botezat, Lorraine Underwood, Cameron Norris, Mayank Sharma, Graham Morrision, Dave Astels, Andy Clark, Oliver Jones, Paul Freeman-Powell, Alex Eames, Sophy Wong, Marc de Vinck, Les Pounder

PUBLISHING

Publishing Director
Russell Barnes
@ russell@raspberrypi.org

DISTRIBUTION

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

SUBSCRIPTIONS

Select Publisher Services
Ltd, PO Box 6337, BH1 9EH
☎ +44 (0)1202 586 848

Mann Enterprises Ltd,
Unit E, Brocks Business
Centre, CB9 8QP
hsmag.cc/subscribe

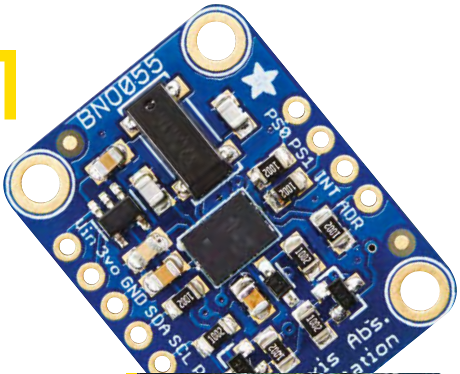
This magazine is printed on paper sourced from sustainable forests. The printer operates an environmental management system which has been assessed as conforming to ISO 14001.

HackSpace magazine is published by Raspberry Pi (Trading) Ltd., Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised.

ISSN: 2515-5148.

Contents

88



118

Cover Feature

ESSENTIAL TOOLS

Upgrade your life with a fresh set of tools. Now, what to choose…? Decisions, decisions!

32

Tutorial

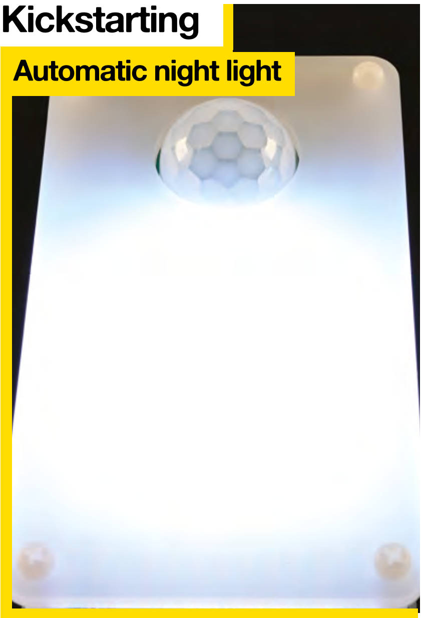
School of making: Google AIY Vision robot sentry

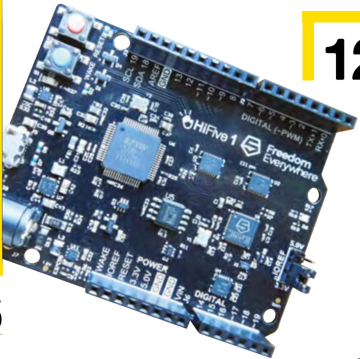
82

Defend the garden with an intelligent water pistol

Kickstarting

Automatic night light





Interview

Tim Richardson

69 FORGE

115 FIELD TEST

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.

Binary CMOS clock

By David Watts

hsmag.cc/AmubiW

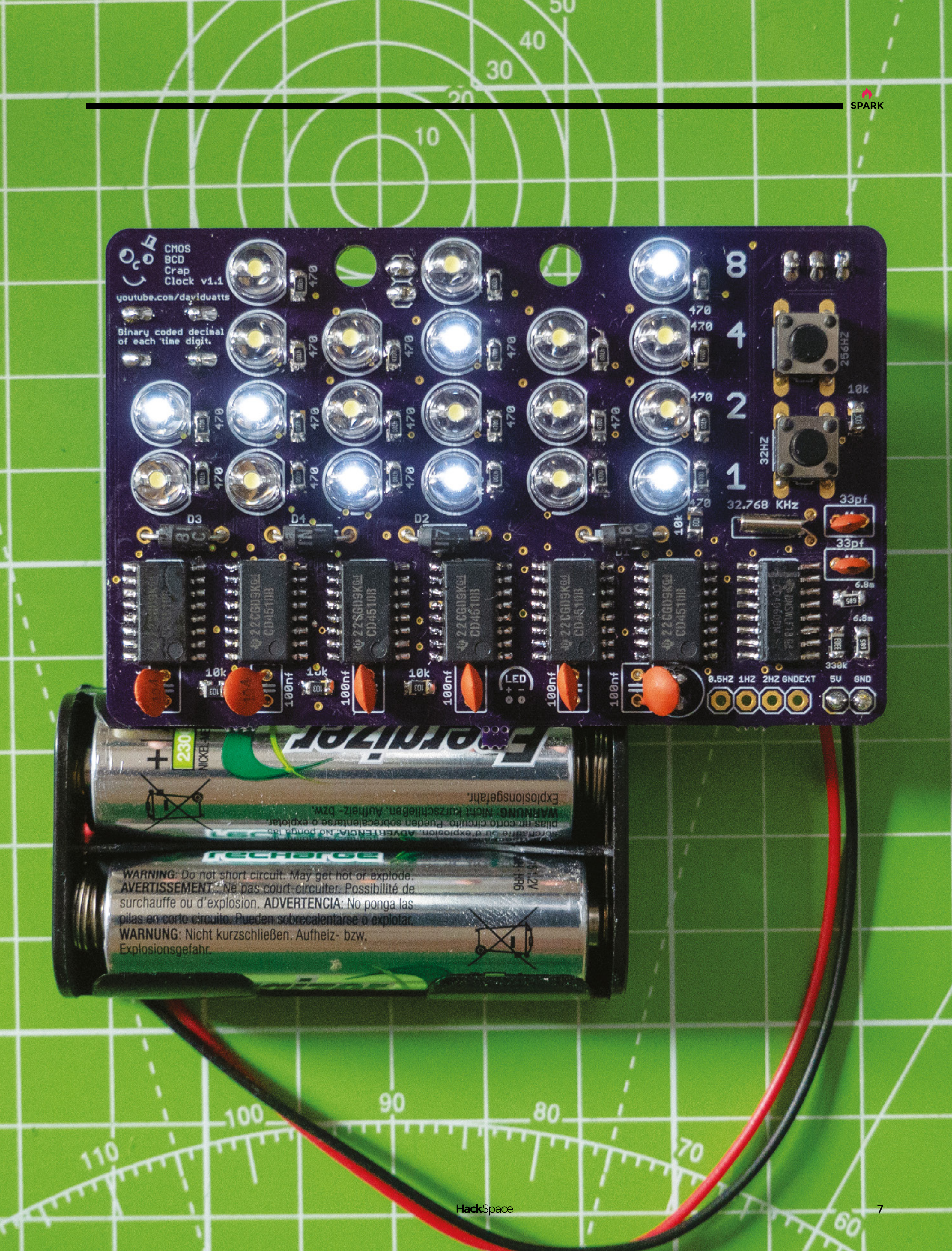
I am David Watts (yes, I have heard the song), and I am a self-styled maker/tinkerer. I learnt how to wire a plug at school, but that is about as far as my electronics education extends.

I call this the 'BCD Crap Clock' – why? Well, it lowers expectations with the hope of exceeding them. BCD stands for Binary-Coded Decimal – you see, it isn't a true binary clock.

This is a fairly crap 'binary' clock kit made with CMOS logic, a simple design without a microcontroller, and it has the unique feature of being both difficult to read and difficult to set the time! It is a curiosity rather than a real timepiece, but it is at least better than guessing the time. Consider it an exercise in learning about CMOS logic and making my first kit that would go on to sell out in a day on Tindie.

It uses quite a few ICs for something that could be achieved with one PIC or ATmega but hey, there are advantages. It runs down to about 2.7–3V, consuming 0.5mA with LEDs enabled or ~6µA with the LEDs disabled, meaning it could keep time for decades on a few AA cells. Also, I get 'no microcontroller' bragging rights. □

Right As the old joke goes, there are 10 types of people: those who understand binary, and those who do not


CMOS
BCD
Crap
Clock v1.1
youtube.com/davidwatts
Binary coded decimal of each time digit.
8
4
2
1
256HZ
32HZ
32.768 KHz
0.5HZ 1HZ 2HZ GNDEXT 5V GND

REGULAR

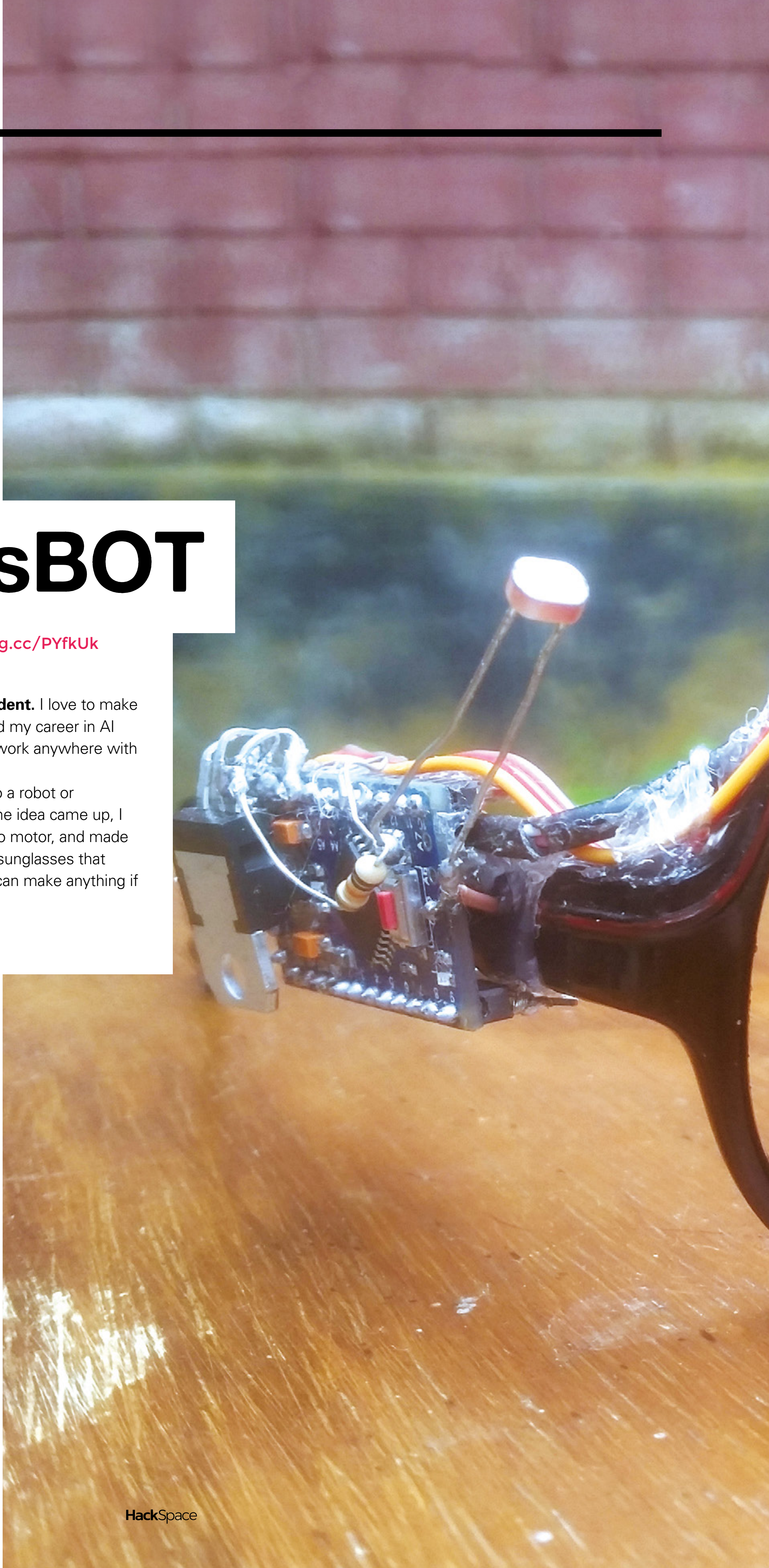
SunGlassBOT

By Ashraf Minhaj

hsmag.cc/PYfkUk

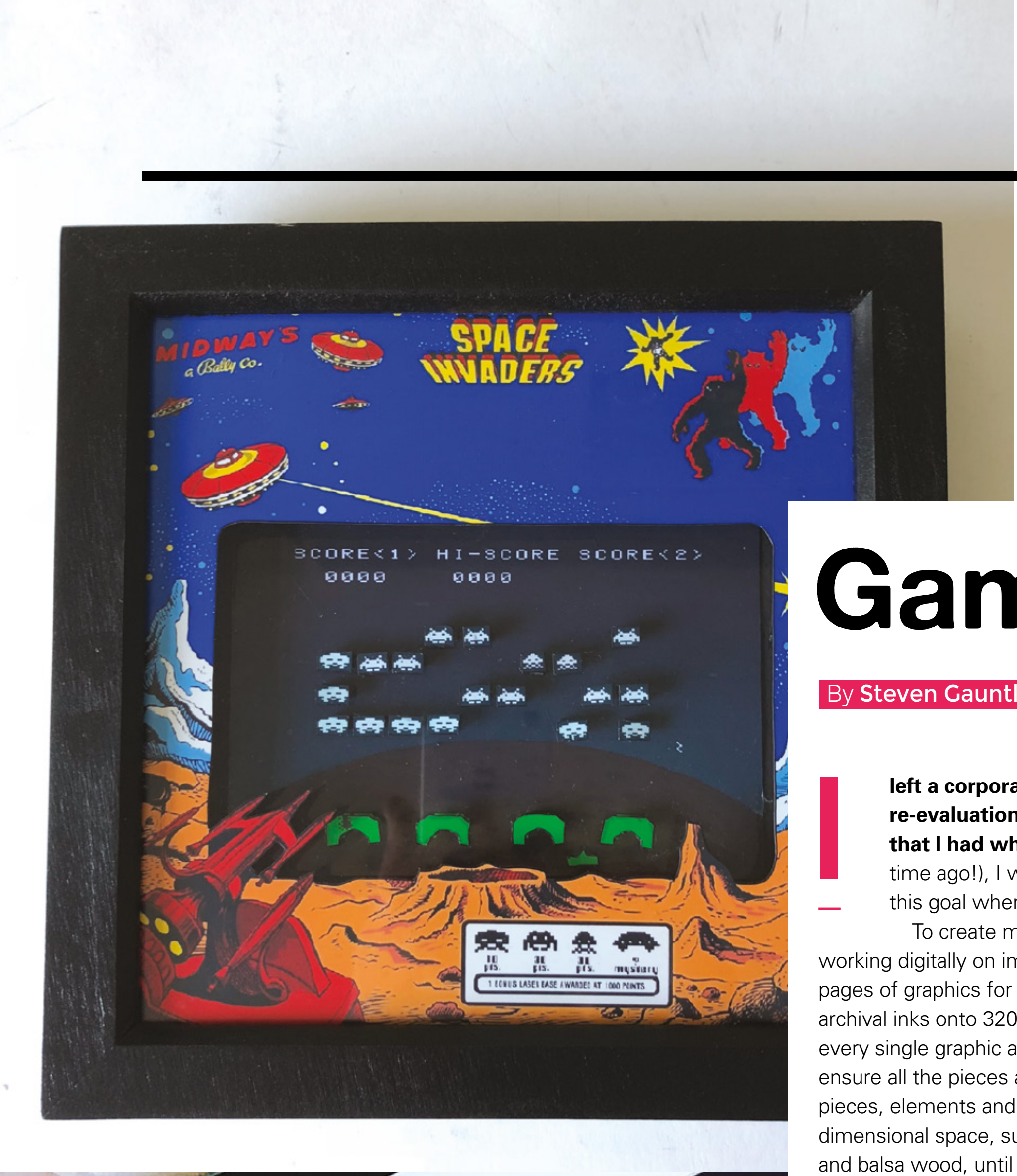
I am a first-year Computer Science student. I love to make robots and program them. I want to build my career in AI and robotics, and am also interested to work anywhere with anyone in robotics.

I believe everything can be turned into a robot or automated machine.That's why, when the idea came up, I took an Arduino a 10kΩ resistor, LDR and a servo motor, and made this – an automatic, robotically-controlled pair of sunglasses that protect your eyes if it gets too bright. I believe I can make anything if I have an idea. □



Right We predict commercial success for Ashraf's design, at least on a par with that of Google Glass

MicroServo
DXW90
9g



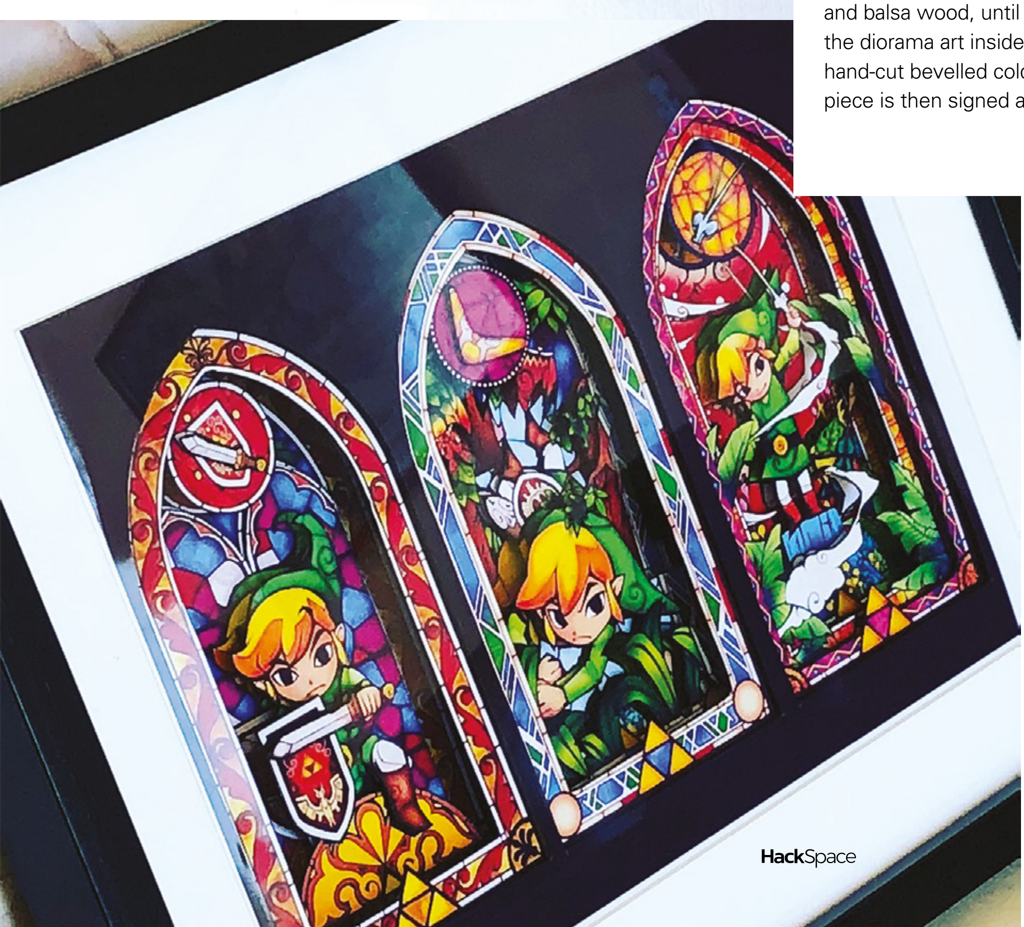
Game art

By Steven Gauntley

8bitboutique.co.uk

I left a corporate job eight years ago now after a major life re-evaluation, and I started to redevelop the artistic skills that I had when I was younger. When I was little (a long time ago!), I wanted to be an artist, and so I decided to pursue this goal when I turned 30.

To create my 3D art, I first of all spend time on Photoshop, working digitally on imagery to create approximately twelve different pages of graphics for each piece of art. I then print the graphics using archival inks onto 320 gsm ultra-gloss card-stock, and then hand-cut every single graphic and detail, and add hand-touched elements to ensure all the pieces are perfect. I then assemble the many individual pieces, elements and layers, building the picture up into a three-dimensional space, supporting the layers with foam-core art board and balsa wood, until the final 3D diorama is achieved. Then I house the diorama art inside a deep, wood shadow box, and mount using a hand-cut bevelled cold press card mount. Each limited edition finished piece is then signed and numbered on the reverse. □



Left
Steven's immersive artwork takes us back to the good old days

Wimblebot

By Ross Atkin

hsmag.cc/aGklzF

I'm a product designer based in London. I work mainly on projects aimed at making life easier for disabled people, either in assistive technology or accessibility. I think it is really important that kids get to experience being creative with technology, and I'm horrified by how this is disappearing from our education system. Because of this, I started The Crafty Robot, a small business that makes robot kits with lots of creative possibilities.

Our latest robot kit is called Smartibot and we're running a campaign on Kickstarter, so we can put it into production. I used the Smartibot circuit board, battery box, and motors, combined with some tennis balls and the tube they came in, to build this Wimblebot. I controlled it using the app we've created for Smartibot. □

Right Smartibot (seen here in its Wimblebot form) gives you an easy way to use AI in robotics

Above
You like glowing LEDs? Turn to page 46 for even more blinkenlights

Bucky Glow

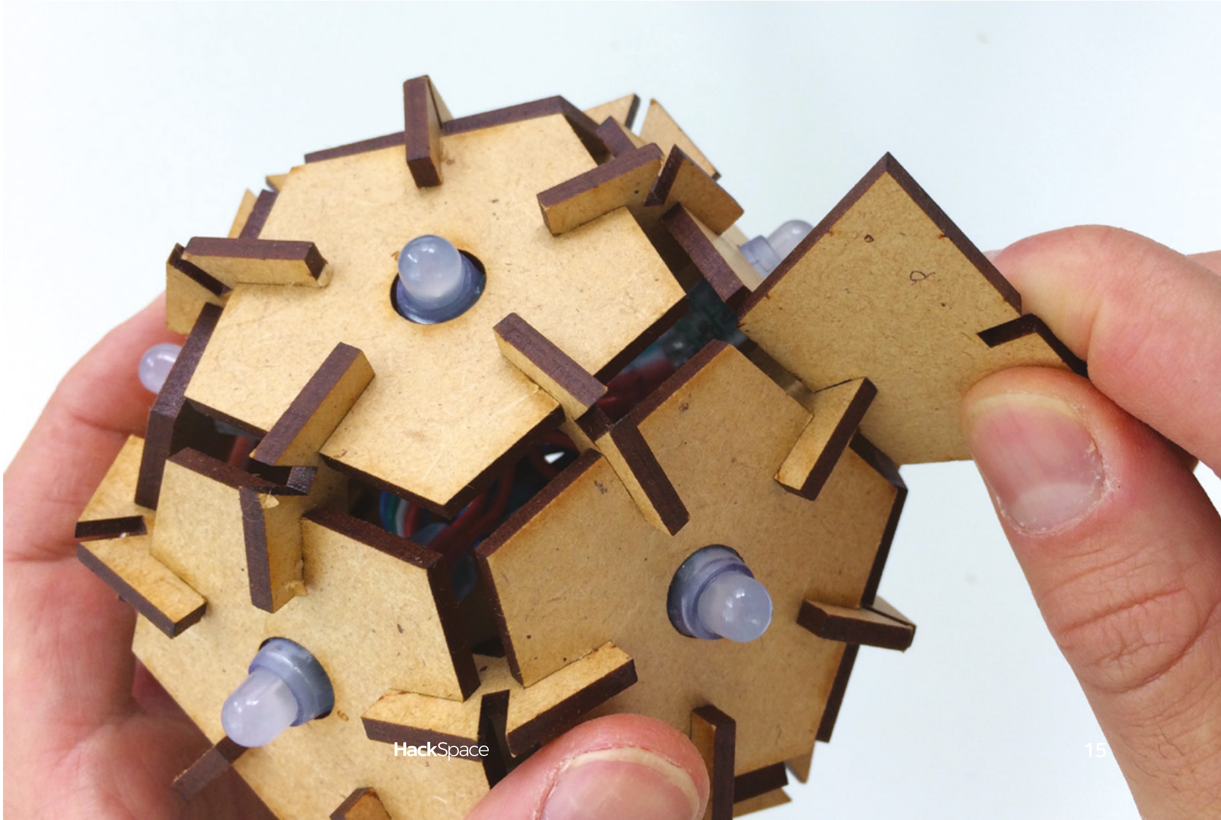
By Jon Bumstead

jbumstead.com

A few years ago, I built a one-metre diameter LED geodesic dome that was touch-sensitive and played music. However, it was a tough, time-consuming build with lots of soldering, cutting and measuring joints, and programming.

So, I designed and built a smaller, less complicated interactive LED sphere made out of MDF and plexiglass that locks into place. I am calling it the 'Bucky Glow'. It's a dodecahedron consisting of eleven LEDs controlled with an Arduino Nano. Using its breakout pins, you can connect the Bucky Glow to sensors, motors, and MIDI jacks, and then customise it to be musical and interactive.

My hope is that anyone can build and then program the Bucky Glow for light displays, or for learning more about coding and electronics. The only tool required for assembly is one Allen wrench. No glue! I am still working on making the Bucky Glow kit, but for now I am selling completed Bucky Glows on Amazon and Etsy (as 'One Bit Kit'): **hsmag.cc/GbhKzZ**. □



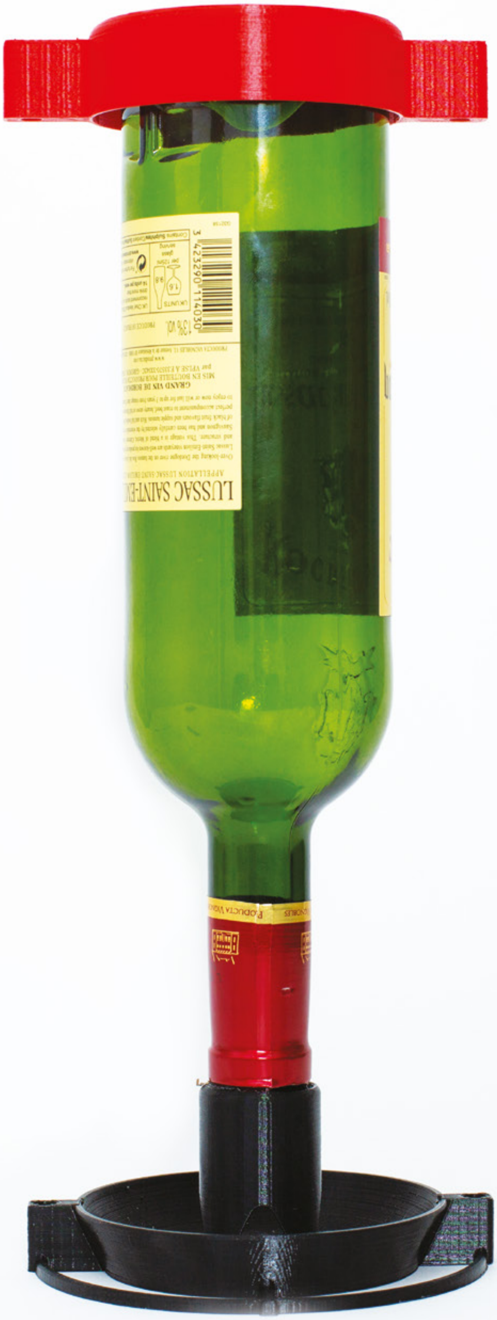
Objet 3d'art

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

Nature is well and truly up the metaphorical creek. Due to a combination of agricultural monocultures and pesticide use, flying insect numbers have declined by a catastrophic amount. This is having knock-on effects on all sorts of wildlife, including birds. With nothing to eat, they're dying out. Farmland birds in Britain have declined by over 50% since 1970, and the decline is even more pronounced in countries with bigger farms, such as Germany and France.

We can do something about it though. John Beardon has produced a hanging bird feeder that combines our love of nature and alcohol, being built as it is around a screw-cap wine bottle. He's even made a bespoke funnel for filling the bottle with nuts and seeds. □

hsmag.cc/pPAYgY



You might have noticed that it's been a bit warm recently. Uncomfortable as it is for humans, for hairy dogs without our next-level homeostatic capabilities, it can be fatal. Do your dog a favour by printing one of these bottle holders, which also works as a bowl, created by Chad Ruble.

Before you fire up your printer, beware: this design is meant for a 16.9 fluid ounce bottle, which is 480ml. You'll have to make this a couple of centimetres longer if your dog rehydrates in metric. □

hsmag.cc/tTiohC

Meet The Maker: Rachel 'Konichiwakitty' Wong

Proving that electronics can be cute

Rachel Wong, also known as Konichiwakitty, is well-known, both in the online maker scene, and at conferences and exhibitions across the UK. We caught up with her to learn about her maker business, where she gets her inspiration from, and how she wants to help the next generation get making. This is what she told us…

I spent a year away from work, not doing anything apart from recreation. Basically, I was a workaholic and didn't know what I liked apart from my work, so I spent a lot of time trying to know what I liked. I tried various things – card making, paper art, and a lot of reading as well. As part of jewellery making, I was already making elaborate headpieces [before I got started with electronics].

I was also playing video games at the time and I wanted to get hold of the Nintendo Classic Mini, but they were completely sold out, but you could make one with the Raspberry Pi. That was when I first heard of the Raspberry Pi. I got it, and went to the Raspberry Pi fifth birthday event. From there, I got to know more people and more electronics – it was really easy to get into. I think the hardest part is programming, but in terms of how circuits work and that sort of thing, it was very easy. I started incorporating it into my jewellery.

> "One of the reasons that I do conferences and exhibitions where there are children is because **I want them to know that when I make with electronics, it doesn't have to look bad**"

I like things that are kawaii – that's cute in Japanese. I like the Harajuku style – it's usually very bright and quirky. From that, it only makes sense to try and enhance clothing which is bright already by adding more reflective material, as well as adding LEDs and making electronic fashion look good.

ON BUSINESS

I sell bears – they were originally called Gummy Bears because they look quite gummy (these are made from resin), but I have rebranded them as Kira-Kira bears. Kira-Kira means shiny in Japanese. I sell them on Etsy. Each bear is unique – it has its own serial number, and an authentication certificate signed by me.

Because I'm doing a PhD in stem cell research – that's my full-time job – I haven't done as much in terms of making as I have sales. I'm planning on doing more exhibitions and things to promote my products. I'll be at Hyper Japan, which is a Japanese convention in London that happens twice a year. Last year, when I wore this to Hyper Japan, a lot of people were asking me where I got them from – the ears! This year I'll be selling there because it's a very targeted audience, and a lot of people who go there like the same things.

One of the reasons that I do conferences and exhibitions where there are children is because I want them to know that when I make with electronics, it doesn't have to look bad. I feel like a lot of young girls think that it's going to look bad – just a bunch of wires. But I believe you can →

Above
A wardrobe full of LEDs (oh, and some clothing)

make it look aesthetically pleasing, and be a fashion statement. Electronics can look good – be girly or just look good. Also, a lot of girls think when they see me, 'oh you do fashion, you do beauty', but that's not the case. I want them to know that you can be incredibly intelligent and nerdy and do electronics or fashion or anything else. I'm doing a PhD in stem cell research – I use stem cells to grow eyes in the lab. A lot of people don't think that when they first see me. People thing scientists are grey old men. That's not the case.

I try to do [outreach crafting work] through Raspberry Jams or workshops to encourage more girls, but I was actually quite surprised – a lot of boys like to do crafting as well. I feel that because they're deprived of a chance to craft, alot of boys want to spend time crafting. If you give them a bunch of craft materials, they want to be able to be free with it and craft with it. In electronics, there are certain rules that you have to follow.

If I give you a bunch of materials like cardboard, coloured paper, pompoms, googly eyes – what are you going to do with it? What are you going to build? I was so amazed that the boys spent more time making craft than the girls did. I don't know if [the girls are] so used to seeing crafting done in a certain way that they'd do it in a way where everything has to be symmetrical, whereas the boys are like 'I'm going to make this cyborg that has a wand and is going to kill the other cyborgs.' There's the whole fantasy world, and they'd spend three hours while the girls would be done in an hour because they're kind of afraid to explore. I think it's great – through these workshops – because you

Right
Electronics don't have to be complicated to look good

get to study and understand what the kids want. For me – I have this scientist mind to look out for this kind of thing – I would cater to and style the workshops in such a way that I can adapt it to both the boys and the girls.

LOOKING TO GO PRO?

Personally, I'm not sure I can give advice because I'm not super-successful. I've attended some talks. I think one of them was by Rob Ives [at the Guild of Makers launch event]. He said that all you need is 1000 fans. If you have 1000 fans supporting you, you'll have enough to have a business that will support you. You don't have to sell something so cheap you have 100 000 buyers. All you need is a group that loves what you make and will support what you make. □

Above
When normal LED strips aren't enough, there's always the Pimoroni Unicorn to increase the density of flashing lights

FOLLOW ON

You can follow Rachel's adventures on Twitter at **@konichiwakitty**, or see her products on Etsy at **hsmag.cc/vdwoaF**

Three monthses

Listen up amigos, great things come in threes...

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**

"You can do anything for just three months. And Life is just a succession of three-monthses." Tweeted Nessa Cason (**@SuperScienceGrl**) recently.

The phrase 'three monthses' has a lovely comfortable Winnie-the-Pooh feel about it. They're simple to visualise. Things of the unexpected, or unknown unknowns, are less likely to occur in that time. In three months, things are pretty much going to be how they are now, with maybe a slight variation in the weather. So to change something in my life and 'only have to do it for three months' is possible. There are various '100 day' challenges out there, but, to me, three months sounds a lot less daunting. There's only three of them, for a start.

Nessa readily agreed that the 'three monthses' was "shamelessly stolen from Malcolm Tucker's 'life is just a succession of five-minuteses'." But threes are better. Lots of things come in threes – wishes, primary colours, little pigs, and blind mice, to name a few. In photography, there's a rule of thirds. In storytelling, the 'rule of threes' is known as a triad, and 'the three R's' used to be Reading, Writing, and Wroughting (making). If only that had remained the case...

I have found it also only takes three days to go from 'this is really hard, awkward, strange' to 'this is okay, I can do this'. And by a day, I probably mean eight hours. A while ago I completed a bushcraft survival course where, using nothing but our skill, knowledge, and determination – and a knife – we made fire, built a shelter, foraged for plants, and tried to work out how to catch a fish. After three days (and because we had found some carbohydrates), we were almost comfortable. Everything we did after that made life a little more pleasant.

The same when I started soldering – I had designed and ordered a large pile of Open Collector Driver PCB's (see p44 **hsmag.cc/YSwxAq** for the circuit) so I could control small motors from a Raspberry Pi. After three days (I took breaks!) my soldering skills had improved from 'I need to gear myself up to do this' to 'oh, the soldering iron is already on and in my hand' without me thinking about it.

Lots of things come in threes – wishes, primary colours, little pigs and blind mice, to name a few

And to make this a triad, there's the first three days of a new job. Day one is daunting, you're learning important things like where the toilet is, what the WiFi code is and so on. By day three, you even know the name of the person you are sitting next to and may even be recognised at reception.

These rules are just guidelines, of course. You won't get the third degree if you break them... □

What is it good for?

What a trade war means for makers and educators in the US

Bunnie Huang

@bunniestudios

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

The new 25% tariffs that went into place on 6 July are bad for US-based makers and educators. The list of goods taxed includes the import of basic components, tools, and sub-assemblies from China that form the bedrock of any workbench or classroom activity (see **hsmag.cc/GAqXsB** for a full list). The list includes soldering irons, primary batteries, oscilloscopes, capacitors, resistors, LEDs, wires, connectors, many types of transistors, and assembled circuit boards, like certain models of Arduino and Chibitronics products. Fortunately for the Raspberry Pi community, Raspberry Pis are made in the UK, and so will be exempt from the trade war, for now.

While many of the components could be sourced from other countries, China often provides the least expensive option for classroom staples like LEDs, batteries, and resistors. This means that tech classrooms operating on a fixed budget will see substantial cost increases for basic consumables this coming semester. Unfortunately, most public school teachers do not have flexible budgets; budgets already tightened by Trump administration reforms to the US educational system have fixed limits set by voters, with budget increases typically set through a voting process that is reactionary at best. As a result, these price increases will simply mean less material for teaching, or perhaps the outright cancellation of some science and technology curricula. This reduction in access to science and technology education could hurt America’s long-term competitiveness in technology.

Ironically, the new tariffs “do not include goods commonly purchased by American consumers such as cellular telephones or televisions”, as stated in the USTR’s press release (the full release is also viewable online at **hsmag.cc/xkmPIM**).

This policy is short-sighted. Big companies, with the resources to organise thousands of overseas workers making TVs and mobile phones, will have their outsourced supply chains protected, but small companies that still assemble valuable goods from basic parts inside the US are about to see significant cost increases.

> **While many of the components could be sourced from other countries, China often provides the least expensive option**

Ultimately, this policy encourages companies of all sizes to build their products somewhere outside the US.

Normally, one would expect finished goods to be taxed, with proceeds used to subsidise the cost of raw materials, improving the competitiveness of domestic manufacturing, and encouraging the onshoring of IP and jobs. Thus, it seems this trade policy is not designed around any sound economic principle; rather, it’s meant to appeal to voters.

Therefore, from the standpoint of the Trump administration, the metric for success or failure of its trade war policy is going to probably just be the outcome of the November mid-term elections, and little else beyond that. □

Letters

ATTENTION ALL MAKERS!

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

WE ARE THE ROBOTS

If you've not done so already, Google 'hall of the mountain king Manchester robot orchestra'. It took me right back to the nights and days I wasted playing *Frontier Elite 2* on the Amiga, so much so that I had to keep looking over my shoulder in case I was being attacked by an Imperial Cruiser. Seriously, what a classy way to get people interested in engineering. Hats off to all concerned.

David Smith
Leeds

Ben says: We've wanted to get the Robot Orchestra in the magazine for a while now. It's a brilliant piece of work, especially the hard drive bass section.

STREAMLINING

Dr Rogers's advice for streamlining your life so you can get more done is sound. Every morning I do a Pomodoro for 40 minutes. It doesn't sound like a lot, but it's a kind of mental anchor for the day, so that if I get distracted I've got that micro-win to focus on. I've also found it useful to keep a bag packed all the time. It's rare that I'd ever go anywhere that I can't just buy a toothbrush, but knowing that there's one in my bag already means I never have to waste a moment thinking about it. Again, it doesn't sound like much, but it takes away mental white noise.

Sarah McEwen
Akron, Ohio

MELTING!!!!

Many thanks for a timely how-to on dry ice ice cream. I'm now adding dry ice to anything I can get my hands on, which is making life much more comfortable now that London has literally transformed into an oven.

Richard
London

Ben says: After my mishap with getting cement in my eye (hsmag.cc/lMdDiZ), I'm more safety conscious than ever. Make sure you know what you're doing when handling dry ice!

CROWDFUNDING NOW

BUYER BEWARE

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

RasPiO night light

Never tread on LEGO bricks in the stumbling darkness again

From £26 | rasp.io/nightlight | **Delivery:** October 2018

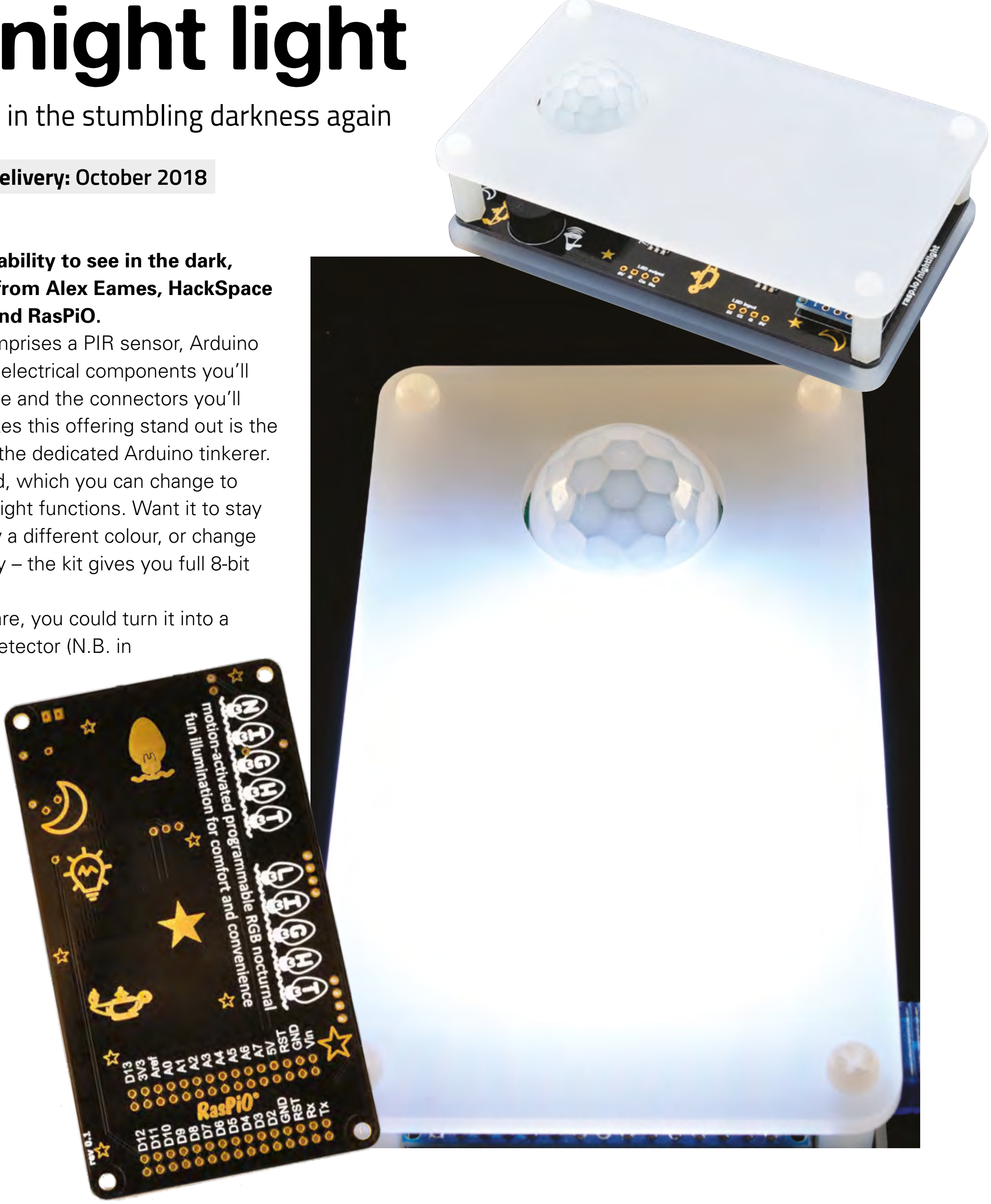
If you're not blessed with the ability to see in the dark, you'll appreciate this project from Alex Eames, HackSpace contributor and the man behind RasPiO.

It's pretty simple – the kit comprises a PIR sensor, Arduino Nano clone, custom PCB, and all the electrical components you'll need to fit it all together, plus the case and the connectors you'll need to mount it to a wall. What makes this offering stand out is the degree of customisability it offers to the dedicated Arduino tinkerer.

There's an Arduino sketch provided, which you can change to configure any aspect of the way the light functions. Want it to stay on longer? Glow more brightly? Glow a different colour, or change colour when it's about to go off? Easy – the kit gives you full 8-bit control of the LEDs (0–255).

If you feel like adding extra hardware, you could turn it into a burglar alarm, dog detector, smoke detector (N.B. in addition to your smoke alarm – we strongly recommend that you use properly safety-certified equipment to keep yourself safe).

It comes in kit form, so you'll need solder, a soldering iron, and a pair of snippers. You'll also need a computer to put the software on to the Arduino, and a 5V power source. Oh, and check out that gorgeous PCB design. □


NIGHT LIGHT
motion-activated programmable RGB nocturnal
fun illumination for comfort and convenience
RasPiO
rev 0.1
rasp.io/nightlight
LED input
D13 3V3 Aref A0 A1 A2 A3 A4 A5 A6 A7 5V RST GND Vin
D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 GND RST Rx Tx

Right One tough design decision was to use a non-WiFi-enabled board, as it interferes with the PIR sensor, creating false positives

Space of the month: Ipswich Makerspace

Ipswich Makerspace

ipswichmakerspace.com

ipswichhackspace

@IpswMakerspace

We started off as a group on Facebook, originally Ipswich Hackspace, talking about how to get a hackspace up and running with the occasional meeting in a pub, bringing our projects to show off, demonstrate, and talk about.

Things changed though when Steve Chalkley hired out a church bungalow for an evening every couple of weeks. We had use of a small room with two tables and barely enough room to walk around. Having a dedicated space to meet encouraged a few of the Facebook group members to attend, and we regularly filled the room and had people show up with all sorts of projects from cannons, bat detectors, RGB light strings, and robots. Oh, we also had someone who showed up with an ultrasound generator, and they immediately teamed up with the bat detector so they could each test out their own devices. This showed us instantly the value of getting a hackspace/makerspace up and running. People working on projects on their own is all well and good, but as soon as people team up, share knowledge, experience failures and successes, this is where the magic happens. It's a win-win for all.

A CHANGE OF NAME

We renamed ourselves Ipswich Makerspace, as we felt it sounded a bit more friendly. We quickly migrated into the church hall, which had more room and tables. Again, we had some good meetings and started to acquire some equipment, including a 3D printer. Hiring the hall out for discrete meetings did not give us much flexibility, and not having permanent storage meant having to lug all our equipment and projects to the hall for each meeting. Although this was okay for the die-hards, it was an effort and was putting people off from attending.

With some luck, we found someone willing to let us have a permanent residence for a short duration to get ourselves together. We now had two rooms and a kitchen. We fitted out the two rooms, which could be permanently set-up with equipment ready to use. For the first time, we had Raspberry Pis available for use, along with a small electronics work station and the Velleman K8200 3D printer.

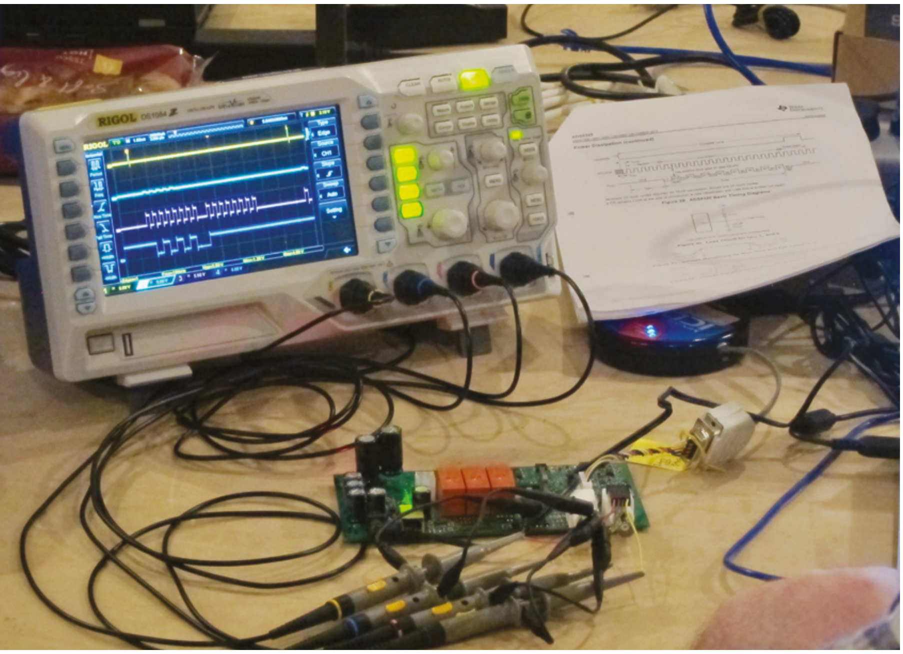
This did us well and we lived here for about one year. Having a permanent space allowed us to open more frequently, with sessions every Thursday evening, some Tuesday evenings, and the occasional Saturday. However, we outgrew the space and the opportunity arose to rent our own space, rather than 'borrow' a couple of rooms.

In March 2017, after raising significant funds through selling Founders' Bonds, we took the plunge and paid for one year's rent for our current space at Dove Street Studio. The space has really flourished since we moved in, and we are now open every Tuesday and Thursday evening and most Saturdays during the day. We also have a significant amount of equipment, including a wood workshop, laser cutter, several 3D printers, electronic work stations, Raspberry Pi stations, and lots of associated tools and equipment.

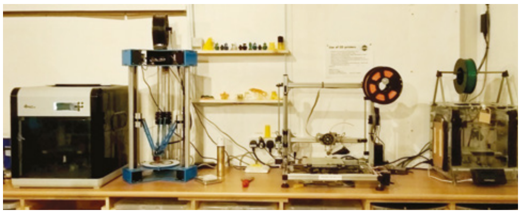
> "For the first time we had Raspberry Pi's available for use, along with a small electronics work station and the **Velleman K8200 3D printer**"

It's very tough keeping the makerspace up and running, with a few of the key founders doing ninety five percent of the work. Membership has increased steadily, and we need this to continue if the space is to continue in its current location. We have been kindly donated many bits of kit and equipment, but the running costs such as rent, internet access, insurance etc. cost so much. We are looking for any kind of sponsorship, members, or any other input that could help. If you like the sound of being part of a lively, friendly makerspace with lots of members with a wealth of experience, pay us a visit. →

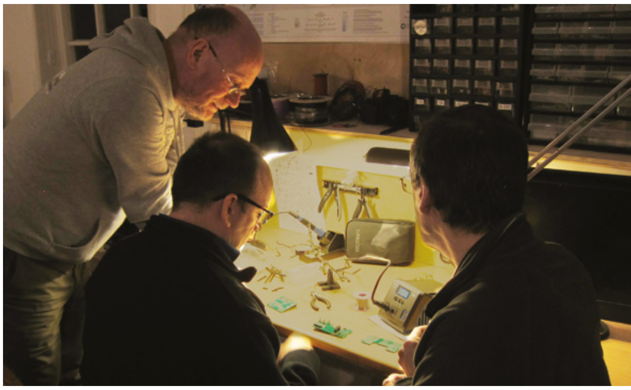
Above
All makerspaces need a vacuum cleaner and someone who isn't afraid to use it



Above You can never have too many laser cutters



Below
Pound-shop hacking challenges were instrumental in getting Ipswich Makerspace off the ground



CONTACT US

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

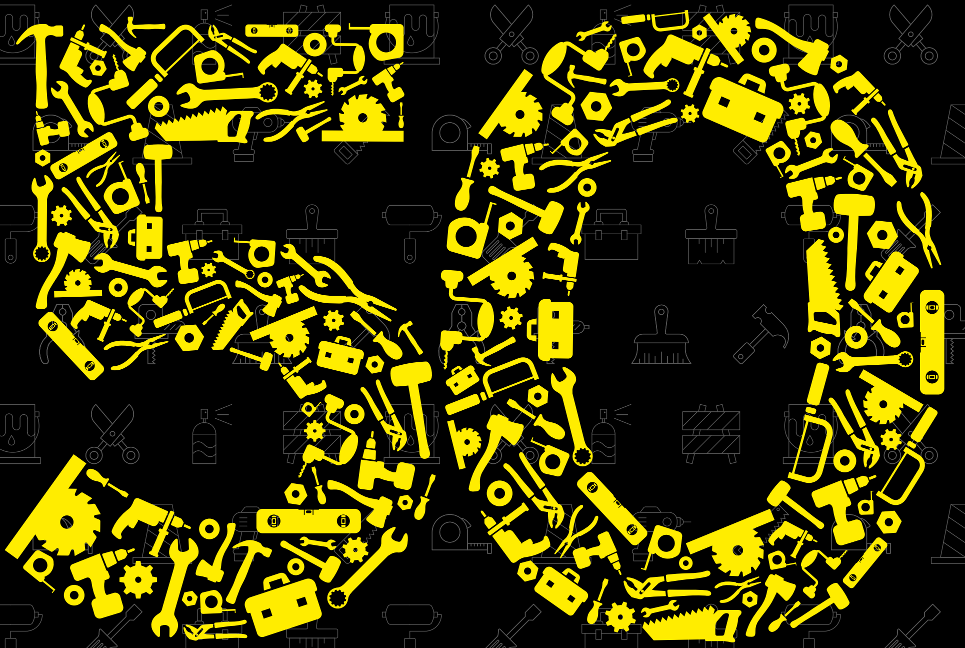
Over 10,000 Different Boards and Modules In Stock
NO ONE
KNOWS
BOARDS LIKE
DIGI-KEY!
Open Source Community Boards
Evaluation and Development Boards
Wireless Modules
Single Board Computers
Reference Designs
Programmers, Emulators and Debuggers
DESIGNS START HERE!
DIGIKEY.COM/BOARDS
Digi-Key ELECTRONICS
MAKER.IO POWERED BY DIGI-KEY
#MAKEWITHDIGIKEY

HackSpace
TECHNOLOGY IN YOUR HANDS

LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future



TOOLS FOR BETTER MAKING

Everything you need to build (almost) anything

Jen Botezat

@neurojenerator

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.

We searched high and low – from small studios to young ventures to vast makerspaces – on the quest to discover which tools makers regularly use to create cool things.

Makerspaces have been popping up everywhere. They range from well-funded co-operative initiatives, to small garage workshops. The diversity of makers, and spaces where they create, is so huge that we have only included tools that are most commonly used, based on our research. All photos are of tool models found in maker studios.

Here, we have distilled our research into five broad areas, each with its specific top ten tools. Your project will lead which specific tools you get. Only purchase new tools if you can't get the job done without them!

If you're interested in woodworking, we incorporate tips from globally renowned woodworking artist Anastasya Martynova, and traditional string instrument maker Andreas Hudelmayer.

For desktop fabrication, we visit successful 3D printing studio Champion 3D to determine which tools are key. We also look beyond 3D printing, at all the different ways you can make working prototypes and functional products right from your workbench.

To compile the electronics section, we consult with outstanding hardware hacker 'Lady Ada', Limor Fried, creator of pioneering open-source hardware company Adafruit Industries.

Textiles and wearables draw inspiration from Becky Stern's work, with a special look at specific machines and tools preferred by makers.

And, if you'd like to try your hand at ceramics, we visit seasoned ceramic artists at The Kiln Room Clay Store in London, and discover what a co-operative ceramics studio can offer its members.

We have focussed mainly on hand tools here, as these are the most accessible to beginner makers. Although you may eventually opt for power tools, such as a drill-press or band-saw, to help you get the job done faster, these aren't strictly necessary when you're starting out.

There are also many ways to acquire these tools, depending on your budget. Donations from companies, the wider community, and your members, if you are running a large space, are great places to start (one makerspace received a broken laser cutter as a donation, and went on to successfully repair it). Discount stores will provide you with all the expendable essentials like scissors, hot glue gun, and sundry materials to make with. If all else fails, the dump offers a myriad of items perfect for recycling, like old furniture. It's all about getting creative with the things you have to hand or in your local area. →

> “The diversity of makers and spaces where they create is so huge that we have only included tools that are most commonly used, based on our research”

WOODWORKING

We whittle down the essential tools for crafting with wood, one of the oldest human pastimes

CHISELS

To start working with wood, you need surprisingly few tools. A sharp chisel set will be indispensable among these, lying at the very heart of your woodworking toolkit. At their most basic, chisels chop off hard material, like wood, when struck with a mallet. Chisels are extremely versatile, with edges (or bevels) of different sizes, heights, and slopes. This allows you to work down from slashing off large chunks of wood, to progressively thinner and finer shavings. Chisels have names like bevel-edge chisel, mortise chisel, fishtail chisel, and gouges that are each suitable for specific applications. For decorative items and artwork, use a carving chisel set as your starting point. For joinery and furniture-making, invest in some quality bevel-edge chisels.

The key to success with chisels: make sure they are really sharp! A dull chisel will cost you much unneeded effort and won't create the effect you imagined. Especially if you have inherited your chisels or bought used ones, sharpen them well before doing any work with them. Pro tip: if you work really patiently, as this traditional craft requires, you shouldn't have to sand your finished piece. Rather, use the right, progressively smaller chisels.

"I don't have much use for sandpaper in my studio," string instrument maker Andreas Hydelmayer says. "Rather, I take the laborious, but ever finer route, of carving the wood with the right chisels. This preserves the natural grain of the wood."

CLAMPS

Clamps secure two pieces of wood in place for drilling, sanding, or gluing. One of the most useful tools in the workshop, clamps come in a variety of types. Bar clamps span long distances. C-clamps are strong and tight. Corner clamps hold pieces at a right angle. Spring clamps are very general and have different sizes. Vices attach to the workbench. You can never have too many clamps!

SHARPENING STONES

It's critical to keep your chisels razor-sharp. Accomplish this by sharpening them regularly with metal-grinding sharpening stones. Start sharpening with the more aggressive diamond plate stone, then the Indian stone, then the Arkansas stone. Finish by rubbing your tool over leather (an old leather belt works). If your chisels are really dull, use a belt grinder beforehand to recreate their shape.

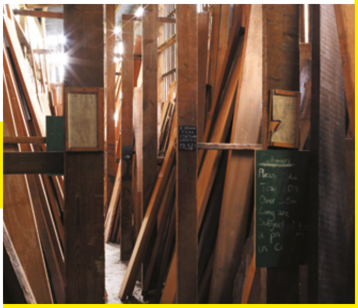
HANDSAW

In woodworking, you start with a block of wood and cut away excess material in progressively finer ways. A handsaw will be your first step to roughing out your desired dimensions and shape. Handsaws come in two types: rip saw and cross-cut saw. Cut along the grain with a rip saw and cut across the grain with a cross-cut saw. As you gain experience, you can use power tools to make cuts easier, but these are not strictly necessary when starting out.

MALLET

Working with a mallet and chisel lies at the very foundation of woodworking. When you whack a chisel handle with a mallet – which is a type of hammer – it drives the chisel into wood, chipping it away with successive blows. As such, you can't be without a good set of variously sized mallets. Start with a large mallet to 'rough out' your shape and chip away large wood chunks. Decrease the mallet size as you reach more intricate details. Use wooden mallets for wood carving, which deliver soft, controlled blows. Never whack your chisels with a metal mallet as this will deform the chisel tip!

WOOD



The type of wood you choose will depend on your specific project. In general, you will use either hardwoods, which come from deciduous trees, or softwoods, which come from evergreen, coniferous trees. For furniture-making, hardwoods like oak, cherry, and maple are ideal, because of their strength and resistance to warping. For wood-carving, lime (basswood) is perhaps the most suitable wood species. It's easy to work – being soft and light – and has a fine, even texture. You can find this wood at a local timber yard, a specialty shop, or a home improvement store. If you're feeling adventurous or working on a tight budget, why not recycle some old furniture wood?

WORKBENCH

A sturdy workbench is another woodworking essential, whether you purchase one ready-made or build your own. Workbenches are most commonly made of a solid hardwood. Make sure that it reaches waist height when you stand. This way, you will avoid back pain and have a solid platform on which to conduct your woodworking experiments. Add a vice to your workbench to secure planks of wood, and a planing stop for plane work.

> **"For furniture-making, hardwoods like oak, cherry, and maple are ideal, because of their strength and resistance to warping"**

DRILL

A standard power drill will come in handy to rough out the shape of your wood, especially where you will have grooves and depressions. Specific drill bits have been designed to cut wood cleanly and accurately. A brad-point bit will be ideal for most jobs. It has a pointy tip which helps to position it on the wood and prevents the bit from sliding off while drilling.

"I use the drill to make initial holes in the wood where needed, which makes chiselling a bit easier," wood-carving artist Anastasya Martynova says, showing a work with cut tree stumps. "I used the drill to create the holes between the stumps."

PLANE

Flattens and smooths wood. Lay your board on the workbench, secure with planing stop, and glide the plane over it with long straight strokes.

COMBINATION SQUARE

Checks, measures, and marks 90 and 45 degree angles. Also useful for angling saw blades and measuring router height.

DESKTOP FABRICATION

These additive and subtractive technologies fit on your table and make it easy to design and prototype your next project

3D PRINTER

A 3D printer is your indispensable tool when it comes to desktop fabrication. No other tool can convert your wild design ideas into physical objects quite as quickly or as effectively. In fact, the rise of 3D printing is widely considered critical to what is known as 'The Fourth Industrial Revolution'. So what makes this machine so special?

In part, 3D printing is an additive technology that builds up a physical object by adding layer upon layer of material. This leads to less material waste than traditional 'subtractive' techniques like sculpting and milling. Moreover, 3D printers give extreme power to makers to prototype and quickly iterate their designs. Feed it a digital model, and watch your 3D printer fabricate something that was previously hard or impossible to make. All by itself. In a matter of hours. Now that's just awesome!

The material choices are vast when it comes to FDM 3D printing, the most readily accessible 3D printing technology. Chief among these is your standard bioplastic PLA (polylactic acid), which is biodegradable. Then there are metal composites like brass, copper, steel, and bronze. You can also 3D-print with cork, wood, carbon fibre, conductive, and glow-in-the dark filament. Material choices are constantly expanding, and you can now even order custom filament that matches your desired Pantone colour. Pro tip: keep your filaments sealed and dry to keep them printing well.

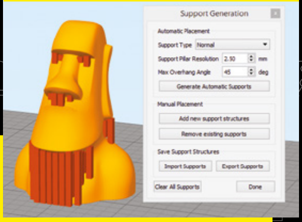
Note that 3D printers will often require replacement parts, such as spare nozzles, PTFE tubes, fans, and glass beds.

CNC MILL

A very popular tool in makerspaces around the world, CNC mills are often the best option for projects that involve solid metal or wooden parts. CNC (computer numerical control) means a computer precisely controls the motors moving the machine head. These mills have the ability to cut deeply into the model, following G-code instructions which you feed in through software. Mills with three, four, and five axes are common, and they will cut through plywood, plastic, metal, and foam. Note: A CNC router is similar to a CNC mill, but designed for softer materials.

VINYL CUTTER

Use a vinyl cutter to cut myriad different shapes and patterns out of vinyl. Start by drawing your design as a vector image and pass it on to the vinyl cutter software. Vinyl is a shiny, adhesive material with an adhesive back, and usually comes in rolls that are twelve inches wide. The software will guide the vinyl cutter blade to precisely cut out your design, which you can then peel off easily from the backing.



Support Type: Normal
Support Pillar Resolution: 2.50 mm
Max Overhang Angle: 45 deg

ADOBE ILLUSTRATOR SOFTWARE

Many times, your design will be two-dimensional: an illustration, a drawing, or a type font that you will want to cut out. Adobe Illustrator will get you started quickly, with very useful drawing and text tools. Easily create vector images that you can later scale as necessary and input into your laser or vinyl cutter.

HAND-HELD SCANNER

Bring real-world objects into the digital world with a hand-held scanner. To save money, you can convert a smartphone into a 3D scanner. Do this by taking photos of your object at many different angles and patch them together with photogrammetry to digitally reconstruct the object.

HAND TOOLS

Having a variety of hand tools around will help tremendously in working with your desktop fabrication machines. Screwdrivers, wrenches/spanners, Allen keys and files are a good start, but they will vary depending on your specific project. For 3D printing, for example, use pliers to easily remove supports and a BuildTak spatula to wedge completed prints off the print bed.

LASER CUTTER

A laser cutter is super-useful when it comes to manufacturing and engraving flat shapes. The laser can be adjusted in resolution and intensity, which will then either cut or engrave sheets of wood, glass, felt, acrylic, leather, marble, or foam. Laser cutters are perfect for making signs and boxes, engraving glass and wood with custom artwork. Some commonly used models are: Epilog Laser Mini, Rabbit Laser QX-80-1290, and Redsail X1390.

HOT GLUE GUN

An inexpensive but extremely useful addition to your toolkit! Use it to quickly bond together wood, easily upholster objects, construct prototypes, and create flexible moulds. To make a mould, squeeze a thick layer of hot glue onto a silicone surface, press an object into it, and let it dry. Once dry, the object and the glue should peel off easily to create your finished mould.

CRAFT KNIFE

For precise and heavy-duty cuts with a razor-sharp blade. Perfect for slicing off unneeded blobs.

BLUE TAPE

Great for getting your 3D prints to stick to the print bed. Create a smooth, continuous surface by laying strips side-by-side.

ELECTRONICS

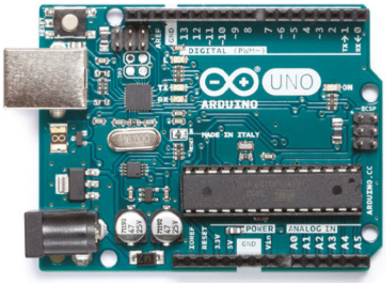
We connect all the dots to bring you this electronics essentials kit. How will you wire it up?

DEVELOPMENT BOARD

You have likely heard of microcontrollers, circuit boards, and inputs/outputs before, but what really makes all these elements powerful is when they are brought together into one easy-to-use device. The two most popular such devices for makers are the Raspberry Pi and Arduino.

The Raspberry Pi runs as a standalone computer with an operating system allowing it to execute multiple complex tasks, but the platform must be booted up and the code read from inside a software application. Use a Raspberry Pi if your project is focussed on software, web interaction, and requires the completion of multiple tasks.

An Arduino is not a self-contained computer and requires access to one in order to read code. It usually does one specific task, but gives you direct access to the input and output pins. Use an Arduino if your project is hardware-driven, requiring lots of sensors and real-world interactions.



The Arduino Uno controls motors, lighting, cameras, and simple robots, while more powerful models have WiFi and Ethernet. Arduino works through a well-documented IDE (integrated development environment) on Windows, Mac, and Linux. It has a massive support community online and hundreds of Arduino expansion options (shields). These are circuit boards that plug into the main Arduino, and extend its capabilities. For example, Adafruit shields control motors and servos, without the need for motor-control circuitry.

Start with a development board that fits your project, and you will soon be on your way to amazing electronics projects.

SHRINK TUBE & HEAT GUN

Much more reliable than electrical tape, use this set to tidy up soldered wire ends and secure connections. The tubes come in many colours too!

DESOLDERING BRAID & PUMP

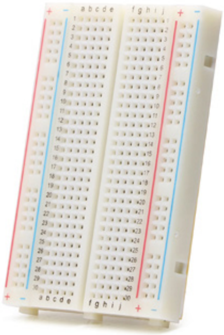
Not everything goes perfectly, and you will often need to correct or change your soldering. Use a desoldering braid to whisk away old bits of solder. Place the braid, which is copper wire coated with rosin flux, over the solder you'd like to remove and heat it up from the top with the soldering iron. Soon you will see it sucking up the old solder via capillary action.

TEST LEADS

Test your circuit using these before applying solder. Some have alligator clip ends, so you can easily attach them to your circuit elements.

BREADBOARDS

Breadboards are ideal for quickly prototyping and testing circuits without soldering. To make a circuit, simply plug in your components and wires in the right sockets to connect them. It's good to have several breadboards on hand to experiment with different circuits. The sockets running along the sides of the breadboard are connected lengthwise, while the internal sockets are connected in rows.



DIGITAL MULTIMETER

Easy-to-use and affordable, a multimeter will be essential to measuring the fundamental parameters in electronics: voltage, current, resistance. It will usually come with two probes: plug the black probe into the ground (COM) port, and the red into the voltage port to check voltage. Easily check your batteries' life by measuring DC voltage.

PLIERS

A variety of pliers will be indispensable as you delve into electronics. Use needle-nose pliers to bend and straighten leads, and hold parts in place while soldering. Diagonal pliers (or snips) will be essential for cutting wires, especially those pesky excess ends. Wire-strippers will remove the insulation off your wires quickly and hassle-free, once you get the hang of it. For precision work, use tweezers to grip and position small bits. Tip: remember to oil your tools periodically with a drop of oil at the metal joints.

SOLDERING STATION

Called the 'mother of all electronics tools', the soldering iron will be essential to permanently connecting up your electronics circuits. To get started, opt for an affordable soldering iron with a stand and a sponge, or brass wire pad, to clean the tip. Use rosin-core 60/40 solder; a 100g spool will be plenty for most electronics work. Note: avoid breathing in soldering fumes by using a fume-extractor, or working in a well-ventilated area. Some solder contains lead, a known carcinogen.

THIRD HAND

Any electronics desk would be incomplete without the third hand. This tool is essential for holding parts securely in place while you solder. It's just the little helper you need, especially when working alone. The magnifier lens helps you see tiny electronics components, and the sturdy alligator clips firmly hold and rotate the parts. "This tool is good for holding small PCBs and wires for tinning or soldering to parts," says 'Lady Ada' Limor Fried. "We have one next to our Panavise Jr (mini-vice). They complement each other well."

> "A varied screwdriver kit will be vital for electronics and repair projects"

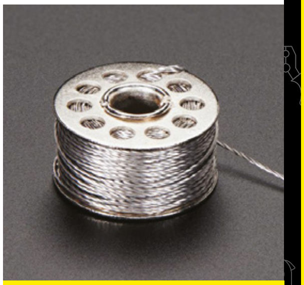
SCREWDRIVER

A varied screwdriver kit is vital for electronics and repair projects. Often coming with interchangeable heads, choose the right one for your project. Adjust connections, secure batteries, and attach circuit elements – all with this handy tool.

TEXTILES & WEARABLES

We pin down the main tools to start you off making fun things to wear on your body

FABRICS & THREADS



In wearable electronics, the wire component of a circuit can be replaced with conductive fabric or thread. ShieldIt, for example, is a super-conductive nickel and copper-plated fabric. Meanwhile, Velostat is a plastic material that changes its resistance when you press on it. You can make some really interesting wearables with these e-textiles.

To sew circuits directly onto fabric, use 100% wool felt as your base material. This will make your circuits soft and flexible. Connect up your circuit with conductive thread, commonly made from stainless steel or silver-plated nylon. Try 3-ply conductive thread by Adafruit, as this will be thick enough for most general wearable applications. This thread serves the function of wires in traditional electronics circuits; however, it is not insulated to begin with.

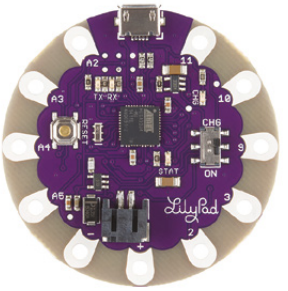
You can check your thread's conductivity with a multimeter. Keep in mind that conductive thread can have significant resistance, unlike copper-based wire, which you should account for when designing your circuit.

You can make your project glow with electroluminescent (EL) elements like EL wire, tape, and panels. These are wearable elements coated with electrically sensitive phosphor. A small amount of alternating current applied through the EL elements makes them glow, and you can easily incorporate them into wearables by sewing them on.

SEWING MACHINE

Repair old clothes and make new ones quickly with this handy tool. Although you can sew a lot of wearable elements on by hand, a sewing machine offers many benefits. Most sewing machines will be able to use thin conductive thread, so you can make long wearable circuits in as many designs as you can think of. Newer machines offer a variety of stitch patterns and designs, which you can program in.

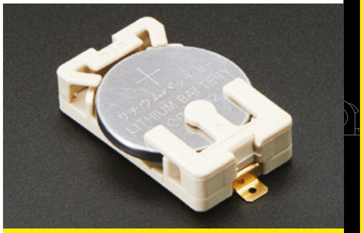
WEARABLE CIRCUIT BOARD



A wearable electronics platform with a built-in microcontroller will be the brains of your circuit. It can read data from sensors, manipulate it, and produce an output. Some popular options for these are the LilyPad USB by SparkFun and the Flora by Adafruit, which are both Arduino-compatible. They come with pins for attaching sensors and components. The pins have holes so you can easily sew them on with conductive needle and thread – put in at least three stitches per hole to make a secure electrical connection.

SEWABLE BATTERIES

There are several wearable battery options, and the one you choose will depend on your project. Firstly, check the maximum current draw of each of your circuit components to determine which battery will supply enough power. There are coin cell batteries with sew-on coin cell battery holders, as well as rechargeable lithium-ion batteries, NiMH batteries, and USB battery packs which plug into the USB port and pack a powerful punch. The battery holders will have metal loops so you can easily sew it onto your fabric with conductive thread and power your circuit. "If you're working on a wearable electronics project, alkaline packs are great," Becky Stern suggests. "These batteries are easily replaceable and the hard plastic keeps everything safe."



LEDS

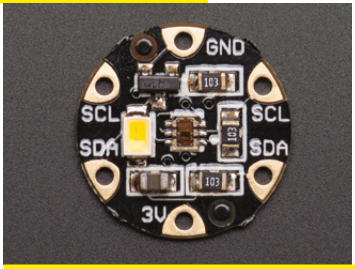
Light it up with sewable LEDs! Both LilyPad and Adafruit offer sewable LED options. Connect them up to your circuits with conductive thread for instant style. Another fun way to light up your project is using fibre-optic strands. These are thin strands of glass or plastic material that quickly transmit light, either at the ends or along their entire length. Usually sold in strands, you can light them up by attaching an LED with a power source firmly to one end of them.

"What's a wearable project without LEDs?" says Ladyada. "Sew a little sparkle in with an Adafruit LED Sequin or the popular Flora NeoPixels. Adding LEDs is a piece of cake!"

FABRIC SCISSORS

The long blades make it easy to cut patterns and long swaths. They also cut cleanly through multiple layers of fabric.

SENSORS

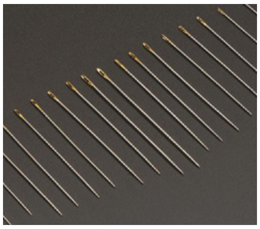


A variety of sensors can be hooked up to your circuit board to give your wearables amazing abilities. Sensors detect anything from sound to temperature, to even your unique fingerprints. With LilyPad and Flora platforms, you can use flex sensors, accelerometer/compass sensors, GPS modules, and colour sensors, among countless others, and program them with the Arduino IDE.

OUTPUTS

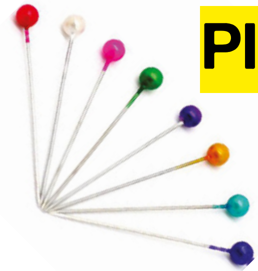
Your wearables platform not only senses the environment but can respond to it via a number of outputs. These are mini-actions that you'll be able to see or feel powered by your wearable perform. For example, the Vibe Board by SparkFun features a small but powerful vibration motor – just give it 5V. You can use this kind of output to make your own anti-slouch device that alerts you when it senses a slouchy back.

NEEDLES



Available in many lengths and sizes, you will want needles that work with standard and conductive thread! Conductive thread is quite thick and frays easily, so use embroidery or chenille needles, which have a sharp tip and large, round eyes, making them ideal for easy threading.

PINS



Super-useful for holding fabrics together, fixing a pattern to a fabric, and just keeping things in place, you will use straight pins constantly for sewing projects.

CERAMICS

Everything you need to get started with one of the oldest human crafts

CLAY

Clay can be found in almost every region of the world. It forms naturally when rocks break down into fine particles. As the particles travel, they accumulate various ions and eventually collect in riverbeds, lakes, and streams – giving each region its own characteristic clay varieties. For ceramics projects, you will come across three main types of clay. Earthenware, such as terracotta, is a low-fire clay that reaches optimum hardness when fired between 950°C and 1100°C. It is a very common clay that is great for construction projects, but is porous and water-absorbent. Stoneware is a mid- to high-fire clay that needs firing at up to 1300°C to harden. It is stronger and more durable than earthenware, and firing makes it water-resistant. Porcelain clay is the toughest and most durable clay, but also the hardest to work. It has a distinctively light colour and translucence after firing, allowing it to transmit light. Pure porcelain requires firing temperatures up to 1800°C, so it is often mixed with other clay types to bring down the temperature. Firing is what turns malleable clay into hard and durable ceramics.

When choosing a clay, check the temperature at which it 'matures', or hardens, as your equipment should be able to achieve that temperature. The maturation temperature of clay is often given in 'cones'. Clays also differ in plasticity and shrink differently with successive firings. Check that the clay matches your requirements before buying.

KILN

A kiln is an insulated chamber that heats up objects, much like an oven. Yet kilns used for ceramics reach temperatures up to five times higher. This high temperature hardens the clay and sets its shape by chemically removing all the water trapped inside the clay. Your piece may need several firings, depending on the effect you are aiming for. After the first firing, the clay will convert to bisque state. You can apply glaze at this stage. The second firing fully sets the shape, and fuses the glaze to the body. You can even build your own kiln with a brick chamber and a sheet-metal roof, but it won't quite reach the high temperatures needed for some clays, although does produce an interesting surface finish.

POTTER'S WHEEL

Building with clay is an extremely versatile craft. One option is handbuilding, which uses techniques like pinching, coiling, and slab building to construct your shape. Yet, for exponentially more options when it comes to building symmetrical vessels and functional items, invest in a potter's wheel. 'Throwing' on a potter's wheel takes some knack, but once you have mastered the basics, you'll be able to make an array of gorgeous objects quickly.

RIB TOOLS

Indispensable for working with clay, a rib tool will be your go-to for shaping clay. Use it to smooth the surface, skim, and compress the clay. Rib tools can be made of plastic, metal, or wood and have straight, curved, or jagged edges (like a metal scraper). They come in a variety of shapes and sizes. Use the round edge inside vessels when throwing to create rounded internal walls. Apply even pressure when working with this hand tool.

WIRE CUTTER

A ceramics essential, wire cutters are basically a thin piece of metal wire attached between two wooden or plastic handles. Use them to cut small chunks of clay from a larger lump and to remove pieces off the potter's wheel when you're done throwing. Press the wire taut against the wheel-head and drag along firmly underneath your piece.

CALIPERS

Calipers are the ideal tool for checking the inner and outer dimensions of your piece. This is especially critical for establishing symmetry and making sure different parts come together correctly. Use them to fit a lid to a pot, or a cup to the depression in a saucer. Once you have measured the width, tighten the caliper to secure the dimension for later use.

ROLLING PIN

Roll out clay slabs with a rolling pin. To adjust slab thickness, position the clay between two wooden slats. Remember to rotate the clay as you roll it.

WOODEN SCULPTING TOOL

Wooden tools and spatulas are like an extension of your finger, but better. They're great for smoothing your object and blending together different pieces. You can also use them for making marks and cutting into clay. Use a wooden spatula or paddle to beat the surface of your clay into shape.

CARVING LOOP TOOLS

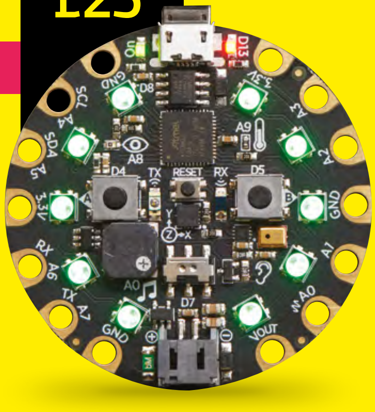
All loop tools remove clay from your clay body, creating concave shapes. They are extremely useful for carving, and you can even make your own out of brass tube and metal wire. Use them to carve into your clay, blend pieces together, and form vessel 'feet'. Pro tip: outline the shape you want to carve out before digging in with a loop tool. To carve, just 'loop, scoop, and pull'.

SPONGE

Use a sponge to keep your clay moist as you shape it.

PICK YOUR TOOLS

What have we missed? Do you have a go-to tool that you simply can't live without? Let us know at hackspace@raspberrypi.org or on the twitter web at @HackSpaceMag



MAKE | BUILD | HACK | CREATE
HackSpace
TECHNOLOGY IN YOUR HANDS
hsmag.cc
August 2018
Issue #09
ULTIMATE LED CUBE
The challenges of building a giant illuminated box
Remap project
Use your maker skills to help others
Electronics for beginners
Discover the might of the electron!
Pi Wars
Inside the ultimate Pi challenge
50
TOOLS FOR BETTER MAKING
Everything you need to build (almost) anything
BUILD A DRUM MACHINE
Kraft homemade electronic beats
AUTOMATIC WATER CANNON
Blast your way to garden supremacy
YOUTUBE
Use a machine to count your online followers
RACHEL WONG
Meet the woman mixing fashion and LEDs
HYDROPONICS MASTER MIDI NAIL CLIPPERS MEASURE INERTIA

How I Made CUBERT

Take flashing lights to the extreme

By Lorraine Underwood

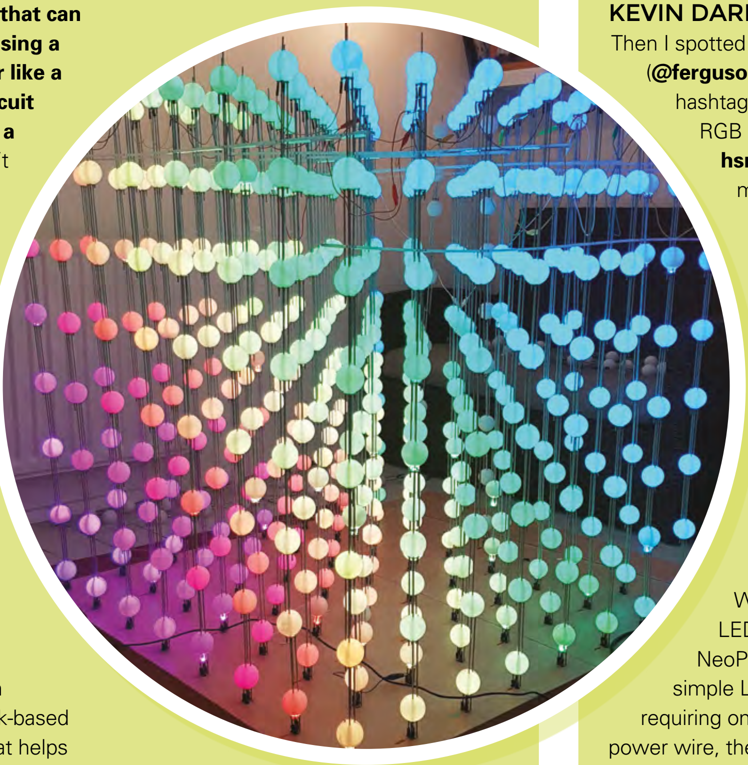
Cubert is a 1 m³ 8×8×8 RGB LED cube that can be controlled using a microcontroller like a micro:bit, a Circuit Playground, or a Raspberry Pi. The build wouldn't have happened without certain people who I've mentioned below and it also almost didn't happen due to several problems I also mention below!

My goal was to build a cube that could be flat-packed into a boot of a car and reassembled easily at events throughout the country, like Pi Wars and Maker Faire UK.

INSPIRATION

In July 2017, 14-year-old Josh Lowe was up for a Tech4Good award for all his amazing work in creating EduBlocks – a free block-based program created with Python that helps students transition from blocks to text programming: **edublocks.org**.

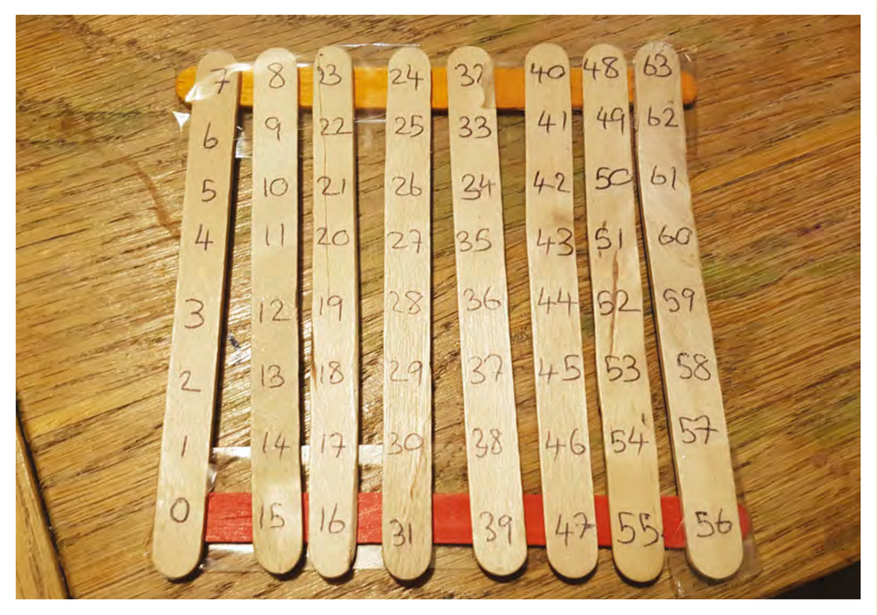
Part of the competition was to tweet #T4GEduBlocks as much as humanly possible. As a huge fan of Josh and all his hard work, I soon got down to business…



Above Cubert in action, showing the full range of colours

DEDICATION 1: DAVID FERGUSON, KEVIN DARRAH & HARI WIGUNA

Then I spotted David Ferguson's tweet (**@fergusondavid6**) which showed his hashtag scrolling through an 8x8x8 RGB LED cube. Check it out at **hsmag.cc/acFkOZ** – it blew my mind! I had seen LED cubes before, but never in RGB. I instantly wanted to make one. David pointed me to Kevin Darrah's website which detailed how to build the cube – **hsmag.cc/JGskjM** – and my heart sank. The cube was made using four-legged RGB LEDs, complicated wiring, and lots of resistors. I had recently fallen in love with the WS2812B LED – a chainable LED that the popular Adafruit NeoPixel is based on. These simple LEDs can be chained together requiring only one ground wire, one power wire, then a single wire between each LED for data. But they were flat LEDs and I knew they would not look as good as 5 mm domed LEDs. So, I shelved the idea until I came across Hari Wiguna's YouTube page: **hsmag.cc/nTKrSy**. Here, I saw that Hari had already made an 8×8×8 RGB LED cube



Left
Step 1 in many great projects is eating a lot of lollies to get the parts for your prototype

using WS2812Bs, but more importantly he made the cube using the wiring as the frame *and* had placed a ping-pong ball on top of every light so it diffused the light, making it stand out better than it would as a flat LED! There also wasn't a resistor in sight. This was it, this was going to be my cube and I have Hari to thank for the hours, the money, and the blood, sweat and tears I've spent on this project!

HOW NOT TO BUILD AN RGB LED CUBE

I started to order my equipment from the far reaches of the internet following Hari's video as a guide. He didn't give step-by-step instructions, but they were enough to get the project started.

Problem 1: The 2.5 mm galvanised steel wire only came in reels, so had to be cut, then straightened using a drill. It was fun for the first 25 metres. The next 175 metres, not so much.

Problem 2: Ping-pong balls are very difficult to drill manually. Like Hari, we made a 3D-printed jig to hold the balls in while we drilled them using a hand drill. Waiting on PCBs to arrive from China, I went ahead and drilled four holes in the top and four holes in the bottom of about 100 ping-pong balls. When it got to threading the balls and the PCBs through the wire, everything was crooked and twisting. We knew the columns might not be straight and might need extra supports, but this was way beyond crooked. Wires were twisting round between lights, and entire columns felt stiff and immovable once all eight PCBs and ping-pong balls were threaded on.

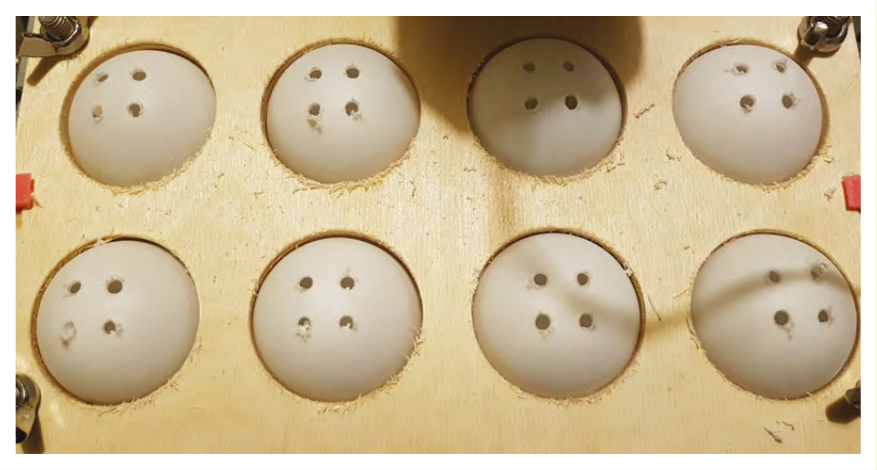
To fix this tangling problem, we decided to cut a hole in the bottom of each ping-pong ball, just big enough for the LED to sit in and to rest on the PCB. This way, the holes would not have to be as accurate. However, a drill was not going to give us this perfect hole and the 100 balls I had already drilled were now useless.

So, of course, we had to buy a CNC miller. We got a fantastic one for around £200 which was also a laser engraver. Luckily, my husband is an electronics nerd and a CNC miller was always on his wish list. So now we had an excuse to buy one together. The miller was a godsend.

HOW WE DRILLED 512 PING-PONG BALLS

This isn't the easiest thing you can be asked to do – certainly, not the least time-consuming. So to make things easier, the next thing we did was that we…

- Used the miller to drill eight 38 mm diameter holes into two pieces of A5 wood
- 3D-printed six holders to hold the sides of the jig – you can get the design at **hsmag.cc/PlmfCD**
- Coded one file to drill four holes into each ball: **hsmag.cc/gDyYZc**
- Coded another file to cut out a single hole into the bottom of each ball: **hsmag.cc/CEAxvH** →



Right
Using a jig can make repetitive tasks easier (or even automatic)

Prepping and Making, Step by Step

The cube is 8×8×8. There are 64 columns to make. Prepping to make eight columns took about four hours. We had to:

- Solder the 64 LEDs onto the 64 PCBs
- Test each PCB
- Drill 64 ping-pong balls, both sides
- Straighten and cut 30 metres of wire
- Crimp bullet terminals onto the feet of each column
- Solder a pin onto the top PCB of each column

Building eight columns took about four hours. We had to:

- Thread eight PCBs and seven balls onto ground and power columns
- Thread data wires through
- Solder the PCBs onto the wires
- Add a capacitor to the top and bottom of every column
- Test every column

The whole process took about six minutes for eight ping-pong balls. It made a lot of noise and mess, but I think overall it was worth it. We now had a way to perfectly drill 512 ping-pong balls!

Problem 3: You cannot solder onto galvanised steel. We had done several prototype columns to test the length of each column, whether the signal would carry through the wire, whether it would fit in the boot of my car (it doesn't, hence I now drive my husband's car), whether it was strong enough to hold eight ping-pong balls and eight PCBs without bending… yet somehow we didn't notice how difficult it was to solder onto galvanised steel. Solder just rolls off it. This was such a deal breaker. We had spent so much money on ping-pong balls, a CNC miller, 600 PCBs, 600 RGB LEDs. We were halfway through drilling the ping-pong balls. The wire had been straightened and cut for at least 16 columns. A quick plea to Twitter told us that you could use flux to help you solder onto galvanised steel. And it worked! Phew! Quick save. Until about… 20 columns later, we realised that flux melts PCBs. The first few PCBs we had done were rotting. They still worked but they looked bad. So, we started to wash each PCB with a toothbrush and some flux remover. Our process was now:

1. Thread eight PCBs and seven balls onto ground and power columns
2. Thread data wires through
3. Add flux to every solder point, using a small paintbrush
4. Solder the PCBs onto the wires
5. Spray flux remover onto every single solder point
6. Scrub every solder point with a toothbrush to remove the flux
7. Test every column

BLACK TUESDAY

When we got to 16 columns, we made a test base to see how everything worked together. This night shall forever be known as Black Tuesday.

My husband wired the test base up and placed eight columns in. Black smoke started to pump out of the base. All eight columns were blown. He fixed the wiring and put another eight columns into the base (why?). He wanted to double-check the wiring, so lifted the base up and it flipped over, crushing and snapping all eight columns. In one evening he had wasted 16 columns – about 20 hours of work.

DEDICATION 2: PHIL UNDERWOOD

I couldn't have done this project without my husband and partner in crime, Phil. He created the PCB files, he coded the 3D files and the miller files. He did the bulk of the soldering of the PCBs onto the wires. He worked tirelessly with me on Cubert throughout early 2018. He never gave up!

Problem 4: The cube would only go red, green, and blue. The problem with your test code only flashing red, green, and blue is that you don't spot for a very long time that your cube doesn't have the capacity to turn any other colour! Our maths had gone wrong at some point and the cube *did* need capacitors. We now had to add one to the top and bottom of every column. A new eighth step in the process above…

DEDICATION 3: THE PEOPLE OF TWITTER!

I started to build the cube in January 2018 and didn't finish it until March. But it wasn't

Below
Inspiration can come from anywhere

Lorraine Underwood
@LMcUnderwood

5 year old David using #Makaton to vote for 13 year old Josh! @edu_blocks #T4GEdublocks @MakatonCharity Retweets count as a vote :)

9:52 AM - 29 Jun 2017

Lorraine Underwood
@LMcUnderwood

#T4GEduBlocks Next time can we get a shorter hashtag?? @all_about_code

8:15 pm - 28 Jun 2017

Lorraine Underwood
@LMcUnderwood

#T4GEduBlocks. Vote for Josh @all_about_code. Retweets count!

10:01 pm - 26 Jun 2017

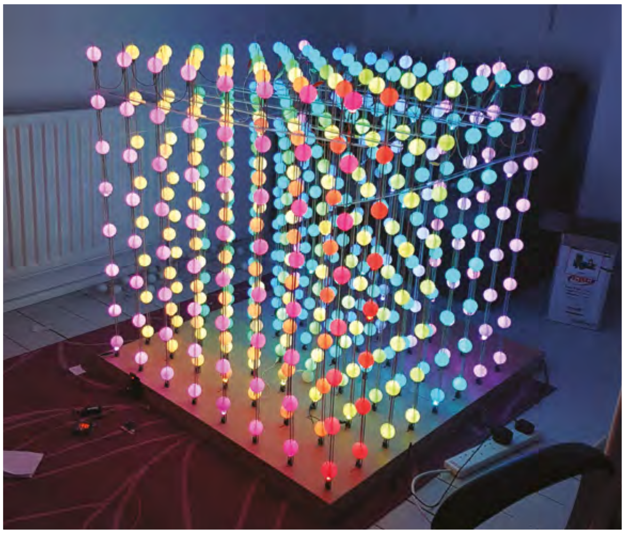
top secret. I tweeted constantly about my failures and my frustration. There were many times in the project I could have given up. My followers felt like it was a soap opera with suspense, drama, and maybe even a murder. They tweeted their cries of support, frustration, advice, and admiration. This cube was truly a community-built project. I couldn't have finished it without the support of my Twitter followers, far and wide.

THE FUTURE

Cubert does work. It fulfilled all the requirements mentioned at the start. But it was a disaster building it! That's why I've built Cubert2.

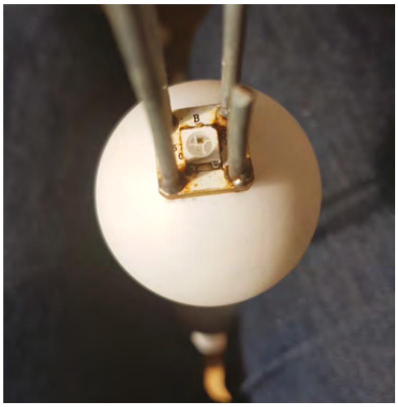
DEDICATION 4: PIMORONI

I wanted to build a more reliable cube, one that would travel well and not need too much upkeep. I found copper-coated iron rods that would be sturdier and easier to solder onto. But to prevent failures, I needed a machine to solder my LEDs onto the PCBs. The cost for this was beyond my budget. Pimoroni stepped up; they redesigned the PCB, got 1027 made, and then soldered the lights on at their HQ in Sheffield-On-Sea. They also added a capacitor to every LED (genius). They did all this, for free. I am so grateful. I have now made Cubert2, Baby Cubert (4×4 ×4) and have plans for micro:Cubert (5×5×5). What a fantastic company. □



Right
The ping-pong ball diffusers make the lights really stand out

Below
A successful test can be as exciting as England winning on penalties (almost)



Above
The four poles slot through holes in each circuit board

Making assistive devices with Remap

Remap is a UK charity that works with skilled volunteers to make personalised assistive devices for people with disabilities

Cameron Norris

@cameronsnorris

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

All Images
Remap CC-BY-SA

Remap, which stands for Rehabilitation Engineering Movement Advisory Panel, was formed in 1964 by Royal Engineer Pat Johnson, with the aim of helping people with disabilities to achieve greater independence and quality of life. Pat's charity now helps over 3500 disabled people across the UK each year, with more than one thousand volunteers designing and fabricating one-off assistive devices, completely free of charge. The charity's aim is to fill the gap where suitable commercial devices are not available.

Before Remap was formed, Pat's sister contracted poliomyelitis, often called polio or infantile paralysis, which causes severe muscle weakness, resulting in an inability to move. After her husband passed away, she increasingly relied on her children and neighbours to help with day-to-day living, and found it particularly difficult to manoeuvre her wheelchair down the steps of her front door to leave her home.

PUTTING MAKER SKILLS TO USE

To help restore his sister's independence, Pat built a wooden ramp up to the front door and installed a simple electric hoist and track system in the hallway, bathroom, and toilet. The hoist and track system was contained entirely within the roof space of the bungalow, with only the hoist cable and controls projecting through a thin slot in the ceiling. Pat was surprised at how effective this relatively simple installation had been for improving his sister's quality of life. It immediately provided her with an increased level of independence that she had previously been unable to experience. Following the success of the ramp and track system, Pat pledged to continue using his skills to help others suffering from the effects of physical impairment.

Today, makers, engineers, and healthcare professionals volunteer at Remap's 80 branches across the UK. These volunteers are tasked with helping disabled people achieve independence and a better quality of life, by designing and making assistive devices for the people's individual needs, which can also include taking part in leisure or sports occupations which would otherwise be impossible. These volunteers work in small local groups, which enables them to discuss projects with others and receive support and advice from occupational therapists to ensure that the solutions they create are appropriate for the people they intend to help.

For those interested in receiving an assistive device, referrals can be made with a local Remap panel, who arrange visits to discuss each individual's challenges and to explore what commercially available solutions have already been considered. Remap volunteers are then tasked with creating an assistive device to solve the specific problem.

THE SNOWDON PUSH

The Snowdon Push is a fundraising challenge organised by the Back Up Trust, which supports people with spinal cord injury and their families. The challenge involves teams of up to 16 people taking on Mount Snowdon, which is the highest mountain in England and Wales, at over 3500 ft. One member of each team is required to be a wheelchair user, with participants pushing, pulling, climbing, and wheeling over approximately eight miles of steep, mixed-surfaced terrain to reach the summit and back down again. Cambridge Remap volunteer, Fred Harrison, was asked to help Rosie Tween, a wheelchair user participating in Women of Altitude, the first all-female team to attempt the challenge.

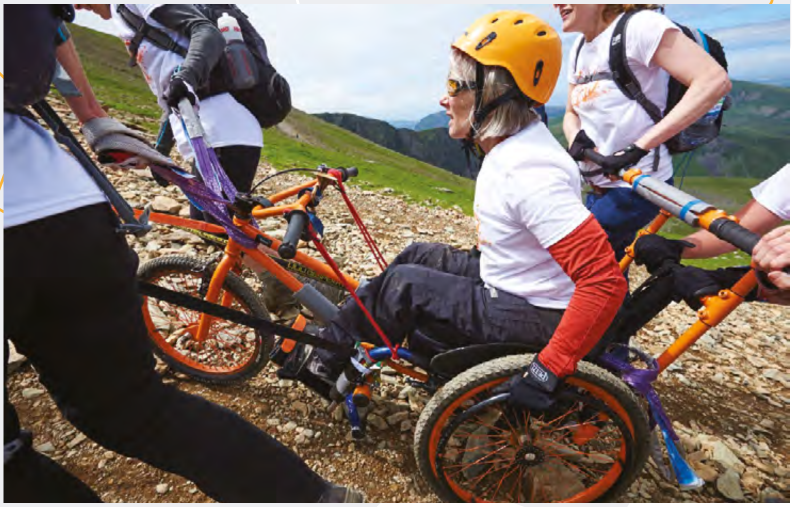
"I was cycling to work one morning when I suddenly noticed I couldn't push the pedals round on my bike; from losing the power in my legs first thing in the morning, I was told later that same day that it was unlikely I would ever walk again," explained Rosie, who suffered a spinal stroke in 2013, before volunteering for the Back Up Trust's mentorship programme and wheelchair skills training group.

EIGHT MILES OF MOUNT SNOWDON

Rosie's everyday wheelchair had to be modified to manage Snowdon's harsh mountainous terrain, while also enabling fellow team members to push, pull, and carry her, up and down the mountain. During her initial discussions with Fred, it became clear that no permanent changes could be made to the wheelchair, as it was essential for Rosie to maintain use of the wheelchair for everyday living.

SIR EDMUND

Fred's creation was nicknamed 'Hillary' after mountaineer and explorer, Sir Edmund Hillary, who conquered Everest with Tenzing Norgay in 1953. Hillary, along with other members of the Everest expedition, trained during the winter of 1952/53 to hone their skills before heading to Nepal. When news of the successful ascent reached the Pen-y-Gwryd hotel near Llanberis, where the team stayed during their training, owner Chris Briggs ordered all the guests out of bed to drink a glass of champagne in celebration.



Above ⬗ **Rosie and team making their way up Mount Snowdon**

Fred, a retired engineer, looked at the possibility of using a bicycle frame as a removable front wheel for safely steering, braking, and lifting the wheelchair during the challenge. After numerous experiments, a modified BMX bicycle frame was securely fastened to Rosie's wheelchair with U-bolts and pipe clamps, enabling the Women of Altitude team to assemble and disassemble the wheelchair attachment during training sessions quickly. A lifting and pushing frame was directly connected to the anti-tip sockets at the rear of the wheelchair and, for the finishing touch, the entire assembly was painted orange at Rosie's request. "It's an ambitious challenge – in fact, you could say it's barking mad – but Remap exists to make things happen for people by making things that aren't available commercially," added Fred. "For me, it was also a real challenge to come up with a customised chair that met Rosie's needs within a pretty tight time frame, so I'm really pleased we've been able to come up with a design that works".

"I was cycling to work one morning when I suddenly noticed I couldn't push the pedals round on my bike... I was told later that same day that it was unlikely I would ever walk again"

RECORD-BREAKING FUNDRAISING

The challenge took seven hours and 33 minutes to complete, and Rosie's team raised a massive £18000 on the day of the ascent, the highest amount raised by any Snowdon Push team. "Your generosity will have a huge impact on the lives of so many people affected by spinal cord injury in the UK. A massive thank you to you all," said Rosie following her team's successful completion of the challenge. →

A LASTING LEGACY

Although the great man died in 2002, it's estimated that over 80000 people have received personalised assistive devices from Remap over 50 years; a stunning testimony to Pat and his life's work.

Remap Awards 2018

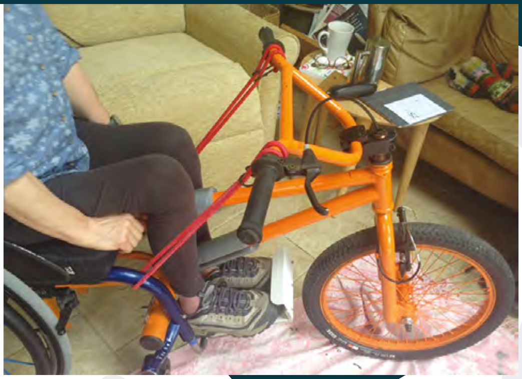
During the Remap Awards 2018 ceremony, held in London on 27th June 2018, the achievements of dedicated Remap volunteers were celebrated with awards presented by British Paralympian, Dave Henson MBE, who highlighted the importance of enabling technology to allow people to achieve their ambitions, whatever their physical limitations might be.

MAKING SURE EVERYONE CAN ACHIEVE THEIR AMBITION

Martin Rees, from the North East Wales branch of Remap, was among the award winners. Martin took on the challenge of modifying a three-wheeled walking aid for Eilian, an elderly man with an upper limb amputation that made it difficult to control a commercially available rollator.

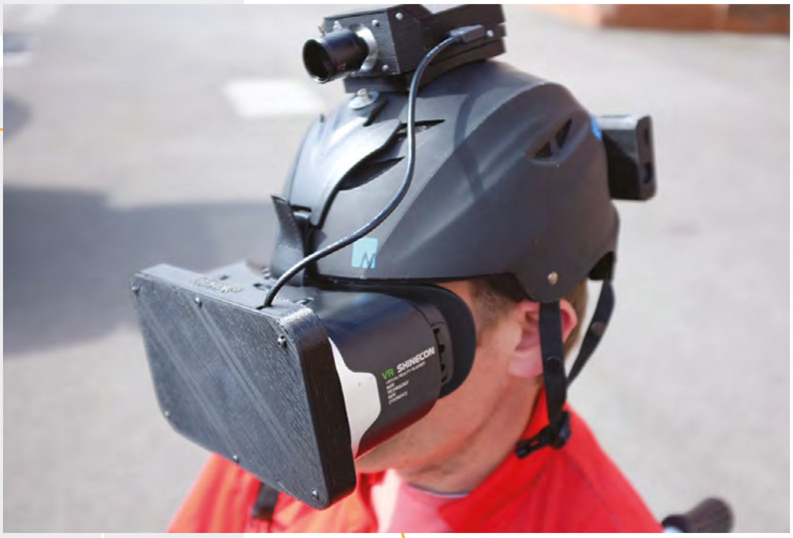
First, Martin modified the brakes for one-handed operation using an off-the-shelf cable doubler, enabling both brakes to be actuated simultaneously using a single lever. A cup-type receptacle was designed to accept Eilian's residual limb, which was later 3D-printed. The finished cup was lined with an adhesive-backed felt layer, to provide additional comfort and support during use.

Another award-winning Remap volunteer to take advantage of 3D printing was Rupert Powell, who



Above
Rosie inspects Frank's work at her home in Cambridge

Below
The modified VR headset is mounted to a ski helmet to display a high-contrast image



THE THIRD DIMENSION

David Martin, who is responsible for the for the day-to-day running of Remap, explained that "3D printing really transforms both the speed and the process of producing… that's probably the biggest impact at the moment, and more of our groups are investing in 3D printers to help them with their projects."

Above ◈
Remap's CEO, David Martin

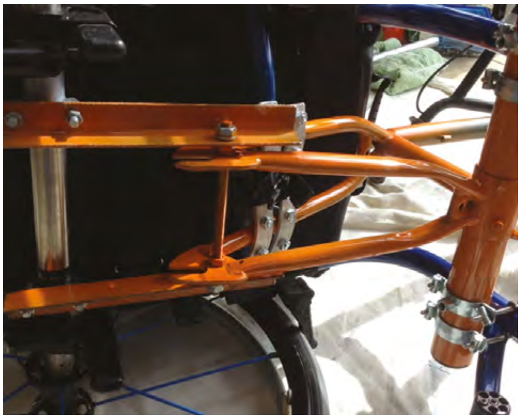
Left ◈
Eilian immediately took to the road with his modified three-wheeler

was given the challenge of providing avid football supporter, Ian – who suffers from severe visual impairment due to mitochondrial dysfunction – a way to watch his home team, Liverpool, play at Anfield Stadium.

Ian's condition means that he is only able to see large, high-contrast images at close range. As a result, Rupert's solution was to modify an existing virtual reality headset to incorporate an LCD video display, using a Raspberry Pi for image processing, and a digital camera with a powerful telephoto lens to enable Ian to zoom in on the action. The modified VR headset was then mounted to a ski helmet, to display a high-contrast image onto an internal LCD screen. Finally, a wireless PlayStation controller provided Ian with control over zoom, focus, contrast, edge detection, and other functions, enabling him to tweak the device's settings to enhance his vision in any given situation.

> “Surprisingly for Rupert, not only did the headset enable Ian to enjoy his time at Anfield Stadium more thoroughly, but it also enhanced his personal independence by enabling him to travel and shop with less assistance from caregivers than he had previously found possible”

Below ◈
Frank used U-bolts and pipe clamps to secure the attachment to Rosie's wheelchair



RASPBERRY PI-POWERED VR HEADSET

Surprisingly for Rupert, not only did the headset enable Ian to enjoy his time at Anfield Stadium more thoroughly, but it also enhanced his personal independence by enabling him to travel and shop with less assistance from caregivers than he had previously found possible.

During the opening speech at the Remap Awards 2018, Dave Henson said, “I am blown away by the ingenuity of Remap's engineers. They are coming up with some really unique, creative solutions, which have a massive impact on people's lives.” We couldn't agree more.

If you're interested in volunteering for Remap, or finding out more about any of the projects mentioned in the article, you can visit their website at **www.remap.org.uk**. □

HackSpace magazine meets...

TIM RICHARDSON

The return of Pi Wars!

Some makers make for the hell of it. Some make because it's their job. And some makers want to make things, but need a reason. It's for this latter group of people that Pi Wars was created.

Pi Wars, for those of you who haven't heard of it, is an almost-annual event based on a set of challenges for robots. Anyone can enter, with the caveat that their robot must be based on a Raspberry Pi. The robots compete in a series of challenges, there are prizes for the winners, and everyone goes home happy.

This simple premise has grown over the last few years into one of the biggest and most important maker events in the UK, thanks to Mike Horne and Tim Richardson, the originators, the organisers, and the builders of Pi Wars. We'll never know how many makers they've inspired, but after our chat with Tim, we do know what they're up to next, how not to win, and why the Pi is such a hit with robotics enthusiasts. →

Above
Pi Wars started humbly, but is now a fixture in the UK maker calendar

HackSpace Can we go right back to the beginning of Pi Wars – it's obviously inspired by *Robot Wars*, isn't it?

Tim Richardson Mike [Horne] and I started CamJam in about May 2013. We both live in the same town. I went along and helped him for the first two Jams, saw what he was doing, and thought we could do more with this. I wanted to bring vendors on board, do workshops and a show and tell, as well as talks. So, in September we ran our first Jam together, and we brought in people like Pi Hut, had a few vendors, lots of talks. I was looking around the event thinking, "I'd like to do more." Because of the workshops where we put things together, I wanted to do something with building things.

I spoke to Mike and said "Do you like *Robot Wars*?" "Yeah I love *Robot Wars*," he said.

I used to watch it with Jeremy Clarkson, and the first series was very different from the other series they've had since. Part of the program was an obstacle course. There were swinging balls, there were drills, there were house robots, and I had the idea that we could do something similar, but with Raspberry Pi-controlled robots. People could make their own robots, on a small scale compared with *Robot Wars*, and complete against each other.

At the end of 2013 we set a date, December 2014 to do it. The reason it took such a long time between setting the date and putting it on is that we had to give people time to build their robots. We had a venue by then at the Department of Astronomy [at Cambridge University], so we went round that and assessed what we could fit in; it turned out that there was room for seven courses, so we started designing to fit the space.

We wanted the robots to be partly autonomous and partly controlled. Autonomous because, well, that's what a robot's about. The controlled side was because not everyone can do the autonomous programming, but hopefully they can do the remote control programming.

HS It's also a great access point for people who want to get into programming an autonomous robot later on.

TR We sat in my garage having our usual beer, deciding what course to do. The classic was line follower. We had a straight-line speed test. We wanted something to do with autonomous control getting a robot to move in a repeatable fashion, which became the three-point-turn test. We did robot golf. We did a proximity challenge, getting as close to a wall as possible.

> "We wanted something to do with autonomous control getting a robot to move in a repeatable fashion, **which became the three-point-turn test**"

Once we decided on those, it was a matter of building them. I'm fortunate enough to have a woodwork area in my garage with a router, various electrical equipment too to make things, so I set about designing and making the course.

My father was very practical. He was an electronics engineer. He made my first bed, see-saws. Once my daughters were born, I made a toy chest for one of them, and made them a large doll's house out of MDF. I've got a very practical 3D design mind. I did CDT at school and learned a lot of it there, but my practical skills were never good, though I am getting better. I like making things, I enjoy making things with my hands. It's so different from my everyday job.

HS There's something very liberating about not staring at a screen when you've been doing it all day long.

TR I design everything in CAD or Visio, or whatever is close enough to what I need to do. I always design it in software first before I cut anything out. You still make mistakes, it just means you need to modify things.

The line follower was me printing out a load of A4 sheets and sticking them together the night before with Sellotape. A lot of the courses were on table tops, and we marked them out using electrical tape, but we had to build the straight-line speed test because we wanted walls on the side. That was about 7 metres long, so it was a matter of finding materials that would be easy to store and also fairly cheap. B&Q have lengths of wood that are fairly cheap, and come in lengths of 2.4 metres, so all our courses are 1.2 metres wide, or multiples of that.

The obstacle course was really Heath Robinson to put together. The first year we just wanted to get something working that hopefully would last the days. By that time we'd got to know Tim Freeburn at PiBorg, who built a laser timing system for the speed test. We sprayed our logo in the middle with a template that I'd 3D-printed.

A couple of parts of the other courses were this ply sheet that I'd bought and mounted on legs. One of the sheets had holes in, the other didn't, because one was for robot golf and one was the proximity test.

For the second time round, one of the things we wanted to do was to improve the look, to make things look more professional. There was a lot of sawing. I do all the building; Mike is not particularly practical. He does so many other things at CamJam and Pi Wars that I can't do. It's just the way it works, we're a good team.

We decided, in 2016, that we'd change the date from December to April, so we missed out 2016 and moved it to April 2017. We wanted to give more schools the chance to get involved, and from the start of the year in September to December isn't a long time to design and build a robot. If you go back to the reason we do Pi Wars, →

Above ↗
Robot golf was just one of the challenges in 2018's Pi Wars

Above
Tim's top tip: don't go cheap on the controller board when building a robot, to keep the magic blue smoke away

we want to give people a reason to make something. You need a reason to do something usually, whether its a genuine reason… I put up LED lights in my cabin and control them simply because I can. I don't need them, but I want to do them. It's to give people a reason to do something, so we wanted to give schools a bit more time, which also gave us more time.

The third instalment came along, we doubled the size of the obstacle course. I decided I wanted a laser cutter. I'd already got a 3D printer by that time, but I wanted a laser cutter. It's big, a 500×700 bed. So I ended up designing most of the course parts to be built using a laser cutter. None of it's really rocket science. The fun bit for me is designing things and turning them into reality. Making them challenging for people, making them robust so that they'll last. A lot of the things we reuse year after year.

HS Do you share the designs with other makers who want to create robot obstacle courses?

TR I make them available if people want them. They're not pretty, they're not perfect. I suffer from being a perfectionist. Mike has to control me here because he knows that I will design and rebuild courses until the cows come home. He has to stop me from doing that. As long as people don't mind them being in all sorts of odd formats, people can use them.

HS What you said about having a reason to use the Pi resonates with me, because the first time I used the GPIO pins on a Pi was thanks to one of the EduKits that you put together for learning GPIO input and output. Before that I was messing about and using it as a desktop computer.

TR Excellent! That's what we wanted. With the Pis at the time, they were slow, and there wasn't an awful lot you could do. There was a fairly early version of Sonic Pi, which was really good. Sam Aaron has done a fantastic job with Sonic Pi.

One of the reasons the Pi is so good is because of the GPIO pins. You just need something to use them with. Mike and I trawled the net trying to find ideas for a workshop or two. We found the bits from China, mailed out the kits ourselves. People booked onto the workshops and paid pretty much what it cost us to get all the parts. There were loads of spreadsheets – that's what I tend to do.

Jamie Mann from the Pi Hut was at the Jam and convinced us to produce a kit for general consumption. I wrote all the worksheets. I'd never used GPIO before, so I had to stand on the shoulders of giants, but the worksheets were all my own language; I wanted to make it as simple as I could so that people could follow along at home. What the parts are, how to do the code, etc., etc.

> **We have enough Pi people applying.** We had over 150 people applying for 76 places last year, all Pis, and that's enough

HS I think you labelled the resistors as well, which I found a great boon, being colour-blind.

TR That's a very good point actually, that's not something that I'd really thought about. But when you've got 330 Ω and 4.7 kΩ [resistors] on you, the 330 will be two colours and then a third colour, whereas 4.7 is different colours on each band. So, if you can tell greys and can count…

One of the things with the EduKits is that we took feedback from the general public and tried to improve the worksheets from there. If there were mistakes or something that didn't work as we wanted it to, we'll redo them.

HS Back to Pi Wars. What software do you use for designing the courses?

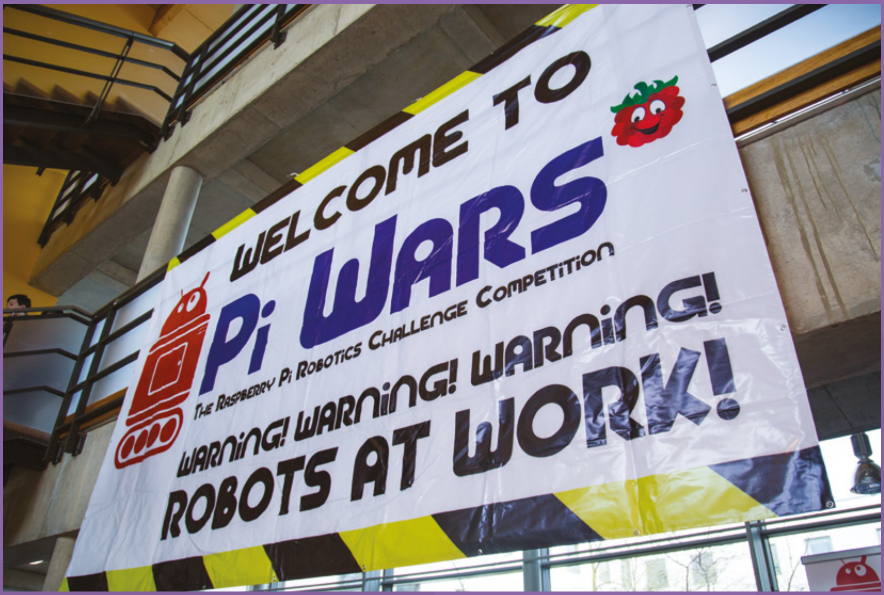
TR For the laser cutter, I used to use Inkscape. It's an open-source product for line drawing. It's a pretty powerful piece of software, even though it hasn't got to version 1 yet [it's at 0.92.3]. But it is a very good piece of software. However, at the last Pi Wars, I met Dominic Morrow, who's part of Smoke and Mirrors [the laser cutting company] and he suggested this other piece of software that now controls my £1600 Chinese laser cutter, which I bought on the strength of a couple of videos online that I saw. Luckily it works!

For the 3D printing side of it, I started out with 123D Design, which I scrapped in favour of Tinkercad, which is a really simple but powerful piece of online software from Autocad. Okay, it's not parametric, which would be good, but it's simple enough that you can design something and then quickly get it out on your 3D printer quite quickly. I do want to use Fusion 360 for all sorts of reasons, but it's just getting time to learn it.

HS Do you have any plans to open Pi Wars out to non-Pi platforms, such as Arduino or micro:bit?

TR We have enough Pi people applying. We had over 150 people applying for 76 places last year, all Pis, and that's enough. If we did open it to other devices, it would mean even more people [being] disappointed. But we have been speaking to quite a few people who have heard about Pi Wars now, as it's become more heard about around the world. They say, "We'd like to do it, but Pis are not that popular in our area; could we do it with Arduinos or micro:bits, that sort of thing?" You can't call it Pi Wars, but apart from that, yeah, why not? Arduino Wars, or whatever.

There are other world robotics events, (there's Robotex). They are any robot of any type, but usually the robots are only doing a single event. So, they're designed specifically for football, or for climbing a wall, that sort of thing, Whereas the Pis in *Robot Wars* – I mean Pi Wars – I keep →


WELCOME TO
Pi WARS
THE RASPBERRY Pi ROBOTICS CHALLENGE COMPETITION
WARNING! WARNING! WARNING!
ROBOTS AT WORK!

Left
The event has moved from December to April, to enable more teams from schools to take part

getting that wrong. I even once printed out hundreds of labels with *Robot Wars* written on instead of Pi Wars. Because they have to be diverse, they have to compete in a number of different courses, they have to be fairly general.

HS Is that also in the UK, Robotex?

TR Not yet. We've been contacted by the guys in Estonia who run that. They've been thinking about getting us involved. We've been contacted by the guys in India wanting us to help run theirs. Turkey at the moment are about to start doing theirs; America, that's another thing.

But Mike and I, because we started out in the Pi community, we like the Pi. We love the Arduino and micro:bit, nothing against them, it's just that's our focus, so that's where we're staying at the moment. Because we started off in the Pi community, we want to stay in the Pi community, because there's enough people applying for it. It's a great little computer.

Below
Sharing = learning. Carry on!

HS It's a great community as well.

TR Yes, that's one thing that's different from most of the other communities. I don't know much about the Arduino community, from a meeting point of view, because I've never seen anything advertised for Arduino people getting together. Same with micro:bit, even though the micro:bit is very popular now in schools. They can hook it up to their current computers without having to buy extra equipment, screens, etc., like they would for the Pi.

HS It's funny how there seems to be a nucleus of people who will work with whatever's cool and interesting at the time. The Southend Raspberry Jam, for instance, grew out of the local Linux User Group. There must be something about the Raspberry Pi that makes it an easier sell than Linux?

TR Yes it did. They came to find out how we run CamJam so they could put on their own events, which has developed into what they're doing now.

HS Have you got any tips for anyone who's thinking about building a robot themselves?

TR If you're not sure that you're going to enjoy doing it, there are some pretty cheap kits out there. CamJam EduKit, of course, I've got to say. It's £18, you've got to build it, you need to supply the Pi yourself and a battery. So that gets you started. You're not going to compete in Pi Wars and get very far with it, although we do have teams competing in Pi Wars.

There are lots of kits out there to make robots, but part of the fun is also making them yourself, so you need to find a good controller board that's going to power the sort of motors that you're going to use. Don't go too cheap on those, you'll burn them out. You can get some cheap ones from China; some of them are okay, some of them... we've known people on Pi Wars who've bought a number of them and not been able to compete because they're burnt out. They've just not been able to take the power when they've stalled a motor or something like that.

But there are some really good ones out there for Raspberry Pi especially. PiBorg are my current favourites, they do fantastic boards. They're really robust. They make them for themselves basically, to do what they want to do.

4tronix do a wonderful range of controller boards. But 4tronix also do controller boards for micro:bits as well, and they're fairly similar. And you get things like the Explorer pHAT from Pimoroni, for controlling small motors.

Set yourself a budget to start with, decide whether you want to buy a kit or build the chassis yourself – the fun part is building the chassis I think. And it also depends what equipment you've got. My first robot I built for the Raspberry Pi, I didn't have a 3D printer or laser cutter, but I did have a router [tool]. So, I designed myself a little box to take the battery that I wanted to use. It had those cheap yellow motors that you can buy (not matched for speed) and glued them on the bottom. You don't need expensive tools to make them, You can make them out of anything. I've seen robots built out of a clock, with motors on the bottom. But you don't need masses of equipment; just make something to do what you want.

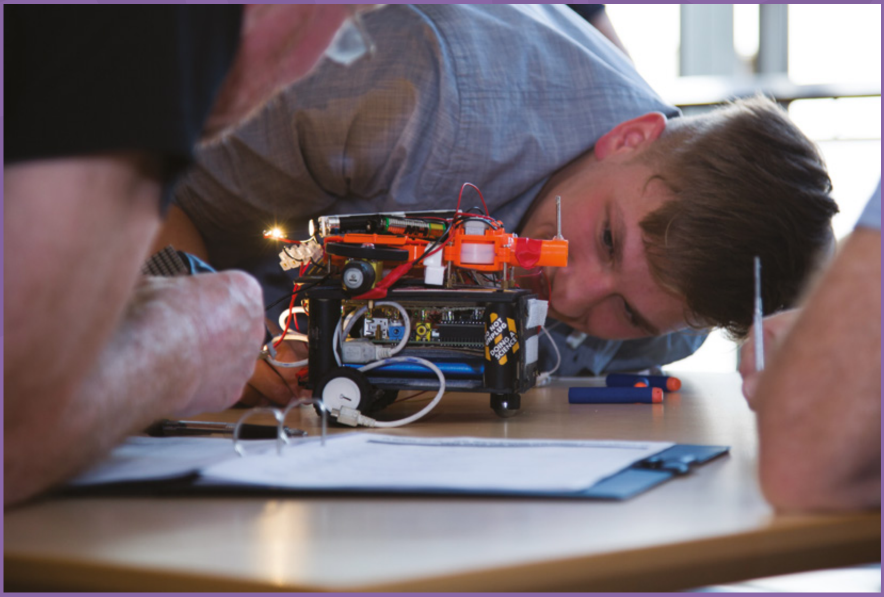
And the biggest piece of advice is to drive it, drive it, drive it. Something that Brian Corteil always advises. He started out as a hobbyist making robots for Pi Wars, and is now running his own small business, making kits. He started making his one robot, designing his own boards, and went from there.

HS Does anyone go to the trouble of fabricating their own custom PCBs?

TR We've had a few with custom PCBs, a few people trying to enter with custom PCBs. Not always succeeding... one of the things they often forget is reliability. You've got to have something reliable, and you've got to be able to replace a part that just burns out. Magic blue smoke is not too frequent at Pi Wars, but it happens. And if you've got a custom PCB, unless you know exactly what you're doing, its so easy to get things wrong. For example, when the motor stalls you get the back surge of current; unless you've protected your circuit well enough it's going to burn out. I had that experience when we designed the new EduKit controller board. When it was originally designed it didn't have a large enough diode for in case the motor stalled. The back pressure on the controller itself just burned it out — fortunately before it went into production, so we caught it.

> "If you want a robot that you're going to drive round your local park, track or six-wheel drives are great. **But on a stickier surface it's not going to do so well**"

Yes, making your own is fantastic, do whatever you like, but there are so many out there right now that are so good, I don't think you need to. There's no reason to unless there's something really specific that you want.



HS Does that mean that the skill in winning Pi Wars now is in the code rather than the construction?

TR Construction definitely helps. Stability, and ability to drive over different surfaces. If you can't get up a slope, you're going to struggle on a few of the courses. If you can't get over a rough surface, you're going to struggle.

There are some robots that are great for grip for the obstacle courses, but when they try to turn, their grip is so good that they can't turn as fast. So there's a real balance to be struck. Do you want to do well at course X, Y, and Z, and know that I'm not going to be brilliant at A, B, and C because of my choice of traction? Track drive, two-wheel drive, four-wheel drive, there are advantages and disadvantages for each.

If you want a robot that you're going to drive round your local park, track or six-wheel drives are great. But on a stickier surface it's not going to do so well.

When the Mini used to rally, they were front-wheel drive and did really well, but then the four-wheel drive cars came in, things like the Audi Quattro, and that started wiping the floor with the Mini. So they decided to build a four-wheel drive Mini. They built it in a weekend, a week, something like that. And when it was built they found that they could only drive it on gravel or a loose surface. Because they'd built it so quickly, the ratios were not brilliant; if they drove it on ground that was too grippy, it used to break the transmission. On gravel it was fine, because you'd just get wheel slip. But on tarmac, they weren't that good. I don't know how many races it went in, not many I don't think, but that shows how you've got to balance grip with lack of grip. □

Right
As well as the challanges, robots are assessed on the quality of their code

NAIL CLIPPERS

A personal care product that maintains your hygiene and doubles up as a nifty little tool

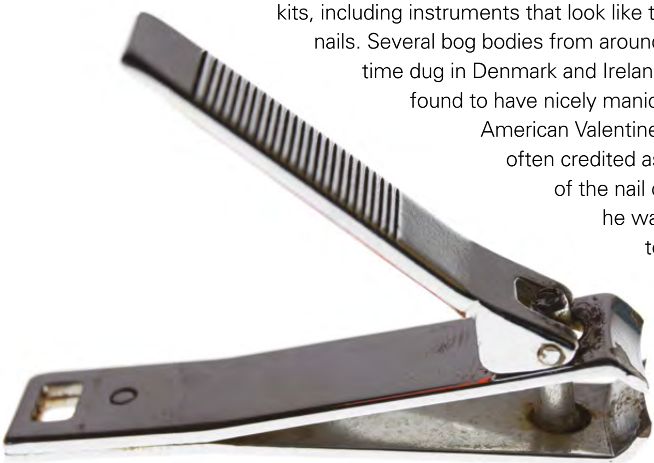
Mayank Sharma

@geekybodhi

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

Nail clippers are one of the few ubiquitous devices to be used by everyone around the house, from your nanna to your cute mutt. While the modern nail clipper is a fairly recent invention, the idea has been around a long time. Although there's no direct record, researchers have found literary references to nail clipping in texts dating back to ancient Rome! And no, they weren't biting their way to well-groomed nails. The use of scissors to cut nails was very popular with barbers in the ancient status-conscious Rome. Closer to home, archaeologists digging through ancient grave-sites around Britain found them replete with hygiene kits, including instruments that look like they were for nails. Several bog bodies from around the same time dug in Denmark and Ireland have been found to have nicely manicured nails.

American Valentine Fogerty is often credited as the inventor of the nail clipper, since he was the first to receive a patent for the device in 1875. He was followed by William Edge in 1876 and in 1878 by John Hollman. While both were improvements to the previous designs, the early clipper looked more like tweezers. It took a couple more years when in 1881, Eugene Heim and Oelestin Matz were granted a patent for their clamp-style clipper that added a lever to the design. But these early nail clippers didn't really take off, since they were expensive to manufacture. This changed in 1947 when William Bassett employed improved manufacturing techniques to cut down the costs and produced the clippers on the cheap. He also tweaked the design to make them more resilient and easier to use. Some of his changes are visible in the modern nail clippers. One such element is the jaw-style design that he improved by adding two nibs near the base of the file to prevent lateral movement. He also made slight modifications to the lever to better catch the thumb, and replaced the pinned rivet with a notched rivet.



The most common nail clippers are made from stainless steel. The various components of the clipper are fabricated automatically, but are assembled, sharpened, and tested manually. Besides the clipping jaws, a nail clipper also has a few other tools, pretty much like a Swiss Army knife. There's one to clean dirt from under the nails and a miniature file to manicure the roughly clipped edges. And after they've used them to do their nails, innovative makers will use them for creative purposes…

ACCESSIBLE CLIPPERS

Hisashi met Kuriki in 2012 in the open fabrication laboratory in his university and was pulled by his positivity, despite a cerebral infarction in 2009 which left Kuriki hemiplegic. In the three years since, Kuriki had adapted to his disability and relearned many major tasks. "But he faced minor obstacles with everyday life tasks like typing and clipping nails," recalls Hisashi. So he made him an accessible nail clipper that Kuriki could operate with minimal support from his paralysed hand.

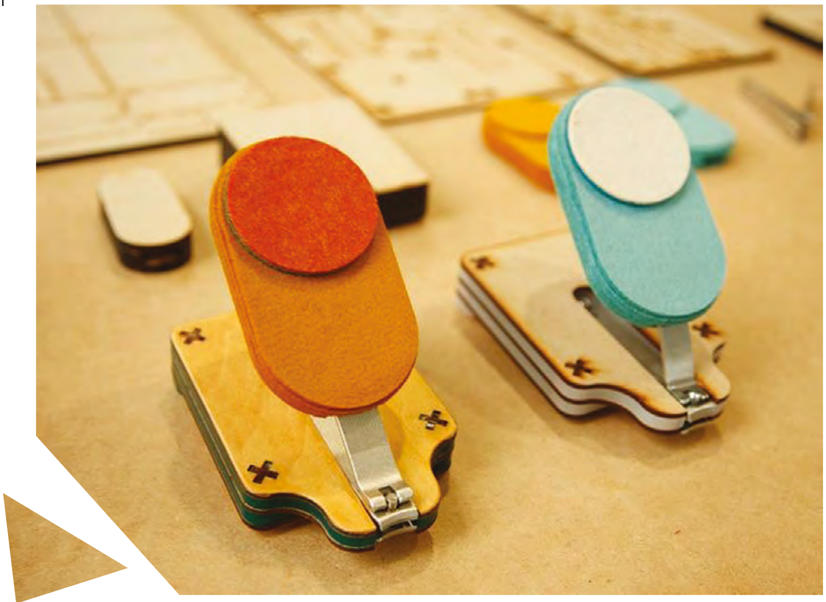
Hisashi sketched patterns for the different components of the clipper and cut them on a 2.5 mm medium-density fibreboard (MDF), a 5 mm acrylic sheet, and a 3 mm felt, using a laser cutter in his university's fabrication lab. He then stacked the cut MDF and acrylic sheet parts to assemble the base that has a slot to hold the clipper. Next, he glued the felts to create an extender that gave Kuriki a much larger support area to operate the clipper's lever. Hisashi's Instructable is short on textual description, but it is well illustrated and easy to follow. He has also shared the EPS files that you can use to fabricate the different components to make the accessible clippers. □

> "He made him an accessible nail clipper that Kuriki could operate with minimal support from his paralysed hand"

Project Maker
HISASHI KUN

Project Link
hsmag.cc/kKvMwJ

Below Hisashi's extension can be used by anyone who has trouble operating the tiny levers on the standard nail clippers



FEATURE

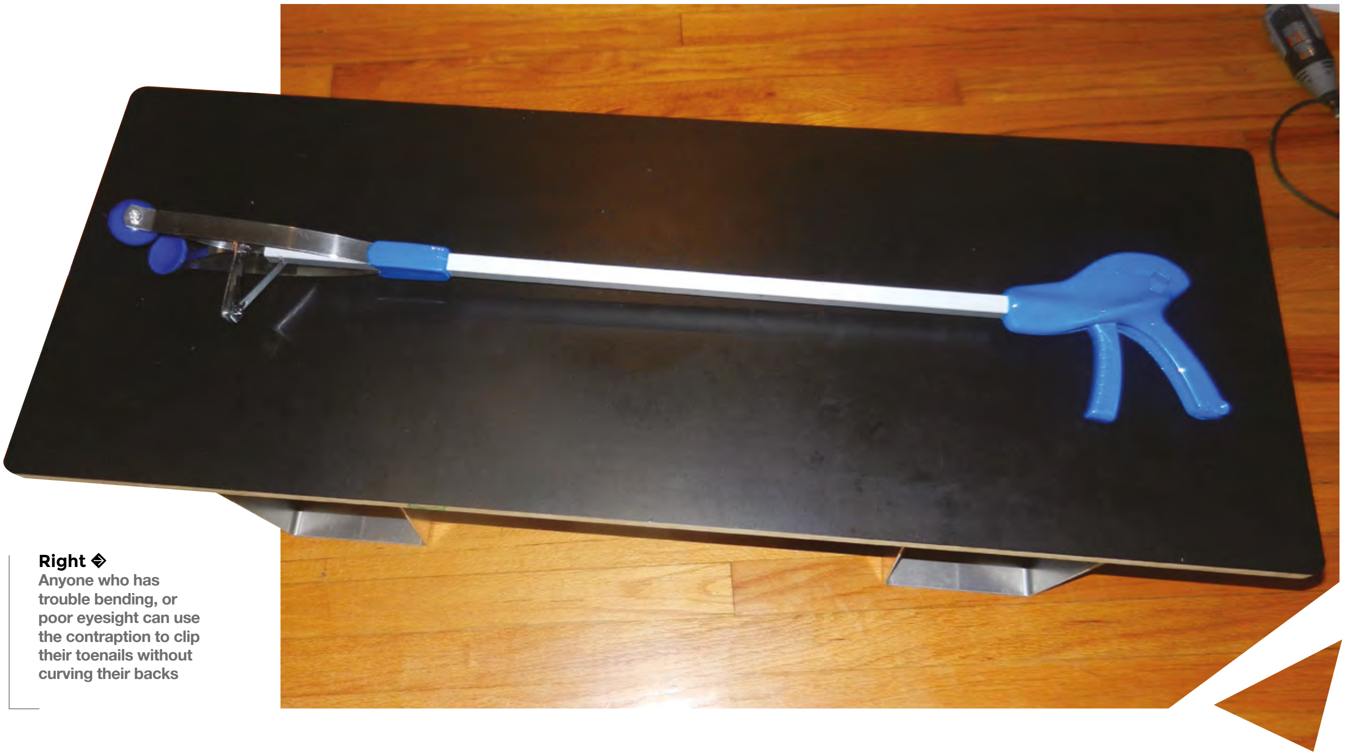
TOENAIL CLIPPER

Project Maker
ROBO PANTHERS

Project Link
hsmag.cc/nwztgR

Anyone who has seen seniors struggle to clip their toenails will jump at this contraption by the ingenious makers at Robo Panthers. They call it the EZ-Reacher Toenail Clipper because it uses an EZ-Reacher and grabber with suction cups. Although you can find toenail clippers for the elderly off the shelf, they are quite expensive and aren't half as convenient as the Robo Panthers device.

First, use pliers to remove the two suction cups from the spring arms of the EZ-Reacher. Then use a hacksaw to create a slit at the top of handle where you'll later insert the leverage arm on the clipper. Next, drill a couple of holes in the spring arm to hold the base of the nail clipper and use a Dremel to cut a slit in it. Then use the pliers again to remove the handle from a binder clip and follow the instructions to shape it so that you can use it to fasten the nail clipper to the spring arms of the EZ-Reacher. When the two are connected, you can reattach the suction cups and fasten the two spring arms to complete your EZ-Reacher Toenail Clipper. All the steps are clearly explained and illustrated and should take less than 30 minutes to put together. □



Right ⯑
Anyone who has trouble bending, or poor eyesight can use the contraption to clip their toenails without curving their backs

SHARPEN A NAIL CLIPPER

Nail clippers are very versatile tools. In addition to helping you maintain personal hygiene, you can use them to remove staples, cut guitar strings, trim candle wicks, and more. When Phil needed to strip a thin copper wire, he had no other option besides the nail clippers his father gave him shortly before Phil left for college: "Unfortunately, the wire left a nick in one side of the cutters. I felt bad about damage to a good cutting tool." Phil is a prolific maker with about 350 Instructables under his belt. A retired Lutheran pastor in his 70s, he has been doing DIY things since he was 10. So when he found a book on sharpening all sorts of things, he thought of the old damaged nail clipper. "All I can say is that my sharpened nail clippers work very well now." □

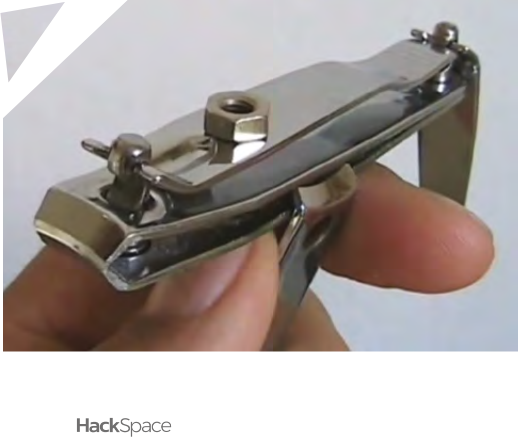
Project Maker
PHIL BOHLKEN

Project Link
hsmag.cc/CfGSyO

Left ⬗
Phil has posted detailed instructions on taking apart the nail clipper and the various strategies for grinding its jaws

NAIL CLIPPER CATAPULT

When makers like Tuan look at a nail clipper, they see a device that has three energy storage mechanisms – tension, torsion, and gravity, pretty much like a catapult. Tuan takes two different-sized nail clippers and disassembles the small one by first removing the pin and then the top lever. Then he stretches the two clip arms and wraps the small clipper around the lower lever of the bigger clipper. Watch the video to follow the last step, where he takes the pin from the smaller clipper and runs it through the holes at the end of the bigger clipper along with the disassembled top lever. When you're done, pull back the top lever from the bigger clipper and hook it up to the reinserted pin. The catapult is now primed and you can place small objects on this lever. A slight tuck will release the top lever and hurl whatever you've placed on it towards the target. □



Project Maker
TUAN DJEEM

Project Link
hsmag.cc/cUfFHs

Left ⬗
Tuan is an industrial designer and YouTuber who posts DIY projects built with everyday things

HackSpace
TECHNOLOGY IN YOUR HANDS

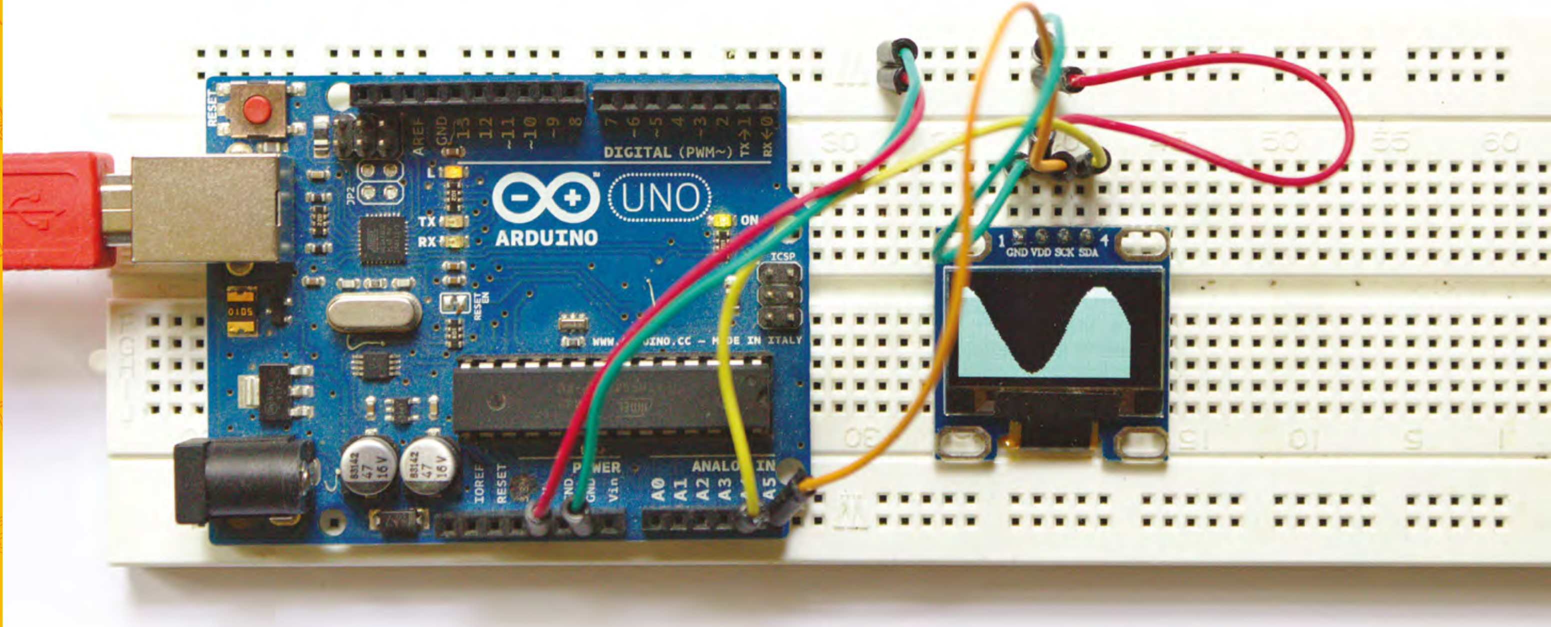
FORGE

HACK | MAKE | BUILD | CREATE

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

Arduino programming: pointers and linked lists

Upgrade your skills and demystify two of the most arcane aspects of Arduino and C



Graham Morrison

@degville

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

We were ambitious in our previous tutorial, creating a scrolling histogram of temperature data on a simple display with less than 80 lines of code. In the process, we also introduced two fundamental programming concepts: classes and stacks. We used classes to abstract both data and functions into a single object, and stacks as a kind of data structure where you can push values onto the top and pop them back off again. In this tutorial, we'll be revisiting these ideas and the same code, but introducing a couple of new fundamental concepts that are equally important – and one you've likely heard about if you've read or seen anything about C programming: pointers and linked lists.

Pointers, especially, are a little more theoretical than the practical nature of classes and stacks, especially on the Arduino platform. But they're important because they're closer to how the hardware works. They're also a fundamental part of the C programming canon. But there's a serious caveat. Most – if not all – programs can be written without them, and there's a good argument for avoiding them completely. They add a potentially catastrophic level of complexity to your code that most beginners don't need to contend with when they're just trying to get stuff done. They can cause your code to crash unpredictably, they can introduce subtle problems that are difficult to track down, and they can be difficult to predict. These are problems you avoid with 'static' types and data structures, such as a two-dimensional array.

In previous tutorials, we've been using types that implement static memory allocation. This means the compiler and the Arduino know exactly what the memory demands of our code are going to be before the code starts to run. In the `int ourList[MAXSTACK];` line, for instance, we're declaring exactly how many integers the `ourList` array is going to hold, which is a value defined by `MAXSTACK`. This is a very common approach with embedded systems, such as the Arduino, because the programmer needs to stay in absolute control of the resources being used, to make sure the hardware isn't asked to store data it doesn't have the memory for – a situation that would cause all kinds of unpredictable behaviour and, ultimately, cause your code to crash the Arduino.

But knowing what pointers are and what they're capable of is an essential step in any programmer's journey, and there are instances where they can be used to elegantly solve some specific problems. In particular, pointers can be a very effective way of passing large sets of values between functions without your program or hardware having to allocate extra memory or spending time copying that data. But they're also great at implementing something called a linked list.

POINTER SISTERS

Up until now, the inner operations of your hardware have been abstracted away by the programming language. Pointers pull the curtain away from this by making you think more like a CPU. To create a pointer, you simply insert an asterisk (*) symbol before the variable name:

```
int *example;
```

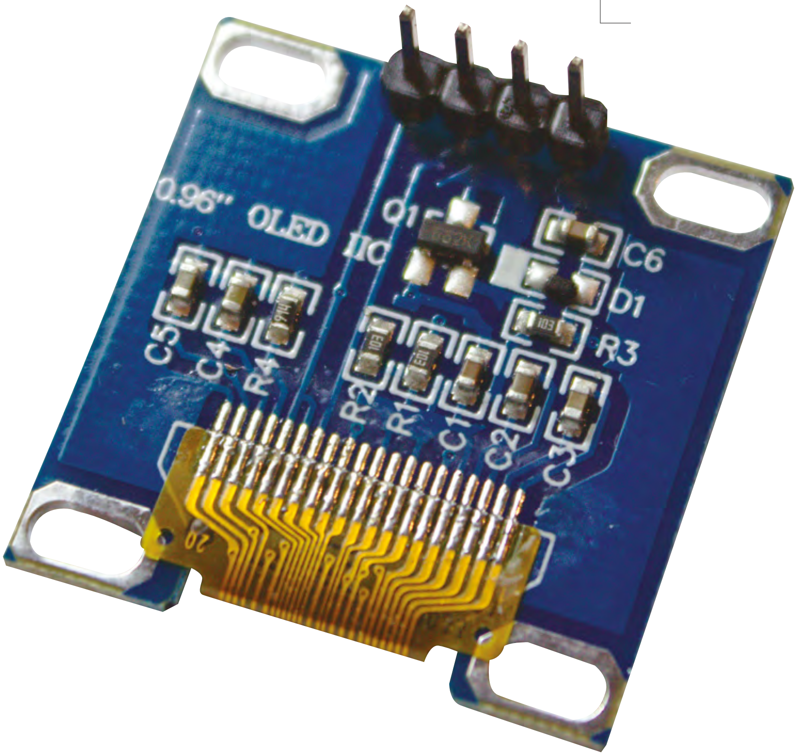
Without the asterisk, the compiler would make sure that enough space was allocated to hold an integer value referenced by the 'example' name. With the asterisk, however, we're now saying 'example' is a pointer to a memory location that holds an integer. 'Example' doesn't hold the integer value, but the memory location of where that value is stored. This is the key to understanding pointers, and it can take a little bit of time to get your head around. It means that, regardless of the value being held at the memory location, in our example an integer, the pointer will only ever hold the value of the memory location itself. On an Arduino, where every memory location can be addressed via a two-byte value, this means pointers use two bytes of storage, regardless of the data type or structure on the end of the dereferenced pointer.

> **"Knowing what pointers are and what they're capable of** is an essential step in any programmer's journey"

To prove this point, we're going to rewrite the stack class from the previous tutorial to use both pointers and a linked list to re-implement the class that surrounded the static array we originally used. A linked list is a very common form of dynamic data structure, and it can also be very simple. As a minimum, it's a structure that holds two elements: a value to be stored and a link to the structure that holds the next element in the list. The link is a pointer. The structure can obviously be augmented with many more components, such as holding a class value instead of a simple variable, and another pointer for the previous element in the list. But for our usage, we're going to keep it as basic as possible with just a pointer and an integer data value: →

YOU'LL NEED

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



Below The SSD1306 display is tiny, requiring only four pins to connect, and yet is immensely useful for all kinds of output

QUICK TIP

The asterisk used by a pointer is known as the dereference operator because it returns the memory location where a variable is stored.

SCOPE RESOLUTION OPERATORS

In the previous tutorial, we defined the functions that belonged to the class within the { and } brackets that delimited the class definition. Normally, these class definitions would be separate to the implementation code. This makes the class easier to understand at a conceptual level without having to resort to the code, and it's why you often get the class definition in the header (.h). But you can do the same thing even when you're working within the same file – you just need to create your functions using the scope resolution operator. Scope is a fundamental concept in many languages. It allows you to have variables of the same name in different classes, or global variables that won't interfere with function variables with the same name. Two colons are used to define the class you're assigning the function to (or a variable, though that's rarer). Here's the constructor function for our new linked list method, using 'stackList::' to tell the compiler it's a member of the stackList class even though it's outside of the bracketed scope of that class:

```
stackList::stackList()
```

We've followed this new protocol for all the class-bound functions in the code.

> "These pointers will allow the class to keep track of the elements at the **beginning and the end of the list**"

```
struct stackNode {
  int value = 0;
  stackNode *next;
};
```

With the above 'struct', we've created our own data structure holding the value we want to store and the pointer to what will be the next element in our linked list. This would have been the next element in an array, if we were still using arrays. To hold the functionality of our linked list, and to provide the same transparent functionality of our class from the previous tutorial, we're going to augment this structure with a new class to do all the hard work for us.

```
class stackList {
  protected:
    byte stacksize;
    stackNode *top;
    stackNode *tail;
  public:
    stackList();
    void push (int);
    int peek (int);
};
```

You'll see that the above code constructs a class that's almost identical to the array-based class we used last time, with the exception replacing the array with pointers to two 'stackNode' types, as defined by our new structure: top and tail. These pointers will allow the class to keep track of the elements at the beginning and the end of the list. Similarly, we'll be using 'stacksize' to hold the number of elements within the list. But the big difference between this implementation and the implementation using the array is that the class is no longer storing the values in the stack, it's simply holding pointers to the beginning and end elements. The values are going to be stored somewhere in memory, and it's going to be our job – and not the job of the compiler or the Arduino – to keep track of where each of these elements are and how many we've created.

CLASS ELEMENTS

```
stackList::stackList() {
      stacksize = 0;
      top = NULL;
      tail = NULL;
}
```

The above code is run whenever we create a stackList within our code, and we use the constructor to define default values and initialise variables. It's identical to the constructor we created when working with arrays, except we no longer need to run through the array to define default values. Instead, we assign the value of 'NULL' to the two pointers we create. NULL is a special value that effectively makes sure nothing is assigned as a value. Anything can be assigned a NULL value, but it can be most useful with pointers because there's always a possibility that, without being initialised in this way, they contain some random memory location that's a vestige of a previous run cycle. Assigning a NULL value is the pointer equivalent of setting a variable to 0.

PUSH AND PULL

We're now going to tackle the main function in our new class, the push function. As before, this takes an integer value and adds it to the stack that we're constructing to hold the values that we're measuring.

QUICK TIP

The arrow coupling of '->' is really just a short-cut to using an asterisk. For example, `ptrtmp->value = item` is functionally equivalent to `(*ptrtmp).value`.

The difference this time is that we're going to use pointers and a linked list. Here's the first part:

```
void stackList::push(int item) {
    stackNode *ptrtmp = new stackNode;
    ptrtmp->value = item;
    ptrtmp->next = NULL;
```

There are only three new lines of code above, but they encapsulate everything you need to know about pointers and linked lists, how they work, and how they can be used. Everything else we're going to add is basic maintenance of what we create above.

The first line after the function name (`stackNode *tmp = new stackNode;`) is creating a new node to hold this new value, and we're calling this 'ptrtmp'. It's a pointer to the new node. But the most important part here is the 'new' keyword. Without this, the pointer would be created but there would be no memory allocated for the data we want stored. Using 'new' handles this for us automatically, setting aside memory for a 'stackNode' element and its contents. There a vital difference between this and creating a normal type, such as by using `stackNode tmp`. In our example, even when `ptrtmp` no longer exists and we've left the scope of the function, the data it holds will still be in memory and reserved from anything overwriting it. As long as we've still got a handle on its location, which is what a pointer is, we can get at the data.

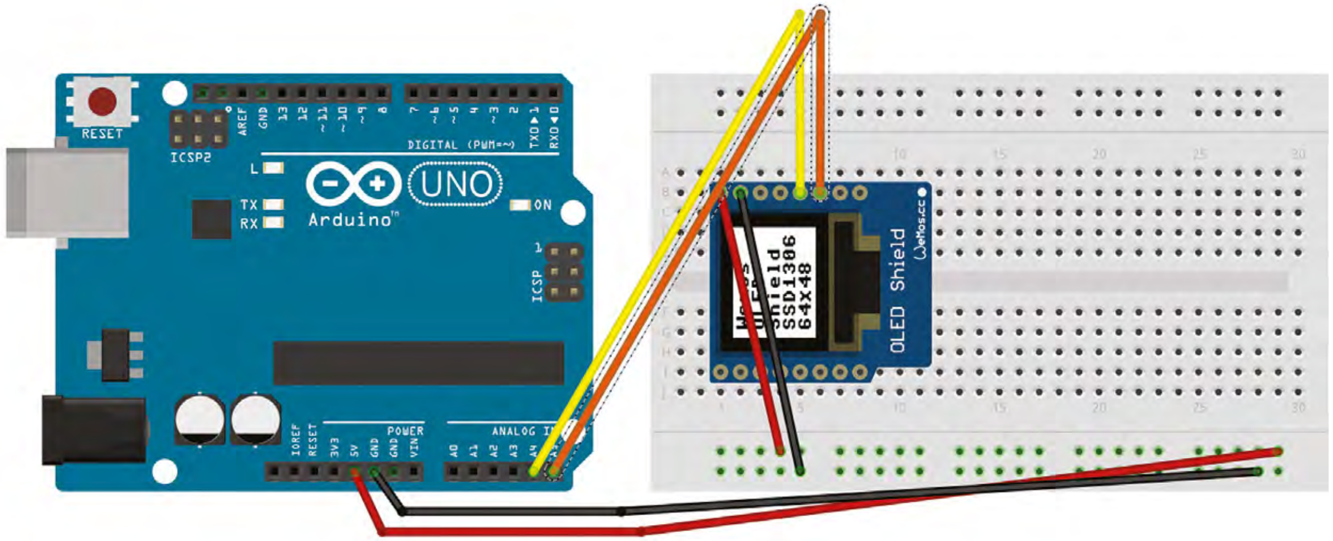
> **"This takes an integer value and adds it to the stack that we're constructing** to hold the values that we're measuring"

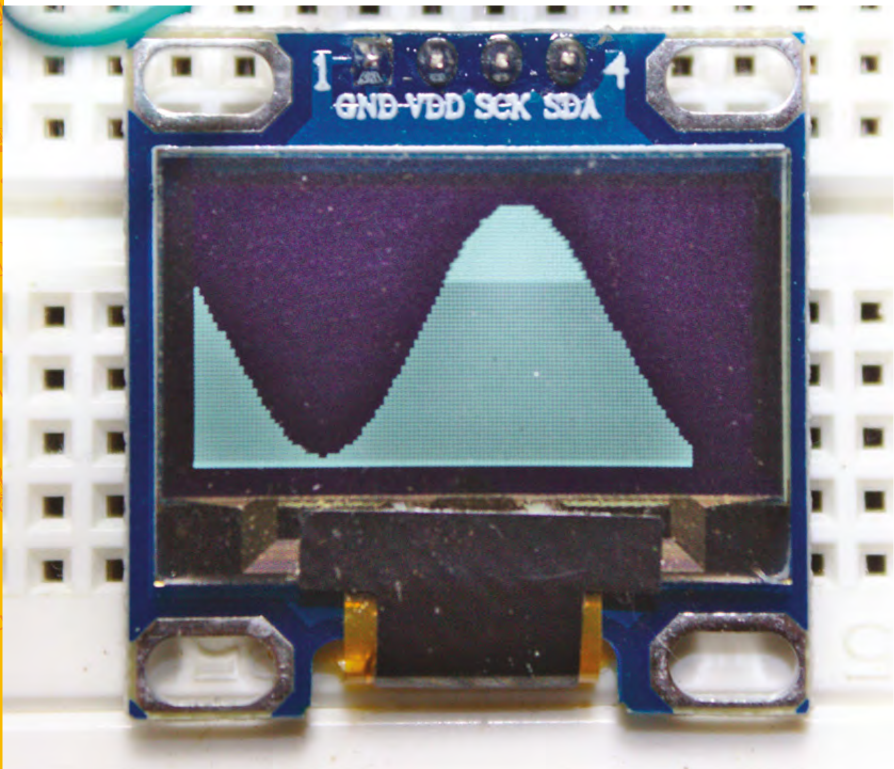
The use of the '->' characters is a short-cut to what the pointer is referencing, allowing you to change the value of what's being stored in the memory location the pointer is pointing at. The element that holds this pointer will be the previous element in the linked list, which is what the `ptrtmp` pointer holds. However, as we currently know nothing of what the next element might be, the pointer to the next element is created with a NULL assignment. We now deal with situation where the first element is added to the list:

```
if (tail == NULL)
        tail = ptrtmp;
      else
        top->next = ptrtmp;
top = ptrtmp;
```

The above illustrates the awesome power of pointers. Firstly, by asking whether the 'tail' pointer is still NULL, we're checking to see whether this is the first element we're adding to the list, as this needs special consideration. If it is, then we point the tail pointer at the new ptrtmp element we've just created. If it isn't, then we know there's at least one element already in the list and the previous element to be added will be pointed to from 'top'. Now we're adding a new element, top's 'next' pointer needs to point at the new one we're creating, which we can simply do with `top->next = ptrtmp;`. Remember, →

Below ↙
The circuit and wiring for this tutorial is identical to the previous one, except you can forgo the temperature sensor if you update the graphics routine





GRAPHICS UPGRADE

It didn't seem fair to leave this month's project generating exactly the same output as last time. As an added bonus, and to investigate mathematical functions a little, you can forgo the temperature sensor and replace the main loop code with the following:

```
void loop() {
  if (counter > 180)
    counter = -180;
  temp_stack.push((sin(counter * 3.14 /
180) + 1.1) * 29);
  counter = counter + 2;
  displayChart();
  display.display();
  delay(1);
}
```

You will also need to add `int counter = 0;` as a global variable outside the scope of the `loop()` function. The above uses `sin()` to generate the sine function for the angles between -180 and +180, as counted by `counter`. The output will be a regular sine wave drawn and scrolled along on the OLED display, but you can play with the numbers to change both its frequency and its amplitude.

Above
Replace the static temperature reading chart with a sine wave – you could even modulate its frequency with temperature changes!

these are just pointers – we're not moving the values they're storing, we're just assigning the memory location. The node holding the value we want to be stored isn't moving.

The next chunk of code stops our linked list expanding further than your Arduino has the capacity to store:

```
  if (++stacksize > MAXSTACK) {
    ptrtmp = tail;
    tail = ptrtmp->next;
    delete tail;
    stacksize--;
  }
}
```

In the above code, we're checking to see whether there are now more elements in the list than we want, as defined by MAXSTACK. If there are, we store the memory location of the oldest node, the tail, in the ptrtmp pointer. This is so we can then make the new tail pointer point to the next item in the list while before deleting what was the oldest element linked to within the list. You will also need to reduce the value of MAXSTACK in your code too, typically to between 60–100 or you'll run out of RAM.

The **delete** command here is the opposite of the **new** command we used earlier, freeing up the memory that we allocated to hold the structure. We can now tackle the final function that returns a value for the node in the list we request. This is more complex than the array equivalent as we're unable to directly address the value stored by the list at location 'x'. Instead, count through the positions until we reach the correct node and return the value it holds:

> **"This is more complex than the array equivalent** as we're unable to directly address the value stored by the list at location 'x'"

```
int stackList::peek(int x) {
  int pos = 0;
  stackNode* current = tail;
```

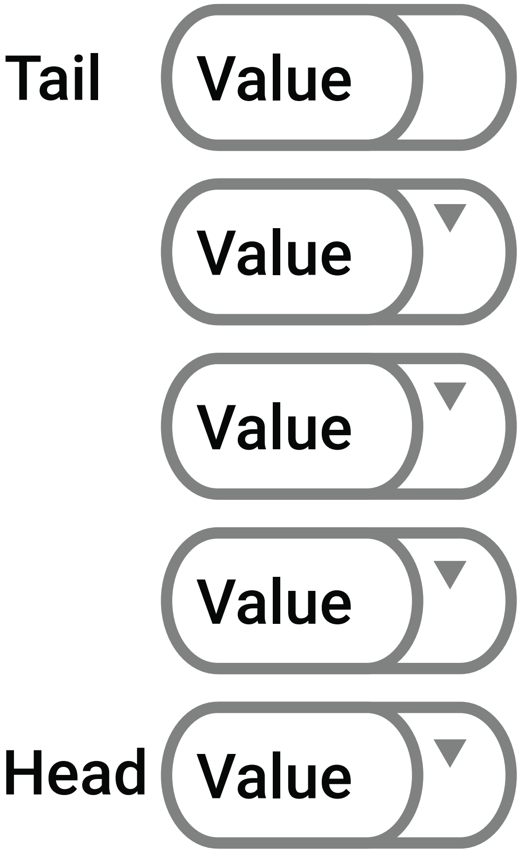
QUICK TIP

The code for this project can be found here: **git.io/fSGkD**

```
    while ((pos < x) && (current != top)) {
      current = current->next;
      pos++;
    }
    if (x > pos)
      return -1;
    else
      return current->value;
    }
};
```

The only other part of the code we're going to touch is the part that draws the graph. We're doing this because there's no longer enough RAM on an Arduino Uno to hold the entire linked list, so we're going to only map values to the screen when there's a node, and stop the redraw process when there are no more elements left to render. However, we'll keep the same sliding window logic from the previous tutorial as this looks rather good:

Below
A linked list is a dynamic data structure where each element contains a link – using a pointer – to the next element in the list



BEYOND THE LIST

Here, we've looked at a linked list because it's one of the simplest data structures (and serves the purpose we need), but there are lots of others that you can create, built on the same principle. Once you've mastered this technique of using pointers to link elements together, you can adapt it to make the others. In each case, you have a structure with a pointer showing the links to other nodes…

- **Trees:** In this data structure, a single root element has one or more children, and each child has children, and so on. Think of something like a family tree, but it can hold almost any sort of data. A common variation of this is the binary tree where each node has, at most, two children. This setup can be useful for searching as each node can represent a value and all left children can be lower, and all right ones can be higher.
- **Graphs:** In computer science, graphs have nothing to do with diagrams showing how a value changes over axes. Instead, they're collections of nodes that can be joined in any way. Think of it a bit like the linked list, but each node can be joined to many nodes, not just one. This can represent many things, particularly the structure of the internet, as this is made up of a large number of servers and data centres with various connections between them.
- **Heap:** These are similar to trees except that there's a heap property such that a parent node has to be either explicitly higher or lower than every node below it. One of the most common uses of a heap is a priority queue, where each node is more important than those below it.

```
void displayChart() {
  char x = 0;
  int value = 0;
  value = temp_stack.peek(x);
  while (value != -1) {
    display.drawLine(x, display.height(), x, 0,
BLACK);
    display.drawLine(x, display.height(), x,
display.height() - value, WHITE);
    value = temp_stack.peek(++x);
  }
}
```

With that done, you can run the code. With a bit of luck, you'll be rewarded by seeing absolutely no difference in the temperature mapping chart from last month – see the 'Graphics Upgrade' box to change this to a different rendering algorithm. But the way your code is working is now completely different – using pointers and a linked list rather than an array – and you've now mastered one of the most arcane and misunderstood aspects of the Arduino and C programming environments. □

QUICK TIP

Try changing `int` definitions to `byte` when you know the value it holds will be between 0 and 255. This will save a whole byte of valuable Arduino memory and give you a larger potential stack size.

Electronics 101.1: Electricity basics

In the first instalment of this series, we will get familiar with what electricity is, some terms, some rules, and a few basic components

Dave Astels

daveastels.com

My career started in the 8-bit days, with the Z80 and 6502, and I've been working with computers ever since. I do some writing at **daveastels.com** and **learn.adafruit.com**

YOU'LL NEED

- **Solderless breadboard**
- **5 V power supply** or battery pack that can be connected to the breadboard
- **LED**
- **Resistors of various values**
- **Breadboard-friendly variable resistor**
- **Digital multimeter** required to do the measurements in some of the exercises

Electricity is fascinating. Most of our technology relies on it: computers, lights, appliances, and even cars, as more and more are hybrid or electric. It follows some well-defined rules, which is what makes it so very useful.

According to Wikipedia, electricity is 'the set of physical phenomena associated with the presence and motion of electric charge'. And what's electric charge? That's the shortage or excess of electrons.

Let's go back (or forward, depending on where you are in life) to high school science and the atom. An atom is, at a very simplified level, a nucleus surrounded by a number of electrons. The nucleus is (again, viewing it simply) made up of neutrons and protons. Neutrons have no charge, but protons have a positive charge. Electrons have a negative charge. The negative charge on a single electron is the exact opposite of the positive charge on a single proton. The simplest atom, hydrogen, is made from a single proton and a single electron. The net charge of the atom is zero: the positive charge of the proton and the negative charge of the electron cancel – or balance – each other. An atom's electrons aren't just in an amorphous cloud around the nucleus: you can think of them as being arranged in layers around the nucleus... rather like an onion. Or perhaps an ogre. This is a very simplified visualisation of it, but it suffices for our purposes.

In a more complex atom, say copper, there are more protons, neutrons, and electrons, and the electrons are in more layers. By default, a copper atom has 29 protons and 35 neutrons in its nucleus, which is surrounded by 29 electrons. The way the electrons are distributed in their layers leaves the copper atom with a single electron in the outermost layer. This is represented in **Figure 1**. Without getting further into subatomic physics, let's just say that having that single electron in the outermost layer makes it easier to manipulate. When we put a bunch of copper atoms together to make copper metal (e.g. a wire) it's easy to move those outermost electrons around inside the metal. Those electrons moving around is electricity. The amount of electrons moving over a period of time is called 'current'.

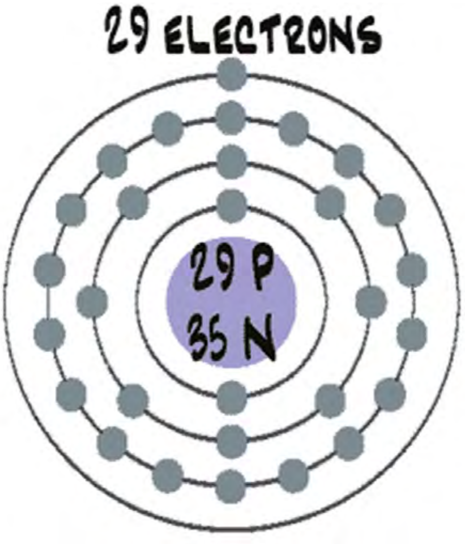


Figure 1
A very stylised representation of a copper atom, with its electron shell

We started by talking about electrons and charge. Look back at the Wikipedia definition: 'presence and motion of electric charge'. Charge is measured in coulombs: 1 coulomb is approximately 6.242 × 10^{18} electrons. That's 6 242 000 000 000 000 000 electrons. They're very small. Actually, this would be -1 coulomb. +1 coulomb would be that many protons (or really, the net lack of that many electrons).

That's charge. Now let's consider moving charge, which is far more useful in general (unless your goal is to stick balloons to the wall). Consider some amount of charge moving through a wire. The

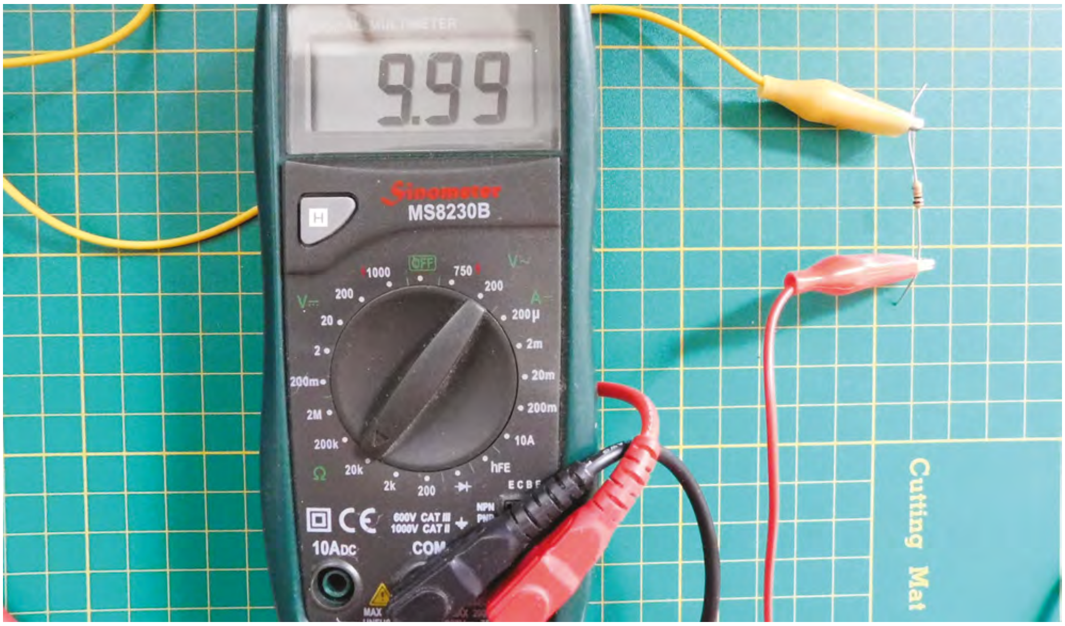
amount of charge that moves past a specific point (and thus through the wire) over a period of time is called 'current' (just like the current in a river) and is measured in amperes, generally just called amps. Specifically, one amp is equal to one coulomb flowing past a point in one second.

Another common term is voltage. You can think of voltage like water pressure; it's the pressure pushing the electrons (i.e. charge) through a material. The higher the voltage (measured in volts), the faster charge is pushed through, i.e. the higher the current.

The final term is resistance, measured in ohms. Resistance is just what it sounds like. It's a measure of how much a material resists the movement of electrons. We said that copper allows electrons to move freely. That's what makes it so common for wires, PCB traces, etc. We say that it is a good conductor. Glass, on the other hand, locks its electrons in place, not letting them move. It's an example of a good insulator. There are materials that are in between: they let electrons move, but not too freely. These are crucial to making electronics work. There's an interesting (and useful) relationship between voltage, current, and resistance called Ohm's Law (Georg Ohm was the fellow who explored and documented this relationship): the current (denoted I, in amps) flowing through a material is equal to the voltage across the material (denoted V, in volts) divided by the material's resistance (denoted R, in ohms). I = V/R. This equation is foundational and, as such, very handy.

> "You can think of voltage like water pressure; it's the pressure pushing the electrons (i.e. charge) through a material"

Below
A single 10 kΩ resistor reads almost 10 000 ohms (no electrical component is perfect)

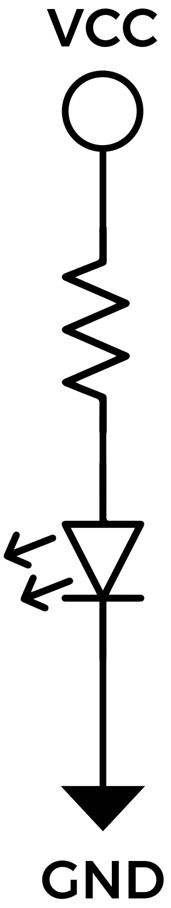


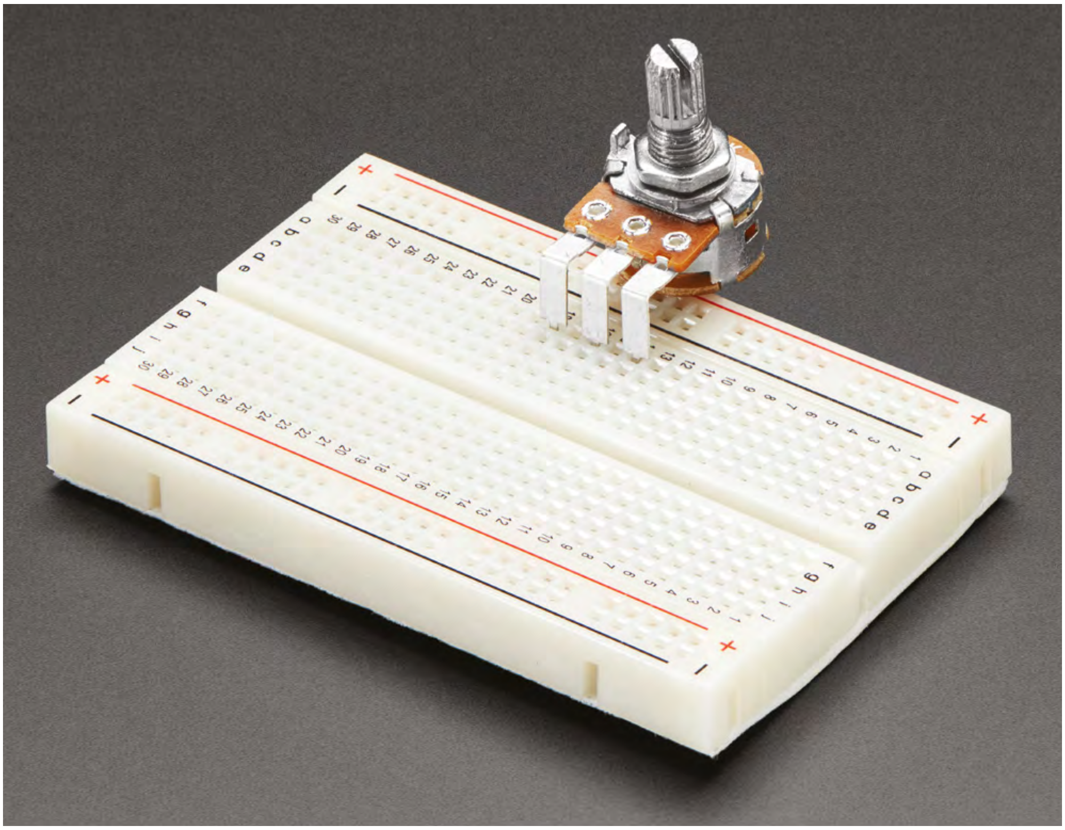
LIGHTING UP

There aren't many electronic devices that don't have at least one LED on them somewhere, especially not gadgety ones. If you look at a simple Arduino Uno, it has LEDs for power, Tx, Rx, and pin 13. The first program that most people try is one to blink an LED. →

Figure 2
The colour spectrum

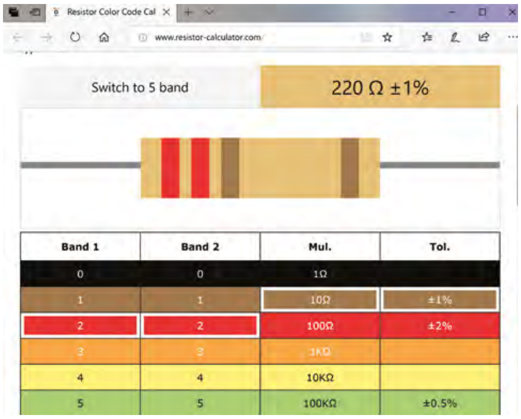
Figure 3
An LED with a current-limiting resistor





> **An LED will have a specific Vf** (f is for forward, as in 'forward voltage') that will be defined in its data sheet

LED stands for light-emitting diode. We'll come back to diodes in a later instalment; all we need to know right now is that a diode has to go the right way around. So that leaves 'light-emitting'. That simply means that it gives off light: it lights up. Specifically, it lights up when enough current flows through it. Be careful, though. Put too much current through it and it'll likely crack in two. Seriously, we've done it. Best case scenario, you'll get a bright pulse of light as it burns out. How much current do they like? 20 milliamps (20 mA) is typical. Because an LED is a diode, i.e. a semiconductor (we'll look at these in more detail in a future instalment too), it defies Ohm's Law. How? It always has the same voltage across it, regardless of the current flowing through it. An LED will have a specific Vf (f is for forward, as in 'forward voltage') which will be defined in its data sheet. The voltage varies with the colour of light that the LED emits, but usually between 1.8 V and 3.3 V. Vf for red LEDs will typically be 1.8 V, and for blue LEDs 3 V–3.3 V. As a rule, LEDs with a higher frequency colour will have a larger Vf. **Figure 2** shows the colour spectrum. Colours on the right end are lower in frequency and LEDs emitting those colours will have a lower Vf, while those on the left end have a higher frequency and a higher Vf.



220 Ω ±1%

Band 1	Band 2	Mul.	Tol.
0	0	1Ω	
1	1	10Ω	±1%
2	2	100Ω	±2%
3	3	1KΩ	
4	4	10KΩ	
5	5	100KΩ	±0.5%

So an LED will have a fixed Vf and a typical LED that we'll use likes about 20 mA of current. An LED won't do anything to limit how much current is flowing through it. That's what we meant when we said it defies Ohm's Law.

If we took a blue LED and hooked it to a 3.3 V power supply, it will shine happily. Do the same thing with a red LED, and it will blink and burn out. So how do we deal with that? How do we use 3.3 V or 5 V to

Above
Potentiometers can change resistance

Right
Resistor colour bands show the resistance. Online calculators can help you learn the values

make an LED light up without burning out? We simply limit the current flowing through it. And for that, we need a resistor and Ohm's Law.

GETTING PROTECTION

If we want to power a red LED from a 5 V source, we know the following information: current has to be 20 mA, Vcc will be 5 V, and the voltage across the LED will be 1.8 V. Consider the circuit in **Figure 3**. The voltage across the resistor will be Vcc – Vf, i.e. 5 – 1.8 = 3.2 V. We said the current through the LED should be 20 mA. Since there is only one path through the circuit that goes through the resistor as well as the LED, all current has to flow through both: whatever amount of current flows through the resistor has to flow through the LED, no more, no less. This is the crucial thing to realise. We can calculate the value of the resistance needed using Ohm's Law: R = V/I = 3.2 V / 20 mA = 3.2 V / 0.020 A = 160 ohms.

The resistor should have a value of 160 ohms to allow 20 mA of current to flow through the LED. Knowing that the 20 mA and 1.8 V values are approximate and that resistors are not exact (+/- 5 or 10 percent are the most common), we chose a slightly higher-value resistor. Considering common resistor values, go with 180 ohm or 220 ohm. A higher-value resistor will allow slightly less current through, which might result in a slightly dimmer light. Try it and see. For practical purposes, simply using a 220 ohm resistor usually works fine.

PARALLEL LINES

In the previous section we connected a resistor and an LED end to end. That's called a series circuit. If we connected them side by side it would be a parallel circuit. Consider the circuits in **Figure 4**.

Below
A multimeter can read voltage, ampage, and resistance

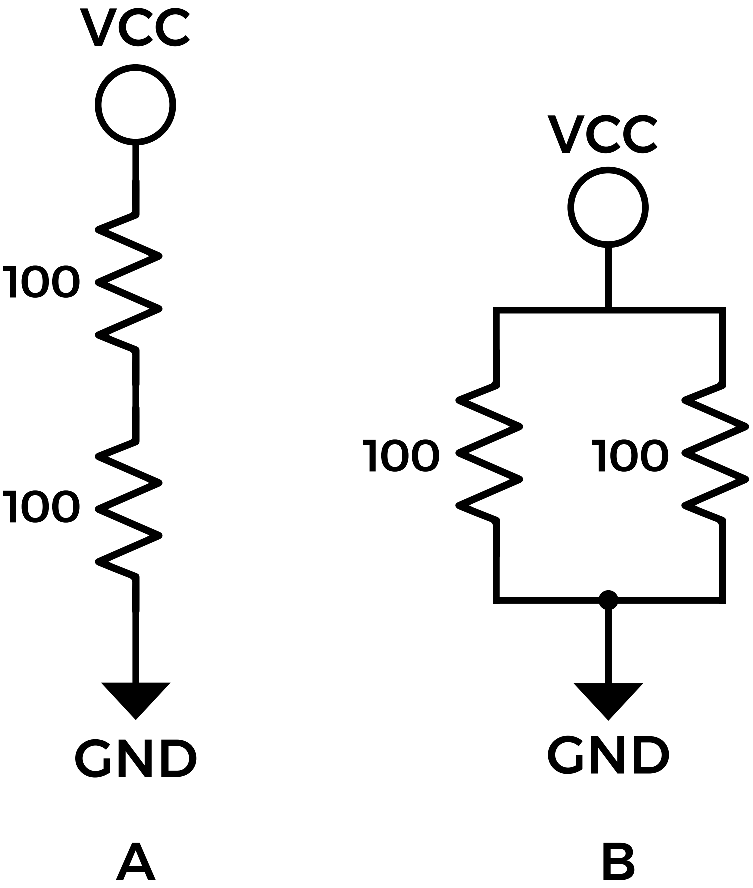


Figure 4
A – series circuit
B – parallel circuit

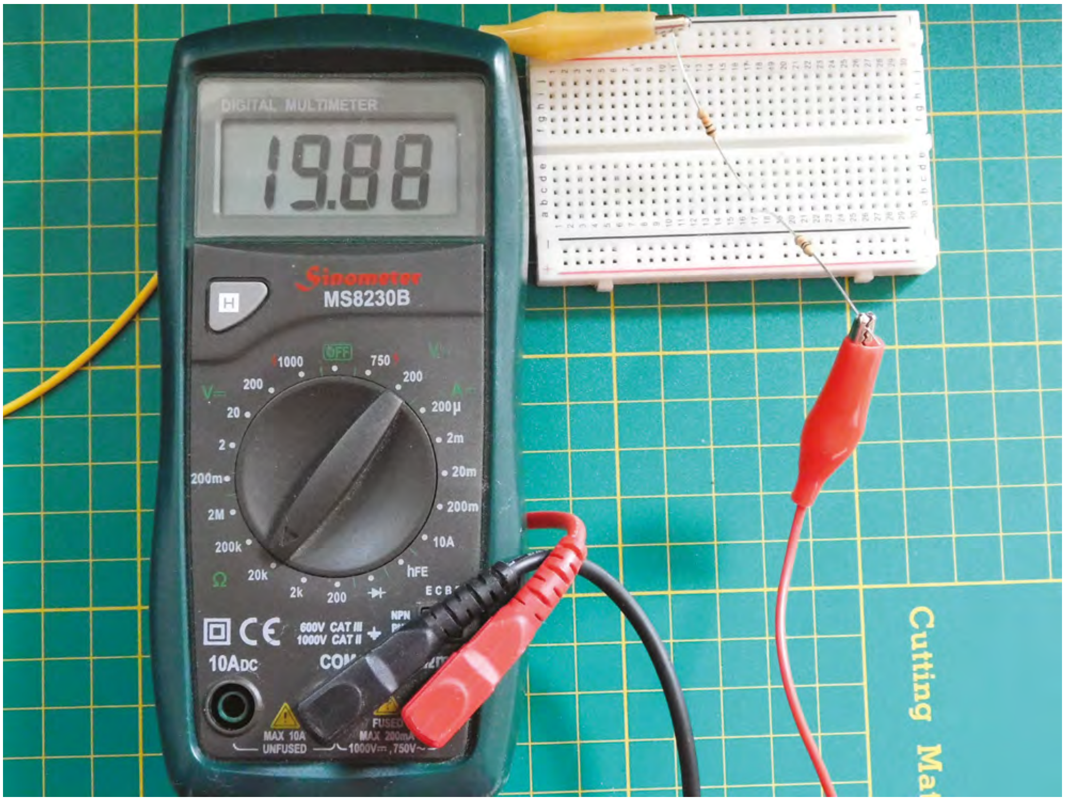
We'll use 5 V for Vcc. What is the total resistance between VCC and GND in each circuit? How much current is flowing through each circuit? What is the voltage across each resistor?

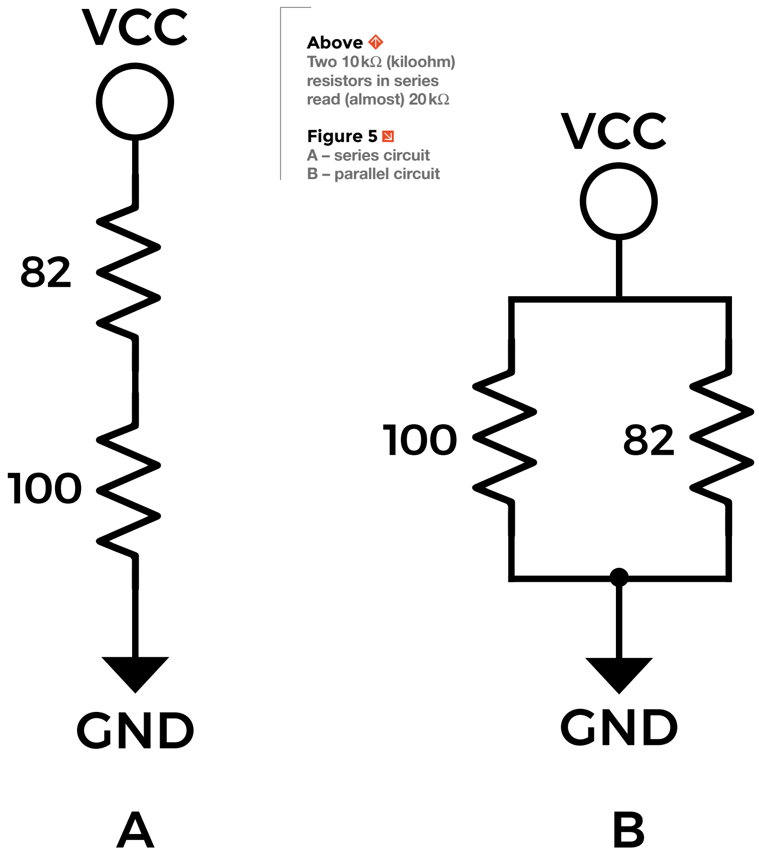
When resistors are connected in series, as in circuit A, the resistances are added. So the two 100 ohm resistors in series have a total resistance of 200 ohms.

When resistors are connected in parallel, as in circuit B, it's more complex. Each resistor provides a path for current to flow through. So we could use an indirect method to calculate the total resistance. Each resistor is 100 ohms, and has one end connected to 5 V and the other to 0 V (GND), so the voltage across each one is 5 V. The current flowing through each one is 5 V/100 ohms = 0.05 A, or 50 mA. That flows through each resistor, so the total current is 100 mA, or 0.1 A. The total resistance is then R=V/I, R=5 V/0.1 A = 50 ohms. A more direct way is to use the equation 1/Rt = 1/R1 + 1/R2 + … + 1/Rn, where rT is the total resistance, and R1, R2, etc. are the values of the individual resistors that are in parallel. Using this, 1/Rt = 1/100 + 1/100 = 2/100 = 1/50. So Rt = 50. This is a quicker way to do it, and only involves the resistor values. →

CHALLENGE

Calculate the resistances, voltages, and currents in the circuits shown in **Figure 6** (overleaf). Measure them to confirm your results. We'll work through it in the next instalment.





Above
Two 10kΩ (kiloohm) resistors in series read (almost) 20kΩ

Figure 5
A – series circuit
B – parallel circuit

Now for current. We know that the series circuit has a total resistance of 200 ohms, so the current will be I = V/R = 5V/200 ohm = 0.025A = 25mA. For one 100 ohm resistor the current is 5V/100 ohm = 0.05A = 50mA. This is expected: if the resistance is lower, there is less 'resistance' to current flowing, so with the same voltage, more current will flow. We already computed the current for the parallel circuit: 100mA. This is higher because we know that each resistor has 50mA flowing through it. In a parallel circuit the currents are added.

The final question is what voltage is across each resistor. Let's look at the parallel circuit first. One end of each resistor is connected to 5V, and the other end of each is connected to 0V (GND). So clearly, the voltage across each one is 5V. In a series circuit it's different. We can use Ohm's Law because we've calculated the current flowing through each one (0.025A), and that current flows through both resistors. Each resistor is 100 ohm, so the voltage across each one will be V = IR = 0.025A × 100 ohm = 2.5V. This makes sense intuitively, since the resistors have the same value and the same current is flowing through both. If makes sense that the voltage across each would be equal, and half of the total. Remember that it's unlikely to be exactly half, due to the slop in the resistor values.

Let's do this one more time with unequal resistors. See **Figure 5**.

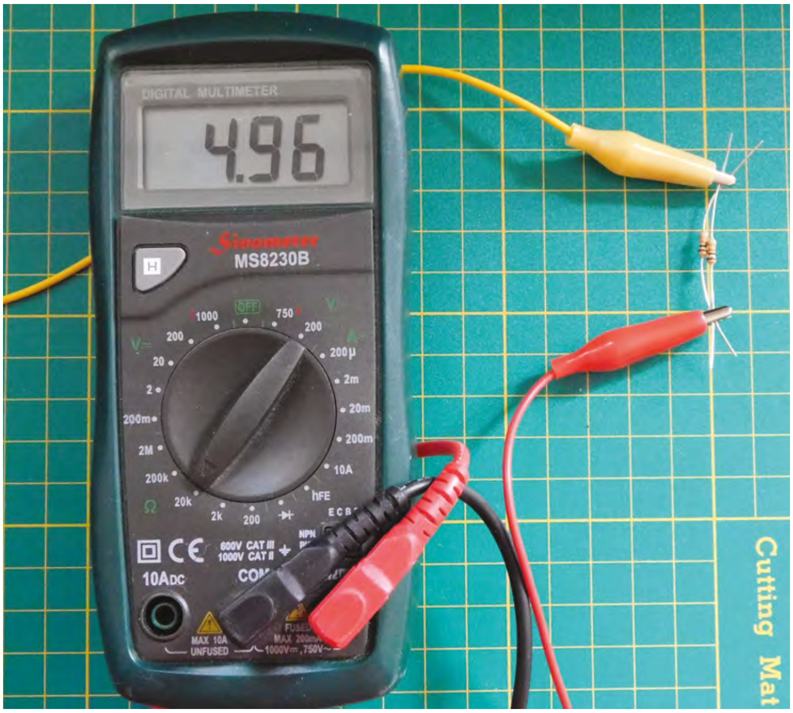
For the series circuit, we simply add the resistances: 100 ohm + 82 ohm = 182 ohm. The current is 5 V/182 ohm = 0.0274725 A = 27.4725 mA. Because resistors are inexact, it's safe to call this 27.5 mA. The voltages are 100 ohm × 0.0275 A = 2.75 V across the 100 ohm, and 82 ohm × 0.275 = 2.25 V. The voltages always have to add up, accepting rounding errors. Relative to ground, the voltage at the point between the resistors is 2.75 V. What will happen if we make the top resistor smaller (i.e. have a lower resistance)? The total resistance goes down, the current goes up, so the voltage across the 100 ohm resistor goes up. This is what's generally called a voltage divider.

For the parallel circuit we can use 1/Rt = 1/100 + 1/82 = 82/8200 + 100/8200 = 182/8200 = 1/45, so Rt = 45 ohm. The total current is 5 V/45 ohm = 0.111 A = 111 mA. For the individual resistors, the currents are 5 V/100 ohm = 50 mA and 5 V/82 ohm = 61 mA. Add these up and we have the total current of 111 mA. Parallel resistors act as a current divider.

I encourage you to create these little circuits on a breadboard and measure the resistances, voltages, and currents for yourself.

RESISTORS IN SERIES FORM A VOLTAGE DIVIDER, RESISTORS IN PARALLEL FORM A CURRENT DIVIDER

Consider what happens if we replace the resistor connected to Vcc in a series circuit with a variable resistor. The voltage between the resistors will vary as the value of the resistor does. As the resistance goes down, the voltage goes up. The reverse is true as well: as the resistance goes up, the voltage goes down.

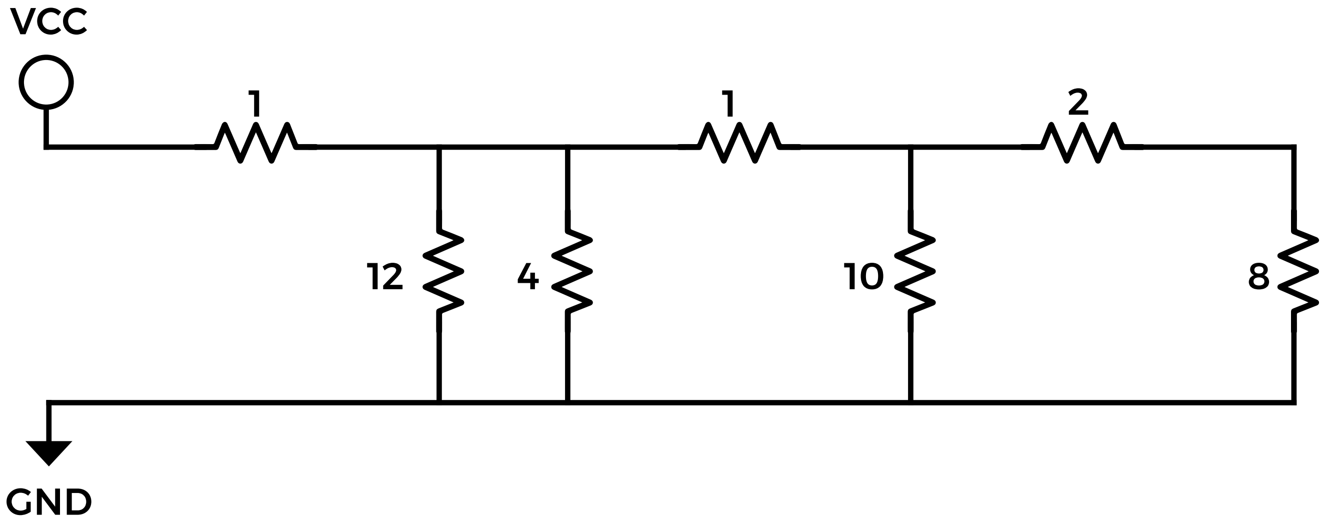


One use of this is to replace the variable resistor with a photoresistor. A photoresistor is a resistor whose value depends on how much light is shining on it (i.e. how many photons are hitting it, to be precise). More light = lower resistance. Now the voltage divider can be used to measure the strength of light. All you need to do is connect the point between the resistors to an analogue input and read it.

We've had a brief look at the basic concepts of electricity: charge, current, voltage, and resistance. We've also had a closer look at resistors and ways of combining them. We finished with a practical example of a series resistor circuit being used to measure light. □

Above
Two 10 kΩ resistors in parallel read (almost) 5 kΩ

Figure 6
Combined parallel and series circuits



Power a robotic sentry with facial recognition and Google AIY Vision

Using the Google AIY Vision Kit and some plumbing parts, we built this robot, which is just as fun to play with as to make

Andy Clark

workshopshed

YOU'LL NEED

- Google AIY Vision Kit
- Waterproof box
- Waterproof button
- Big servo
- OnOff SHIM
- Battery
- Waste pipe
- Solenoid valve
- N-Channel MOSFET
- Garden sprayer
- Hosepipe fittings and nozzle
- Funky foam

Until now, garden sprinklers have been dumb pipes with predictable motion. Fun to play in, but not much of a challenge. This project gives the sprinkler a brain and vision so that it can target passing humans for a soaking they will never forget.

BUILDING A BRAIN

The brain and eyes of our robot are a Google AIY Vision kit. These are for sale in the USA at the time of writing and should be on sale in the UK soon.

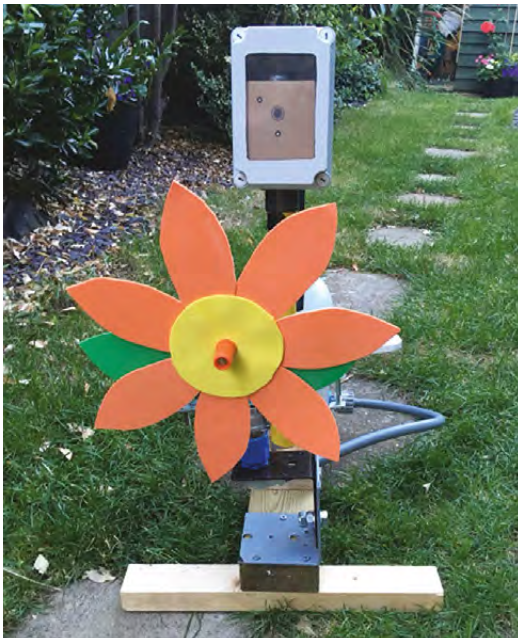
The kit consists of a Pi Zero W, Pi Camera V2, Vision Bonnet, a switch and a buzzer, along with assorted cables. Google has provided detailed instructions on assembling the AIY Vision Kit which we followed carefully. We found the ribbon cables particularly fiddly, and there are two different kinds of zero insertion force connectors for these.

WE HAVE THE POWER

To make our game portable, and to reduce the risk of a nasty accident with electricity, the project runs from batteries, naturally.

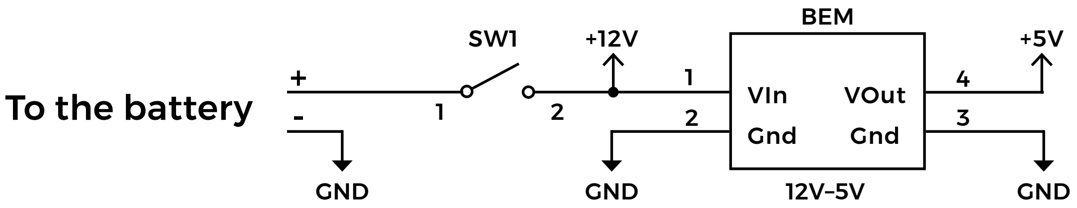
We used a three-cell LiPo battery designed for radio-controlled vehicles; this outputs 11.1 V, which is high enough to activate the mechanical parts of the project. Along with this we added a 'battery eliminator module' (BEM) which is the RC world's name for a DC to DC converter or regulator. The BEM drops the voltage down to 5 V for the Pi.

The Vision Kit does not have a simple way to shut down the Pi, and suggests logging in and using the `shutdown` command before powering off. To make the project a bit more practical, we added a Pimoroni OnOff SHIM power switch. This is wired up to the waterproof button on the top of the case. To drive the digital servo, we needed I^{2}C, so extra wires were added for that. The OnOff SHIM comes with software to safely shut down the Pi before turning off the power.



Above Our robot water sentry, cunningly disguised as a flower

WARNING: the default install seems to affect the AIY software, which is sensitive to being updated. So the daemon to shut down the software should be installed manually.



Left Battery circuit

We also added a waterproof toggle switch to isolate the batteries from the circuit.

SOLENOID VALVE

A solenoid valve is a mechanical valve that is actuated via a magnetic coil. The GPIO pins are not capable of providing sufficient current to directly activate this solenoid. For this we used a driver circuit, consisting of an N-Channel MOSFET and some discrete components. R1 limits the maximum current from the GPIO, R2 ensures the MOSFET is turned off when power is removed, D1 provides a path for the stored energy of the coil when the circuit is turned off.

WATER WORKS

The plumbing aspect of the project is quite simple but is worth spending time on to avoid leaks. The driving force for the water is a garden sprayer to make the project portable. Hosepipe quick disconnects were used, allowing the project to be quickly packed away.

So that the head of the sprayer can be turned, we used another quick-release hose connector. This was mounted on some metal brackets, with a smaller bracket attached to act as a lever. The lever was linked to the servo horn using some coat-hanger wire.

To disguise the weaponry, we added a flower made from funky foam.

We found a solenoid valve with ½-inch threads that connected to the screw threads of the hose fittings, and used a jubilee/hose clip to hold the tubing onto the sprayer.

BOXING IT UP

As mentioned, electricity and water don't mix. Neither does cardboard. So, our Vision Kit needs some protection. We took a waterproof box and added a window using some thin polycarbonate sheet and Sugru to seal. Next, a tank connector and waste-pipe were added and sealed. This raises up the electronics and camera to get a better image. Wires were threaded through the pipe and out through a grommet.

HARDWARE SPECS

- Intel Movidius Myriad 2 MA2450 vision processing unit (VPU) chip
- Microchip SAM D09 (same as in the Arduino Zero, MKR1000, and MKRZero boards)
- Four GPIO pins controlled by the MCU
- Pi Camera V2

JUMBO SERVO

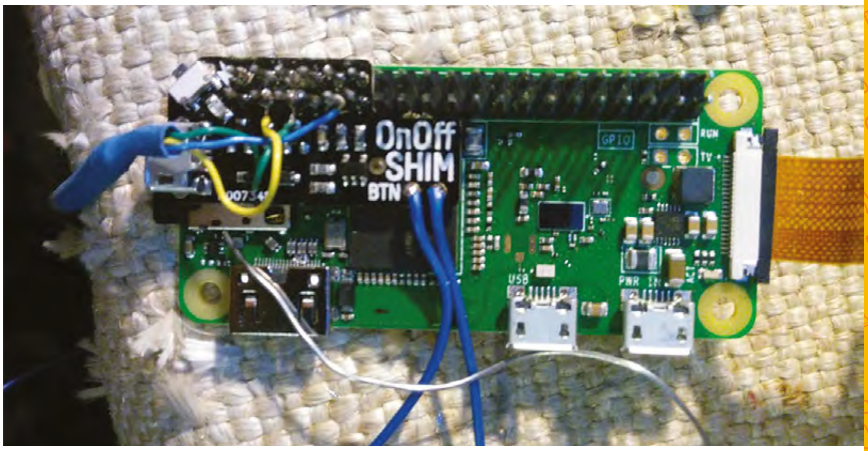
To move our nozzle, we need a motor and a way of knowing the position it's in: in other words, a servo. Some quite large servos can be bought from robotics or RC model suppliers, but these are also expensive. We already had a big 12 V motor with a reduction gearbox, so a linear potentiometer and 3D-printed gear were added for the position. This was wired up to a small Arduino board to control the motor. For communication to the Pi, we decided to use an I^{2}C protocol rather than the standard analogue PWM approach used for off-the-shelf servos. The reason for this is that it allows us to read back the current position into the Pi and avoid the twitching movement on startup that happens with analogue servos.

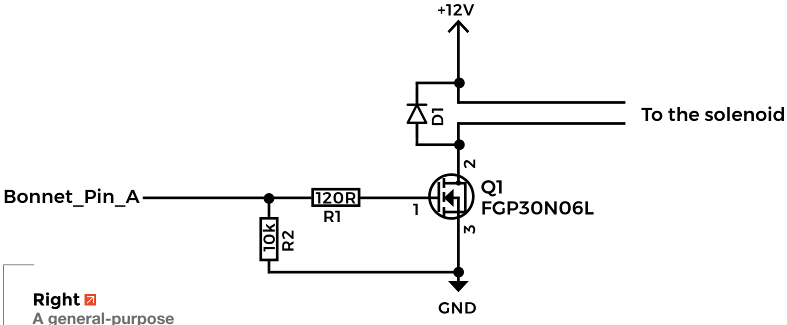
We used an Adafruit ItsyBitsy as the control board. An H-bridge controls the speed and power of the motor, and a potentiometer provides the position feedback.

The servo code consists of the standard Arduino setup and loop, along with some extra functions for handling the I^{2}C commands and requests.

The start of the code consists of constants for the pins and I^{2}C parameters. The variables that could →

Below OnOffSHIM and power





Right A general-purpose low-side driver circuit

be changed by the wire events are marked as volatile so that the compiler does not optimise their use by swapping in constant values.

```
#include <Wire.h>

const int slaveAddress = 8;

const int sensePin=A0;. //Needs to support
AnalogRead
const int drivePin1=3;
const int drivePin2=4;
const int speedPin=5;   //Needs to support PWM

enum readRegisters { Target = 0,
                     Running = 1,
                     Position = 2,
                     Speed = 3,
                     Direction = 4};

enum commands { CmdStop = 1,
                CmdAngle = 2,
                CmdSpeed = 3,
                };

volatile int registers[5];
volatile int readCmd;
```

In the setup, we configure the pins used to drive the motor and ensure the motors are off.

`Wire.begin` defines the address used by the servo. If you had multiple servos, you could use an EEPROM to store this address.

`Wire.onReceive` and `Wire.onRequest` registers the functions that are called when data is received on the I^{2}C bus. These are often called event handlers or callbacks.

```
void setup() {
  stop();

  pinMode(drivePin1, OUTPUT);
  pinMode(drivePin2, OUTPUT);
  pinMode(speedPin, OUTPUT);
  pinMode(sensePin, INPUT);

  Wire.begin(slaveAddress);          // join i2c
bus with address #8
  Wire.onReceive(receiveEvent);  // write data
  Wire.onRequest(requestEvent); // requests to
read data
}

void stop() {
  digitalWrite(drivePin1, LOW);
  digitalWrite(drivePin2, LOW);
  analogWrite(speedPin, 0);
  registers[Running] = false;
}
```

The loop is used to record the position of the potentiometer and check if we have reached the target position. Once the target position has been passed, the motor is stopped until a new position command is sent.

```
void loop() {
  registers[Position] = analogRead(sensePin);

  if (registers[Running]) {
     analogWrite(speedPin, registers[Speed]);
    if (registers[Direction] == 1) {
      if (registers[Position] >=
registers[Target]) {
        stop();
      }
    }
    if (registers[Direction] == -1) {
      if (registers[Position] <=
registers[Target]) {
        stop();
      }
    }
  }
}
```

The `receiveEvent` function takes the data from the host computer and acts on the various commands. It is important to read all of the data sent, or the handler will not get called again. Each of the messages consists of three bytes: the first is the command and the second two are the high and low byes of an integer parameter

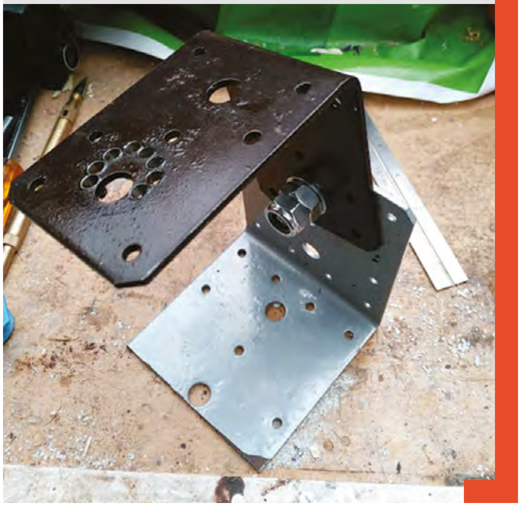
```
void receiveEvent(int bytesReceived) {
  char command;
  int value;

  if (bytesReceived == 1) { //Read request
register
    readCmd = constrain(Wire.read(),0,5);
```

GPIO

The Microchip MCU on the Vision Bonnet handles the buttons, LEDs, and buzzers in the kit. It connects to the Pi via I^{2}C so it can handle several I/Os without using the Pi's GPIO pins. Software handles pulsing lights and debouncing switches.

CHAIN DRILLING

Large holes can be made by drilling lots of smaller holes in a circle and cutting through with a saw. We used a half-round file to finish the edges.



Above
How to cut a large hole

```
}

 if (bytesReceived == 3) {
  command = Wire.read(); // receive byte as a
character
  value = Wire.read();
  value = value << 8;    // shift high byte to
be high 8 bits
  value |= Wire.read(); // receive low byte as
lower 8 bits

  switch (command){
   case CmdStop:
    stop();
    break;
   case CmdAngle:
    registers[Target] = value;
    registers[Running] = true;
    if (registers[Target] >
registers[Position]) {
     registers[Direction] = 1;
    }
    else {
     registers[Direction] = -1;
    }
    break;
```

```
   case CmdSpeed:
    registers[Speed] = value;
    break;
 }
  }
 else
  {
   //Invalid command throw buffer away so we
can read again
   for (int a = 0; a < bytesReceived; a++) {
     Wire.read();
   }
  }
}
```

The full list of commands which are supported by the motor are:

Status: Sets the status for when request is called

Stop: Stop the motor

Move to: Sets the target position

Speed: Sets the speed of the servo

The **requestEvent** returns the value of the register as already selected by the **CmdStatus** command.

```
void requestEvent() {

 String value = String(registers[readCmd]);
 char message[2];
 value.toCharArray(message, 2);

 Wire.write(message,2);
}
```

If you can locate a powerful servo, you can control that using GPIO Zero's AngularServo class. See the ObjectMeter example from the Vision Kit – you can find it on GitHub at **hsmag.cc/GCEMXg**. →

WHAT IS TENSORFLOW?

TensorFlow is an open-source software library for numerical computation using data-flow graphs. A tensor is a multidimensional array of numbers. The Vision Bonnet works by having trained models that take an image in the form of an array, and produce a list of objects in the scene and a score as to how confident the match may be. For example, one model is trained to tell you what a plate of food contains, or if the image contains any smiling faces.

Below
An IP67-rated box should keep water out

BRAIN POWER

The Google AIY Vision Kit has limited documentation, but there are good examples and plenty of source code. The software on the Vision Kit was made by combining the various Python examples. There are many prebuilt TensorFlow classifier models that identify different kinds of images. We used one that can detect people, cats, and dogs; our code just responded to people.

Many of the examples use an on-screen preview, but as we were not using a display, this proved not to be necessary.

The `import` lines bring in all of the various premade libraries to recognise images and talk to the hardware.

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

from aiy.vision.inference import CameraInference
from aiy.vision.models import object_detection
from aiy.vision.leds import Leds
from aiy.vision.leds import Pattern
from aiy.vision.pins import PIN_A
from gpiozero import OutputDevice
from picamera import PiCamera
import time
import smbus
```

Variables are then configured for the I²C bus, solenoid, and the LEDs:

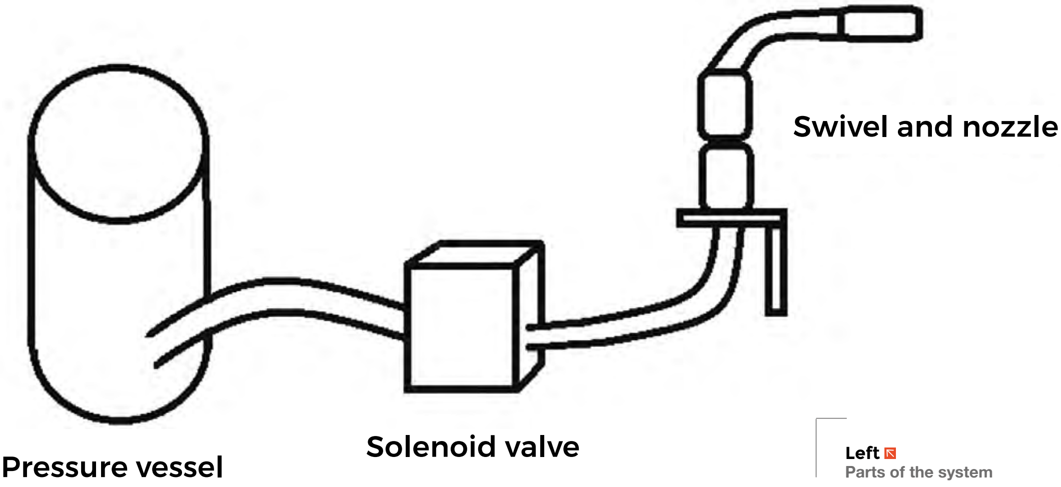
```
RED = (0xFF, 0x00, 0x00)
leds = Leds()
leds.pattern = Pattern.breathe(1000)
bus = smbus.SMBus(1)
slaveAddress = 8
solenoid = OutputDevice(PIN_A)
```

We start by turning on the button light so you know the code is running. Next, we start the image recognition from the camera. This is configured for 1640×1232 mode, which matches the training for the Vision Bonnet. If something is detected then we check if it is a human (`kind == 1`) and there is a good match (`> 0.7`). The position of the detected human relative to the centre of the image is then sent to the squirt routine, which is scaled by the width of the screen.

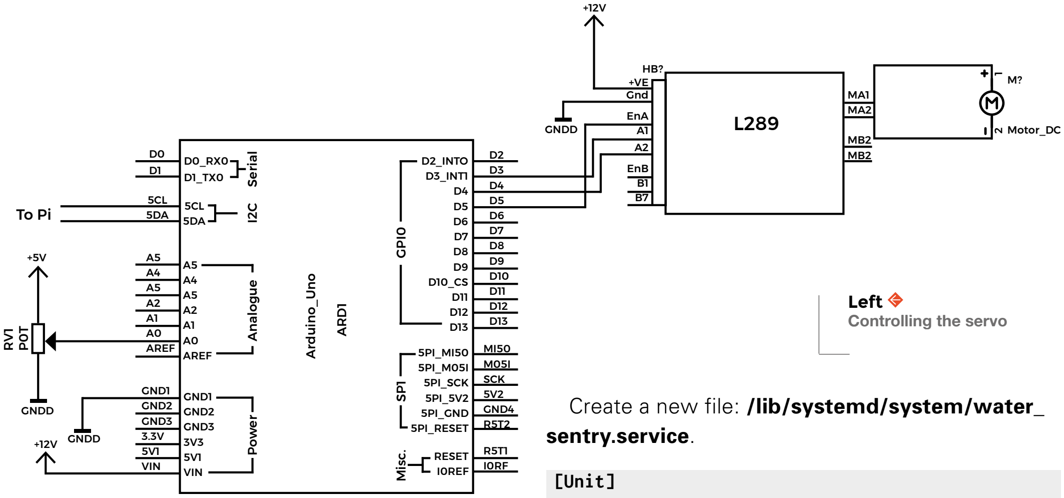
```
def main():
   # Turn on the LED so we know the box is ready
   leds.update(Leds.rgb_pattern(RED))

   with PiCamera() as camera:
       camera.sensor_mode = 4
       camera.resolution = (1640, 1232)
       camera.framerate = 30

       with CameraInference(object_detection.
model()) as inference:
           for i, result in enumerate(inference.
run()):
               for i, obj in enumerate(object_
detection.get_objects(result, 0.3)):
                   if  obj.score > 0.7 and obj.
kind == 1:   # Person
                       x, y, width, height = obj.
bounding_box
                       squirt((x + (width/2) -
(1640 / 2 )) / 1640)

if __name__ == ‘__main__’:
   main()
```

OTHER IDEAS

- Use the width of the bounding box to estimate distance and get better targeting
- Train your own TensorFlow to squirt foxes or squirrels
- Train your own TensorFlow to look for wilting plants, and water them automatically
- Build an intelligent tennis trainer that can give you practice on forehand or backhand
- Instead of the Vision Bonnet, you could use a Movidius Neural Compute Stick to analyse your images



Left Parts of the system



Left
Controlling the servo

The squirt routine calculates the angle. When checked, the image distance at 2.5 m is approximately 2.9 m wide. It then sends a command to rotate the servo and reads the **IsRunning** register until it stops moving. The solenoid is activated for 1s, and the light is turned to solid red. Finally, the solenoid is turned off and the light is set back to the breathe pattern.

```
def squirt (position):
    angle = math.degrees(math.atan((position/2.9)
* 2.5))
    bus.write_byte_data(slaveAddress , 2, angle )

    while bus.read_word_data (slaveAddress, 1) !=
0:
        time.sleep(0.1)

    leds.update(Leds.rgb_on(RED))
    solenoid.on()
    time.sleep(1)
    solenoid.off()
    leds.update(Leds.rgb_pattern(RED))
```

TURN ON AND OFFABLE

When the Vision Kit is first installed, it comes with a smile detector as default. We want to swap this out with a service to run our code.

First, we disable the smile detector.

```
sudo systemctl stop joy_detection_demo
sudo systemctl disable joy_detection_demo.service
```

Then we create a file that specifies the script we want to start. This consists of a title, plus parameters that determine that it runs as Pi and executes a Python script. Finally, the last parameter determines when the script is started; **multi-user.target** is run when most of the operating system is ready for use and the machine is ready for login.

Create a new file: **/lib/systemd/system/water_sentry.service**.

```
[Unit]
Description=Robot Water Sentry

[Service]
Type=simple
Restart=no
User=pi
ExecStart=/usr/bin/python3 /home/pi/human_
detection_camera.py

[Install]
WantedBy=multi-user.target
```

The last step is to install and enable our startup service.

```
sudo systemctl enable water_sentry.service
sudo systemctl start water_sentry.service
```

PLAY THE GAME

To prepare the game, you fill the sprayer with water and prime it by pumping it with air. The sentry is then activated by pressing the button on the top.

At this point, the camera is activated and the Vision Bonnet starts looking for humans. If it locates a human, the co-ordinates are passed on to the control software which then moves the nozzle to point at the human. The controller then activates the solenoid valve, squirting the person.

If you've just got a Pi Zero, why not add a remote control to manually target the water? You could also use a Pi 3, Pi Camera, and OpenCV to create a software-based vision system. □

LEARN MORE

- developers.google.com/machine-learning/crash-course
- projects.raspberrypi.org/en/projects/getting-started-with-picamera

Joystick MIDI controller

Use the Arduino platform to build a prototype USB MIDI controller

Oliver Owen

@oliverjowen

Oliver Owen is a computer musician based in Bristol, UK.

As an experimental electronic musician, I've found Arduino to be a great platform for building interesting interfaces to control my software instruments.

This tutorial will explain how to build a prototype MIDI controller fo use with digital audio workstations, software synthesizers, effects, or DJ software applications.

You'll use two 'thumb' joysticks as sensors, and will send the readings from these as MIDI control change data over USB.

Your controller will include the function to change the MIDI channel being used, and will also include an LCD display to give visual feedback to the user.

Right This prototype USB MIDI controller will allow control of software synthesizers, effects, and much more

Starting with the physical components, you'll want to solder some male headers to your Pro Micro, and some female headers to the LCD display.

You'll also need to solder the joysticks to their breakout boards, and afterwards also solder some male headers to these.

With headers in place, you can start to assemble your components as per the Fritzing diagram (see **Figure 1**, overleaf).

Your joysticks will both need to connect to the VCC and GND pins on your Pro Micro, and to analogue pins A0 to A3, where you'll measure readings from each joystick in the horizontal and vertical direction.

Connect from SEL on your joysticks to digital pins 7 and 8, which you'll use for detecting the button press function.

The LCD display connects to VCC and GND in a few places, including via a 220 Ω resistor, and a 10 kΩ potentiometer to adjust the brightness of the display.

You'll also need to connect the display to digital pins 2, 3, 4, 5, 9, and 10, as shown in the diagram.

With that done, you hopefully have something that looks neater than our skeleton build over on page 91.

Now, let's look at the code you'll need to make the controller work.

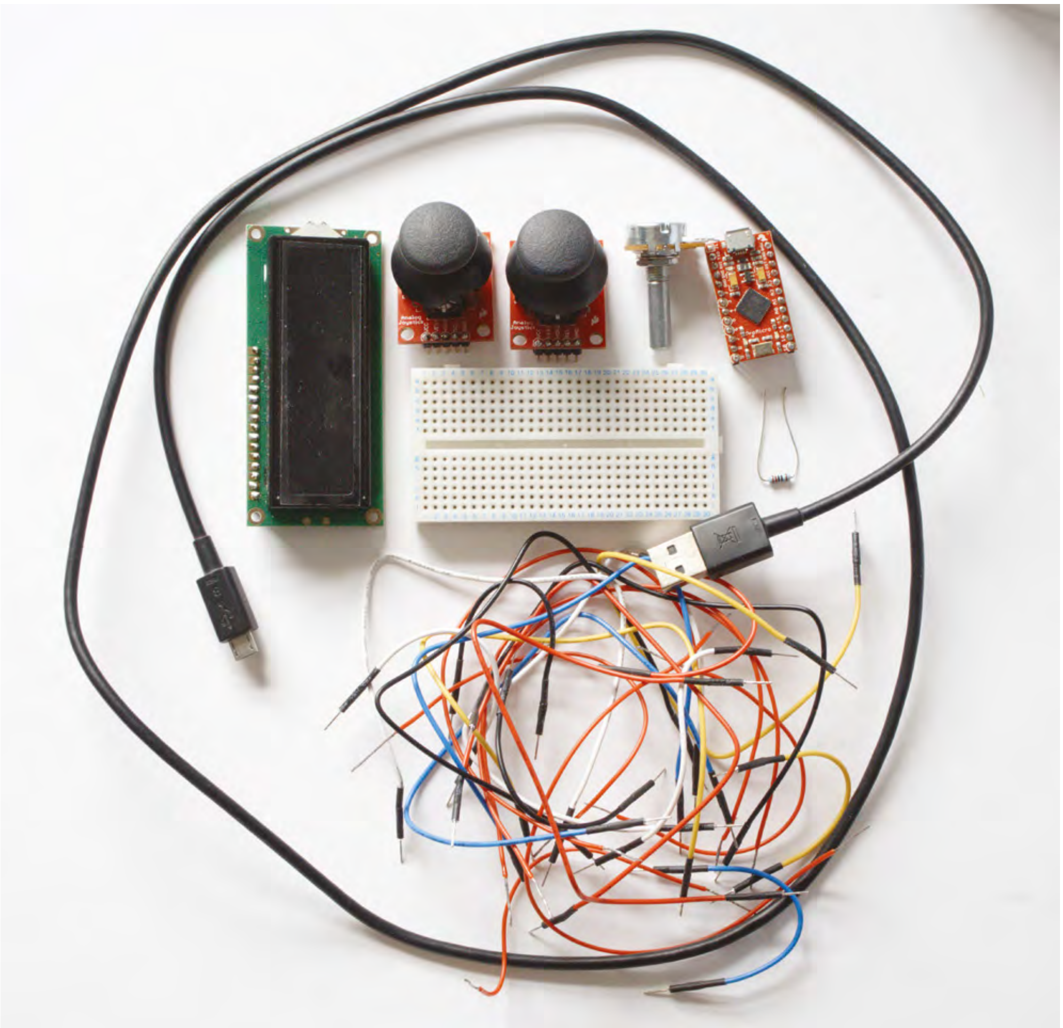
To begin with, there are two libraries that you'll be using: the MIDIUSB library, which facilitates sending MIDI messages from your board over USB, and the LiquidCrystal library, which lets you use your LCD display.

The LiquidCrystal library is already included in the Arduino IDE, but you'll need to search for and install the MIDIUSB library from the Library Manager.

Once you've done that, include them both in your code:

```
#include <MIDIUSB.h>  //Include library for sending
MIDI data from Arduino over USB
#include <LiquidCrystal.h>  //Include library for
interfacing with LCD display
```

Now declare the global variables needed throughout the code, including which pins on the Pro Micro will be used for what.



Left You don't need much to make your own MIDI controller

You'll want variables to store readings from each of your joysticks – in both the vertical and horizontal axes – and variables for the button presses and managing your channel data.

Also declare that you'll be using digital pins 7 and 8 for button presses and digital pins 2, 3, 4, 5, 9, and 10 for your LCD display.

```
//Declare variables & pins to be used for
joystick 'select' buttons
const int Stick_1_Switch_In = 7;
const int Stick_2_Switch_In = 8;

//Declare other variables to be used
int Stick_1_V;
byte Ctrl_1_V;
int Stick_1_H;
byte Ctrl_1_H;
int Stick_2_V;
byte Ctrl_2_V;
int Stick_2_H;
byte Ctrl_2_H;
byte Channel;
byte Channel_Num;
byte Channel_Print;
byte Button_State_1;
byte Button_State_2;

//Declare pins to be used with LCD display
const int rs = 10, en = 9, d4 = 5, d5 = 4, d6 =
3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
```

In your **`setup()`**, initialise your button pins (7 and 8) as inputs, and initialise the built-in pull-up resistors.

Also, set the baud rate for your serial connection – 31250 seems to be widely used for MIDI over USB – and initialise your LCD display, declaring the columns and rows you'll be using (16, 2).

```
void setup() {
  pinMode(Stick_1_Switch_In, INPUT_PULLUP); // →
```

YOU'LL NEED

- **An Arduino-compatible, ATmega32U4 board** (I used the SparkFun Pro Micro)
- **USB to micro USB cable** hsmag.cc/mQPxKb
- **2 × SparkFun thumb joystick controllers** hsmag.cc/RWmwTF
- **2 × SparkFun breakout boards** hsmag.cc/CSILUt
- **A breadboard**
- **An LCD display, compatible with the Hitachi HD44780 driver** hsmag.cc/yomkpT
- **220 Ω resistor**
- **10 kΩ potentiometer**
- **A selection of jumper wires**
- **A selection of headers, male and female**
- **Cardboard** (optional)
- **A soldering iron and solder**
- **A PC with the official Arduino IDE installed**

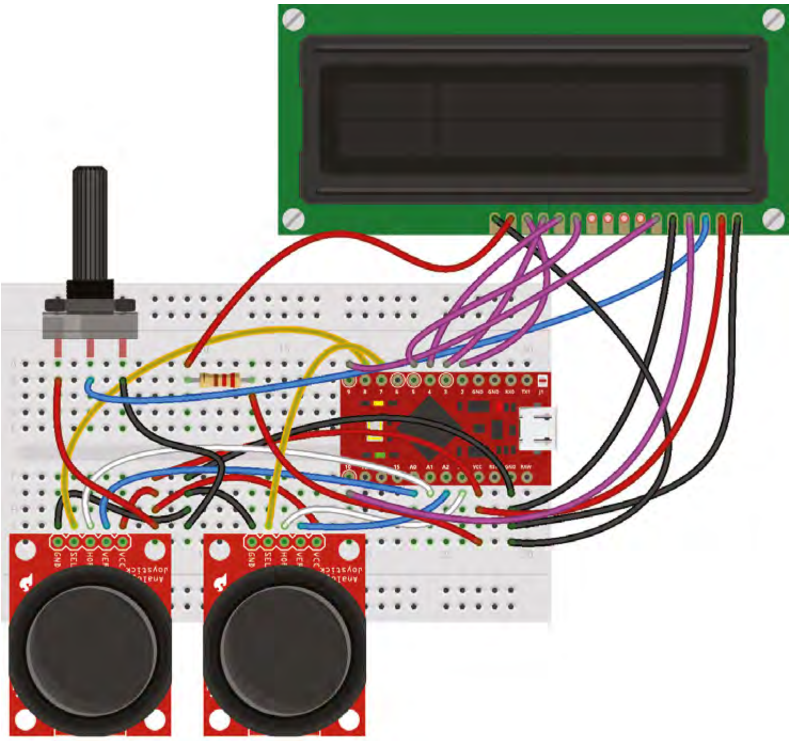


Figure 1 With headers in place, you can start to assemble your components as per this Fritzing diagram

```
Initialise ‘select’ pins
  pinMode(Stick_2_Switch_In, INPUT_PULLUP);
  Serial.begin(31250);  //Set baud rate for MIDI
over USB
  lcd.begin(16, 2); //Initialise LCD display &
declare size of display (columns, rows)
}
```

Now is a good time to declare two functions you'll want to use.

The **controlChange()** function is included in the MIDIUSB library. It takes an argument for channel, 'controller number', and 'value', and will package and send this as MIDI control change messages over USB.

It's worth mentioning that the function expects these arguments to be delivered as 'byte' data types.

```
//Declare ‘controlChange’ function to ‘package’ &
send MIDI data over USB
void controlChange(byte channel, byte control,
byte value) {
  midiEventPacket_t event = {0x0B, 0xB0 |
channel, control, value};
  MidiUSB.sendMIDI(event);
}
```

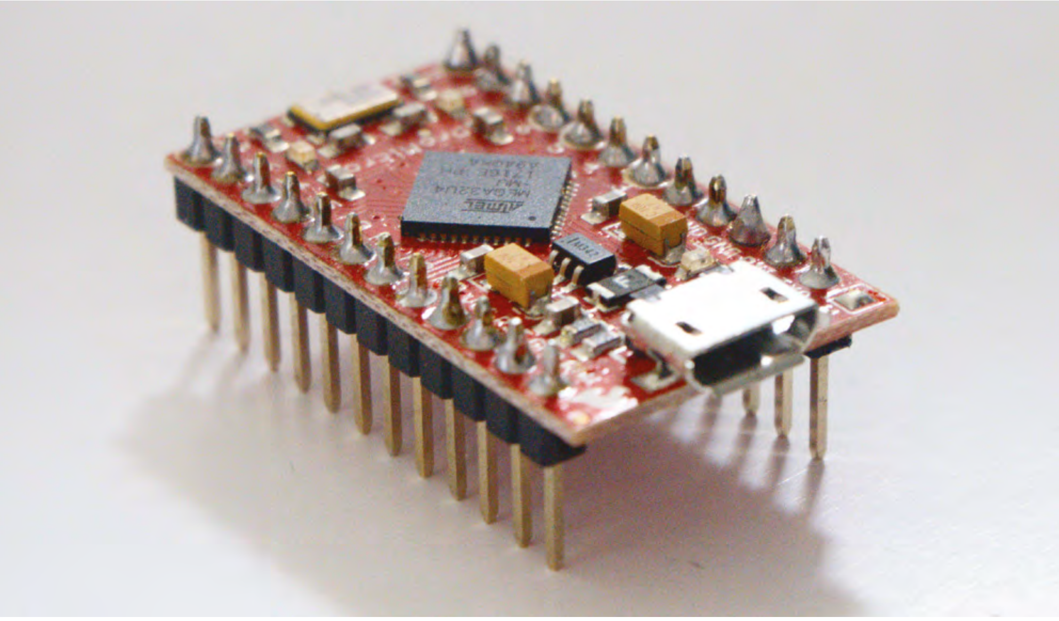
You'll now declare your own function **ChannelChange()** to handle the changing of the channel value in your code when a joystick's button is pressed.

The function will take two arguments – a variable for changing the channel up, and one for down.

Within the function, declare two variables to handle the input from the arguments.

```
byte ChannelChange(int Input_Up, int Input_Down)
{
  int Stick_1_Switch = digitalRead(Input_Up);
  int Stick_2_Switch = digitalRead(Input_Down);
```

Now your function will use an if/else statement to process your button presses.



Right The SparkFun Pro Micro has a tiny form factor and can plug directly on to your breadboard, once headers are in place

First, check if the button is currently being pressed. If it isn't, listen for a button press, incrementing the channel up or down when one is detected.

If the button is actively being pressed then do nothing until it is released.

```
  if (Button_State_1 == 0) {
    if (Stick_1_Switch == LOW) {
      Channel_Num ++;
      Button_State_1 = 1;
    }
  }
    else if (Stick_1_Switch == HIGH) {
      Button_State_1 = 0;
    }

  if (Button_State_2 == 0) {
    if (Stick_2_Switch == LOW) {
      Channel_Num --;
      Button_State_2 = 1;
    }
  }
    else if (Stick_2_Switch == HIGH) {
      Button_State_2 = 0;
    }
```

MIDI supports up to 16 channels (0–15), so you'll want to 'wrap' your channel number to stay within that range.

Now, your **Channel_Num** variable is a 'byte' data type, which will already be 'wrapping' from 0–255 if it goes below 0, so check whether the variable is greater than or equal to 255, and if it is, set your channel to 15.

If the variable goes above 15, the channel is set to 0.

The function returns the channel number for use later in the code.

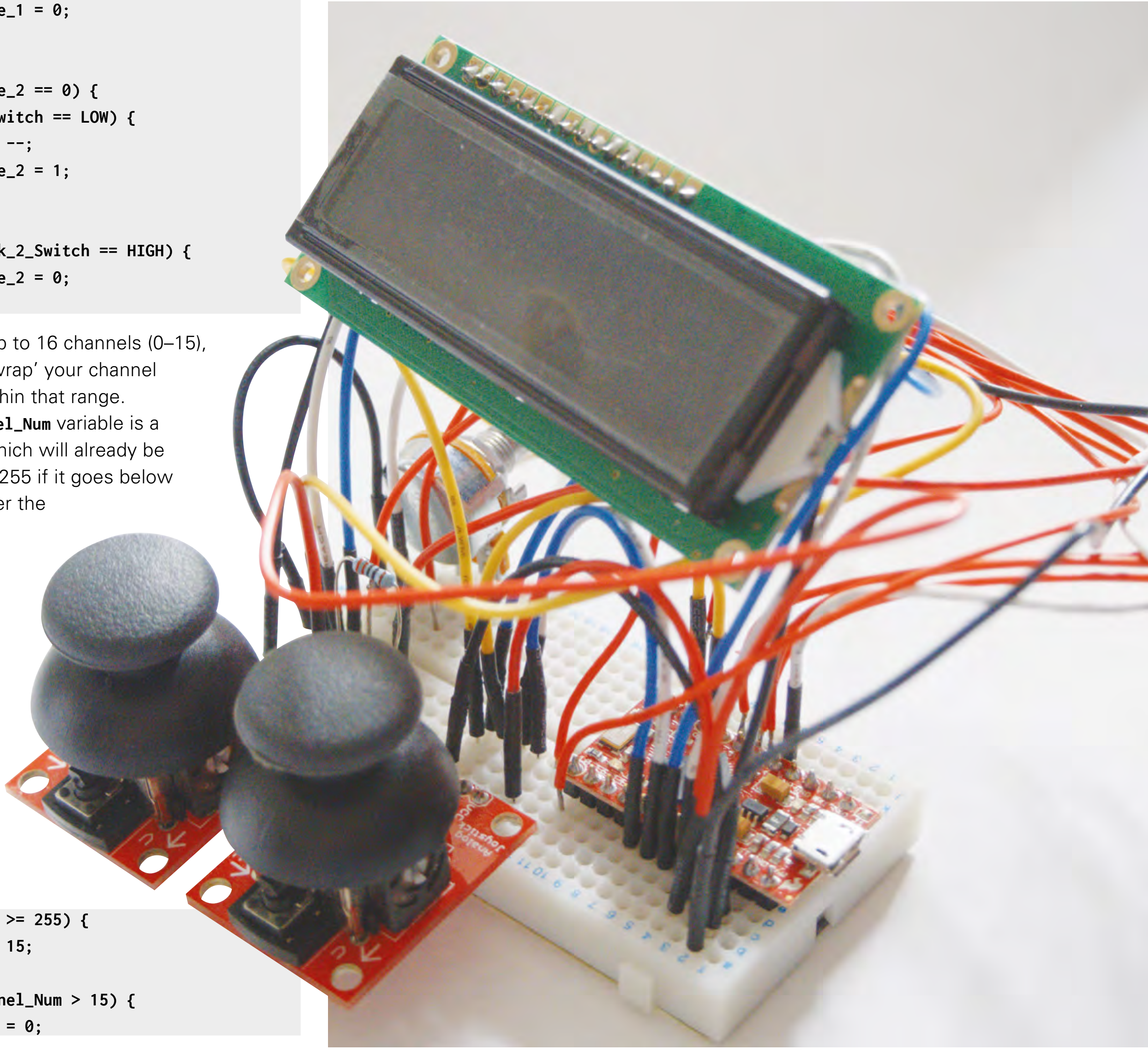
```
  if (Channel_Num >= 255) {
    Channel_Num = 15;
  }
    else if (Channel_Num > 15) {
      Channel_Num = 0;
    }

  return Channel_Num;
}
```

With variables and functions declared, now move on to the main loop of your code.

Start by calling your **ChannelChange** function.

```
void loop() {

  Channel = ChannelChange(Stick_1_Switch_In,
Stick_2_Switch_In); //Call ‘ChannelChange’
function
```

→



Below Despite the chaos of jumper wires, all the components are in place for your controller

Above Soldering the joysticks to breakout boards will simplify your prototyping

Next, we need to take a reading from each analogue pin in use. It will be in the range of 0–1023. A MIDI value should be in the range of 0–127, so you'll have to scale your reading, multiplying it by 0.124144673.

Now you need to call the `Control_Change` function, giving your channel and scaled joystick readings as arguments, along with a controller number argument – we chose 85–88 to avoid other commonly used values. Then use `MidiUSB.flush()` to send your message.

MIDI

MIDI (Musical Instrument Digital Interface) is a standard and protocol for sending data between musical instruments, devices, and computers. It was conceived in the early 1980s, with the specification first published in 1983.

MIDI can communicate a message over 16 individual channels, and a message might contain information relating to note selection, note on or off, velocity (how hard the note was pressed), or perhaps a controller number and value.

Values are within the range of 0–127, so the note selection of middle C on a keyboard would be the value 48. A control knob turned to its midway point would be a value of 64.

A 'note on' message would include: an identifier as a 'note on' message, a channel number (0–15), a pitch number (0–127), and a velocity (0–127).

Similarly, a control change message would include an identifier as a control change message, a channel number, a controller number, and then a value.

```
  Stick_1_V = analogRead(A0);
  Ctrl_1_V = (Stick_1_V * 0.124144673);
  controlChange(Channel, 85, Ctrl_1_V);
  MidiUSB.flush();

  Stick_1_H = analogRead(A1);
  Ctrl_1_H = (Stick_1_H * 0.124144673);
  controlChange(Channel, 86, Ctrl_1_H);
  MidiUSB.flush();

  Stick_2_V = analogRead(A2);
  Ctrl_2_V = (Stick_2_V * 0.124144673);
  controlChange(Channel, 87, Ctrl_2_V);
  MidiUSB.flush();

  Stick_2_H = analogRead(A3);
  Ctrl_2_H = (Stick_2_H * 0.124144673);
  controlChange(Channel, 88, Ctrl_2_H);
  MidiUSB.flush();
```

To improve usability, you'll now add an LCD display to the controller to give visual feedback on channel selection and the values being sent from each joystick. The LCD you're using is divided into two rows and 16 columns. To state where on the display you want information to appear, use the command `lcd.setCursor()`, with arguments for column and row.

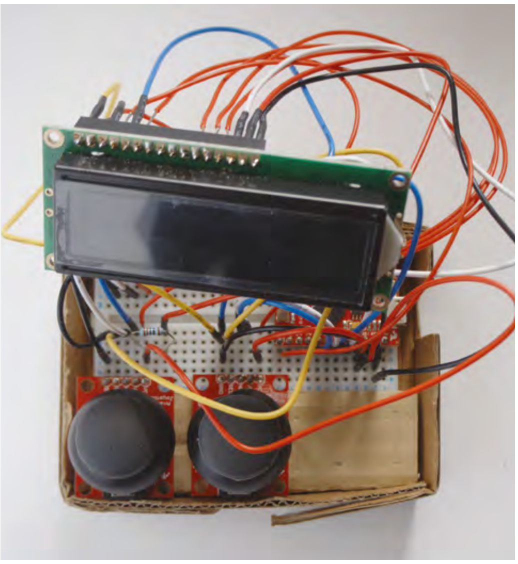
To write information on the display, use the `lcd.print()` command with the information you want to display as an argument, either enclosed in quotation marks for an explicit piece of text, or by giving the variable name for something that'll be updated. As you update `lcd.print()`, you can run in to problems with redundant values remaining on the display, particular when changing from two or three digit numbers to single digit numbers.

You'll work-around this problem using a series of if/else statements, and by displaying zeros before your number – so a single-digit number such as 9 will display as 009, and a two-digit number such as 76 would display as 076, etc.

Before you print a value to the display, use an 'if' statement to check how large the value is. If it's smaller than 10 then you know it will be a single-digit number and so you'll print two zeros to the display, reposition your cursor, and then write the value.

If the value is equal to or larger than 10 then you know it will be a two-digit number and so will only require a single zero to be printed before the value.

```
lcd.setCursor(0, 0);
  lcd.print("Channel: ");
```



Above Don't worry about the great profusion of wires – the next step is to tidy them up and hide them away

```
if (Channel_Print < 10) {
  lcd.setCursor(8, 0);
  lcd.print(“00”);
  lcd.setCursor(10, 0);
  lcd.print(Channel_Print);
}
else if (Channel_Print >= 10) {
  lcd.setCursor(8, 0);
  lcd.print(“0”);
  lcd.setCursor(9, 0);
  lcd.print(Channel_Print);
}
```

The values read from your joysticks will be in the rage of 0–127, so add a third clause to your if/else statements to check whether the value is equal to or greater than 100 before printing.

```
if (Ctrl_1_V < 10) {
    lcd.setCursor(0, 1);
    lcd.print(“00”);
    lcd.setCursor(2, 1);
    lcd.print(Ctrl_1_V);
 }
 else if (Ctrl_1_V < 100) {
    lcd.setCursor(0, 1);
    lcd.print(“0”);
    lcd.setCursor(1, 1);
    lcd.print(Ctrl_1_V);
 }
 else if (Ctrl_1_V >= 100) {
    lcd.setCursor(0, 1);
    lcd.print(Ctrl_1_V);
 }
```

SPARKFUN PRO MICRO ATMEGA32U4 MICROCONTROLLER

For this project, it's important to source an ATmega32U4 board that has native support for USB. It's this detail that makes it possible to send your MIDI message over the USB connection.

I decided to use the 5 V/16MHz SparkFun Pro Micro board, as it has all the analogue and digital inputs required for the project, but in a tiny form factor.

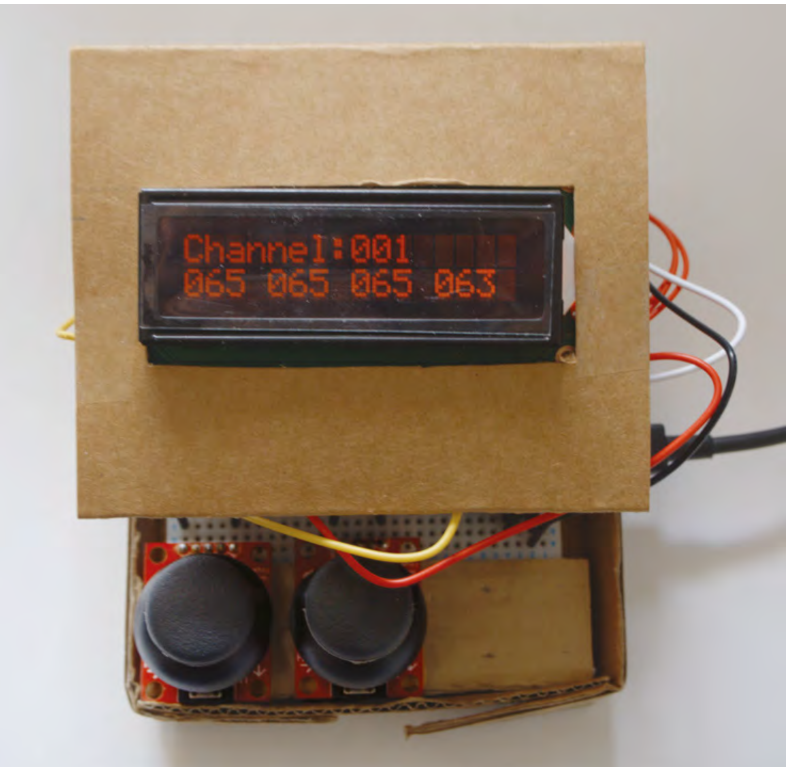
The Pro Micro isn't included in the Board Manager of the Arduino IDE, so you'll need to install it to the environment before you begin. Full details are on the SparkFun website here: **hsmag.cc/AgjkdW**.

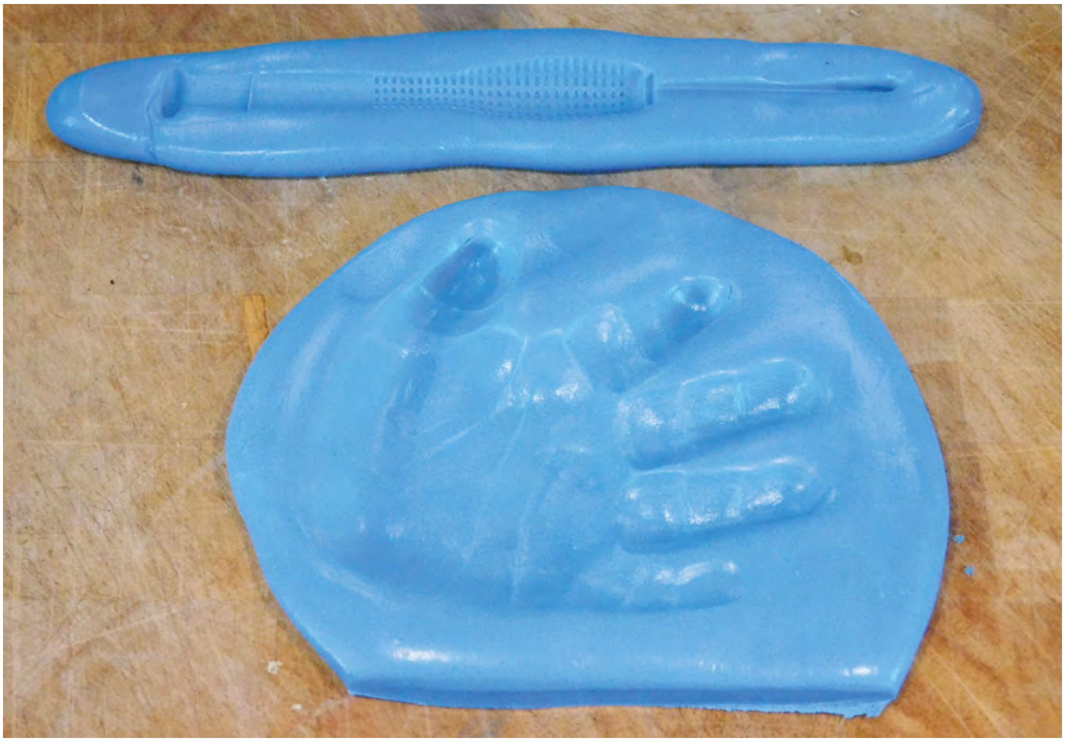
Once complete, the SparkFun Pro Micro board will be shown in the list of available boards in the Board Manager. It's important to select the exact board you are using, i.e. 5 V/16MHz, before attempting to upload any code – selecting the wrong board can have disastrous consequences for your board, and your project.

This workaround is satisfactory for your needs here, where you only need to display whole integer values. If you had a project that required you to display a decimal, you'd need to devise a new method – the number 1.23 is less than 10, but actually needs four characters to display properly.

You've learnt how to work with third-party Arduino-compatible boards, use external libraries to facilitate working with MIDI and LCD displays, as well as a little about the MIDI standard. □

Below Cardboard is great for prototyping; stiff enough for stability, but easy to shape





Right We also tried to mould a screwdriver, but the fine detail didn't come through well in a shallow impression

Making food moulds

Making baby-shaped chocolate mementos

Ben Everard

@ben_everard

Ben makes stuff. We'd like to be more specific, but his interests cover too wide a range to narrow down any further.

There are loads of different methods for making cases for moulding in – 3D printing, carving from foam, lost-wax casting, to name but a few. There are also lots of foods that are mouldable into different shapes such as chocolates, cakes, and ice cubes. Wouldn't it be great to combine the two?

The problem with such things is that it's hard to know which materials are safe to use with food. Some things are known to be toxic, while other things simply aren't well-known – manufacturers aren't required to list all the ingredients on their products, so toxic chemicals could be contained in surprising places. Not only this, but you have to consider the entire build process. For example, not all 3D printer nozzles are safe, even when used with safe filament. Even when working with food-safe materials and a food-safe process, you could still contaminate your make if the equipment has a build-up of non-food-safe materials on it. It's easy to get complacent about handling toxic chemicals when you do it regularly and don't feel ill after using them, but this is a dangerous attitude to take. Many hazardous materials don't make you sick immediately, but build up over time.

All this doesn't mean that you can't, or shouldn't, get creative when working with food, just that you should think about the materials and processes you use when doing it. We're going to look at one simple method for making food moulds using food-safe materials, and no equipment that could be contaminated.

There are a few sources of food-safe silicone for modelling with, and the one we're using is sold under the brand name Silicone Plastique. It's easy to work with, and can be used with both hot (up to 230 degrees C) and cold.

Silicon Plastique comes as a two-part putty. When you want to use it, simply knead the blue and white parts together until you have a mixture with an even colour, and you've got about fifteen minutes to create the impressions that you'll use as a mould.

Your author is a new father, and as such, prone to saccharine displays of sentimentality, so our example uses a baby's hand. While many new parents distribute mugs, plates, and other crockery adorned with their child's handprint, this proud new father felt his friends would appreciate chocolate more. Of course, this is just an example. You could use just about anything to make an imprint for the mould. The only real caveat is that using this simple technique, you can only make an impression from one side of the object. Creating a full 3D mould, showing off all sides, would be a bigger project.

You could simply shape your mould by hand, but in many cases it's easier to do it by pressing the putty over the object to get an impression of its shape. This way the mould will create exact replicas of the donor object with an impressive level of detail.

To make the impression, roll out the Plastique until it's big enough and deep enough (there needs to be enough putty in the thinnest part to avoid the mould breaking when it's being used). Ours was about 1 cm thick, but yours may need to be thicker if you want a bigger mould. Then clean the object that you're making the impression from (if it's a child's hand, this might take some time) and press in firmly. If possible, the object should be left in the putty as it cures, but this isn't always possible. You can remove the object from the putty if necessary, provided the impression is shallow enough to allow this without disturbing the putty.

According to the documentation, Silicone Plastique takes an hour, but we left it for two, just to be safe. After waiting this time, you can start moulding.

You can use your mould for all sorts of food preparation, but for chocolate, simply create a bain-marie, melting chocolate in a heat-proof bowl on top of a saucepan, with a small amount of boiling water in. The steam should then gently heat the bowl and melt the chocolate without burning it. Once it's liquid, pour it into the mould, and push it around to eliminate bubbles. Leave to cool and enjoy. □

OTHER **FOOD-SAFE MOULDS**

If you read the title and expected to find ways of enhancing food with fungal activity, then here's a few of our top moulds for producing flavour:

Penicillium camemberti produces a fine white mould on the surface of cheeses such as Camembert and Brie. The mould also aids the ripening that sees the cheese progress from a flaky cheese into a soft, gooey delight.

Penicillium roqueforti is responsible for the blue veins and strong flavours in cheeses such as in Stilton, Gorgonzola, and Roquefort.

Aspergillus oryzae (also known as Koji) is used to ferment soy beans to make soy sauce and also to convert the starches in rice to sugars so they can then be fermented to alcohol (such as in Sake).

Truly dedicated punsters can combine both types of mould in a hand-shaped blue cheese.

> " **You could simply shape your mould by hand,** but in many cases it's easier to do it by pressing the putty over the object to get an impression "

Left
Isobel's mother wouldn't let us press her entirely into the putty to create a chocolate homage to the *Star Wars* Han Solo-in-carbonite scene, so just one hand had to do

Maker's Toolbox: Oscillating multi-tool

Cut, sand, and scrape with precision and ease

Ben Everard

@ben_everard

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

Above This may look like a cross between an angle grinder and a circular saw, but the blade moves just 3.2 degrees each cycle, making it a much more delicate tool

The phrase 'multi-tool' is usually used to refer to a folding pack of knives, screwdrivers, saws, and other gubbins that you can fit in your pocket. What we're looking at here is a completely different beast. Oscillating multi-tools (also known as power multi-tools) basically consist of a handle with a built-in motor that can oscillate whatever you attach to it. Typical attachments are saw blades, sanders, and scrapers (for removing paint, glue, and so on). This oscillation is a rotation a small amount in each direction (typically 3.2 degrees).

The oscillating cutting action has a fairly unusual property in that it's very inefficient at cutting soft materials (such as flesh), because any material with enough give to move will simply vibrate rather than be cut. Oscillating tools very similar to these are used to remove plaster casts in hospitals because they cut the still plaster cast and not the flesh underneath. We were, however, unable to find a volunteer willing to test if a multi-tool would cut their skin. No power tool can be considered truely safe, but the oscillating multi-tool is one of the safer, provided the usual precautions are allowed for (such as eye and ear protection, and ensuring you have a firm, secure footing before starting).

The cutting blades get through all but the hardest materials. Providing you're using the right blade, wood, metals, and ceramics all yield to these tools.

As well as being versatile in their materials, oscillating tools also allow a wide range of cutting angles. The blade can typically be rotated to any angle to allow it to cut in tight spaces, and the blades can typically be 'plunged' straight into a material without having to start at the edge. As such, it's great for cutting patterns into a sheet of material, or cutting out space for a component.

One issue with the oscillating action is that the blade doesn't move far enough to clear the debris out from between the teeth when cutting, so you have to ensure that you keep bringing the blade out of the cut to allow the swarf to come out, or moving it across the cut in thin materials. These multi-tools are quite slow for cutting, but the counterpoint of this is that they allow for precise incisions. In this way, it's almost the opposite of the reciprocating saw that we looked at last issue.

The sanding action is similar too, and as such, it's great for getting into corners and tight spaces. The scraper tools work much like manual paint scrapers, but the addition of oscillation makes the scraping more effective.

Oscillating multi-tools combine a range of features that mean they're well-suited for makers working with existing objects. The cutting action is great for re-working or adding features to existing wood or metal objects, while the sander and scraper options are good for preparing surfaces for a new finish.

A BUYER'S GUIDE

As with almost all power tools, there's a wide variety of options available at an array of prices. You can pick up an oscillating multi-tool with accessories for under £50, and they go up to over £200.

These tools are obviously heavily dependent on the various attachments you can get for them. Most tools are either OIS or Starlock (with Starlock being the newer option), and there are plenty of attachments available in both standards (and some work with both). Make sure that the tool you get is compatible with the attachments you want before purchasing. Some tools come with attachments, while others don't. This can make quite a difference to the total cost of the setup.

Another important feature is the ability to control the speed of the oscillations. In general, slower speeds are going to be better for more detailed work, while faster speeds get through things faster. A tool that goes down to around 6000 oscillations per minute will allow fine-detailed cutting. But, unless you want to wait around all day for your saw, you'll want a tool that goes up to something in the region of 20 000.

As well as the technical specifications, different tools also inevitably have differing build qualities. Cheap blades will dull quite quickly, and cheap tools often don't stand up to long-term use. As with most power tools, you tend to get what you pay for in terms of build quality, and a multi-tool from a reputable manufacturer will probably last you much longer than a no-name tool if used regularly. However, cheaper tools can be fine if you only need them occasionally.

The tool we used for this article is a Makita TM3000C/2, which represents a good mid-range tool. It uses the OIS attachment as standard, and has variable speed oscillations of up to 20 000 oscillations per minute. It's mains-powered, so there are no batteries to worry about (which may or may not be a plus, depending on your circumstances). The tool itself feels tough and should stand up to workshop use. It doesn't come with any accessories though, so if you do purchase this tool, you'll need to get them separately. □

> "Oscillating multi-tools combine a range of features that mean they're well-suited **for makers working with existing objects**"

Above
It's easy to use tools and accessories from different manufacturers, provided they use the same attachment specification

Build a social media follower counter

A great way to keep track of followers and watch your brand grow!

Paul Freeman-Powell

@paulfp

Paul is a technology enthusiast, geek, video producer, drummer, cyclist, husband, father to three children, including twins, and host of the award-winning Switched On Network YouTube channel. You can find his website at **switchedonnetwork.com**

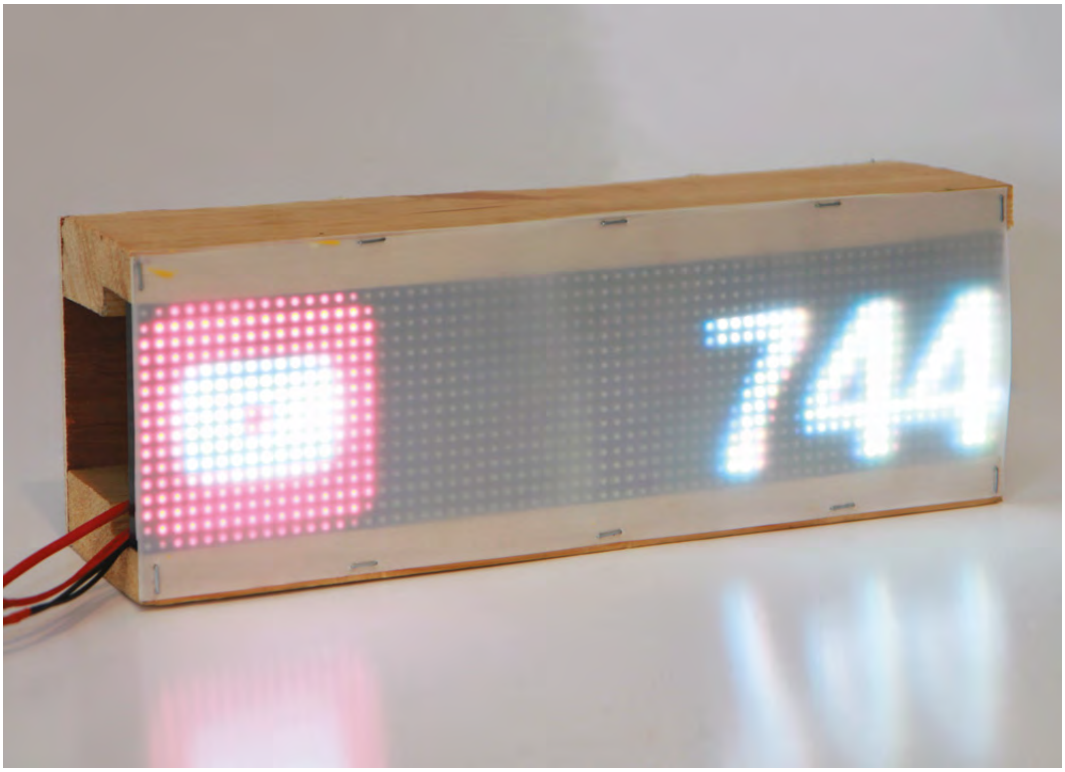
If you run a local business like an electronics shop or a café, or if you just want to grow your online following and influence, this project is a fun way to help you keep track of your progress. A counter could also help contribute to growing your following, if you hang it on the wall and actively ask your customers to like/follow you to see the numbers go up!

You've probably seen those social media follower counters which feature mechanical split-flap displays. In this project we'll build a counter powered by RGB LEDs which scrolls through four social profiles, using APIs to pull the number of followers for each account. I'm using YouTube, Twitter, Facebook, and Instagram; you can, of course, tailor the project to your needs.

This project involves a bit of electronics, a bit of software coding, and a bit of woodwork, as well as some fairly advanced display work as we transfer a small portion of the Raspberry Pi's HDMI output onto the LED matrices.

LET'S GET SOCIAL

First, you need to get your Raspberry Pi all set up and talking to the social networks that you're going



Right The finished build with all components connected, working and installed in the frame ready for hanging on the wall to display. Usually, it's advisable to install Raspbian without any graphical user interface (GUI) for most electronics projects, but in this case you'll be actively using that GUI, so make sure you start with a fresh and up-to-date installation of full-fat Raspbian.

You start by turning your humble little Raspberry Pi into your very own mini web server, which will gather your credentials, talk to the social networks, and display the follower counts. To do this, you need to install a LAMP (Linux, Apache, MySQL, PHP) stack. Start by installing the Apache web server by opening a Terminal and typing:

```
sudo apt-get install apache2 -y
```

Then, open the web browser on your Pi and type **http://localhost** – you will see a default page telling you that Apache is working. The page on our little 'website' will use code written in the PHP language, so install that by returning to your Terminal and typing:

```
sudo apt-get install php -y
```

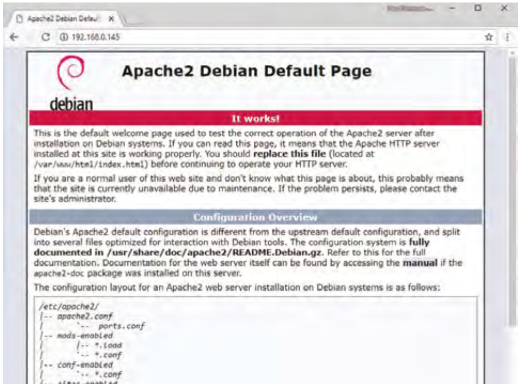
Once that's complete, restart Apache:

```
sudo service apache2 restart
```

Next, you'll install the database to store your credentials, settings, and the handles of the social accounts to track. This is done with the following command in your Terminal:

```
sudo apt-get install mysql-server php-mysql -y
```

To set a root password for your database, type the following command and follow the on-screen instructions:



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

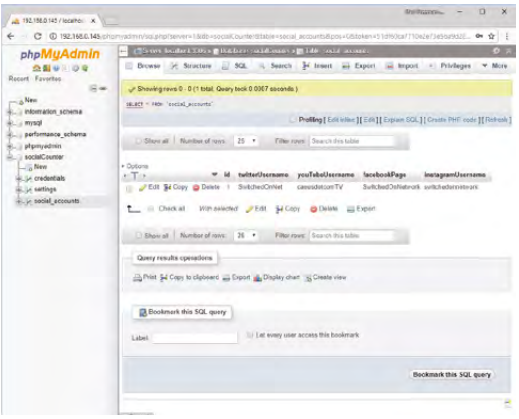
Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
```

Above
With Apache installed, the Raspberry Pi becomes a web server capable of displaying hosting webpages and talking to other services (in this case, social networks)



✓ Showing rows 0 - 0 (1 total, Query took 0.0007 seconds.)

```
SELECT * FROM `social_accounts`
```

id	twitterUsername	youTubeUsername	facebookPage	instagramUsername
1	SwitchedOnNet	caseydotcomTV	SwitchedOnNetwork	switchedonnetwork

Above
phpMyAdmin gives you an easy web interface to allow you to access and edit the device's settings – for example, speed and direction of scrolling, API credentials, and the social network accounts to monitor

```
sudo mysql_secure_installation
```

Restart Apache again. Then, for easier management of the database, I recommend installing phpMyAdmin:

```
sudo apt-get install phpMyAdmin -y
```

At this point, it's a good idea to connect your Pi to a WiFi network, unless you're going to be running a network cable to it. Either way, it's useful to have SSH enabled and to know its IP address so we can access it remotely. Type the following to access Pi settings and enable SSH:

```
sudo raspi-config
```

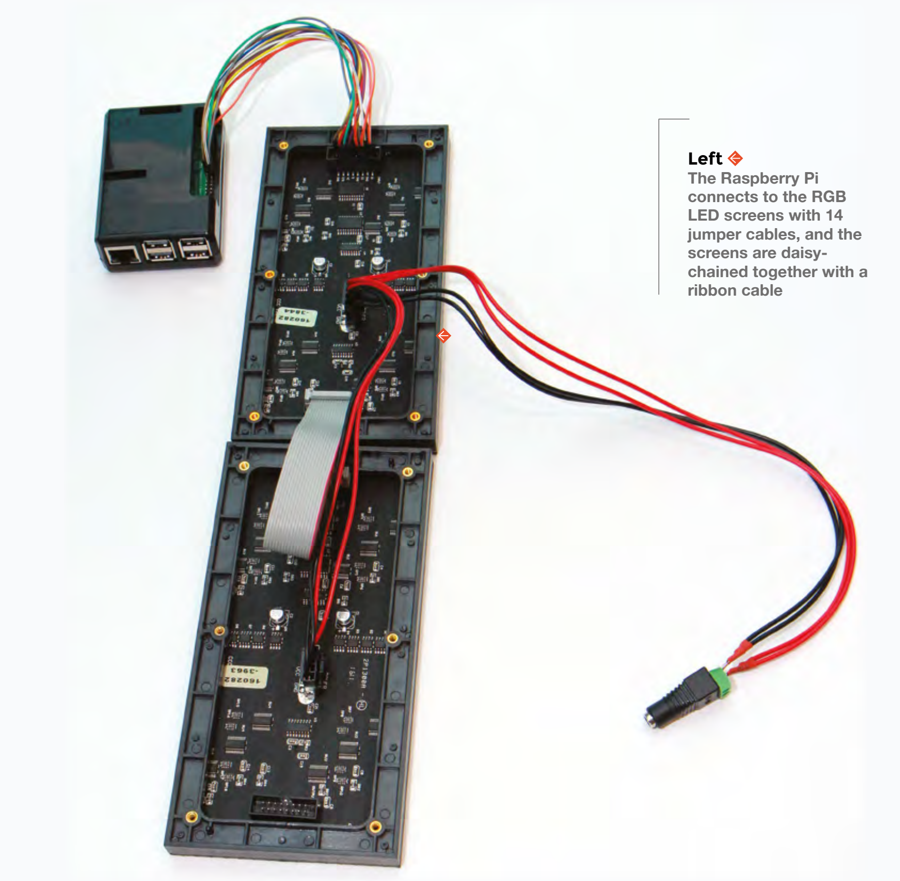
IMPORTANT

Depending on what versions of Raspbian/MySQL you install, MySQL may use the UNIX auth_socket plug-in for access, which prevents you from logging in as root via phpMyAdmin. If this is the case, you can remedy like this:

```
$ sudo mysql -u root -p

[mysql] use mysql;
[mysql] update user set plugin='' where
 User='root';
[mysql] flush privileges;
[mysql] \q
```

YOU'LL NEED

- **Raspberry Pi**
- **2 × RGB LED matrix displays** (I used 32×16 panels – available from Adafruit)
- **14 × DuPont jumper cables** (female/female)
- **1 × 16-pin ribbon cable** (female/female)
- **5 V power supply for the panels** (depending on the size of your display, allow ~3.5 A per 32×32 panel)
- **Female DC power jack**
- **Assorted bits of wood** to build your frame
- **Reflective diffuser** from an old broken LED TV (or any other translucent sheet)
- **Staple gun** and general woodworking tools



Left
The Raspberry Pi connects to the RGB LED screens with 14 jumper cables, and the screens are daisy-chained together with a ribbon cable

> At this point it's a good idea to connect your Pi to a WiFi network, **unless you're going to be running a network cable to it**

To determine your Pi's IP address (which will likely be something like 192.168.0.xxx), type either of the following two commands:

```
ifconfig # this gives you lots of extra info
hostname -I # this gives you less info, but all
  we need in this case
```

Now that SSH is enabled and you know the LAN IP address of the Pi, you can use PuTTY to connect to it from another computer for the rest of your work. The keyboard, mouse, and monitor can now be unplugged from the Raspberry Pi.

QUICK TIP

At the time of writing, Twitter and Instagram don't require credentials. However, any one of these endpoints may change or stop working altogether in the future, so be prepared to update if required.

SOCIAL MEDIA MONITOR

To set up the database, type **http://XXX/phpmyadmin** (where XXX is your Pi's IP address) and login as root with the password you set previously. Head to the User Accounts section and create a new user called socialCounter.

You can now download the first bit of code for this project by running this in your shell window:

```
cd /var/www/html
sudo apt-get update
sudo apt-get install git -y
sudo git clone https://github.com/paulfp/social-
  media-counter.git
```

Next, open up the **db.php** script and edit it to include the password you set when creating the socialCounter user:

```
cd ./social-media-counter
sudo nano db.php
```

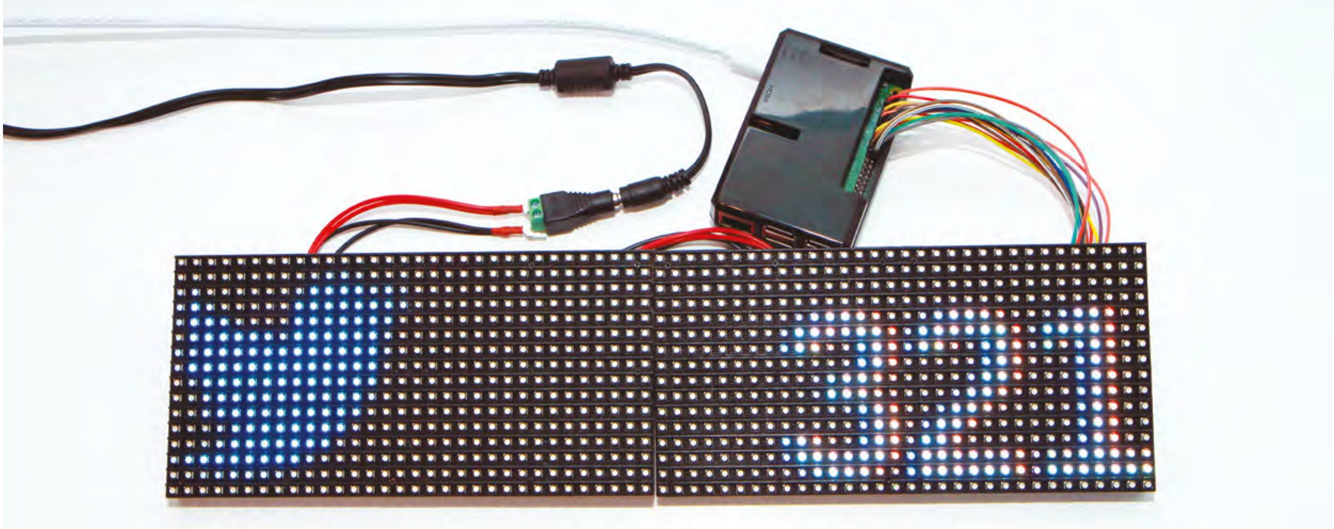
The database, including tables and settings, is contained in the **socialCounter.sql** file; this can be imported either via the shell or via phpMyAdmin, then open up the credentials table. To retrieve the subscriber count, YouTube requires a Google API key, so go to **console.cloud.google.com** and create a new Project (call it anything you like). From the left-hand menu, select 'APIs & Services', followed by 'Library' and search for the YouTube Data API and enable it. Then go to the 'Credentials' tab and create an API key which you can then paste into the 'googleApiKey' database field.

Facebook requires you to create an app at **https://developers.facebook.com**, after which you can paste the details into the facebookAppId and facebookSecret fields. Unfortunately, due to recent scandals surrounding clandestine misuse of personal data on Facebook, you'll need to submit your app for review and approval before it will work.

The 'social_accounts' table is where you enter the user names for the social networks you want to monitor, so replace those with your own and then open a new tab and navigate to **http://XXX/social-media-counter**. You should now see a black page with a tiny carousel showing the social media icons plus follower counts next to each one. The reason it's so small is because it's a 64×16 pixel portion of the screen which we'll be displaying on our 64×16 LED boards.

GPIO PINS TO LED DISPLAY

Now that you have your social network follower counts being grabbed and displayed, it's time to get that to display on our screens. The first step is to wire them up with the DuPont jumper cables from the Raspberry Pi's GPIO pins to the connection on the first board. This is quite fiddly, but there's an excellent guide and diagram on GitHub within Henner Zeller's library that we'll be



Above When placed side by side, the two boards comprise a 64×16 grid of LEDs, with each LED representing one pixel from the top-left of the Raspberry Pi's display

using later, so head to **hsmag.cc/PLyRcK** and refer to **wiring.md**.

The second screen is daisy-chained to the first one with the ribbon cable, and the power connector which comes with the screens will plug into both panels. Once you're done, your setup should look just like the picture on this page.

To display the Pi's HDMI output on the LED screens, install Adafruit's rpi-fb-matrix library (which in turn uses Henner Zeller's library to address the panels) by typing the following commands:

```
sudo apt-get install -y build-essential
 libconfig++-dev
cd ~
git clone --recursive https://github.com/
 adafruit/rpi-fb-matrix.git
cd rpi-fb-matrix
```

Next, you must define your wiring as regular. Type the following to edit the Makefile:

```
nano Makefile
```

Look for the `HARDWARE_DESC=` property and ensure the line looks like this: `export HARDWARE_DESC=regular` before saving and exiting nano. You're now ready to compile the code, so type this and then sit back and watch the output:

```
make clean all
```

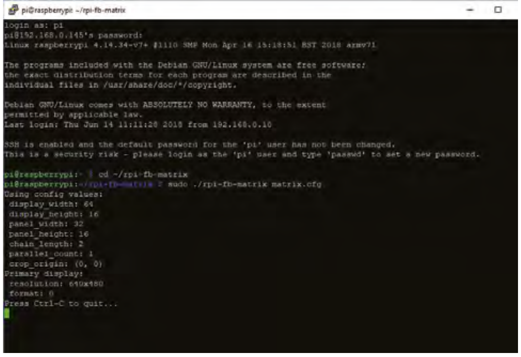
Once that's done, there are a few more settings to change in the matrix configuration file, so open that up:

```
nano matrix.cfg
```

You need to make several changes in here, depending on your setup:

- Change `display_width` to 64 and `display_height` to 16
- Set `panel_width` to 32 and `panel_height` to 16
- Set `chain_length` to 2
- Set `parallel_count` to 1 →

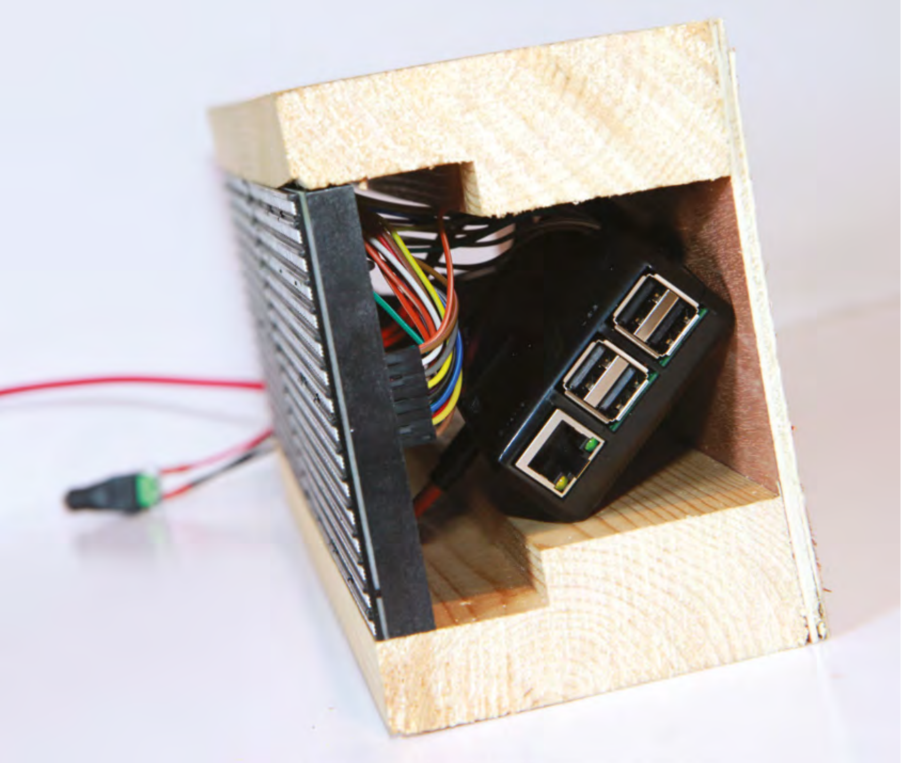
Below Shelling into the Raspberry Pi from another computer via PuTTY enables commands to be run without needing to have a keyboard, monitor, and mouse connected



pi@raspberrypi: ~/rpi-fb-matrix

```
login as: pi
pi@192.168.0.145's password:
Linux raspberrypi 4.14.34-v7+ #1110 SMP Mon Apr 16 15:18:51 BST 2018 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: Thu Jun 14 11:11:28 2018 from 192.168.0.10

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $ cd ~/rpi-fb-matrix
pi@raspberrypi:~/rpi-fb-matrix $ sudo ./rpi-fb-matrix matrix.cfg
Using config values:
 display_width: 64
 display_height: 16
 panel_width: 32
 panel_height: 16
 chain_length: 2
 parallel_count: 1
 crop_origin: (0, 0)
Primary display:
 resolution: 640x480
 format: 0
Press Ctrl-C to quit...
```

Below It's amazing what you can find in your firewood pile! I rescued this unused door frame, which was perfect to hold the two screens tightly in place



Above ◈
It ain't pretty... but it's out of sight. The Raspberry Pi plus the power supply for the screens fit nice and neatly behind the screens. I left each end open to allow airflow

The panel array should look like this:

```
panels = (
  ( { order = 1; rotate =   0; }, { order = 0;
rotate =   0; } )
)
```

Uncomment the **crop_origin = (0, 0)** line to tell the tool that we don't want to squish the entire display onto our screens, just an equivalent portion starting right in the top-left of the display. Press **CTRL+X**, then **Y**, then **ENTER** to save and exit.

Finally, before you can test the output, there are some other important settings you need to change first, so open up the Raspberry Pi's boot configuration as follows:

```
sudo nano /boot/config.txt
```

First, disable the on-board sound (as it uses hardware that the screens rely on) by looking for the line which says **dtparam=audio=on** and changing it to **off**. Also, uncomment the line which says **hdmi_force_hotplug=1**, to ensure that an HDMI signal is still generated even with no HDMI monitor plugged in. Save and then reboot your Raspberry Pi.

Now run the program using the config you just set:

```
cd ~/rpi-fb-matrix
sudo ./rpi-fb-matrix matrix.cfg
```

You should now see the top 64×16 pixels of your Pi's display represented on your RGB LED panels! This probably consists of the Raspberry Pi icon and the rest of the top portion of the display bar.

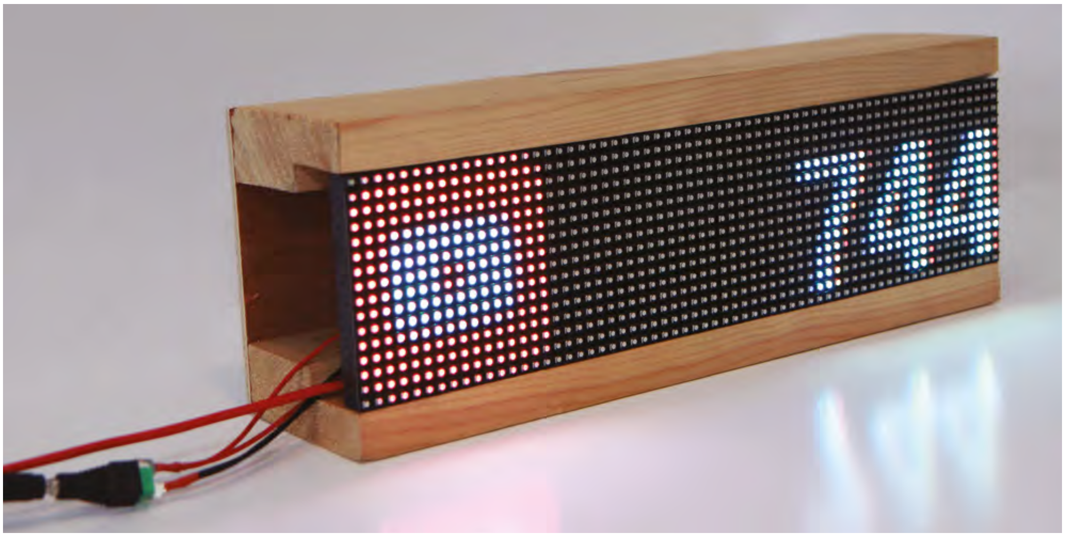
NO SCREENSAVER!

At this point it's important to ensure that there's no screensaver or screen blanking enabled on the Pi, as you want this to display all the time. To disable screen blanking, first install the xscreensaver tool:

```
sudo apt-get install xscreensaver
```

That will add a screensaver option to the Pi's GUI menus, allowing you to disable it completely.

Finally, you need to tell the Raspberry Pi to do two things each time it loads:



Right ◈
Voila – the perfect home furnishing for social media obsessives who can't rest without one eye on their stats

QUICK TIP

In the 'settings' table, you can adjust options like the direction of the carousel scroll, how long each network's count is displayed for, how quickly it scrolls, and how often the values are refreshed.

- Run the rpi-fb-matrix program (like we did manually just now)
- Open the web browser in fullscreen ('kiosk' mode), pointed to the Social Counter webpage

To do so, edit the Pi's **autostart** configuration file:

```
sudo nano ~/.config/lxsession/LXDE-pi/autostart
```

Insert the following two lines at the end:

```
@sudo /home/pi/rpi-fb-matrix/rpi-fb-matrix /home/
 pi/rpi-fb-matrix/matrix.cfg\
@chromium-browser --kiosk http://localhost/
 social-media-counter
```

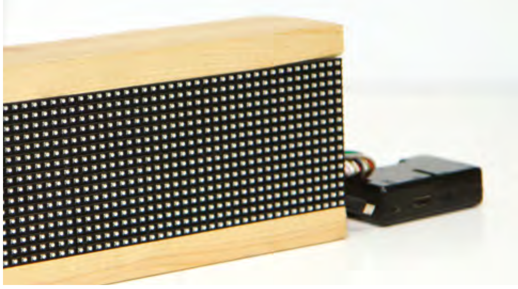
ET VOILÀ!

Disconnect any keyboard, monitor, or mouse from the Pi and reboot it one more time. Once it's started up again, you should have a fully working display cycling through each enabled social network, showing up-to-date follower counts for each.

It's now time to make a surround to hold all the components together and allow you to wall-mount your display. The styling you go for is up to you – you could even go all out and design and 3D-print a custom package.

For my surround, I went for the more rustic and homemade look, and used some spare bits of wood from an internal door frame lining. This worked really well due to the pre-cut recess. With a plywood back, you can screw everything together so that the wood holds the screens tightly enough so as not to require any extra fitting or gluing, making for easier future maintenance. To improve the look and readability of the display (as well as soften the light and reduce the brightness), you can use a reflective diffuser from an old broken LED TV if you can lay your hands on one from eBay or a TV repair shop, or just any other bit of translucent material. I found that two layers stapled on worked and looked great. Add some hooks to the back and – Bob's your uncle – a finished, wall-mounted display!

Phew – that was quite an advanced build, but you now have a sophisticated display which can be used for any number of things and should delight your customers whilst helping to build your social following as well. Don't forget to tweet us a picture of yours! □



Above The finished product, in pride of place on the wall of our office. Now I just need some more subscribers...!

Left The wooden beams are screwed to the back a millimetre or so too small, to allow them to grip the screens nice and tightly

EXPANSION: EASY REMOTE MANAGEMENT

The display settings, API credentials, and social network handles are stored in a database rather than hard-coded. This means that you could write a basic PHP administration page to let you update these settings easily from another computer on your home network without having to log in to the database via phpMyAdmin each time.

meet
build
learn
play
Ultimaker
Ultimaker

Festival of Maintenance

Celebrating maintenance in all its forms

One day festival

University of London Union
Malet Street **London WC1E 7HY**

22 September 2018

Tickets: ti.to/festival-of-maintenance

Teensy drum machine

Are you ready to rock?

Matt Bradshaw

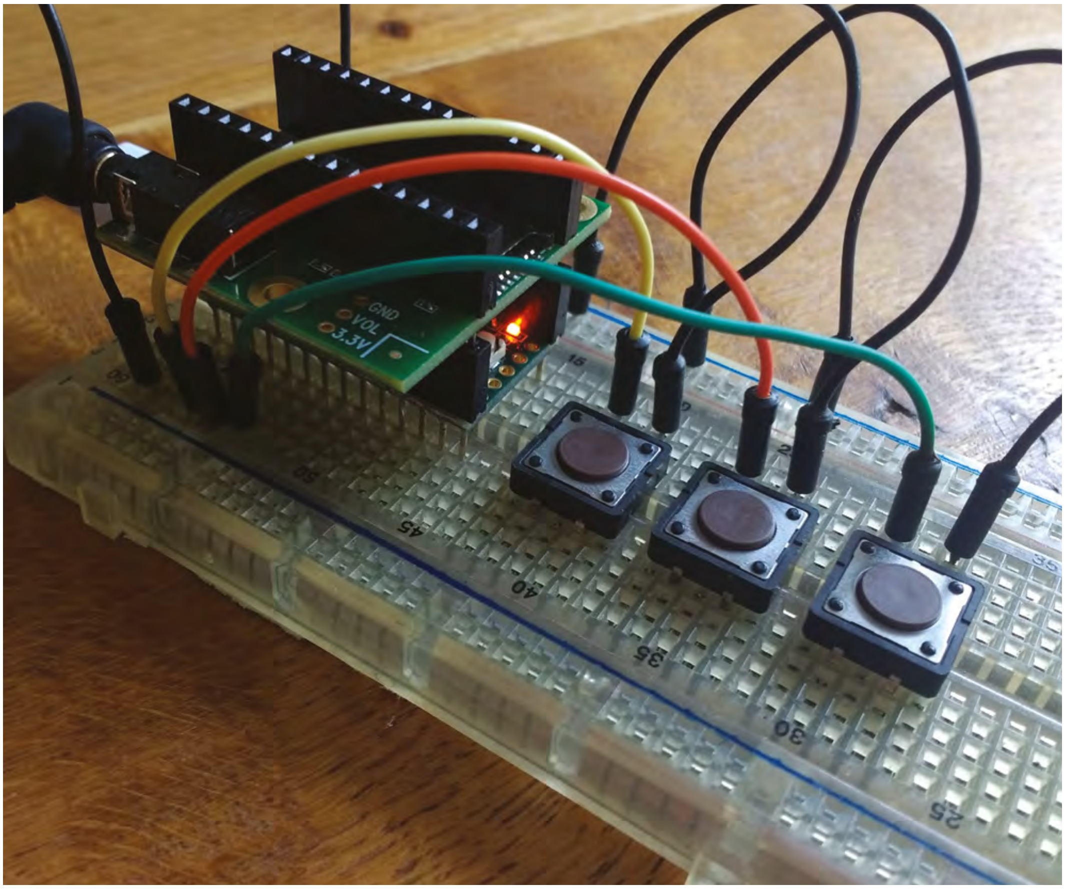
mattbradshawdesign.com

Matt Bradshaw is a programmer, maker, and musician from Oxford. He likes to build instruments to play with his band, Robot Swans. You can find more of his projects at **mattbradshawdesign.com**

Making your own electronic instruments is a fascinating, rewarding, and surprisingly feasible pursuit. The Teensy platform (an extension of the Arduino platform) is a perfect tool for doing this, offering CD-quality audio and a variety of sound sources and effects. This tutorial shows you how to build a playable, standalone drum machine using nothing but a Teensy, an audio adapter, and a few buttons. The finished build will let you play a kick drum, a snare drum, and a closed hi-hat cymbal with your fingers.

The first thing you need to do is install the software to program your board. The Teensy is programmed via the standard Arduino software, but requires an add-on called Teensyduino. This can be downloaded from **hsmag.cc/TWRtkj**, where there are also instructions to help you through the installation process.

Once you have installed Teensyduino, it's a good idea to check that everything is working correctly. Open the Arduino software and go to File > Examples > Basics > Blink. Plug the Teensy into your computer and go to Tools > Board to select the exact version of Teensy you are using. Send the sketch to the board using Sketch > Upload, and after a while you should



Right
The finished drum machine

QUICK TIP

Don't solder the smaller 5-pin header that comes with the stackable header kit – it will prevent the Teensy from fitting into a breadboard.

see your Teensy's LED start to flash on and off. Try adjusting the delay times in the 'loop' section of the sketch and re-uploading the program to double-check that you are communicating with the Teensy.

SOME ASSEMBLY REQUIRED

Teensy boards tend to arrive without headers, meaning you have to solder them yourself before you can connect anything. This can be a bit off-putting if you've been happily breadboarding with Arduinos and Raspberry Pis until now, but it's an easy enough task with the right equipment and a bit of patience. The headers should be positioned with the plastic housing on top of the board and the metal pins poking through below, with the soldering done on the underside of each board.

Once you have soldered the headers to both the Teensy and the audio board, you can attach the audio board on top of the Teensy, making sure that the pin numbers line up. Be careful to apply gentle, even pressure when pushing the audio board's pins into the Teensy – it is easy to fatigue the pins when doing this.

With the audio board attached to the Teensy, you can check that it is working by uploading another example sketch, found in File > Examples > Audio > Synthesis > PlaySynthMusic. When the sketch is loaded, plug in some headphones and you should hear a piece of music.

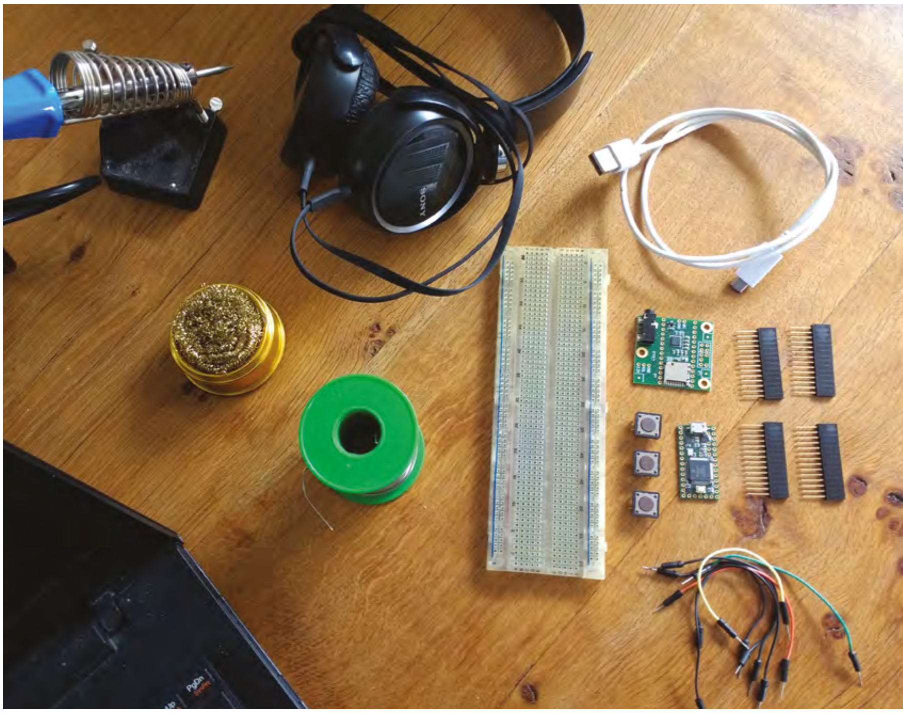
MAKING CONNECTIONS

The first thing you need to do when creating a Teensy audio project is to define what audio components you are using and how they are connected together. Happily, there is a great online tool for helping you do this – it can be found at **hsmag.cc/xiZUDr** and we found it really useful.

Sound sources, effects, and other audio objects can be found in the left-hand menu, and dragged into the main area. Audio objects can then be connected by clicking and dragging to create virtual cables between them.

AUDIO ADAPTER

The audio adapter board is a shield which adds high-quality audio output to the Teensy and features a convenient headphone socket, plus the option to solder your own audio input. It also contains an SD card slot and space to solder an optional memory chip for additional effects and sample playback. Any of the version 3 Teensy boards will work with the audio adapter, but for more ambitious projects, the 3.6 is worth the extra money for its memory and extra pins.



Above
Everything you'll need for the project

For the drum machine, you will need to recreate the system shown in **Figure 1** (overleaf). There are three **playMem** objects, which will play our kick, snare, and hi-hat sounds. These three sound sources are connected to a **mixer** object, where their signals are combined into a single output. The mixer's output is then connected to an **i2s** object, which represents the headphone socket on your audio board.

Finally, add an **sgtl5000** object, which represents the main chip on your audio board. This object does not need to be connected to anything, but the sketch will not work unless you include it.

Having laid out the design of your audio system, click the 'export' button to generate Arduino code. Create a new sketch in the Arduino software (File > New) and paste the generated code right at the top of the file, before the **setup** function. The audio system you designed is now defined in your sketch.

MAKE SOME NOISE(S)

This drum machine requires an audio sample of each type of drum you want to use. A good starting point is to visit **freesound.org**, a website containing thousands of audio samples. You will need to log in to download files, but it's free to set up an account. Search for a kick drum (the '808' kick is the classic if you're overwhelmed by choice), a snare drum →

YOU'LL NEED

- **Teensy 3**
  hsmag.cc/FoXxGn
- **Teensy audio adapter board**
  hsmag.cc/jukWSM
- **4 × 14-pin stackable male/ female headers (two kits)**
  hsmag.cc/TBUhkL
- **Breadboard**
  hsmag.cc/AwjCXt
- **Jumper wires**
  hsmag.cc/fdUnYB
- **3 × large tactile buttons**
  hsmag.cc/lCxFun
- **USB micro cable**
  hsmag.cc/SfFxXv
- **Soldering equipment**
- **Headphones**
- **Computer**

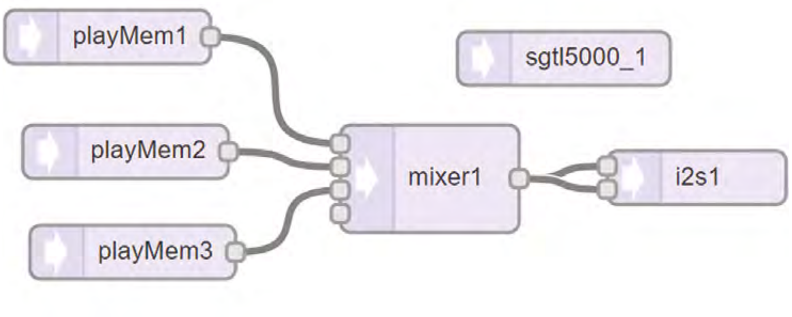


Figure 1 Recreate this flowchart in the online tool and export it as Teensy code

QUICK TIP

If you get stuck with any Teensy project, the official forum at **forum.pjrc.com** is a great place to get help.

and a hi-hat, and download each one. If possible, find sounds that are already in the WAV format, otherwise you will have to convert them to WAV files using an audio editor.

Next, you need to convert your audio file into Arduino code that can be used in your sketch. There is a command-line program called 'wav2sketch' to do this, downloadable from **hsmag.cc/RsSWAw** for Mac or **hsmag.cc/cCstSh** for Windows. Make a folder containing both the downloaded program and the WAV files that you want to convert, then open a terminal window. Navigate to the directory you made (using the **cd** command) then run the program, as shown below.

```
cd /path/to/your/folder/
./wav2sketch
```

Assuming there are no errors, six files should have been created in the folder: two for each audio file. Copy these six files into the same folder as your Teensy sketch.

Now it's time to get writing some quality code. We'll go through the process of getting just the kick drum working first – adding the snare and hi-hat should be easy enough for anyone after that. Starting after the generated code that you pasted from the online tool, add the all-important kick drum audio file that you created earlier, and be sure to use an **#include** statement.

Next, you need to define a couple of variables that will be used later: a Boolean (true/false) value representing the current status of the kick drum button (whether it is pressed or not), and an unsigned long value representing the last time the kick button status changed.

```
#include "AudioSampleKick.h"

boolean previousKickButtonStatus = false;
unsigned long previousKickTime = 0;
```

Writing a Teensy sketch is very similar to writing an Arduino sketch, so you still have two main functions: **setup()**, which runs once at the start of the program, and **loop()**, which then runs repeatedly. In the **setup** function, you need to do a few things: assign the audio library some memory, enable the audio board, set the audio board's volume, and set the kick drum button's pin as an input.

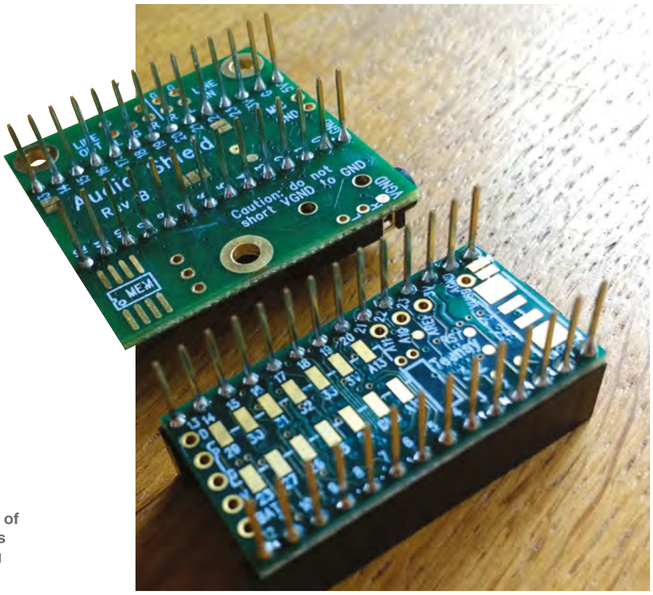
```
void setup() {
  AudioMemory(10);
  sgtl5000_1.enable();
  sgtl5000_1.volume(0.5);
  pinMode(2, INPUT_PULLUP);
}
```

The **loop()** function should simply keep checking whether the kick button is pressed, and trigger the

EDITING AUDIO

When preparing audio files for a sample-based instrument like this, it's useful to have a good audio editor, such as Audacity. It's worth spending some time cleaning up your audio files if you're going to be hearing them repeatedly. Make sure to remove any silence from the start of a file – this will improve latency (the time between triggering a sound and hearing it) and will also reduce your sketch size.

It can also be a good idea to add certain effects in an audio editor rather than using the Teensy's audio effects. This is because some effects, particularly time-based ones such as reverb and delay, are either processor-intensive or memory-hungry. If you find yourself struggling to implement certain effects on the Teensy itself, consider whether they could be baked into the original sample using an audio editor.



Right The underside of the two boards after soldering sample if the button changes from off to on, but there's a catch: when a button does change state, it tends not to do so perfectly, but rather flickers between on and off for a few fractions of a second.

You should be able to overcome this problem by not checking the button for a few milliseconds after a change has been detected – to the initiated, this is known as 'debouncing'.

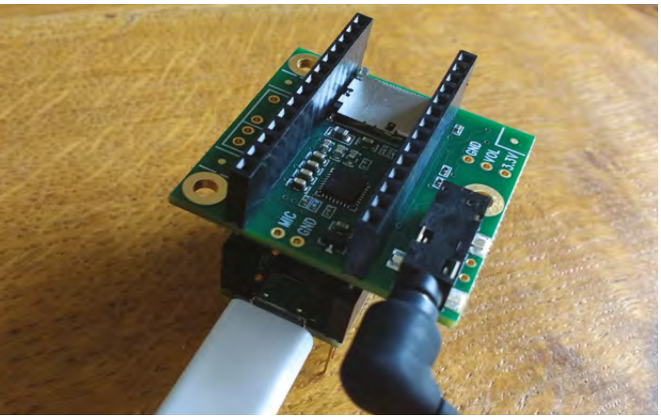
```
void loop() {
  boolean newKickButtonStatus = !digitalRead(2);
  if(newKickButtonStatus !=
previousKickButtonStatus && millis() >
previousKickTime + 30) {
    if(newKickButtonStatus) playMem1.
play(AudioSampleKick);
    previousKickButtonStatus =
newKickButtonStatus;
    previousKickTime = millis();
  }
}
```

BUTTON UP

With the code for the kick drum in place, you can start constructing your circuit. Carefully push the Teensy (with its audio board stacked on top) into the breadboard, then add three tactile buttons. This type of button has two pairs of conductive legs which are only connected when the button is pushed; connect one side of each button to ground, and the other to a digital pin. See **Figure 2** for more details.

Once you've constructed the circuit and uploaded the sketch, plug in a pair of headphones. One of the buttons should now trigger the kick drum sound when pushed. If not, perhaps check that your wiring matches the diagram and, of course, that you haven't made any obvious mistakes in your code. You can also check that the sample playback is working okay by commenting out all the code in `loop()` and triggering a kick sample directly at the end of the `setup()` function. If you are still stuck, try loading some of the Teensy audio example sketches to see if they work.

Below
You can test the audio output before beginning the drum machine code



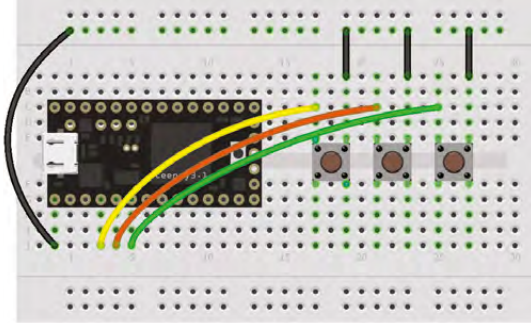


Figure 2
Circuit diagram (audio board omitted for clarity)

> **If your kick drum is triggering correctly, congratulations!** All you need to do now is add similar code for the other two sounds

If your kick drum is triggering correctly, congratulations! All you need to do now is add similar code for the other two sounds, remembering to change the pin numbers and **playMem** object numbers, and your drum machine will be complete...

Well, the basics will be, anyway. Now it's up to you upgrade this project and add more features. Happy, rhythmic hacking! □

NEXT STEPS

This project could be extended in a huge number of ways. You could gear it towards live performance, replacing the tactile buttons with piezo triggers mounted under large rubber pads, meaning you could play your instrument with drumsticks. Alternatively you could make a user-programmable machine which records and stores rhythms to accompany you while you play guitar or synthesizer.

Another logical next step might be to add MIDI connectivity, a standard way for electronic musical instruments to communicate with each other. Adding a MIDI input to this project would allow you to control it from a set of MIDI drum pads or from a computer via a software package like Cubase, Ableton Live, or Pro Tools.

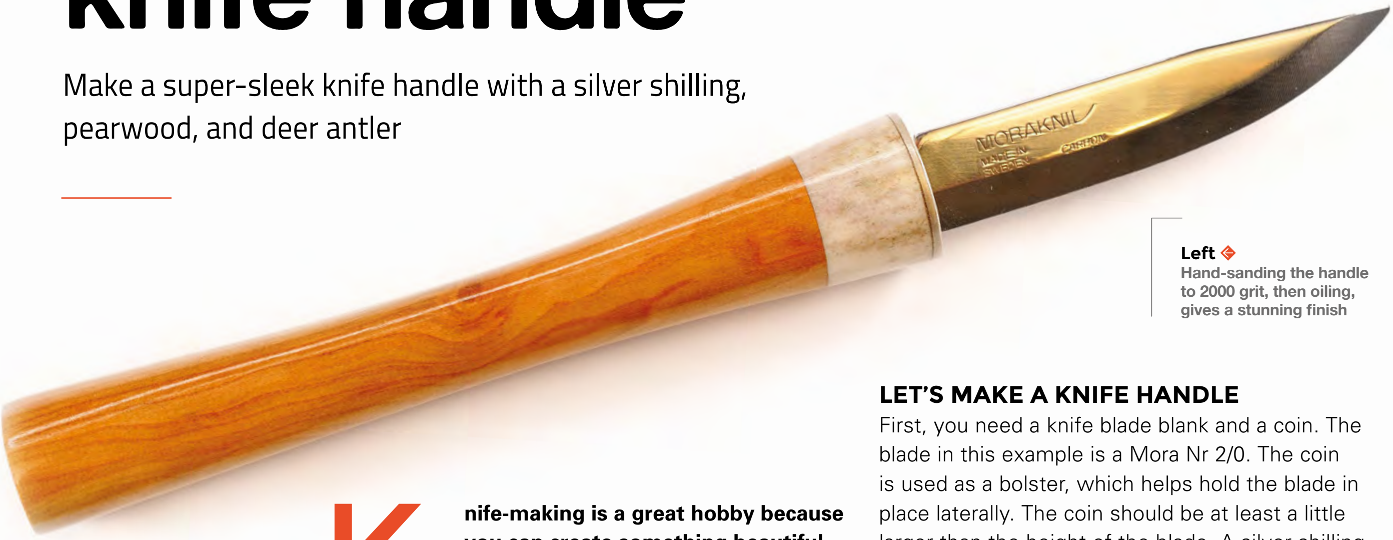
Whatever you end up doing, moving from a breadboard to a more permanent soldered circuit is a good idea once you're happy with your design.

QUICK TIP

Setting a pin as **INPUT_PULLUP** means that the pin will default to reading true/high when not connected to anything. This is useful because it means you don't need a resistor when adding a button to your circuit.

Beautiful, unique knife handle

Make a super-sleek knife handle with a silver shilling, pearwood, and deer antler



Left
Hand-sanding the handle to 2000 grit, then oiling, gives a stunning finish

Alex Eames

RasPiTV

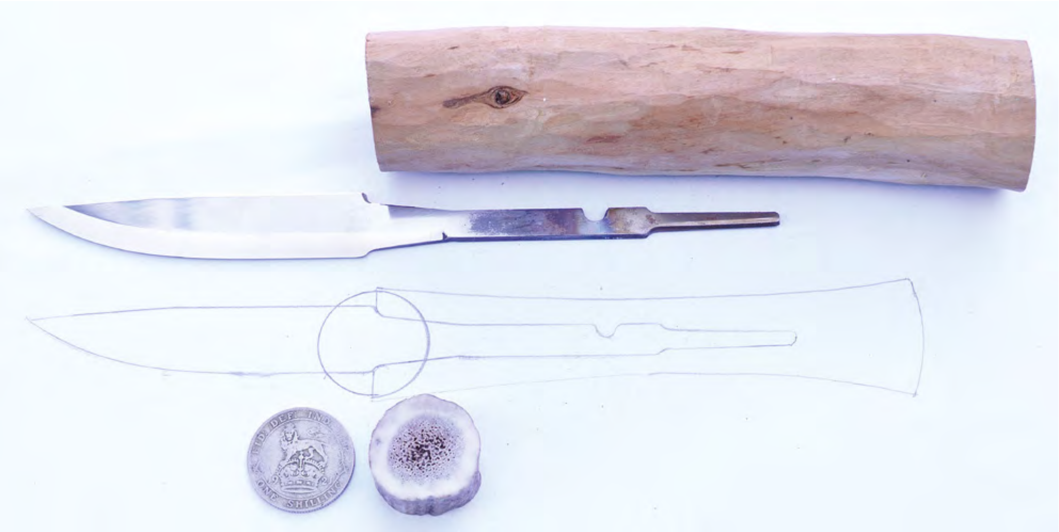
Alex Eames loves making things and regularly blogs/vlogs at RasPi.TV He makes a living designing and selling RasP.iO products.

Knife-making is a great hobby because you can create something beautiful and useful with six to ten hours of effort, over two to three days, for under £20 in materials. It's a chance to get creative, experiment, and make something unique and personal. The technique works with any wood, but hardwood is preferred for knife handles as it's less prone to denting. There's also something special about using wood harvested from your own garden. The example project uses wood from pear tree prunings (seasoned for two years).

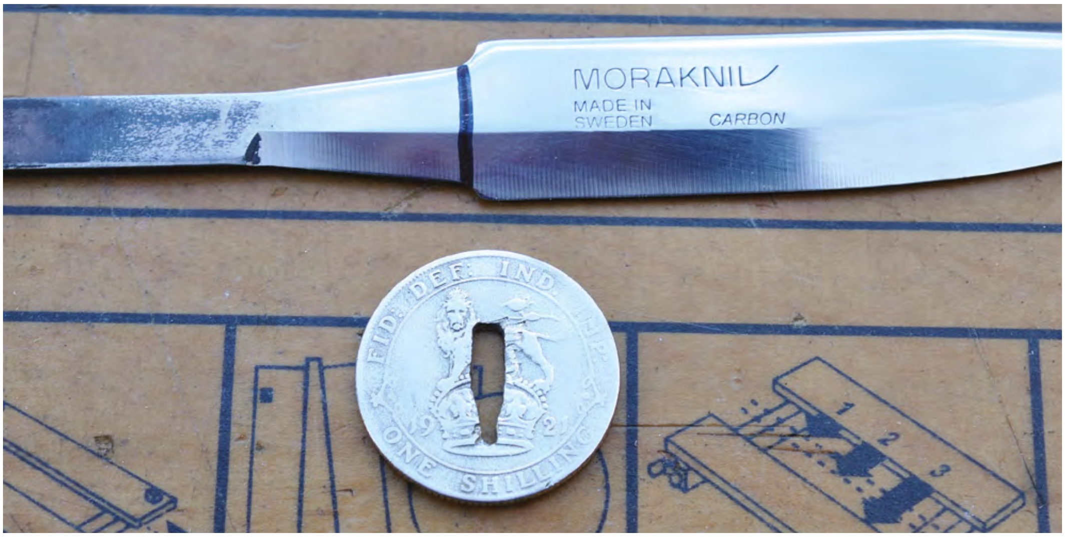
LET'S MAKE A KNIFE HANDLE

First, you need a knife blade blank and a coin. The blade in this example is a Mora Nr 2/0. The coin is used as a bolster, which helps hold the blade in place laterally. The coin should be at least a little larger than the height of the blade. A silver shilling (pre-1947 actually has some silver in) is perfect for this blade size. Silver is also relatively soft and easy to work with.

Take your knife blade, place it on a sheet of paper, and draw all around the perimeter. Placing your coin bolster on the plan also helps you to size one end of the handle accurately. Draw the outline of the handle shape that you'd like to make. You can get inspiration from the internet, your kitchen knife drawer, or make something up. Or you can simply



Right
Design the handle to fit the parts you want to use by laying it all out on paper



Left
A 1921 shilling is 50% silver, 50% copper, so is quite soft and easy to work with

copy the design from **github.com/raspitv/knife** to get started.

Once you've decided on the shape, choose a piece of seasoned hardwood larger than your final design by at least 5 mm in every direction. This gives you some margin for error. It's a lot easier to get the alignment right if you work from a planed square edge block of wood, but this example uses a piece of pear tree branch.

Choose a section of your antler that has a slightly larger diameter than the coin. When sanding to shape later on, the outer 'brown' part is removed, exposing the 'ivory-like' section underneath. So the antler cross-section needs to be chosen to fit the coin. Small deer antlers can easily be bought in pet shops, where they are sold as dog chews. You'll need about 1 cm length of antler, cut with perpendicular ends. Antlers can be cut with wood saws. The ends can be sanded (wear a mask).

BOLSTERS AND TANGS

The thin part of the blade embedded in the handle is called the 'tang'. You need to make a hole in the coin bolster so that the tang fits through until the coin is almost touching the start of the blade 'proper'. Ideally, the bolster should be a tight fit on the tang. Measure the blade cross-section at the top (wider end) of the tang, and draw the approximate cut-out on your coin with a felt-tipped pen. Use a Dremel-like multi-tool, with small cutting wheel and/or engraving bit, to remove material from the coin. It's best to iterate until you have achieved the desired fit. When it fits well, remove the coin and give it a good polish with Brasso, or some other metal polish. It's much easier to do it now than when it's on the knife.

Mark the antler in the position where you want the blade's tang to pass through it. This can be a fairly 'generous' hole and doesn't need to be a tight fit at all. It should be a little higher and a little wider than the cut-out in the bolster, to give you some wiggle room for alignment in glue-up. Drill the hole with a 5 mm drill bit or use an engraver. The middle part of the antler is 'honeycombed' and easy to work with. →

Below
The ring pattern of the wood grain often shows in surprisingly beautiful ways

YOU'LL NEED

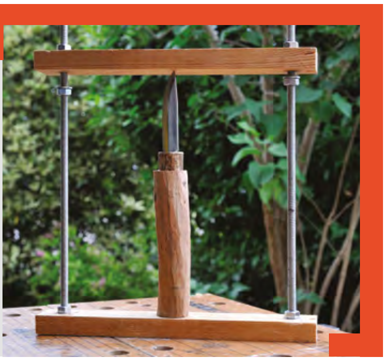
- **Piece of hardwood**
- **Knife blade, e.g. Mora Nr 2/0** hsmag.cc/kbQAjR
- **Piece of deer antler** (often sold as dog chews in pet shops)
- **Coin** (eBay – silver is ideal as it is a soft metal)

Tools

- **5 mm drill bit** (long)
- **Power sander/ detail sander, with coarse and medium abrasive paper** (e.g. 60, 120 grit)
- **Dremel, small cutting wheel, and engraving bit**
- **Wet and dry abrasives 400 to 2000 grit**
- **Varnish, polish, or oil**
- **Glue – Gorilla glue (polyurethane) or epoxy resin**
- **Band-saw, jigsaw, scroll-saw, or coping saw**

MAKING A KNIFE PRESS

A simple knife press can be made very quickly from two lengths of wood, two lengths of threaded rod, and four to six nuts. It's a lot less fiddly than using two separate clamps and pieces of wood. The threaded rod should be 30–40 cm long to allow for a variety of knife sizes. The wood should be something relatively soft, e.g. pine or spruce, to avoid damage to blade or handle. The knife blade will dig into it when clamped.



> **You need to make a central hole in the wooden handle,** deep enough for the tang, and wide enough to allow some free movement for alignment

WOOD YOU HANDLE IT?

You need to make a central hole in the wooden handle, deep enough for the tang, and wide enough to allow some free movement for alignment. A 4–5 mm drill bit (you may need an extra long one) is usually sufficient. Drill two adjacent holes and then merge them together, either using a milling action with the drill, or with small rasps or files. Be careful not to drill deeper than you need to and be sure to get the alignment right. A pillar drill can be a big help here if you have one.

If you need to, you can shorten the tang. But don't take too much off it because the longer the tang, the more glue area holding the blade firmly in the handle.

Rough up the tang with coarse abrasive paper. This is partly to clean it and partly to give the glue something to 'key' to. Then fit the bolster to the tang, slip the antler section over the tang, and trial-fit the tang in the wooden handle. Ensure it can all be made to line up straight. This is your last chance to make any adjustments to the antler section or hole in the wood. Once satisfied, it's time to glue it all together.

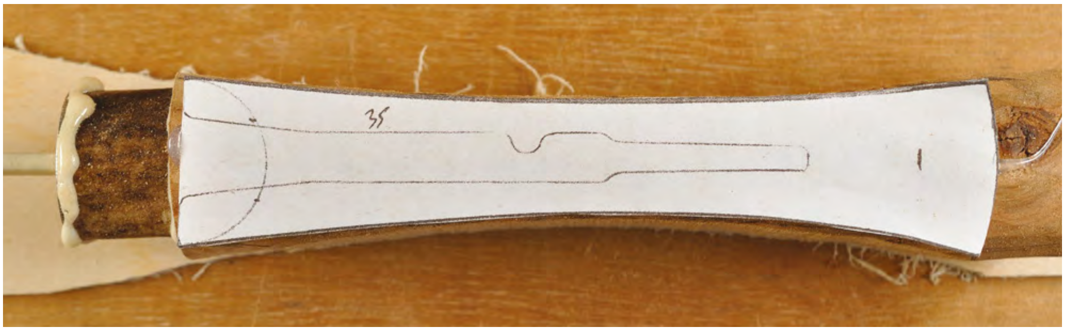
STICKING POINT

Polyurethane glue is ideal for knife making because it expands to fill gaps and is also waterproof. (You can also use epoxy.) Ensure the hole is free from excess sawdust, then squirt glue into the hole. You can use a bamboo skewer to push glue down and make sure it reaches to the bottom of the hole. It doesn't have to be completely full because the tang is also going in there and the glue expands. Ensure the bolster is correctly fitted to the tang. Spread some glue on the top side of the antler section and press it into place against the coin. Then glue the other side of the antler section and fit the whole assembly to the wooden handle. Ensure everything is aligned correctly, then press together using either two lengths of wood and two clamps or a knife press. You have about 30–60 minutes before the glue becomes unworkable.

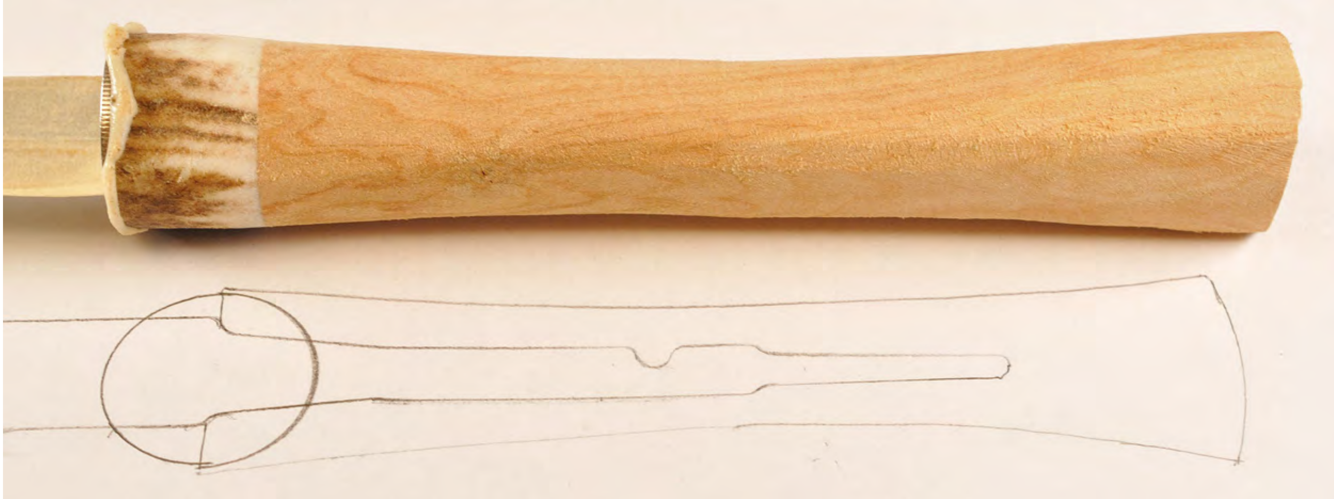
Check alignment and leave clamped for at least four to six hours, preferably overnight. Gorilla glue sets hard overnight and fully cures in 24 hours. Some epoxies may be faster.

ROUGHING IT

Now you're going to rough out the handle shape. Make a copy of the handle section of the plan and stick it to the top side of the wood with double-sided sticky tape, or two blobs of hot glue. This will act as a guide for removing unwanted wood. If you have a band-saw, it's the perfect tool for the job, but it could also be done with a jigsaw or coping saw. It can help – particularly with non-square materials – if you hot-glue the handle to a piece of thin 'sacrificial' sheet material, e.g. 3 mm MDF or ply. Aim to cut away all excess material to within 1–2 mm of the pattern edge. The more accurately you can do this, the less sanding you'll have to do later. Repeat the process for the side profile. The knife in this example is symmetrical, so has the same profile for both top and side view. Other designs don't, so use whatever is appropriate for your chosen design.



Right A scrap of sacrificial 3 mm ply and hot glue hold a 'round' wood blank in position for cutting



SHAPING UP

Now it's time to finalise the shape of the handle. You can take larger chunks off with a sharp knife or chisel, but you have a much better degree of control with sanding. You need to find the right balance between speed and accuracy, which you can only learn by 'doing'. If you take your time, you will end up with a nice result. Shaping of the handle usually takes a couple of hours and is great 'thinking time'.

Put masking tape round the blade to protect both it and yourself. Clamp the blade and use a coarse grit, (40–60) on a power sander or detail sander, to blend the handle and antler to the shape of the coin bolster. Try not to let the sander remain in one place for long. Keep it moving backwards and forwards and change the angle. Blend the whole handle into the desired shape. When satisfied, use a finer grade abrasive paper (120–240) to smooth the handle.

> **Make the handle super-smooth by hand sanding with** 'wet 'n' dry' abrasive paper used dry. Progress up through grades 400, 600, 800, 1200, 1500, 2000 and you'll end up with a wonderful surface

Then make the handle super-smooth by hand sanding with 'wet 'n' dry' abrasive paper, used dry. Progress up through grades 400, 600, 800, 1200, 1500, 2000 and you'll end up with a wonderful surface. It takes time and patience, but it's truly worth it. From band-saw to 'finally smooth' usually takes two to three hours. But it's your knife. It's totally up to you how far you take it.

I'LL GET MY COAT

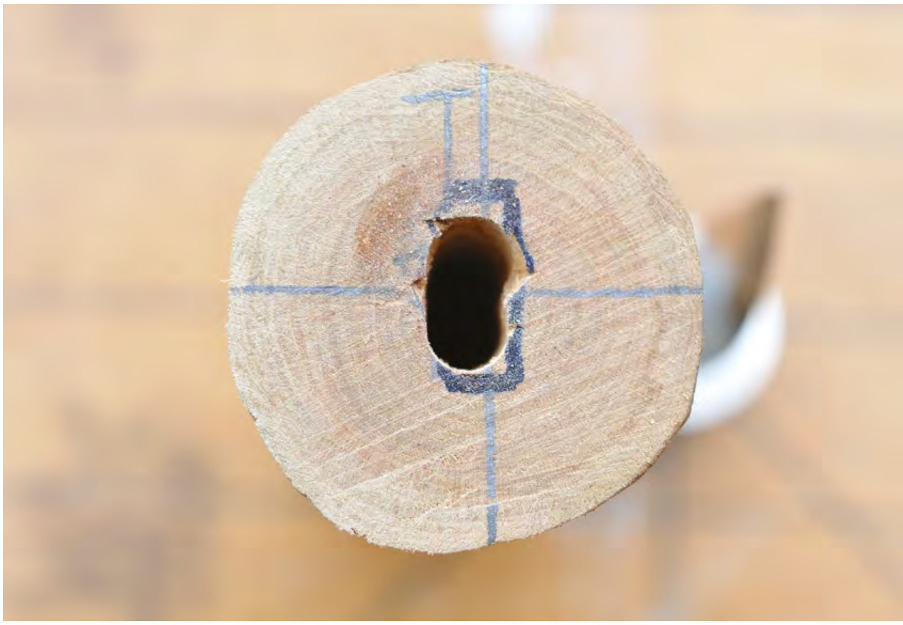
Now it's time to finish the handle with a protective coating. You can use wax polish, varnish, or one of the various oils available for wood finishing. In this example, Birchwood Casey's Tru-Oil is used. Four or five coats of this boiled linseed oil blend, with two hours between coats, brings out the natural beauty of the wood and the antler 'ivory' with a lustrous shine.

Then, all that's left to do is sharpen the knife to your own precise requirements. Most knife blades are sold 'usably sharp', but if you want to carve or shave with it, it may need some further work with oil stones, whetstones, or waterstones.

It takes a little patience and time to handle a knife this way, but you end up with something unique, personal, and beautiful. It's a great project for a weekend make. Happy knife-making. □

Above
Part-way through sanding, the antler still shows some of its exterior

Below
The handle hole should be generously sufficient for the tang and glue, leaving some flexibility in alignment



FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

DIRECT FROM SHENZHEN

Hydroponics

The quick and easy way to get a fresh salad

By Ben Everard

@ben_everard

Below There's something both pleasing and a little unnatural about plants lined up perfectly in two dimensions

DIRECT FROM **SHENZHEN**

Hydroponics – the process of growing plants by applying liquid nourishment directly to their roots rather than via soil – comes in many shapes, sizes, and varieties. Here at HackSpace towers, we don't have space to create vast amounts of produce, but would like a little fresh salad grown on our windowsill, where the pesky slugs that blight the garden can't get at them. Can the simple kits available from direct-from-China sites help us here?

We turned to AliExpress vendor, Garden World Store, and purchased a twelve-holed bubble cabinet for £24.27. This is one of the simplest methods of hydroponics, where the plants are grown suspended directly in the water, and an aquarium pump keeps the liquid oxygenated. Other hydroponic approaches include trickling water over the roots, or applying a mist to the roots as they're suspended in the air. In principle at least, hydroponic systems should allow plants to grow faster and with less intervention than those grown traditionally. They can also be grown more densely, which comes in particularly useful when space is limited indoors.

PICK OF THE CROP

As we're fairly literal-minded, we decided to test our setup with watercress. Hydroponic systems can grow a wide range of plants, but leafy greens are the easiest and quickest to grow, particularly in a simple system like this. They're also a crop that can be expensive to buy and taste better when fresh, so make an ideal candidate for home cultivation.

There's not too much to the assembly – essentially everything just slots together. To start your crop, first germinate some seeds (place them on a damp paper towel for a couple of days until they just start to grow, then transplant them to a foam plug (included in the kit) using the supplied tweezers). Leave these soaking in a small amount of water until they have three leaves, then fill up the tub with a water and nutrient solution (not supplied), and start running.

There's no clear information on exactly how much bubbling the solution needs, as it's hard to measure the amount of dissolved oxygen in water. It will vary, depending on the power of your particular bubbler, the strength of your plants, and a myriad of other factors, so you'll need to experiment to get it right. We've combined our system with a simple timer switch that runs the bubbler when we're least likely to be near it, (so the noise of the bubbles is less distracting).

As well as water and nutrients, plants need light. We've placed our system on a windowsill that's in sunlight for most of the afternoon, and this is working well through summer. If you have less light – or for year-round growing – you might want to partner this system with a grow light to speed up growth. If you do, make sure that you get the right colour grow light for the crops you produce – the balance of frequencies can have a big effect on the plant, positive and negative.

Above With the lid lifted up, you can see the roots hanging in the water

REAP WHAT YOU SOW

Our setup is running, and we took our first crop just three weeks after planting out. We're using a cut-and-come-again approach, where we harvest some of each plant and leave the rest to keep growing. This should mean that our system will continue to supply us with watercress for many months to come. We've found that now everything is going, there's very little to do. About once a week we top up the water levels, but other than that, we've not had to do anything.

So far, this little hydroponic system has delivered a quick supply of salad in a small space with little effort – exactly what it promised to do. Hopefully it will mean that the days of getting halfway through a bag of salad before it starts leaking brown juice are behind us, and we will now subsist on a diet of fresh, green leaves. Whether this gives us better health for better making is still to be determined.

The hydroponics setup is made from cream-coloured plastic that does let a little light through (as opposed to the black plastic of many more expensive hydroponics setups). This hasn't caused any problems for us, however, though it could lead to a build-up of algae in the tank over prolonged usage.

In the world of hydroponics, our setup is tiny, both in terms of the size of plants it can support, and the number of them. This is perfect for a few salad leaves, but it'll struggle with larger plants. Make sure that you match the hydroponic system you get to the crops you want to grow to avoid disappointment. □

ONLY THE BEST

Where am I, and where are we going?

How to easily figure out a change in direction of orientation over time

By Marc de Vinck

@devinck

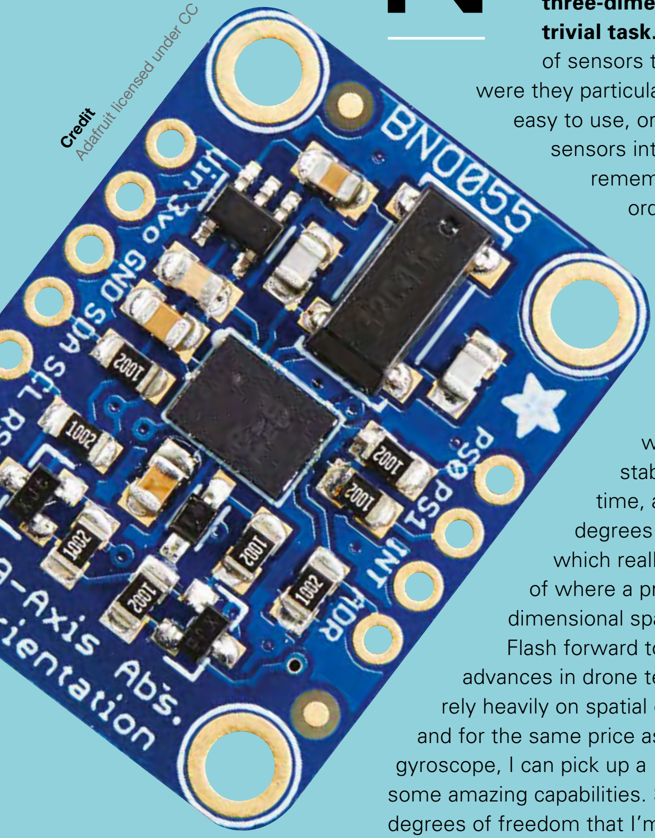
Not that long ago, creating an electronics project that knew its specific orientation in a three-dimensional space was no trivial task. There were a variety of sensors that worked, but rarely were they particularly small, nor were they easy to use, or combined with other sensors into one package. We remember about 15 years ago ordering a MEMs-based accelerometer for a project. It was small, and worked fairly well, but it had its limitations. It would drift, meaning the sensors' output data would not be particularly stable over a period of time, and it also only had two degrees of freedom (2 DOF), which really didn't give a true idea of where a project was in a three-dimensional space.

Flash forward to today, with all the advances in drone technologies, which rely heavily on spatial orientation awareness, and for the same price as that simple three-axis gyroscope, I can pick up a 10-DOF board that has some amazing capabilities. So, what exactly are the degrees of freedom that I'm talking about?

It's fairly simple to understand, and, with all these boards, almost as easy to implement into your next project. A degree of freedom can be thought of as a direction of movement. A door typically swings in one direction or axis. The door rotates back and forth, or has yaw, which is one degree of freedom (1 DOF). For any ridged body to have full free 3D movement, it requires six degrees of freedom (6 DOF) which are categorised into two different envelopes. The three-axis translational envelope consists of up/down, forward/backward, left/right and the three-axis rotational envelope consisting of roll, pitch, and yaw. Together you can move, or in our case sense the movement, in a 3D space.

The inertial measurement unit boards (IMUs) we are looking at today typically have about nine degrees of freedom (9 DOF), but some are described as having 10 DOF, and we've seen a 12-DOF board too, which can seem a bit confusing since you only need 6 DOF for orientation. Typically, we are talking about boards that integrate several sensors to allow it to know where it is in a 3D space, or more specifically its change in 3D space.

IMUs typically do this by leveraging data from a three-axis accelerometer for measuring acceleration, a three-axis magnetometer for measuring magnetic forces to help with orientation, a three-axis gyroscope for changes in orientation, and a barometric sensor, with a temperature sensor, to help account for drift. All totalled, you get nine to ten degrees of freedom, or more accurately described as 9–10 streaming data points from the various sensors.



Credit
Adafruit licensed under CC

Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout

Powered by Fusion

ADAFRUIT ◈ $34.95 | adafruit.com

Being able to sense an object's movement in space isn't particularly difficult until you start to realise that it's a lot of data that needs to be analysed fast and accurately. Movement is one thing, but knowing the precise orientation of an object in a three-dimensional space is a whole other monster to figure out, and that's where Adafruit's 9-DOF Absolute Orientation IMU Fusion Breakout board really shines.

The secret to the success of this IMU is Bosch's Fusion algorithm that blends the accelerometer, magnetometer, and gyroscope data into a stable three-axis orientation output that you can easily implement into your project.

The sensor also integrates an ARM Cortex-M0-based processor that interprets all the different streaming data from the sensors, and spits out the data in quaternions, Euler angles, or vectors. There's some really complicated maths under the hood, but you won't need to worry about that, as this board gives you simple and meaningful data that's easy to understand.

> **There's some really complicated maths under the hood,** but you won't need to worry about that, as this board gives you simple and meaningful data that's easy to understand

You'll need to solder a few header pins on the board prior to using it. As usual, Adafruit isn't leaving you with just a data sheet to try and figure it all out; they have created an extensive 'getting started with' guide that includes everything you would ever want to know about the board, including Arduino and CircuitPython code examples, and some neat graphics examples using Processing. □

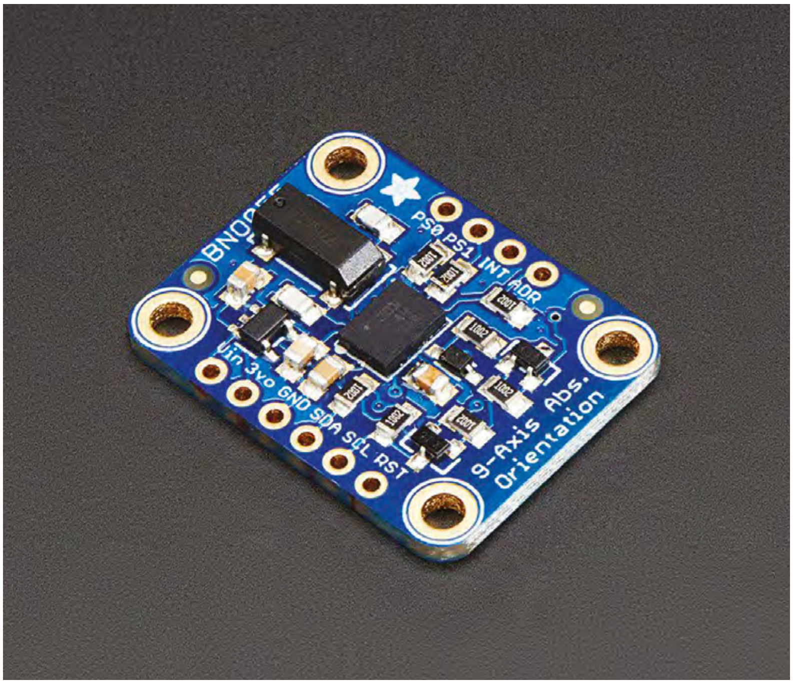
VERDICT

Accurate, easy to use, and great documentation

10/10

Below
Accurate data and easy to use, all from one tiny board

Credit
Adafruit licensed under CC



SparkFun 9DoF Razor IMU M0

Razor-sharp accuracy

SPARKFUN ◈ **$34.95** | **sparkfun.com**

The SparkFun 9DoF Razor IMU M0 gets its power by combining a SAMD21 microprocessor with a 9 DoF MPU-9250 sensor. It's a tiny re-programmable board with a lot of features and accuracy. You can easily use the board to monitor and log motion, transmit Euler angles, sense linear acceleration, angular rotation velocity, and the vectors of a magnetic field.

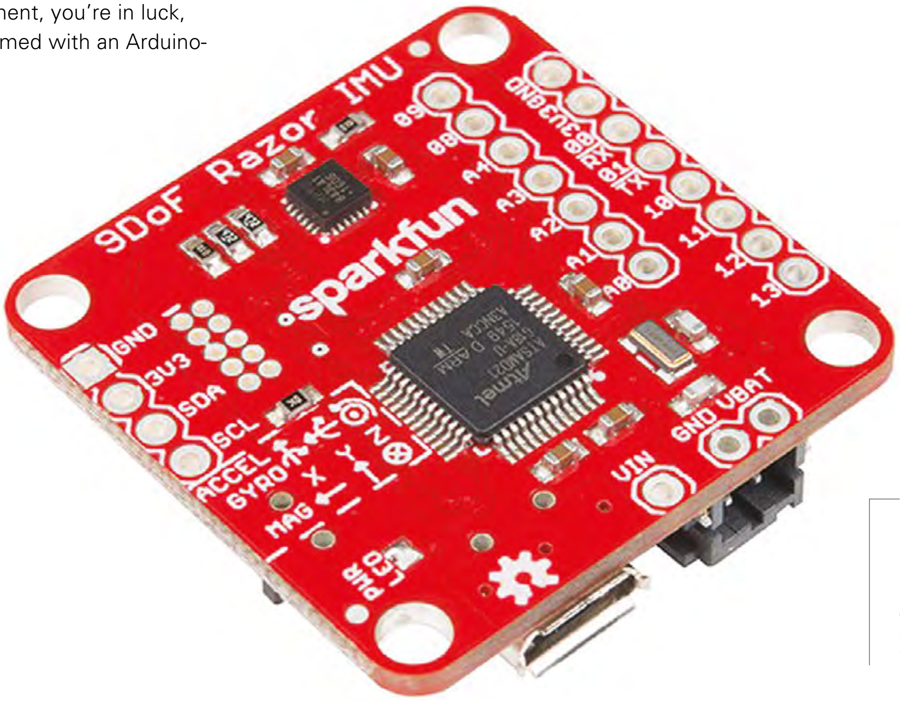
In addition to the onboard microcontrollers, the 9DoF Razor IMU has a micro SD card slot on the back, a LiPo battery charger, a power switch, and it's expandable, thanks to a few extra I/O pin break-outs that are all ready for any additional components your project may need. If you are familiar with the Arduino developer environment, you're in luck, since it comes pre-programmed with an Arduino-compatible bootloader, so you'll be up and running fast. SparkFun also has a great tutorial page to answer all of your questions and walk you through the steps required to get the IMU sensing the world as it moves around you. □

VERDICT

Precise and powerful

10/10

> **"In addition to the onboard microcontrollers,** the 9DoF Razor IMU has a micro SD card slot on the back, a LiPo battery charger, a power switch, and it's expandable"



Left Everything you need in one smart board

Credit SparkFun Electronics CC BY 2.0

Parallax LSM9DS1 9-axis IMU Module

Designed for flight

PARALLAX ◆ $24.99 | parallax.com

Left ◆
A small board that is ready to use and supported in BlocklyProp

Credit
Copyright © 2018 Parallax Inc

The LSM9DS1 9-axis motion sensor module from Parallax is a simple and affordable way of measuring direction, orientation, and motion. The same components that give this board so much power are used in their ELEV-8 v3 Quadcopter's Flight Controller, which should give you some confidence that the board is capable of accurate measurements, as keeping a quadcopter in the air is no trivial task.

This board contains the typical sensors found on many other IMUs, a three-axis accelerometer, three-axis gyroscope, a three-axis magnetometer, and a temperature sensor. The board can be used with almost any microcontroller, and it can be powered by 2.8, 3.3, or 5 volts. Communication works with either a 3-pin SPI interface or an I^{2}C bus, and features an interrupt which can be used for detecting and reacting to free falls or other inputs. It's all ready to use with the BlocklyProp environment, but you can use it with almost any other development environment too. □

VERDICT

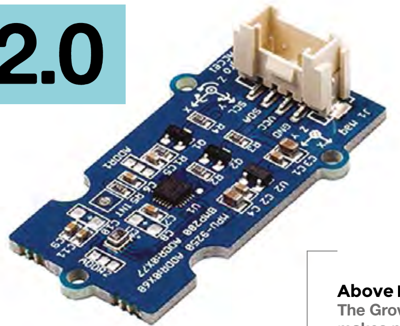
A good choice if you are writing your code in Propeller C or BlocklyProp

8/10

Grove - IMU 10DOF v2.0

Always going one better

SEEED STUDIO ◆ $16.90 | seeedstudio.com



Above
The Grove system makes prototyping new projects really easy

Credit
Seeed Technology Co.,LTD

Seeed Studio has created an interesting no-solder platform known as the Grove system. Having played around with a few of their modules, we can attest to its ease of use when prototyping. They have a variety of modules available and most of the common microcontrollers and single-board computers are compatible with the system through the use of a breakout board.

The Grove IMU 10DOF v2.0 is the latest iteration of the board, which is capable of motion tracking by utilising a three-axis gyroscope, three-axis accelerometer, three-axis magnetometer, and a digital motion processor (DMP), all housed in a compact Grove-compatible package. Another nice feature is its low power draw of only 2.74uA@1Hz.

This is an affordable board that is a must-have if you are already utilising the Grove system. You could easily create a motion-based game controller, wearable – or indeed anything that relies on spatial orientation – with this natty board, and without any soldering required, which could be massively appealing to many people. □

VERDICT

A must-have if you use the Grove system

8/10

Can I Hack It?
An MP3 player?

Can you easily add sound to your project?

Les Pounder

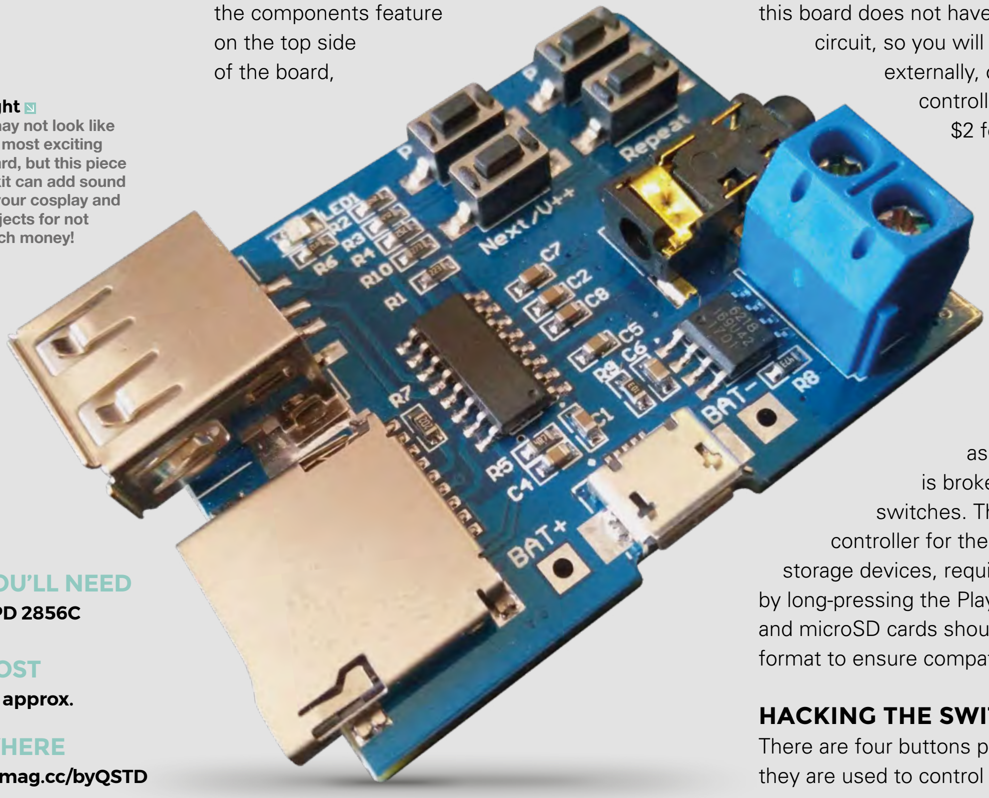
@biglesp

Les Pounder loves taking things to pieces and seeing how they work. He teaches others as part of the Raspberry Pi Foundation's Picademy event. He blogs at **bigl.es**

MP3 players have been with us for many years. In fact, we remember our first, a 64MB USB flash MP3 player that cost £50 in 2003. But thankfully, over the years the prices of MP3 players have dropped and now we can purchase the bare bones of a unit for around £5 in the shape of the GPD2856C. But what can we do with it, and how does it work? Well, let's take one apart!

GENERAL CONSTRUCTION

Measuring just 48 mm by 35 mm, this tiny board packs a lot of components. Constructed of a simple dual-sided circuit board, all of the components feature on the top side of the board, enabling easy access to everything that we need to hack. In fact, the datasheet for this board states, "Module is designed to facilitate modification"– that is great to read, thanks! The board has no mounting holes to secure the board inside a case or chassis, but hot glue will provide a stable platform. For use in cosplay, the unit could be contained in a project box and sewn/fixed, using Velcro, into the costume.



Right It may not look like the most exciting board, but this piece of kit can add sound to your cosplay and projects for not much money!

POWER

The GPD2856C works with voltages between 3.7 V and 5.5 V, and can be supplied from the built-in micro USB port, or via a lithium battery connected to the B+ and B- pins, but you will need to solder your own reusable connectors for these pins. Note that this board does not have a built-in charge control circuit, so you will need to charge the battery externally, or add your own charge controller, such as a TP4056 (around $2 for five from AliExpress.)

ELECTRONICS

The dominant chip on the board is the GPD2856C itself, and this 16-pin chip serves as the audio processor, supplying audio to the 3.5 mm headphone socket and the speaker terminals, and as the control interface, which is broken out to four momentary switches. The chip also acts as a controller for the USB and microSD flash storage devices, requiring us to select the device by long-pressing the Play/Pause/Mode button. USB and microSD cards should be formatted using the FAT format to ensure compatibility.

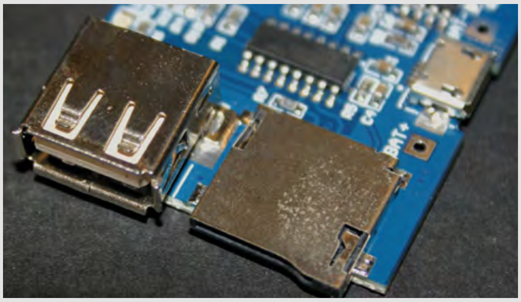
HACKING THE SWITCHES

There are four buttons present on the board, and they are used to control playback and volume for your

YOU'LL NEED
GPD 2856C

COST
£5 approx.

WHERE
hsmag.cc/byQSTD



audio. All four of these buttons operate by default as 'high' running with 3V of voltage. When pressed, the buttons are pulled 'low' via a connection to GND. This triggers the GPD2856C to perform an action, based on the button and duration of the press. Short presses are used to change tracks, play/pause, and repeat. Long presses control the volume and the source of the audio. These buttons can also be easily desoldered and replacements added, either larger more robust buttons for a build, or discreet buttons or sensors for cosplay. We could also connect the buttons to the GPIO of a Raspberry Pi, micro:bit, Arduino etc., and use the GPIO to trigger the audio.

AUDIO PLAYBACK

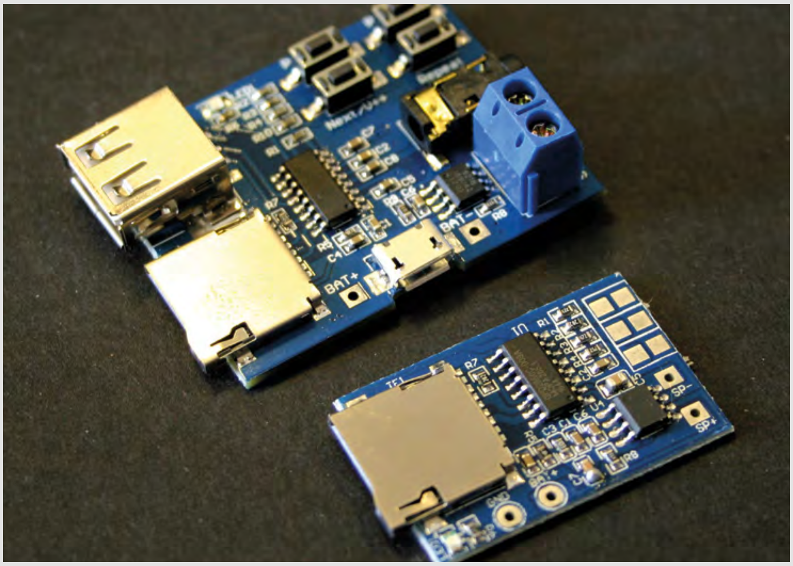
The GPD2856C has decent audio playback. The sound may not be for audiophiles, lacking bass and presence, but it is clear and loud, thanks to a 3W amplifier. Audio, via the 3.5mm headphone jack, works with no issue and can be used with headphones and external amplified speakers. Audio from the speaker terminals is another matter. You really need to use an amplified speaker with the speaker terminals. We tried a number of non-amplified speakers and none of them produced any noise, so we connected an amplified speaker to the terminals and we had sound, but with a caveat: there was a noticeable hum to the audio, which we traced to a bad ground connection. So if you get this issue, solder a wire from the SP- terminal to any GND pin; we used the B- connection.

CONCLUSION

For discreet audio playback on a budget, this is a steal! Cosplayers who need to add sound to their costumes will love this board. It's so easy to hack and work into any design. For example, we could control audio playback using an accelerometer connected to an Arduino to give us that 'swoosh' sound effect for a homemade lightsaber. Or, we could use the board to add sound effects for our M41A Pulse Rifle and motion tracker! The option to use USB or microSD provides flexibility in the field, as does the ability to connect a battery. It just would have been nice to have a JST connector for lithium batteries to connect to, but at this price that would be asking too much. The speaker terminals are handy to have but, in all honesty, 90% of us will just connect an amplified 3.5mm speaker, so we could desolder those terminals to lower the height of the board, and of course add the terminals to our bits box.

The smaller cousin of this board, the GPD2846A, is a viable alternative, being half the size of the GPD2856C; it lacks the USB interface and 3.5mm headphone jack, but it does work with unamplified speakers.

The GPD2856C is a serious contender for your money and deserves a home in one of your projects! □



Above
The GPD2586C (top) offers all the ports and terminals you need, but the GPD2846A (bottom) is half the size, with fewer features, but only £1 delivered!

Above Left
Choose between USB and microSD for your storage needs. You can easily switch between them at the touch of a button

WHICH **BOARD?**

Adding audio to your project usually involves expensive Arduino shields. While this may be the best approach for certain projects, sometimes you just need a nice cost-effective way to add audio. Boards such as the GPD2856C and the GPD2846A offer this feature for a low price. Pair them with an Arduino used to trigger the button presses, and we have a low-cost electronics sound effects setup for our cosplay.

But if you really do need an Arduino shield for your project, you can pick up the Adafruit Wave Shield for around £20, or the Audio Hacker Shield for around £28 from **coolcomponents.co.uk**. Both of these boards are based on the layout of an Arduino Uno, which means they are around twice the size of the GPD2856C, and four times the size of the GPD2846A. So, if you have space and need an Arduino board, these are your best options.

But if you need a small audio board, then there are also other alternatives, but higher in price than the GPD boards. Adafruit has its own diminutive boards, including one designed for the Feather range: the Adafruit Music Maker FeatherWing attaches to the board by stacking on top, so we still have a nice neat package, and a 3.5mm audio output. This board costs £18, much more than the GPD range.

As you can see, there are a number of boards for all price ranges. So, dear maker, the choice is yours.

TinyLily Mini by TinyCircuits

A wearable microcontroller, suitable for dolls

TINYCIRCUITS ◆ **$9.28** | **tinycircuits.com**

By **Sophy Wong** sophywong

Below ◆
Small enough for almost any project

Need a microcontroller for a tiny wearable project? Break out your magnifying glass! The TinyLily Mini by TinyCircuits is an ultra-compact Arduino-compatible microcontroller designed for e-textiles. But don't be fooled by its size – with an Atmel Atmega328P on board (the same processor as Arduino Uno and LilyPad), the TinyLily Mini can drive complex projects and be hidden in the smallest wearables.

At one twelfth the size of a standard Arduino Uno, the TinyLily Mini is smaller than a 1p coin and smaller than a CR2032 battery. In fact it's so small, it's easy to lose – so keep it inside its plastic bag until you're ready to sew it to something. How could a microcontroller the size of a breath mint possibly compete with a full size Arduino? We put the TinyLily Mini through its paces to find out if this tiny board measures up.

WHAT YOU GET

The TinyLily Mini is available on its own for $9.28, or as part of a kit. We opted for the Starter Kit ($39.95) which comes with two TinyLily Mini processor boards, a USB adapter for programming, a battery connector, six sewable LEDs, and one incredibly tiny sewable button. That's more than enough bits and pieces to create some interesting wearables, and more components are available on the TinyCircuits website. The kit also comes with a quick start guide, which is very handy for programming the TinyLily Mini for the first time.

The kit does not come with a battery, and it's worth noting that the included battery connector uses a JST-SH connector, which is smaller than the standard JST-PH connector on many LiPo batteries. If your LiPo batteries came from SparkFun or Adafruit, chances are they will not fit the TinyLily battery connector, and you'll want to purchase a new battery from TinyCircuits.

Although it doesn't come with the starter kit, TinyCircuits also makes a motor shield for TinyLily, which can drive one DC brushed motor. And, because the TinyLily Mini is Arduino-compatible, you can use it with other Arduino-compatible wearable components, like Adafruit's popular NeoPixels and SparkFun's LilyPad modules.

YOU PROBABLY ALREADY KNOW HOW TO USE IT

Aside from its size, the TinyLily Mini shares the same form factor as many other microcontrollers designed for e-textiles. Its familiar circular shape and sew tabs are clearly inspired by Leah Buechley's iconic LilyPad Arduino. If you've used other wearable boards like LilyPad or Adafruit's Flora, you'll be up and running with TinyLily in no time. Programming with the Arduino IDE is pretty standard, just select 'Arduino Pro or Pro Mini (3.3V, 8MHz) w/ATmega328' in the boards menu, and upload your code.

There are eight sewable I/O pins on the processor board; four are digital pins and four are analogue/digital pins. Two pairs of sewable power and ground tabs are conveniently located at the top and bottom of the board. And if that's not enough, there are more solderable pins on the back of the board: three more I/O pins, reset, VCC, and GND. That's an impressive amount of capability for such a compact design, and the solder-friendly gold-plated tabs mean this board is useful for more than just e-textile applications.

FORM VERSUS FUNCTION

To achieve its tiny size, some convenient features that are standard on other wearable boards were off-loaded to separate modules, such as a JST battery connector and a USB adapter for programming. If you want a reset button or on/off switch (highly recommended for any wearable project) you'll have to add these to your circuit on your own. This is fine for many wearable projects, where you'll want to customise the locations of these components anyway.

Keep in mind that although the board itself is tiny, adding the battery connector increases its size, and the connection is a little awkward. The battery connector plugs into the board face-down, which means the connector doesn't lie flat against the surface of the project. This is more of an annoyance than a problem, but you'll want to plan around it in your wearable designs, as the connector has corners that could scratch fabric or skin. TinyLily Mini has no power regulator, so be sure your power source is within the 2.7V – 5.5V operating voltage. Trading convenience for customisation means this board is less beginner-friendly than other wearable boards, and more suited to makers who are already comfortable with e-textiles and Arduino.

The sew tabs on the TinyLily Mini are definitely small, but they're easy to sew to, and require fewer stitches to fill with conductive thread for a good connection. The sew tabs are labelled on the back of the board, likely due to lack of space on the front, and this makes it a bit tricky to be sure that you are sewing to the correct pin. It really helps to use a very small dab of hot glue to hold these tiny components in place while you sew. The board is washable, however you'll want to take care not to bend or snag the male five-pin connector when washing.

SMALL, MODULAR, AND POWERFUL

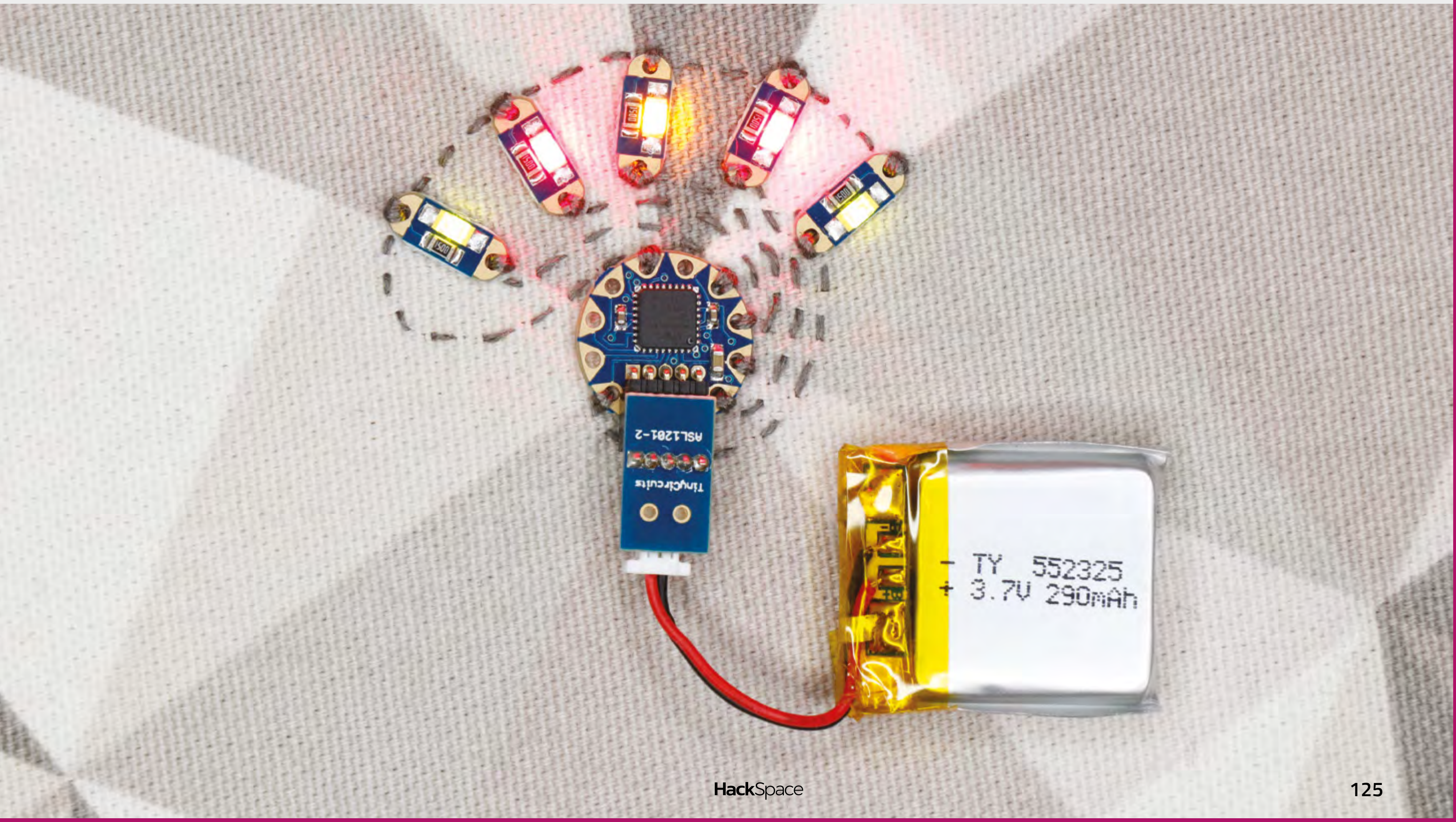
While other hardware companies are adding more features to their microcontrollers, TinyCircuits has stripped features away to create a board that is efficient and effective. Removing non-essential components from the main board results in a system that is customisable for every project. Small components like those in the TinyLily family make it easy to distribute your circuit invisibly throughout your wearable as you see fit, without having to sacrifice function. While it's not the board we'd recommend for your first project, the trade-off in convenience is easily worth it for intermediate and advanced makers seeking to fit electronic circuits invisibly into their wearables. □

Below
The sewable LEDs come in red, green, and amber

VERDICT

Like a magical blue police box, the TinyLily Mini microcontroller packs massive functionality into a mind-blowingly small footprint.

8/10



SiFive HiFive1

The most open microcontroller around

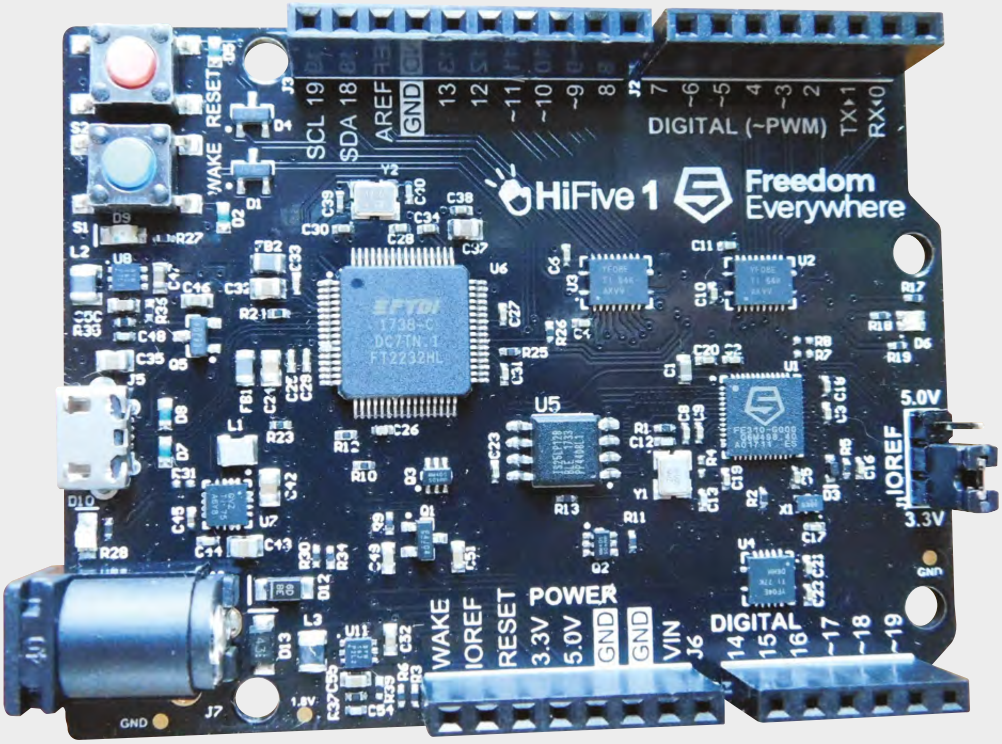
SIFIVE ◈ $59 | sifive.com

By Ben Everard   ben_everard

There's a wide range of microcontroller boards around. You can broadly categorise them by the instruction set architecture (ISA) they use. This is the type of instruction that the processors actually execute. Most hobbyist microcontrollers are based on either AVR or ARM processors, a few PICs still linger on, and the ESP boards use the Xtensa ISA. In general, you don't have to concern yourself too much with the ISA of a particular board, as long as your programming environment understands it.

All these ISAs are proprietary. They're controlled by the companies that make the microcontrollers (or licensed from other companies). For example, you can't make a new ARM chip without ARM's permission. What's more, the circuitry that implements this ISA is kept as a closely guarded secret. While we may use open-source software, it's running on top of closed-source hardware.

The HiFive, however, is different. It's based on RISC-V. RISC-V, unlike almost every other ISA, is open-source. It's released under the BSD licence, which means that anyone can design chips using it. Not only



Right ◈ The form factor will be familiar to anyone working with microcontrollers that, but there are open-source designs for processors using the RISC-V ISA, including Rocket Chip. This means that anyone who happens to have access to a chip fabrication lab can download the designs and knock up some processors. Of course, not everyone has access to chip fabrication, so we have to rely on others to build our hardware for us.

SiFive have brought together various bits of open-source RISC-V design into a microcontroller in a similar way to how Red Hat or Canonical bring together bits of open-source software to make a Linux distribution (after all, you need more than just a processing core). At the moment, they're the only source of actual chips based on RISC-V (though there are other projects, such as lowRISC, that have implementations that can run on FPGAs).

This isn't a case of being open for open's sake though. SiFive is in the business of helping clients make custom-designed chips using the same processing cores. The open-source Rocket Chip cores are the starting point, and they'll help you add whatever you like to them to tweak these chips for your personal needs – of course, this is only for large manufacturing runs, not us hobbyists.

It would be wrong to call the HiFive1 development board fully open. The schematics and processing core are, but there are plenty of other chips on the board that aren't. For example, it relies on an FTDI USB connection. However, the HiFive is the most open development board available today. If you'd like a more open board, the microcontroller chips are available and you can build your own board around this.

OPEN INSIDE

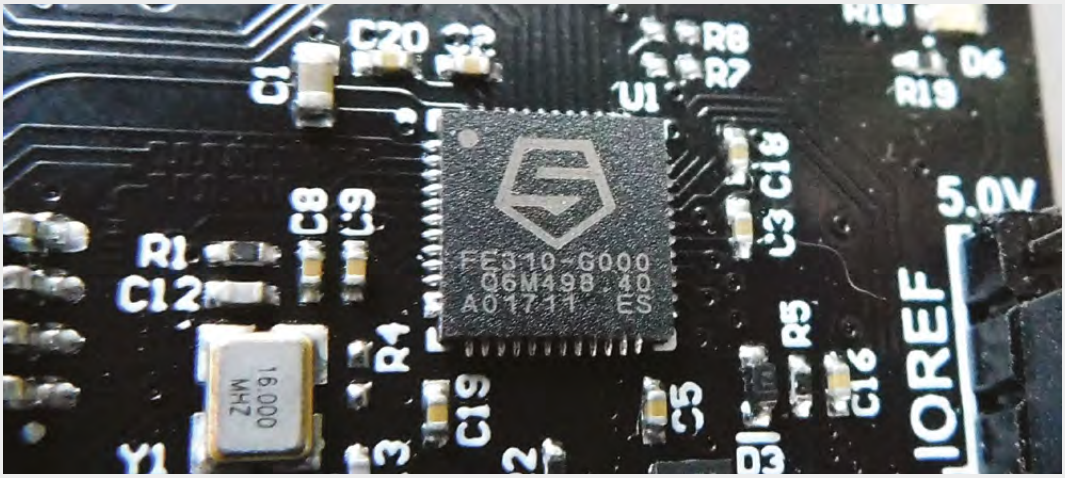
The HiFive1 is powered by an E31 processing core, which is a 32-bit architecture running at 320MHz. The per-cycle performance of the E31 processing core in the HiFive1 is roughly similar to the Cortex-M0+ processors found on boards like the Arduino MKR1000, however the E31 is clocked much higher. The performance is in the same ballpark as the Teensy 3.6, which has a 180MHz Cortex M4 processor which, while slower than the E31, can perform more in each cycle.

There's a range of options for programming the HiFive1, but for hobbyists, the most familiar will probably be the Arduino IDE. Here, as you'd expect, it's just a case of installing the board definition and selecting it in the GUI.

> "There's a range of options for programming the HiFive1, but for hobbyists, **the most familiar will probably be the Arduino IDE**"

The HiFive1 comes in the now familiar Arduino Uno form factor, but it does have a slightly different pinout – most notably, that there are no analogue pins on the HiFive1. This straight away rules it out of some areas where the extra processing power would be useful, such as sound processing.

At $59 (plus $15 shipping, if you're outside the USA), the HiFive1 is a touch expensive by microcontroller standards, but if you're after an Arduino-compatible board with some serious processing power, it's one of the best options out there. We suspect, though, that most people aren't buying the HiFive1 for the performance, and it's the chance to use open-source chips which is the real driver. □



Left You can also buy the chips separately if you want to build your own board around RISC-V

VERDICT

It's great to see RISC-V chips on the market, and this one packs a lot of punch

8/10

SketchUp

Free web-based 3D design software

TRIMBLE INC ◆ free | sketchup.com

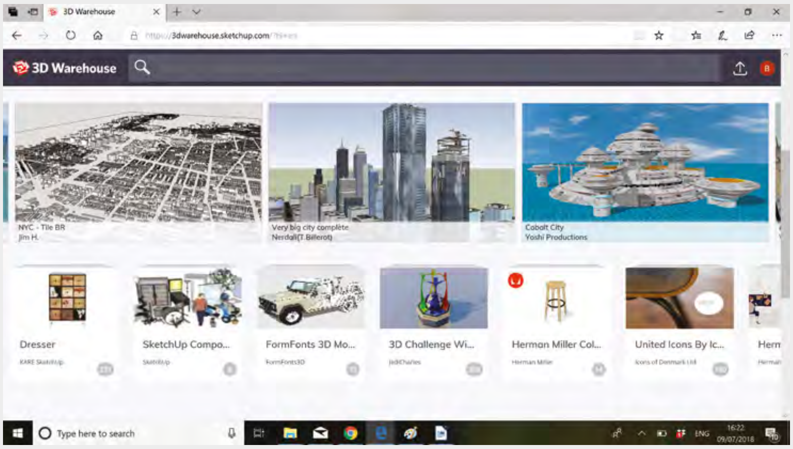
By Ben Everard

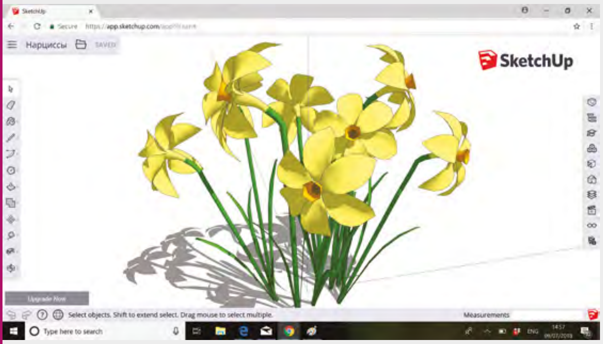
ben_everard

When we first start a project, we sketch out the design with pencil and paper. For some builds, this is all that's needed. In other cases, we need a bit more precision in the plans, and this is when we reach for 3D design software. There's a wide variety of software available for this task, from highly complex CAD systems for designing and even testing mechanical systems, to simple tools for building blocky shapes for 3D printing. Between these extremes lies a world of options.

SketchUp is fairly simple in its basic form – you can draw shapes and extrude them in to 3D objects, paint them, and build up your model. In addition to this, there's a huge selection of extensions that add more functionality. Using these, you can tweak the tool by adding the specific features you need. This makes it easy to get started, and only bring the complexity in when you need it (and have mastered the basics).

You can export STL files for 3D printing with the basic online version (free), but for a wider range of CAD export file types, you will need to upgrade to the Shop version for $119 per year, which also adds unlimited storage, better support, and better tools for working with materials. The Pro version (a one-off payment of $695) brings in an offline version.




SketchUp
Нарциссы
Select objects. Shift to extend select. Drag mouse to select multiple.
Measurements
Type here to search

Above The 3D Warehouse is packed full of inspiration for aspirational 3D designers

Below Flowers designed by SketchUp Warehouse user, DenDroDesign - hsmag.cc/VRLJQQ

3D modelling is quite a complex skill. SketchUp is easy to get started with, but unless this is something that you've worked with before, it will take a little practice before you're able to create complex models. Fortunately, SketchUp has a good selection of videos to help you bring your design skills at **www.sketchup.com/learn**.

CONVENIENT DESIGN

Alongside this learning material, there is a library of models that other users have shared that you can look at, edit, incorporate, or expand at will. You can see these at **hsmag.cc/sJXmWX**. With a good range of extensions, SketchUp is a free-to-use 3D design tool that is easy to use, and powerful enough for many hobbyist users, and has been popular, particularly with woodworkers and for more architectural designs. If you are looking for physical design software, then SketchUp is worth checking out and because it's web-based, there is nothing to download, so you can try it out straight away. Even with its extensions, it is still a little lacking when creating complex designs, but for beginner and intermediate users, it's a great choice. □

VERDICT

Simple to use 3D software, with a good range of extensions

8/10

Junkyard Jam Band

David Erik Nelson ◈ $24.95 | Nostarch.com

By **Andrew Gregory** AndrewGregory83

Making music is enormous fun. Making things that make music should add another layer of awesomeness to the mix, and so it shows in *Junkyard Jam Band*. As the title suggests, this book comprises a collection of how-to guides, showing the reader how to make music instruments of increasing complexity out of cheap junk you might have lying around and inexpensive, easy-to-obtain components.

Each project contained in the book is self-contained, so if you don't want to make an instrument out of a piezo speaker and slinky spring (this awesome 15-minute project is covered in the first chapter and is a perfect example of the book's low-fi ethos), you can skip ahead to something else without missing out. To be honest, though, if you don't like the sound of making a musical instrument out of a slinky spring, there must be something wrong with you.

Other projects include building an amplifier, electronic kalimba (a type of African plucked idiophone, as shown on the book's cover), a pickup for an electric stringed instrument, and a variety of sound effects.

There are also appendices covering electrical components, music theory and expanded versions of the circuits used in the book. These appendices are great reading on their own, even if you're not going to follow the build projects, with the electronics section in particular a useful primer for anyone who wants to doesn't know where to start. Yes, there's technical knowledge in there, including formulae for calculating resistance and capacitance, but the context of musical tinkering means it makes far more sense than it would in a physics classroom.

Junkyard Jam Band is an easy-to-read, easy-to-follow guide to creating homemade musical instruments which assumes zero knowledge and is shot though with enthusiasm like the words in a stick of rock. It takes the reader from very simple making to active electronics, even building a whole synthesizer, with effortless style, and the publishers thoughtfully provide sound samples of the things you're trying to build.

Don't spend thousands chasing your perfect sound – build it yourself! □

VERDICT

Junkyard Jam Band: DIY Musical Instruments and Noisemakers **is a fantastic book that, in the hands of the curious, will give many, many weekends of musical making.**

10/10

next month

issue #10

ON SALE
23 AUGUST

FEATURING

MULTI ROTORS:

BUILD YOUR OWN FLYING MACHINE

ALSO

→ THE HACKADAY PRIZE

→ STEPPER MOTORS

→ NEAR-FIELD COMMUNICATIONS

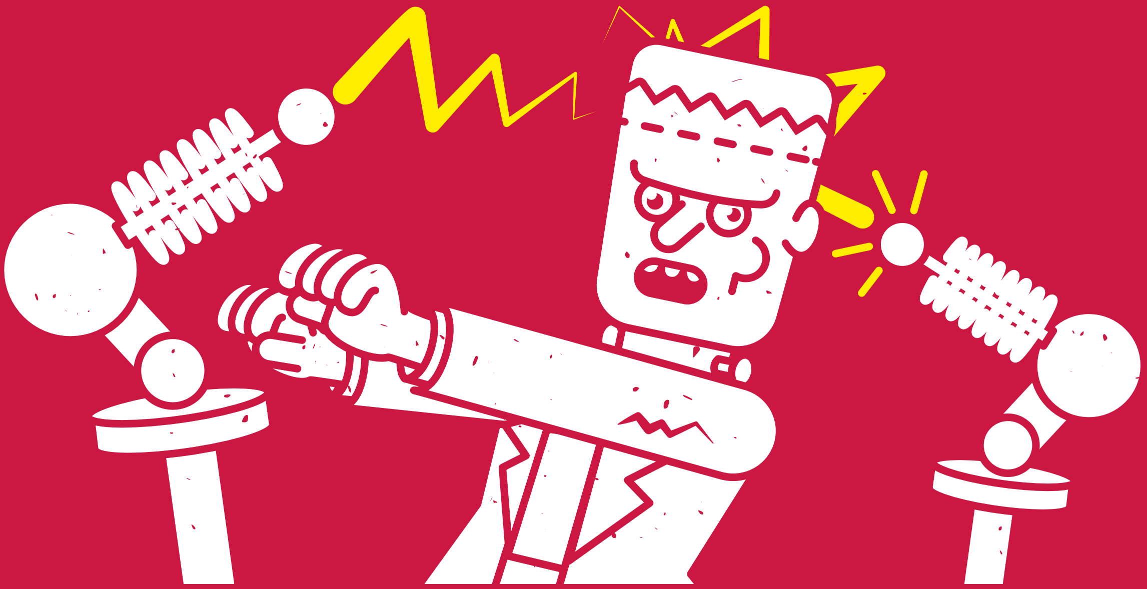
→ BUILD A TANDOOR

→ AND MUCH MUCH MORE

DON'T MISS OUT

hsmag.cc/subscribe

HackSpace
TECHNOLOGY IN YOUR HANDS



“INVENTION, IT MUST BE HUMBLY ADMITTED, DOES NOT CONSIST OF CREATING OUT OF VOID, BUT OUT OF CHAOS”

MARY SHELLEY

hsmag.cc

SanDisk
32GB
micro SD HC
CanaKit Raspberry Pi Quick-Start Guide
CanaKit
2.5A Power Supply for Raspberry Pi