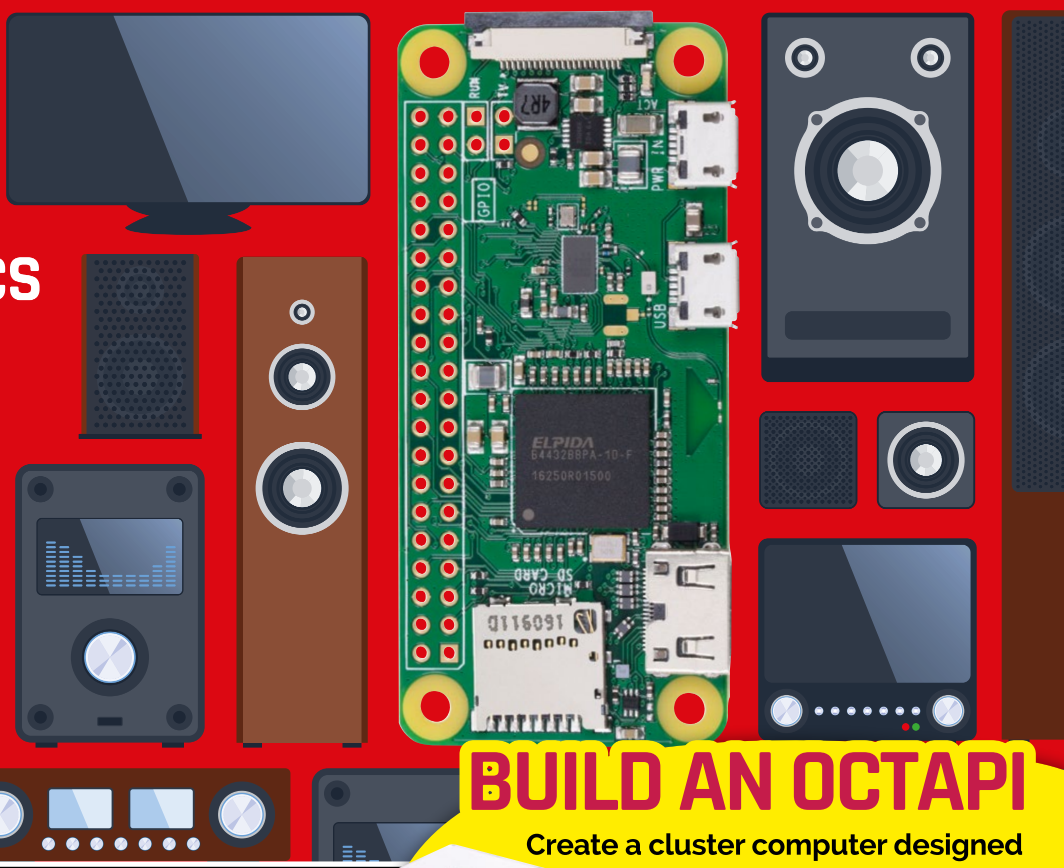
YOUR OFFICIAL **RASPBERRY PI** MAGAZINE

The MagPi

The official Raspberry Pi magazine Issue 66 February 2018 raspberrypi.org/magpi

MEDIA PLAYER PROJECTS

5 definitive guides for Pi-powered home theatre builds



PI ROBOT PLAYS COUNTDOWN

Watches TV game show and finds winning words

PAC-MAN ROBOTICS

Playing real-life Pac-Man with robots in a maze

FANCY A PINT?

Brewing beer with Raspberry Pi

SPECTRE & MELTDOWN

Eben explains how they work & why they don't affect the Pi

BUILD AN OCTAPI

Create a cluster computer designed by spymasters at GCHQ

Also inside:

- > **NEW PI ZERO WH - WITH HEADER - REVEALED**
- > MAKE A PI FRACTAL MUSIC GENERATOR
- > **OHBOT TALKING ROBOTIC HEAD REVIEWED**
- > LEGO MINDSTORMS WITH C-STEM

Issue 66 • Feb 2018 • £5.99

02

9 772051 998001

Raspberry Pi PRESS

WELCOME TO THE OFFICIAL MAGAZINE

The Raspberry Pi is the most versatile computer in existence. Thanks to its low cost, small size, and those handy GPIO pins, you can turn it into just about anything.

But you've got to start somewhere. And many Raspberry Pi owners start out with a media centre project.

This makes sense. With its HDMI socket and small form factor, the Raspberry Pi can quickly become a much more versatile media player than the ones you'd buy in the shops.

Starting on **page 16**, we've got a range of sound and vision projects that can put your Raspberry Pi at the heart of your entertainment. With it you can stream movies, play television shows, and listen to music tracks and radio. It's a great introduction to digital making with a Raspberry Pi.

Further inside the magazine, you'll discover stories from other makers about the incredible things they've built. This month has one of the finest selections of creations on earth. We've got a real-life Pac-Man game built using robots (**page 30**) and a robot called Rosie that can play the TV game show *Countdown* (**page 32**).

Raspberry Pi is a community, and it's the amazing things our readers build and share that make it special. Good luck with your next project and don't forget to share it with us.

Lucy Hattersley
Editor

THIS MONTH:

FIND US ONLINE raspberrypi.org/magpi **GET IN TOUCH** magpi@raspberrypi.org

The MagPi

Available on the App Store
GET IT ON Google Play

EDITORIAL
Editor: **Lucy Hattersley**
lucy@raspberrypi.org
Features Editor: **Rob Zwetsloot**
rob.zwetsloot@raspberrypi.org
Sub Editors: **Phil King and Jem Roberts**

DESIGN
Critical Media: **criticalmedia.co.uk**
Head of Design: **Dougal Matthews**
Designers: **Mike Kay and Lee Allen**
Illustrator: **Sam Alder**

PUBLISHING
For advertising & licensing:
Publishing Director: **Russell Barnes**
russell@raspberrypi.org | **+44 (0)7904 766523**
Director of Communications: **Liz Upton**
CEO: **Eben Upton**

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

SUBSCRIPTIONS
Select Publisher Services Ltd
PO Box 6337
Bournemouth
BH1 9EH | **+44 (0)1202 586 848**

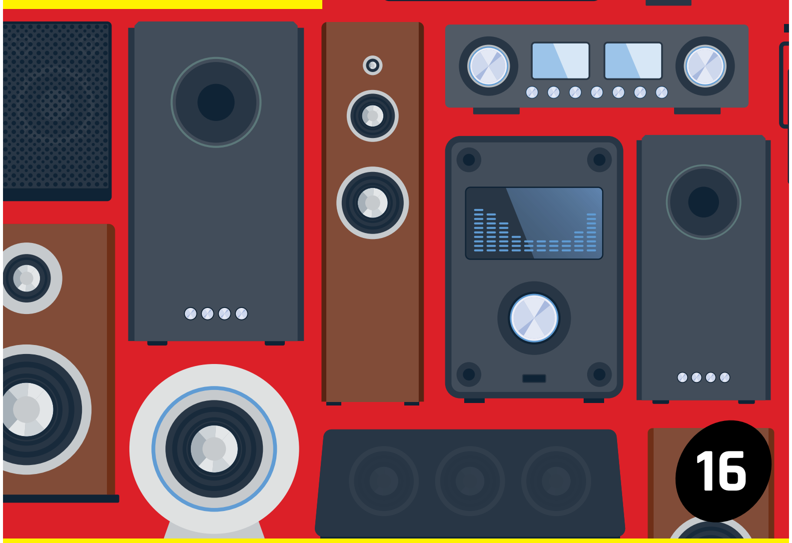
CONTRIBUTORS
Alex Bate, Brian Beuken, Harry H. Cheng, Mike Cook, Kylie Cooper, David Crookes, Kyle Goff, Phil King, Sean McManus, Binsen Qian, Matt Richardson, Laura Sach, Richard Smedley, Clive Webster

Contents

Issue 66 February 2018 raspberrypi.org/magpi

TUTORIALS

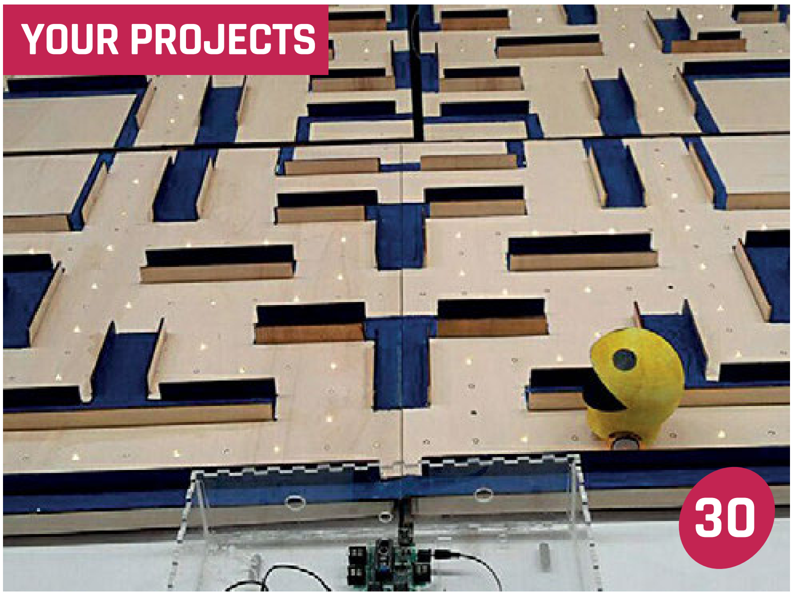
COVER FEATURE

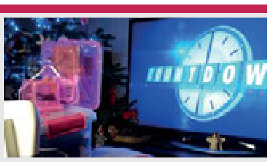


THE BIG FEATURE

WIN MEDIA CENTRE KITS

YOUR PROJECTS





REGULARS

COMMUNITY

REVIEWS

MELTDOWN

SPECTRE

RASPBERRY PI IMMUNE TO SPECTRE & MELTDOWN

It's a good time to use a Raspberry Pi for online banking

Above Discover how the Spectre and Meltdown exploits work later in the magazine

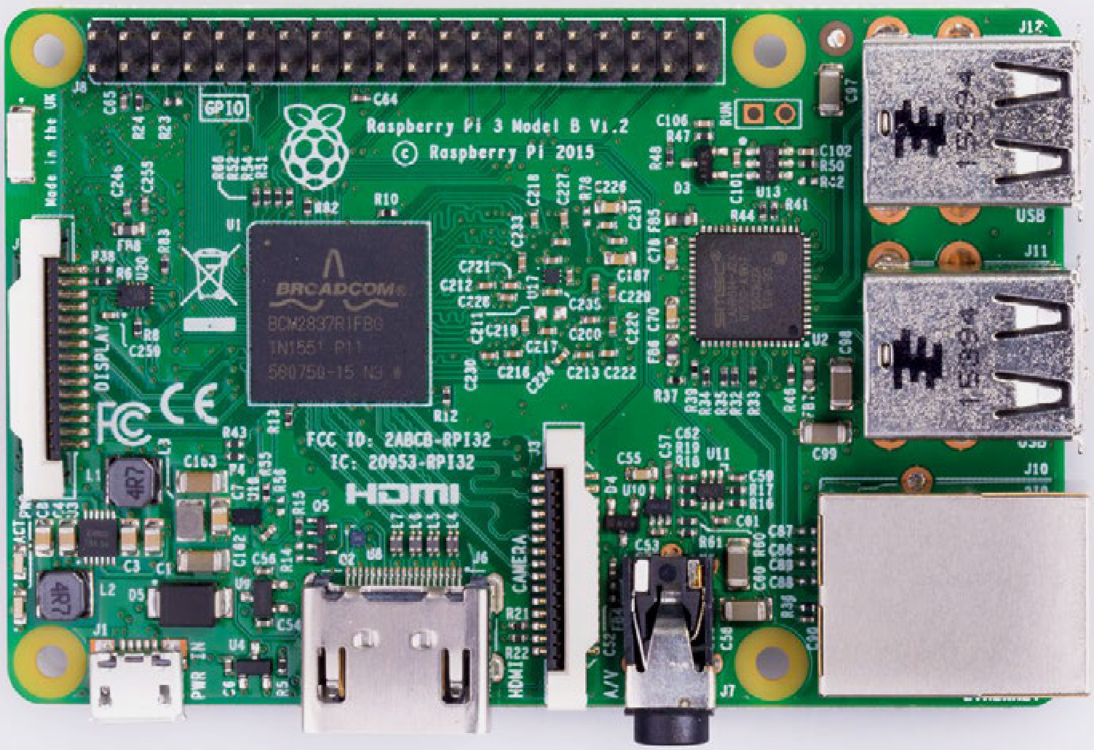
While some ARM processors are vulnerable to the Spectre and Meltdown exploits you may have heard about, every model of Raspberry Pi is not.

Both hacks exploit a weakness in how modern processors increase their efficiency by predicting and reordering instructions.

Very basically, Meltdown and Spectre exploit a processor performance technique known as speculative execution, which tries to pre-execute certain instructions before the CPU knows whether they are really needed. If they are not needed, the results are thrown away, but changes to the state of the processor's data cache caused by the unneeded instructions are not undone during the throwing away process. If an unneeded instruction would have accessed sensitive data, the hacks can read the data by analysing these changes, even though the instruction never really executed. Spooky! Sensitive data can contain personal and private data, including passwords. It's a bit tricky to explain as it

> Stay safe in the knowledge that the Raspberry Pi is not affected

Right Your Pi is immune from Meltdown and Spectre because its processor does not have the hardware these viruses exploit



involves some seriously niche, low-level computing stuff that the vast majority of people would never have to come across – unfortunately, that does not make it any less dangerous.

But we live and breathe this sort of thing, and Raspberry Pi co-founder Eben Upton goes into it in detail in a tutorial this month (see page 48). It's an excellent read that explains one of the most intricate aspects of technology in an accessible way.

Pi CPU is immune

As Arm Holdings revealed (**magpi.cc/2m9Ig6j**), the processors used in every model and version of the Raspberry Pi (the ARM1176, Cortex-A7, and Cortex-A53) are not vulnerable to Spectre or Meltdown attacks. The processors used in Raspberry Pi boards don't feature the hardware that Spectre or Meltdown exploits.

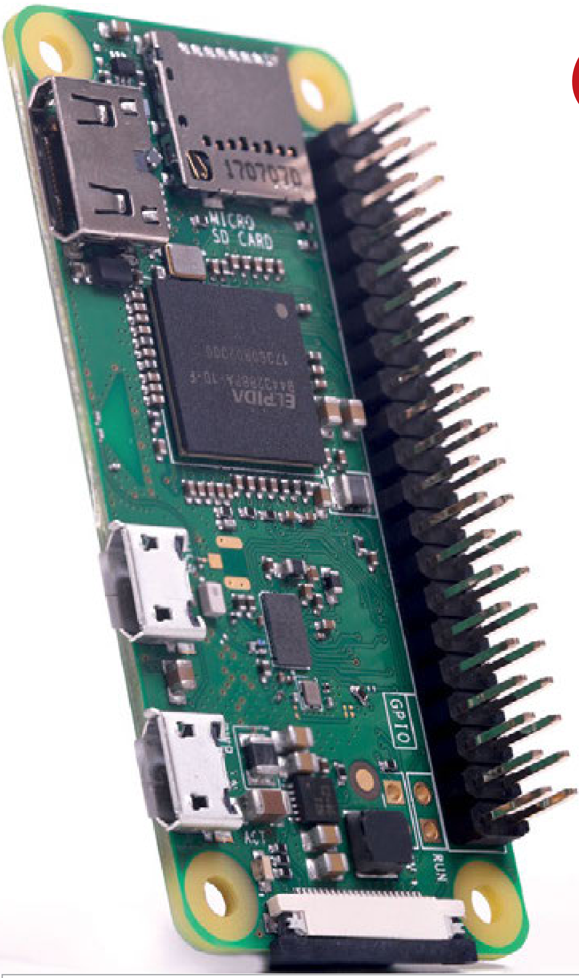
"The lack of speculation in the ARM1176, Cortex-A7, and Cortex-A53 cores used in Raspberry Pi render us immune to attacks of the sort," says Eben Upton.

So, you can stay safe in the knowledge that the Raspberry Pi is not affected – while patches are rolled out for other devices, at least you now know which computer in your house is the most secure for online banking.

For systems that do use vulnerable processors, patches have been issued to create a kind of firewall around the memory that Spectre or Meltdown could potentially snoop. Such a system comes at a performance cost, however, estimated to be something like 5%.

If you are thinking about temporarily switching to the Raspberry Pi as a more secure desktop, in issue #59 we ran a feature where we challenged features editor Rob to use a Raspberry Pi as his desktop PC for a week (**magpi.cc/2u0ZzK3**). There are some great tips in there that should help you set up your Pi as a desktop!

NEW PI ZERO WH LAUNCHED

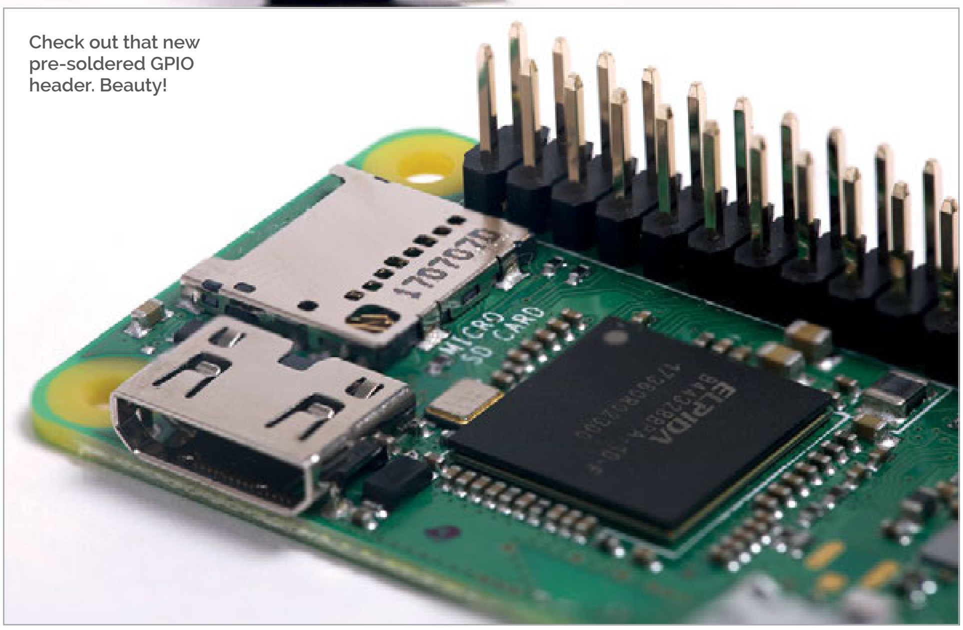


The Raspberry Pi Foundation has launched a new Pi model, the Pi Zero WH. The difference between this edition and the current Pi Zero W is its pre-soldered GPIO header.

Mike Buffham, Raspberry Pi Foundation Director of Product Management, explains that the new Zero WH has been launched "to support those customers who did not want to or feel comfortable with soldering the header themselves."

While Mike clarifies that "it seemed sensible" to solder the GPIO header on during manufacture, the move is "not completely simple." As the GPIO header is soldered to the opposite side to other components, this means "the boards have to go through the solder baths twice."

The new Pi Zero WH should be available through all your favourite retailers by the time you read this.



Check out that new pre-soldered GPIO header. Beauty!

RASPBERRY PI CAFÉS FOR MANCHESTER SCHOOLS

Web hosting company UKFast to invest £100 000

Following a successful pilot in 2015, web hosting firm UKFast has announced five further 'Raspberry Pi Cafés' for Manchester schools this year. The project represents a £100 000 investment from UKFast.

Aaron Saxon, UKFast's Director of Training and Education, reveals, "We are distributing 120 Pis across the five sites: Holy Name RC Primary School in Moss Side, St Bede's Prep School in Hulme, Alderley Edge School for Girls, The Hollins Technology College in Accrington, and The Factory Youth Zone in North Manchester."

Below The five new Pi Cafés will operate much like the pilot site in Broadoak School, Partington

The sites were chosen "where gaps in digital engagement exist", Aaron explains. This includes areas lacking "the resources to deliver cutting-edge digital training, as well as all-girl schools which have traditionally seen low uptake in technical subjects."

"there will be arcade, old-school gaming, and robotics cases" Aaron says. "We're providing the technology for the children in a fun and exciting way."

Paul Grier, Network Manager at St Bede's Prep School (one of the five new sites), adds that "in 20 years' time, 45% of jobs will be done by AI and robots. So if kids today don't understand [these things], they won't understand how the world works."

> We're providing the technology for the children in a fun and exciting way

School café

We asked Aaron how the Pi Cafés would actually operate, and it seems that's largely up to the schools: "Some schools may use it as a creative space, others will use it as their computer science classroom as well as an extra-curricular hub and space for the community."

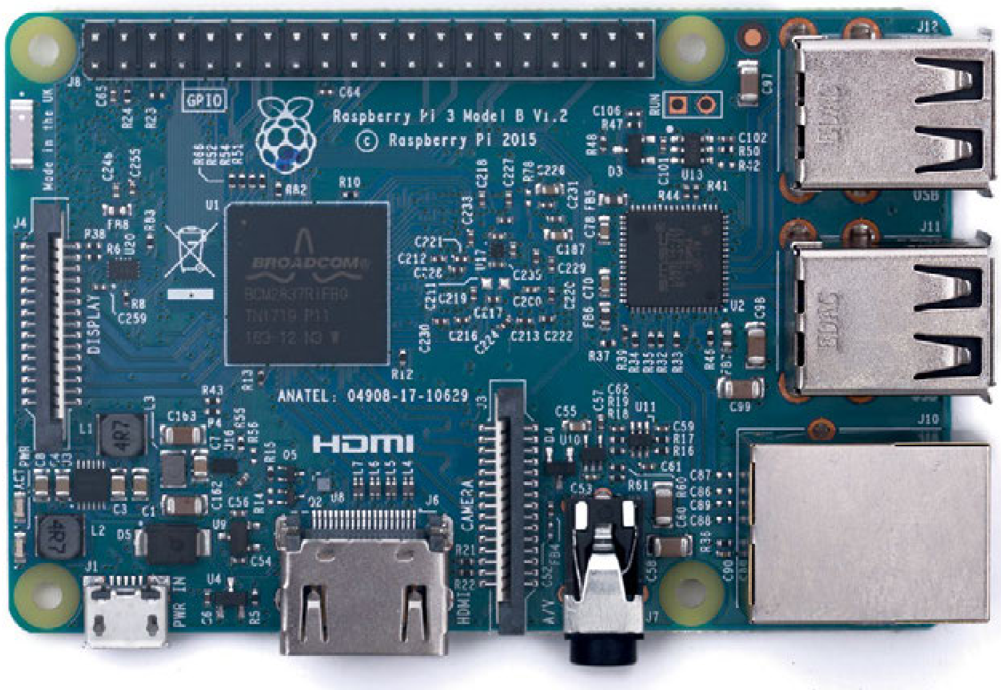
The Pis in question "won't look like traditional desktop units," Aaron tells us, "as we want them to be more computer-science focused." For UKFast, that means

Paul says he hopes the new Pi Café will "allow both children and the staff [of St Bede's] to delve more into computer science." While students and staff of St Bede's "learn ICT, which is processing and spreadsheets," Paul tells us that "programming hasn't taken off as much as I would have liked it to."

MR BLUE PI

Official Brazilian Raspberry Pi is blue

Below Apart from the colour of the circuit board, the Brazilian Pi is identical to any other Pi 3



Brazilians can finally buy an official Raspberry Pi, and to celebrate this breakthrough the Brazilian Pi model has a new, blue PCB.

The delay in delivering Pis to Brazilians was due to a regulatory hitch: all telecommunications products sold in Brazil must be approved by the Brazilian regulatory agency Anatel, and this certification has taken some time.

The reasons for the delay are unclear, especially as the approved Pi is functionally identical to any other Pi. The only difference between a Brazilian Pi and any other is the colour of its circuit board.

The excellently named FilipeFlop (**filipeflop.com**) is the first retailer to stock the new 'Anatel' Pis, with a blue Pi 3 costing R$199.90, or roughly £45.

Blue Raspberry Pi boards will only be on sale in Brazil – even if you order through Filipe Flop, you will only be able to buy the Blue Pi if you are a Brazilian resident.

NEW DUAL-LAYER PROTO BOARD

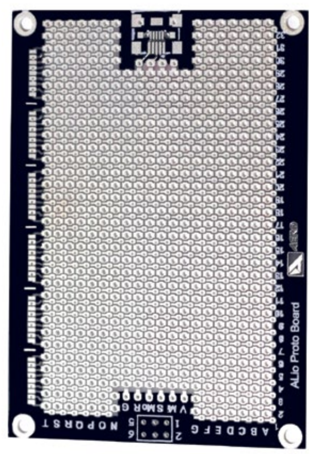
Add chips and USB ports to your quick builds

The ALio Proto Board recently hit its Crowd Supply target (**magpi.cc/2mgVz4X**), and with a pad layout that can accommodate chips (ICs) it should be useful board to get complex builds running quickly. There are variants compatible with Arduino and USB ports.

ALio Lead Engineer (and AERD CEO) Arief Adha tells us, "Currently we are on production for first batch [of ALio boards, but] my next plans are to publish the files [for the boards]. As Arief explains, "Since the board itself is fully open-source, we hope we can make rapid prototyping accessible for everyone who wants to prototype with SMD or PTH."

Arief even says, "All profit that we'll get [from ALio], we will allocate to develop new open-source stuff."

The AERD team are still focused on producing the ALio boards, however, with Arief confirming that "ALio is a good candidate to include in the Digi-Key catalogue." Keep an eye on **digikey.co.uk** if ALio looks useful for your next build.



The ALio Proto Board has pads for chips as well as components, with versions to easily add USB ports or make Arduino shields

VIVALDI COMES TO PI

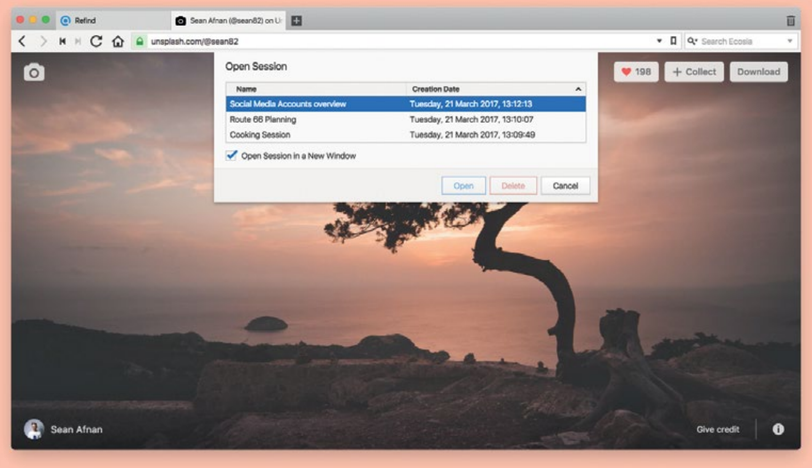
New web browser built for Raspberry Pi boards

The Vivaldi web browser now runs on your Raspberry Pi, bringing interesting tools such as Rewind (returning to the first page of a site), Opera-like Speed Dial, and a 'no-UI' mode for distraction-free surfing.

Vivaldi CEO Jon von Tetzchner tells us why the firm wanted to support the Pi: "We love the Raspberry Pi! We believe we can contribute positively to the Pi community by providing our browser for the platform."

Asked whether he sees the more speed-focused features as being more useful for Pi users, Jon replies: "The Pi is used for a lot of different things." If you use your Pi as a standard PC, having a choice of powerful browsers is handy, but Jon also claims that if you use you Pi "as a media player or a games console… having a fully featured browser in there is a perfect fit."

You can download Vivaldi from **vivaldi.com** to give it a try.

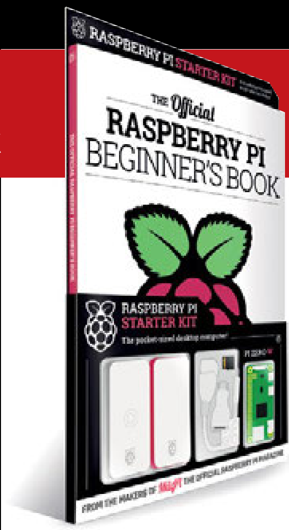


Vivaldi is a fully featured web browser for your Pi

NOW TRENDING

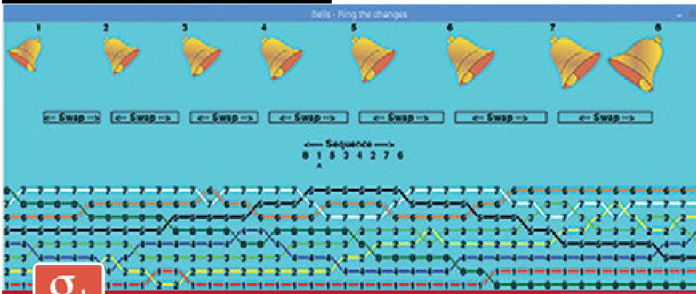
The stories we shared that flew around the world

THE OFFICIAL RASPBERRY PI BEGINNER'S BOOK



magpi.cc/2mf52Kk

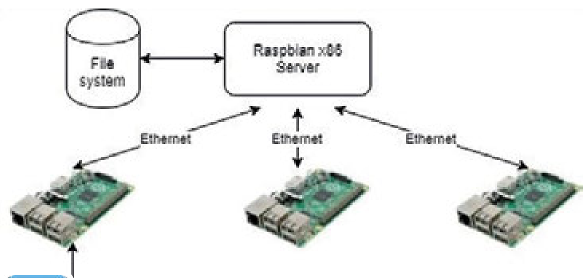
We've released a 116-page guide to getting started with a Raspberry Pi, and as a bonus it comes with a Raspberry Pi Zero W mounted on the front! The book costs £24.99, from **store.rpipress.cc**.



BUILD A BELL-RINGING SIMULATOR

magpi.cc/2lFjlaL

Virtual campanologists rejoice, as Mike Cook's comprehensive guide on building a bell-ringing simulator is available to all. Just head to the link above to practise your (virtual) pulling technique without annoying the entire neighbourhood!



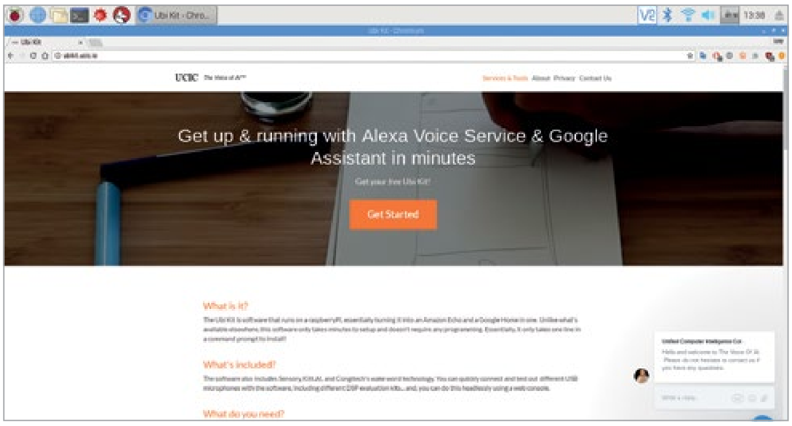
PISERVER MAKES DIGITAL LEARNING EVEN EASIER

magpi.cc/2mhWq5J

A new feature of Raspbian Desktop for x86, PiServer enables you to create and manage a network of Raspberry Pi boards from a single laptop or PC. This makes it easier than ever to demonstrate code and techniques to groups, whether that's an ICT class or a Code Club.

FREE VOICE ASSISTANT DEV KIT

Ubi Kit delivers Alexa and Google Assistant to your Raspberry Pi



Right Ubi Kit combines development for Amazon's Alexa and Google Assistant into one easy-to-use kit

UCIC has released the Ubi Kit development kit, which incorporates both Amazon's Alexa Voice Service (AVS) and Google Assistant into one easy-to-install package. Just fire up your Pi, enter a single install command, and get developing.

The Ubi Kit includes the Sensory embedded speech recognition SDK, Kitt.AI for natural language processing, and Congitech's wake-word technology. You only need to add speakers and a USB microphone (anything that runs on the Pi should work).

Once you've installed Ubi Kit, you can access the Ubi Kit console via a web browser. This means that you can "change WiFi, AVS credentials, and sign in without needing to code or command line," according to UCIC.

While developing, you might be glad that you can set the trigger action to the web button or the Pi's GPIO.

Head to **ubikit.ucic.io** to sign up for your kit.

THE 3000-CORE PI CLUSTER

Add chips and USB ports to your quick builds

Working with the Los Alamos National Laboratory, BitScope has created a Raspberry Pi Cluster of 750 Raspberry Pi 3s in a single box (**magpi.cc/2mh1uqV**). Admittedly, that box is a 35U rack server case, but the computing density is "five to ten times more dense than anything before" according to BitScope CEO Bruce Tulloch.

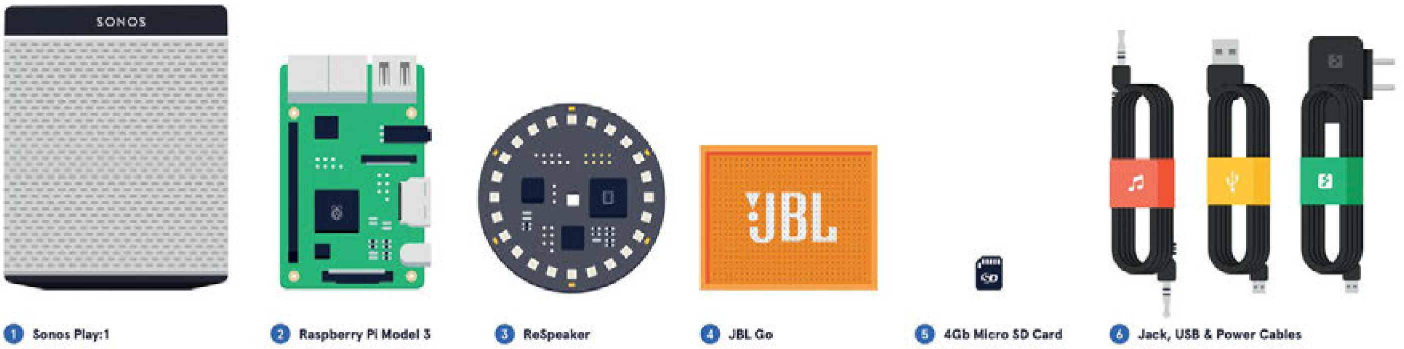
Bruce reveals that LANL works with "clusters of 20 000+ nodes capable of doing the sort of thing that would otherwise require millions of normal PCs to achieve."

LANL's challenge to BitScope was to build a 3000-core 'pilot cluster' as "a test bed for Los Alamos researchers to use to develop their own next-generation computers," Bruce explains.

However, it's "non-trivial to get 750 Raspberry Pis (or any type of computer, really) to work reliably at such high density," continues Bruce: the problems are "power, mounting, and cooling." Bruce confirms that the 750-node Clusters project "needs less than 4kW at full tilt", over ten times lower than a conventional air-cooled setup.

You can learn more about BitScope Pi Clusters by visiting **cluster.bitscope.com**. And don't miss our cluster computing feature this month, starting on page 64.

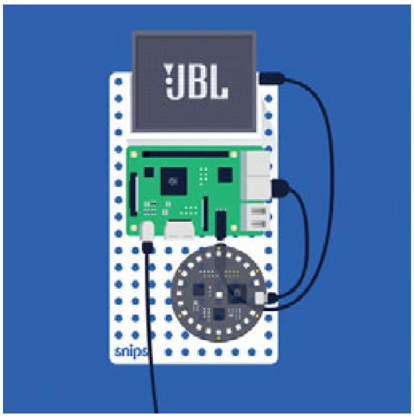
Below Packing 750 Pi 3s into a 35U server rack requires a lot of power and cooling
Credit: BitScope



SNIPS:

A NEW OPEN-SOURCE VOICE ASSISTANT

Voice assistant runs locally to protect your privacy



Left If you want to build voice devices that process voice commands locally, then take a good look at Snips

Snips (**snips.ai**) is a new offline voice assistant that processes commands locally on your Raspberry Pi. "We realised that the main issue of the next decades was the way people and machines interact", says Snips founder and CEO Dr Rand Hindi. "The more devices we want to use, the more effort it takes to use technology… Our mission at Snips [is] to put an AI assistant in every device in order to make technology disappear."

Privacy matters

Snips' offline nature is a response to the concerns of the server-side processing of most digital assistants: "we have come to believe that adding AI to our everyday devices necessarily means sacrificing our privacy, but we are challenging this point of view… Big servers are no longer needed – on-device voice recognition actually works!"

Snips published test results of how its assistant performs in comparison to other voice services. In its own test, Snips was either comparably accurate or better (**magpi.cc/2CUU7zG**).

Training and specialisation

Snips can run offline on hardware because each assistant is built with a specific task in mind. As Rand puts it, "a coffee machine should be good at understanding various types of coffee, but does not have to know movie names."

Snips also uses "very efficient libraries, like Tensorflow, and a very efficient programming language, like Rust," Rand clarifies. Even so, you'll need a Pi 3 to run the full Snips platform.

pi-top

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

POWERED BY RASPBERRY PI

OCR Oxford Cambridge and RSA

The modular laptop with sliding keyboard

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

pi-top

Colors
Raspberry Pi 3 optional

AWESOME INVENTOR'S KIT INCLUDED

20+ projects to explore

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

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

The modular desktop

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

pi-topCEED

Colors
Raspberry Pi 3 optional

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

pi-topPROTO

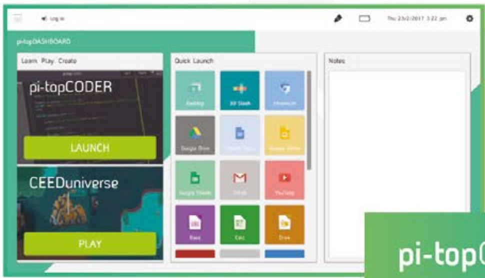
pi-topSPEAKER

pi-topPULSE

Modular **Accessories**

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

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



pi-topOS

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

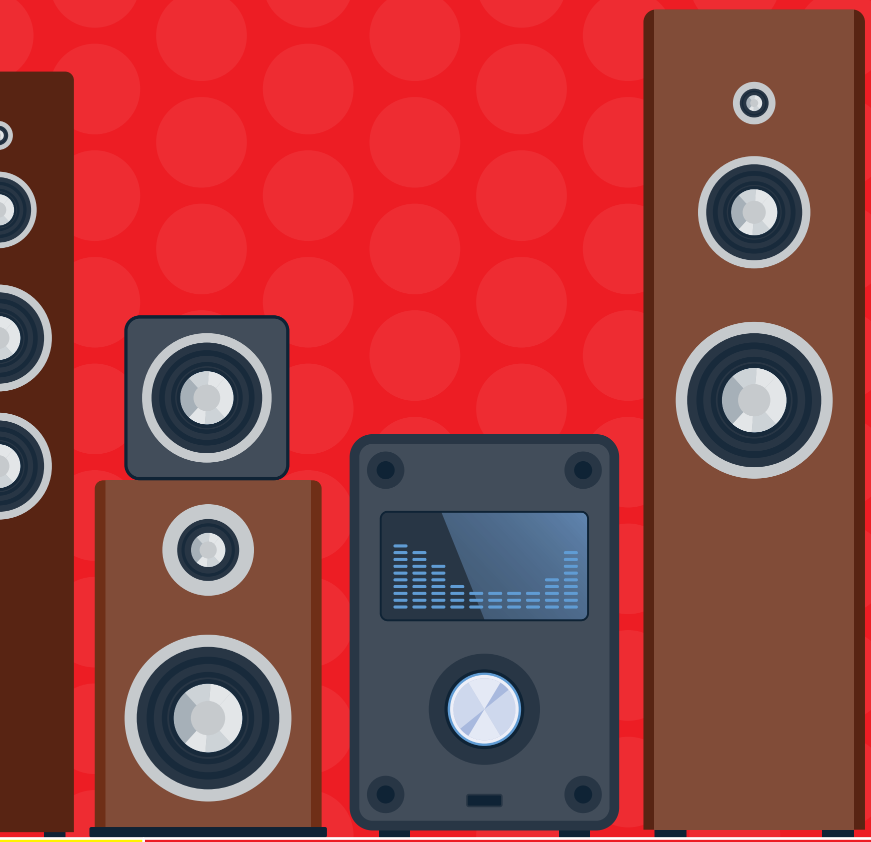
CEED**universe**

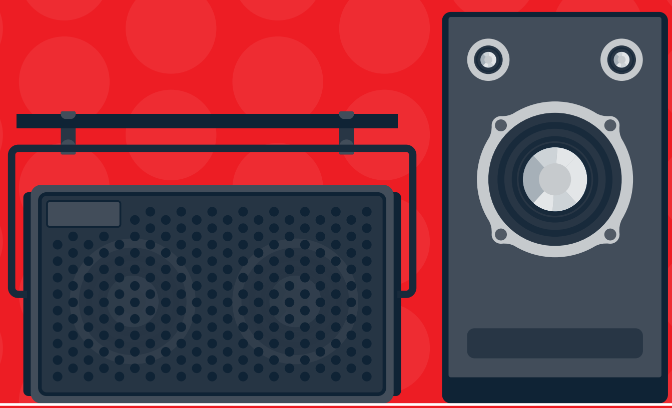
Learn programming concepts through our minigames, for example, learn problem decomposition by solving visual programming puzzles.

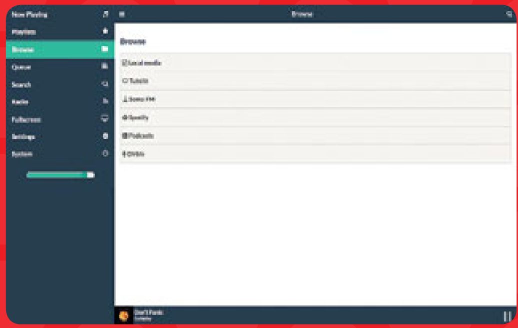
Media Player PROJECTS

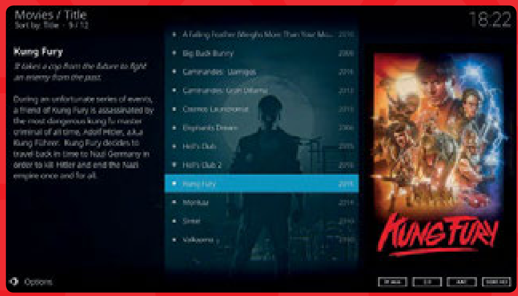
Power all your home media with Raspberry Pi

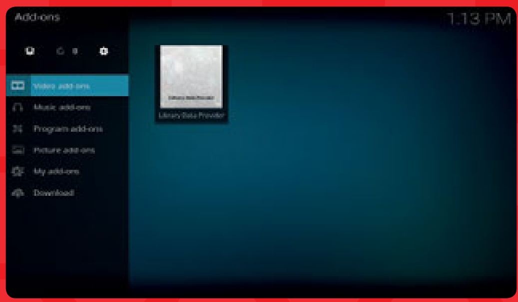
The Raspberry Pi is incredibly versatile. While a lot of people use theirs for fun electronics projects or to learn how to code, many folks use them in a more practical way, such as a media PC for their TV. The media capabilities of a Raspberry Pi are powerful and varied, though, so we thought we'd show you how you could replace (or upgrade) almost all of your media devices with some simple Raspberry Pi projects. Let's enter a new world… of sound and vision.











BUILD A RASPBERRY PI *Media Server*

Keep all your media in one place so every device on your network can access it

YOU'LL NEED

RASPBERRY PI:
Any model will do, but we recommend a Pi 2 or 3

OPERATING SYSTEM:
We're doing this on Raspbian, but it will work on any other Pi OS

OTHER HARDWARE:

PORTABLE USB HARD DRIVE
As big as you can afford

WIFI DONGLE (OPTIONAL)
Only for Pis without wireless LAN

ETHERNET (OPTIONAL)
We recommend using a wired connection if you can

If you're like us, you have several computers in your house, with music and videos and everything spread between them. Some (very) smart folks will have figured out how to use cloud storage to sync it all. However, our solution ended up keeping everything on a centralised computer: a file server. Using a Raspberry Pi for this is perfect as well, due to its size and low-power consumption.

>STEP-01
Location, location, location

Where you put your file server is very important. As well as access to power, you need to make sure it has a good connection to your router. In some houses, this might mean putting it right next to your router. Job done. In other houses, you may not have that luxury. Pick a few places you'd be happy to put it and think whether one spot makes more sense over the others. If all else fails, download a WiFi analyser app for your phone and test each spot to see which one might be the best.

>STEP-02
Basic setup

Before you put your Pi file server in its final destination, hook it all up to a monitor, keyboard, and mouse so you can configure it. Make sure Raspbian is up-to-date, that you're connected to the internet, and attach the USB hard drive you plan to use. If you need to format it, make sure to format it as NTFS. Finally, create a folder on your system where you'd like all your files to live. For this tutorial, let's just create a folder called **Share** in the home folder.

>STEP-03

Configure your Raspberry Pi

There's a few things you need to do before setting up the Pi to share files over the network. First of all, you need to make sure the Pi will automatically connect to the external hard drive after boot. To do so, open the **fstab** file with:

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

Add the following as one complete line to the bottom of the file and then save and exit:

```
/dev/sda1 /home/pi/Share
ntfs-3g rw,default 0 0
```

>STEP-04

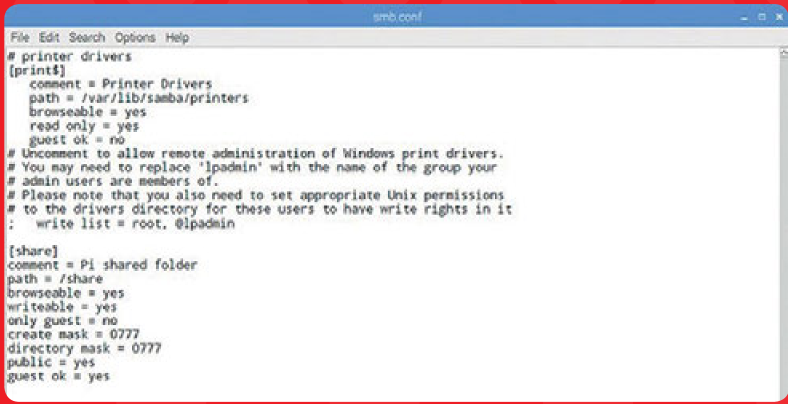
Configuring Samba

Samba is the name of the software that lets you easily share files over a network. On the Raspberry Pi, you need to install it with:

```
sudo apt-get install samba
samba-common-bin
```

You then need to edit the configuration file so it knows where to look. Open it with:

```
sudo nano /etc/samba/smb.
conf
```


smb.conf

File Edit Search Options Help

```
# printer drivers
[print$]
   comment = Printer Drivers
   path = /var/lib/samba/printers
   browseable = yes
   read only = yes
   guest ok = no
# Uncomment to allow remote administration of Windows print drivers.
# You may need to replace 'lpadmin' with the name of the group your
# admin users are members of.
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
;   write list = root, @lpadmin

[share]
comment = Pi shared folder
path = /share
browseable = yes
writeable = yes
only guest = no
create mask = 0777
directory mask = 0777
public = yes
guest ok = yes
```

And add the following to the bottom of the file:

```
[Pi share]
comment = Pi shared folder
path = /home/pi/Share
browseable = yes
writeable = Yes
only guest = no
create mask = 0777
directory mask = 0777
public = yes
guest ok = yes
```

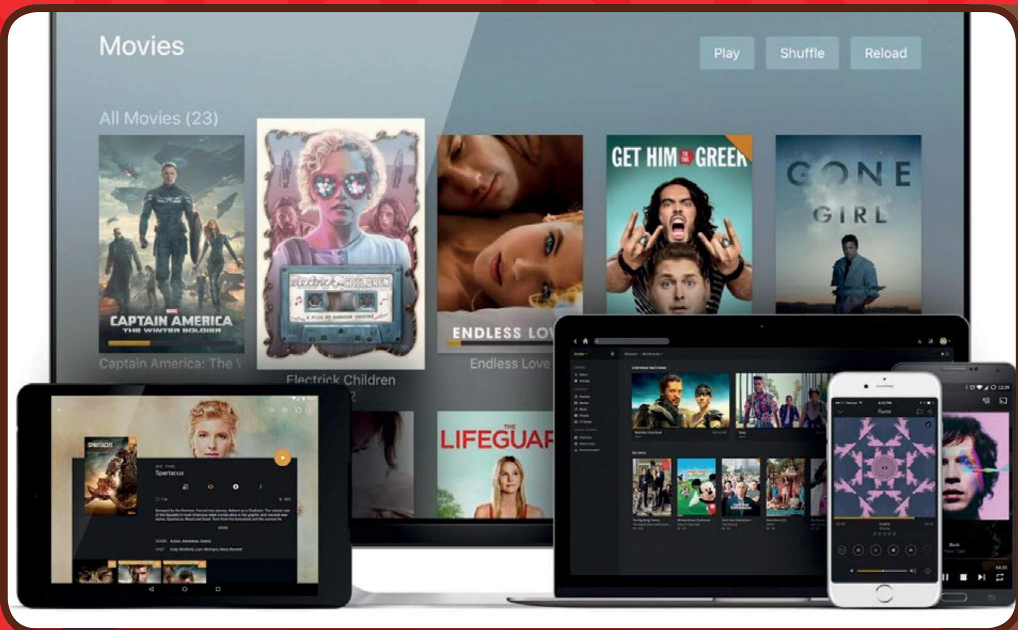
Save and exit. Finally, reset the Samba password with:

```
sudo smbpasswd -a [password]
```

Then restart it with:

```
sudo /etc/init.d/samba restart
```

And you're done!



PLEX ON PI

Plex is a great piece of software for creating a media server as it also includes some interesting online stream options for your media. Setting it up on Pi is not much more difficult than the simple setup we have here – check out this great tutorial on how to do it: **magpi.cc/2m1psGA**.

CREATE A Music STREAMER

Play music from your media server or online services like Spotify

“Music is the soundtrack of our lives,” said famed American music-man Dick Clark. When you think about it, well, of course it is, but it’s nice to have a well-known phrase to back it up. While some people might still listen to the radio, DJs much like the late Mr Clark decide on the music for you. With a Raspberry Pi, you can create your own custom radio that plays the music you want to listen all the time.

YOU’LL NEED

RASPBERRY PI:
Raspberry Pi 3 is best due to its built-in wireless LAN and easy audio options

OPERATING SYSTEM:
Pi MusicBox is the perfect OS for streaming music online and from your media servers

OTHER HARDWARE:

SPEAKER
You can either use one that connects via the Pi’s headphone socket (e.g. magpi.cc/2CZ7sH6) or get a Bluetooth speaker

A CASE
Much easier to carry around

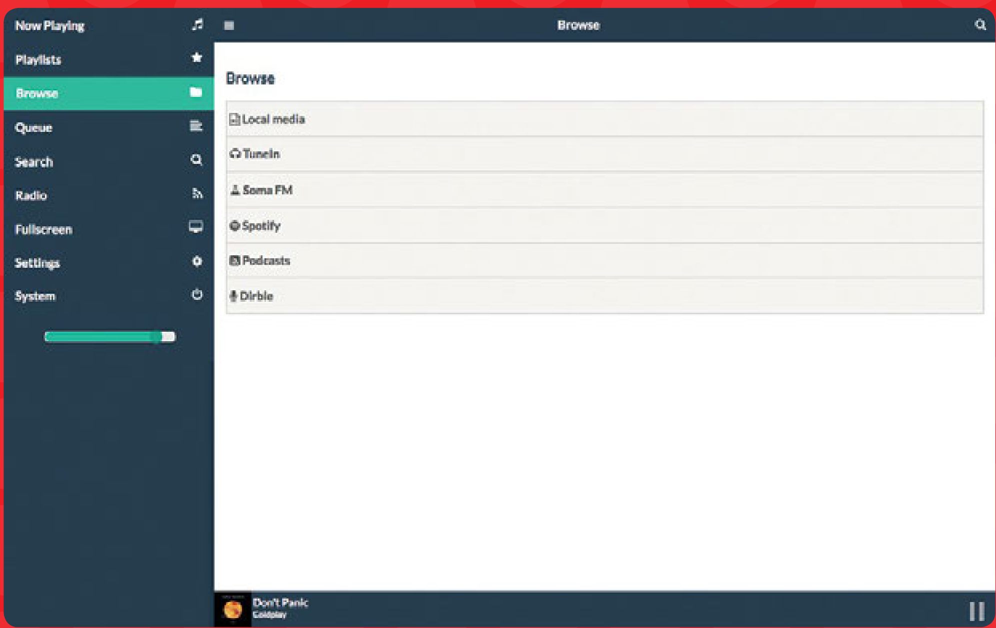
PORTABLE BATTERY (OPTIONAL)
If you move around a lot, this will make carrying it easier

>STEP-01
Make a music box

This is a simple setup really: Raspberry Pi 3 in its case with the speaker attached to it. You can Blu-Tack it to the top of the case if you want a quick and dirty way of carrying it around the house with you, along with a portable mobile battery.

>STEP-02
Install Pi MusicBox

Head to **pimusicbox.com** and download the SD card image. You’ll



need to manually install it using Etcher, and we have a tutorial on that here: **magpi.cc/2fZkyJD**.

Once that's done, you need to do some configuring of the files on the SD card before plugging it into your Raspberry Pi. First of all, navigate to the **config** folder on the SD card and open the **settings.ini** file. From there, look for the Network Settings section and add the name of your wireless network along with the password.

>STEP-03

Configure your music box

Plug the SD card into your music Pi and turn it on. Give it a moment to boot up and then access it on a browser on your PC or smartphone by heading to **musicbox.local**.

Here you can access the play controls, which are very wide-ranging. It will accept web streams, AirPlay, and even lets you set something to automatically start playing when you turn it on. You'll need to manually add the Samba share you created on your file server (something like **\\fileserver\Share** depending on how you named your file server Pi).

>STEP-04

Connect to Spotify and more

You can connect online services to your music box by heading to the settings menu and scrolling down to the services menu. These will guide you through how to set up Spotify, SoundCloud, Google Music, and more. You can even connect to podcast streams as well. For some services, you'll need to have a paid account, though.

Once set up, you'll find them in the Browse menu on the main interface and play what you want from there. Got a nice Synth Wave playlist on Spotify? No problem.

PIRATE RADIO – A BETTER MUSIC CASE

The Raspberry Pi has been out for many years, but there aren't many cases that include a speaker – believe us, we had another very thorough look while writing this article! However, the Pirate Radio from Pimoroni comes very close. It's a bit more limited than Pi MusicBox, but it will connect to Spotify. You can find the whole kit here: **magpi.cc/2nSEOfE**

BUILD A Home Theatre PC

Connect a Pi to your TV and play videos, music, and even show photos on your big screen

Ah Kodi. It's one of our favourite pieces of software, and has been paired with the Raspberry Pi to create wonderful media PCs since the Pi came out. You can get it in various flavours these days, even some Pi-specific ones, but we like the standard version. Here's how to set it up on your TV.

YOU'LL NEED

RASPBERRY PI:
A Raspberry Pi Zero W is well suited for this, as it can be easily hidden behind your TV

OPERATING SYSTEM:
LibreELEC is our favourite way of using Kodi on Pi

OTHER HARDWARE:

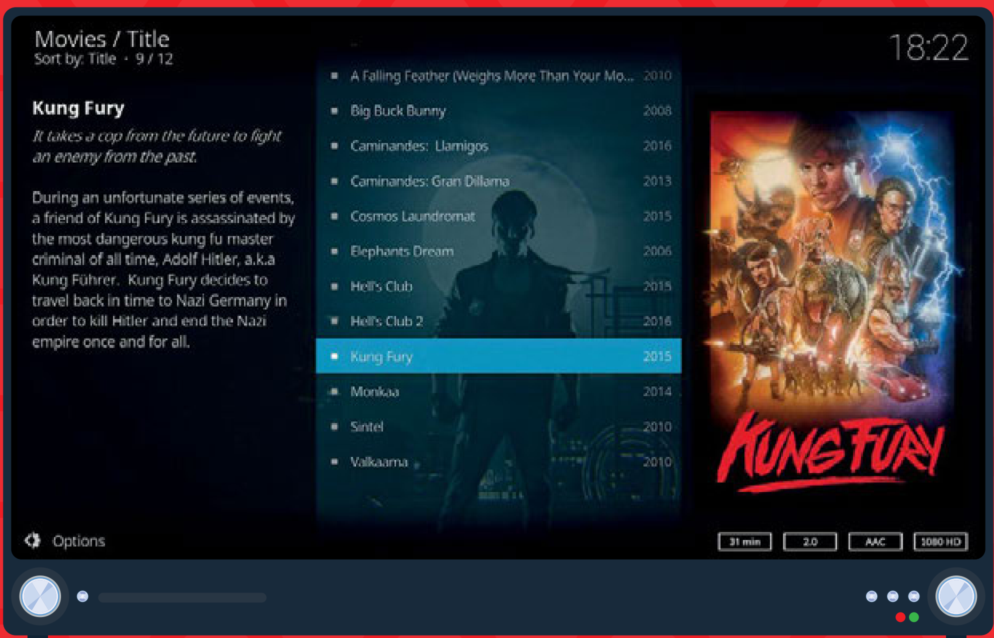
UNIVERSAL REMOTE AND IR RECEIVER
Although you can easily control Kodi via your smartphone

ADAPTER CABLES (OPTIONAL)
If you're using the Pi Zero W, you might need these

>STEP-01

Hook it up

Placement is important. With a Pi Zero W you can hide it behind your TV easily, but you still need to at least power it and connect it to your TV. If your TV has a USB port, you can always try powering the Pi Zero W via it; however, it may not supply enough power



and you'd have to make sure to do a software shutdown of the Pi before turning the telly off. Otherwise we recommend an official Pi power supply and also to invest in a mini HDMI to HDMI cable to cut down on adapters. If you plan to use your smartphone as a remote, you won't need a USB adapter.

>STEP-02

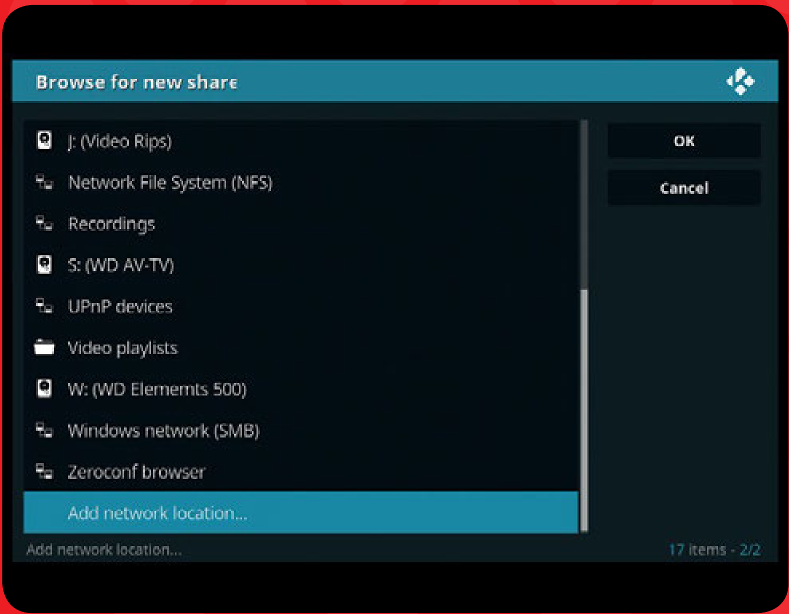
Prepare an SD card

Freshly format a microSD card. If you're not super-confident in using Etcher to burn your SD card, put NOOBS on there (**magpi.cc/2bnf5XF**) and install LibreELEC from the menu. Otherwise, head to the LibreELEC website and download the latest image for your Raspberry Pi (**libreelec.tv/downloads**). Follow our instructions on how to burn an SD card with Etcher (**magpi.cc/2fZkyJD**) and then plug it into your Pi.

>STEP-03

Configure Kodi

LibreELEC will boot up and after doing its own little quick configuration, it will then start and let you put in your details. For this first step, it can be useful to plug a keyboard into your Pi; however, the on-screen keyboard works fine with a remote control.



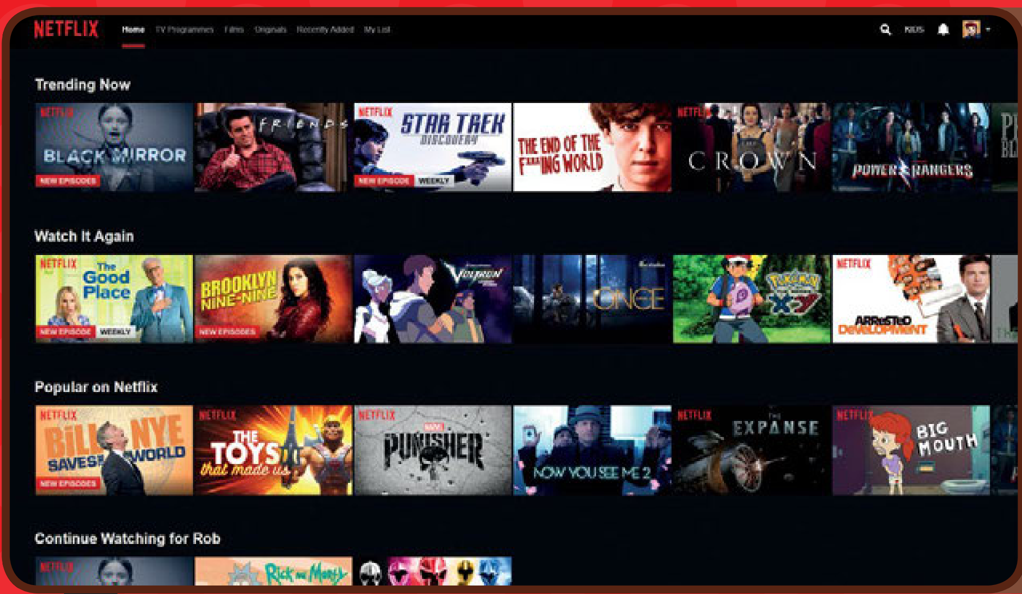
> "If you're not super-confident in using Etcher to burn your SD card, put NOOBS on there"

Connect to the WiFi, set your other preferences, and you're ready to go. Quick and easy.

>STEP-04

Add a Samba share

You can add a Samba share from your media server in Kodi pretty easily. Go down to Videos at the bottom of the main Kodi interface and select Files. Click on 'Add videos' and then go to Browse. Scroll down the choice list to 'Windows network (SMB)' and open it. It should show a list of available shares, including your media server's shared folder. If not, go to 'Add network location' and manually add your media server Pi's network IP address to connect it. Press OK, and you'll be able to access it!



VIDEO STREAMING

At the moment, Kodi does not support video streaming services such as Netflix. It's generally a whole lot better than your average smart TV is at playing video files, though, so pairing a Kodi box with a TV that can play Netflix itself will give you the best of both worlds.

MAKE A Live Photo Frame

Step into the future with this high-tech photo frame

One of those classic science-fiction inventions, like video chat and communicator watches, live photo frames that scroll through photos and play video clips have existed in the minds of authors for a long time. Live frames have been around to buy for many years now, but with the Raspberry Pi you can make an even better version that's much cheaper.

>STEP-01

Another Kodi box

Follow our instructions on the previous pages on how to install Kodi on a Raspberry Pi. Don't plug it into your Pi yet to do the setup, though.

While the SD card is burning, make sure your **Share** folder on your file server Pi has a specific folder for photos, or even one especially created for the photo

YOU'LL NEED

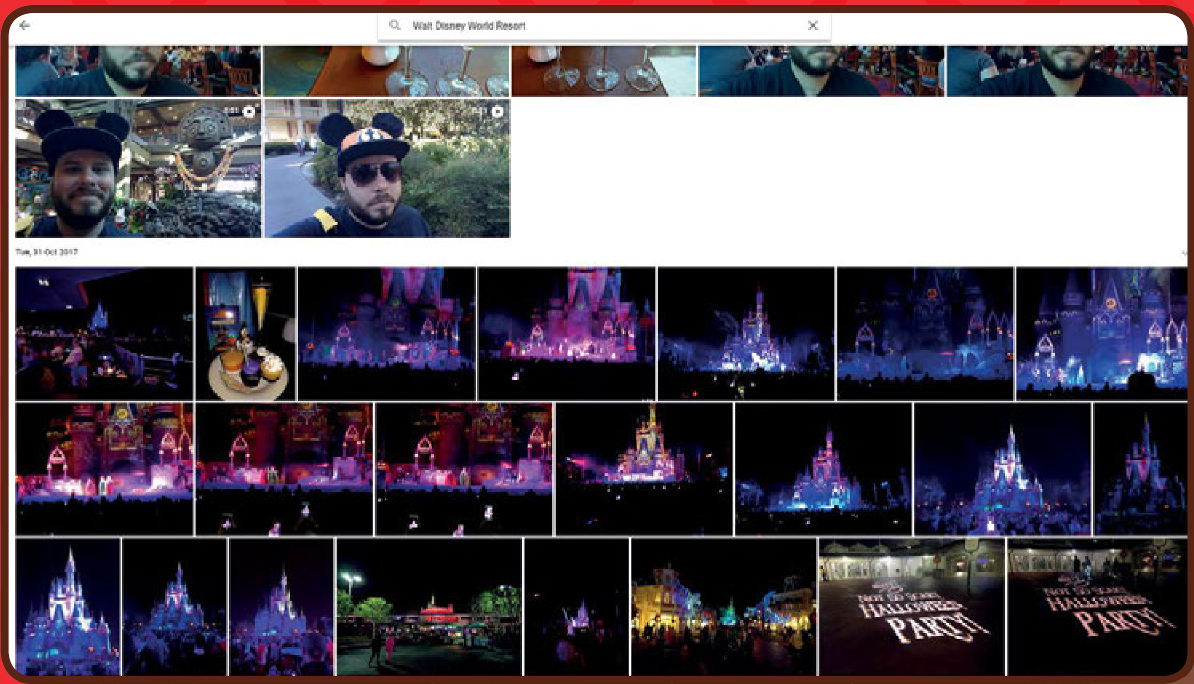
RASPBERRY PI:
You'll need a full-sized Raspberry Pi for this. You could even use an old Model A you have lying around!

OPERATING SYSTEM:
LibreELEC. The media player software from our HTPC? Yup!

OTHER HARDWARE:

RASPBERRY PI SCREEN
A bigger one, say 7-inch, so the photos look nice. The official screen is a good fit for this: magpi.cc/2mftBGX

STAND/CASE
The good thing about the official screen is that there are plenty of cases for it: magpi.cc/2meas8n



frame. Get the photos and such copied into there that you want display on the screen.

>STEP-02
Frame setup

If you want to do some fancy woodworking to create a more traditional frame, now is your chance. We'll assume you know what you're doing there, so good luck, don't cut yourself. For everyone else, connect up your screen to the Pi and install them both in the case. Find a place where you want to keep the live photo frame; as usual, you'll need there to be a power socket nearby so you can turn it on. Once you've got your spot, insert the SD card and turn it on.

>STEP-03
A new configuration

You may need to plug in a keyboard for this part. Do the Kodi setup as usual, making sure to call it a different name to your other Kodi box and other network-connected Pis on your network.

To make it act as a photo frame, we need to do two things. First of all, similar to how we added a video repository folder in the previous tutorial, we need to add a photo repository to the photo section. This should be the specific folder we created in step 2. You can very simply start a slideshow from here once the pictures are added, but let's go a step further and make it do it automatically on boot.

>STEP-04
Screensaver

The trick to this project is using one of the Kodi add-ons, called Multi Slideshow Screensaver. To install it, go to Add-ons > Download > 'Look and feel', and find the Screensaver option. From there, you'll see Multi Slideshow Screensaver. You can then edit the configuration of the screensaver to come on quickly, and to use the source folder we created in step 2 and added in step 3.

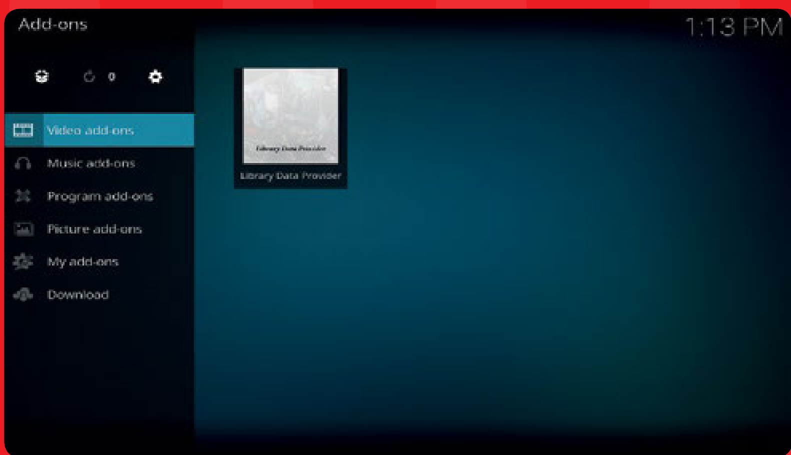


PHOTO STREAMING

Not everyone keeps images on a computer or hard drive any more, especially with both Apple and Google having their own highly integrated cloud photo services these days. Some people have made their own photo frames that stream from these services – check out one of these methods here: magpi.cc/2mhwJSI

Now you can turn off your photo frame and easily get it back to displaying pictures with just the flick of a switch! You could also then use the touchscreen functionality to watch some video with it in the kitchen – it's your choice (but you should definitely do that).

CREATE A Voice Control System

Control your house with your own voice using Alexa and Pi

Alexa is great. While you can buy an Alexa device outright, wouldn't it be fun if you could build your own? Well, you can, as Alexa is available for Raspberry Pi, meaning you can install a little Alexa device into a robot with a Pi Zero W, or control your home with a custom AlexaPi. Let's do the latter.

YOU'LL NEED

RASPBERRY PI:
We recommend a Raspberry Pi 3, as you'll need the wireless LAN, headphone jack, and USB ports

OPERATING SYSTEM:
Raspbian, although we'll modify it to be AlexaPi

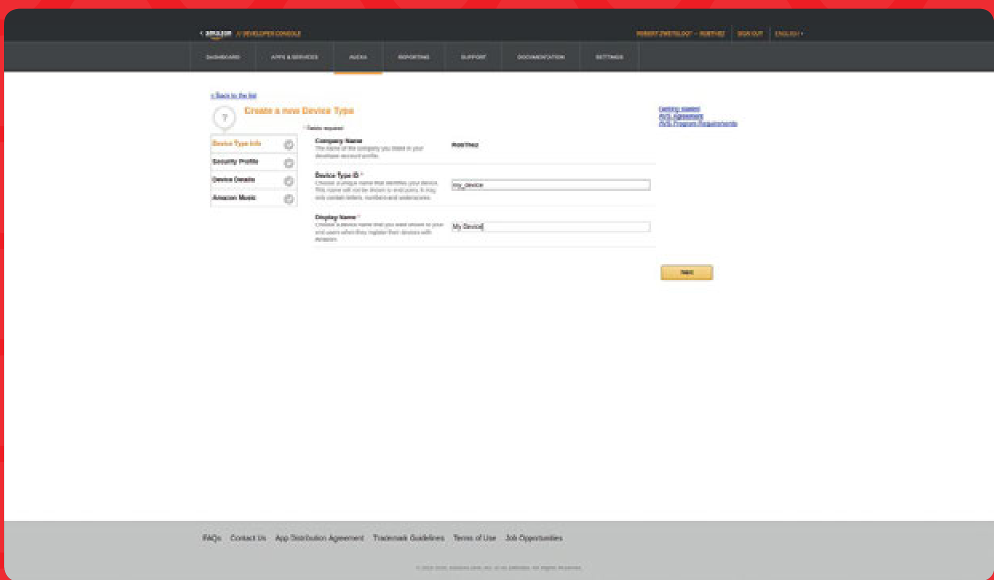
OTHER HARDWARE:

A USB MICROPHONE
So we can talk to Alexa

A SPEAKER
So we can hear her response. Use one like in the music box

>STEP-01
Basic hardware

Get a fresh install of Raspbian on an SD card and connect your Pi up to monitor, keyboard, and mouse. Make sure to attach the speaker and microphone as well. Boot it up to get the first-time configuration out of the way and connect to the WiFi. Once you're ready, open up



the Terminal and find out your Pi's network IP using **ifconfig**. It should look something like 192.168.1.25 – make a note of it and move back to a regular computer.

>STEP-02
Become an Amazon developer

You need to get some extra info to set up your Alexa. Go to **developer.amazon.com**, log in, and go to Alexa > Your Alexa dashboards > ALEXA > Alexa Voice Service. Once there, you'll see 'Register a Product Type' – from there, click on Device.

Under the Device Type Info tab, give your device a name in the Device Type ID and Display Name fields. Click Next to get to Security Profile and, from the drop-down menu, choose 'Create a new profile'. Fill out name and description however you wish and click Next again.

Under Web, you want to add the following…

Allowed Origins:
http://localhost:5050 and
http://[Pi IP address]:5050

Allowed Return URLs:
http://localhost:5050/code and
http://[Pi IP address]/code

Finally, back at the Devices pages, click Manage next to your new device and go to Capabilities. Enable timers and change the Cards option to 'Cards with text only'. Click Update. You're ready.

>STEP-03
Install AlexaPi

Go back to your Raspberry Pi and open up the Terminal again (or SSH in if you prefer). Move to the **opt** folder with:

```
cd /opt
```

Install git with the following so we can get the AlexaPi files:

```
sudo apt-get install git
```

The AlexaPi files can easily be downloaded now using the command:

```
sudo git clone https://github.com/alexa-pi/AlexaPi.git
```

Finally, run the setup script and follow the instructions with:

```
sudo ./AlexaPi/src/scripts/setup.sh
```

Having problems post-install?

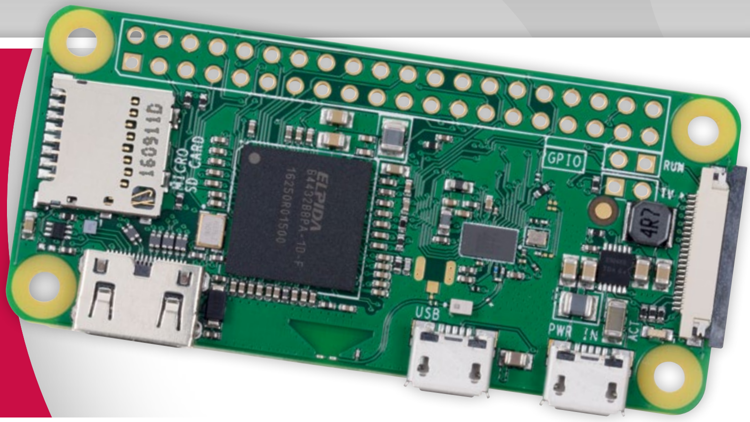
Want to do advanced configuration? Check out the documents for AlexaPi on its GitHub page here: **magpi.cc/2kiyOxO**

>STEP-04
Place your AlexaPi

Once the setup is complete, turn off your Pi and disconnect the monitor, keyboard, mouse, and anything else you don't need for Alexa. Place it in its new home and give it a test run – you may need to edit some of the config files to your liking. Once it's working how you want it to, you can start customising it and adding your own custom commands and functions. Perhaps you can get it to control all your new media devices – the choice is yours!

SUBSCRIBE TODAY AND RECEIVE A

FREE PI ZERO W



PLUS OFFICIAL PI ZERO CASE WITH 3 COVERS

AND FREE CAMERA MODULE CONNECTOR AND USB / HDMI CONVERTER CABLES

Subscribe in print for 12 months today and receive:

- A free Pi Zero W (the latest model)
- Free Pi Zero W case with three covers
- Free Camera Module connector
- Free USB and HDMI converter cables

Other benefits:

- Save up to 25% on the price
- Free delivery to your door
- Exclusive Pi offers and discounts
- Get every issue first (before stores)

Pricing

Get six issues:

£30 (UK)

£45 (EU)

$69 (USA)

£50 (Rest of World)

Subscribe for a year:

£55 (UK)

£80 (EU)

$129 (USA)

£90 (Rest of World)

FREE PI ZERO W!

Get three issues:

£12.99 (UK) (Direct Debit)

$37.50 (US) (quarterly)

How to subscribe:

- magpi.cc/Subs-2 (UK / ROW)
- imsnews.com/magpi (USA)
- Call +44(0)1202 586848 (UK/ROW)
- Call 800 428 3003 (USA)

Available on the App Store

GET IT ON Google Play

SUBSCRIPTION FORM

YES! I'd like to subscribe to *The MagPi* magazine and save money

This subscription is: ☐ For me ☐ A gift for someone*

Mag#66

YOUR DETAILS Mr ☐ Mrs ☐ Miss ☐ Ms ☐

First name .... Surname ....

Address ....

....

Postcode .... Email ....

Daytime phone .... Mobile ....

**If giving The MagPi as a gift, please complete both your own details (above) and the recipient's (below).*

GIFT RECIPIENT'S DETAILS ONLY Mr ☐ Mrs ☐ Miss ☐ Ms ☐

First name .... Surname ....

Address ....

Postcode .... Email ....

PAYMENT OPTIONS

1 DIRECT DEBIT PAYMENT £12.99 every 3 issues (UK only)

Instruction to your bank or building society to pay by Direct Debit

DIRECT Debit

Please fill in the form and send to:
The MagPi, Select Publisher Services Ltd,
PO Box 6337, Bournemouth BH1 9EH

Service user number 8 3 8 7 7 3

Name and full postal address of your bank or building society:

To: The Manager Bank/building society ....

Address ....

....

.... Postcode ....

Name(s) of account holder(s) ....

Branch sort code ☐☐☐☐☐☐ Account number ☐☐☐☐☐☐☐☐

Reference ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ (Official use only)

Instruction to your bank or building society

Please pay Select Publisher Services Ltd Direct Debits from the account detailed in this instruction subject to the safeguards assured by the Direct Debit Guarantee. I understand that this instruction may remain with Select Publisher Services Ltd and, if so, details will be passed electronically to my bank/building society.

Signature .... Date ☐☐/☐☐/☐☐

Banks and building societies may not accept Direct Debit instructions for some types of account.

SUBSCRIPTION PRICING WHEN PAYING BY CHEQUE OR CREDIT/DEBIT CARD

6 ISSUES	☐ UK £30	☐ Europe £45	☐ Rest of World £50
12 ISSUES	☐ UK £55	☐ Europe £80	☐ Rest of World £90

2 CHEQUE

I enclose a cheque for .... (made payable to Select Publisher Services Ltd)

3 CREDIT/DEBIT CARD ☐ Visa ☐ MasterCard ☐ Maestro ☐ Switch

Card number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Expiry date ☐☐ ☐☐ Valid from ☐☐ ☐☐ (if shown)

Issue number ☐☐ (if shown) Security number ☐☐☐
(last 3 digits on the back of the card)

Signature .... Date ☐☐/☐☐/☐☐

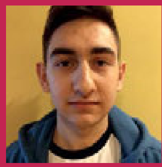
I would like my subscription to begin from issue .... (month + year)

RETURN THIS FORM TO:
MagPi Magazine Subscriptions, Select Publisher Services Ltd, PO Box 6337, Bournemouth BH1 9EH

☐ Please tick this box if you DO NOT want to receive any other information from Select Publisher Services Ltd.

☐ Please tick this box if you DO NOT want to receive any other information from other companies.

☐ Please tick this box if you DO NOT want to subscribe to The MagPi newsletter.



EMANUELE COLETTA

Emanuele began experimenting with robots aged ten and Pis aged eleven. Now 15, he codes in Java and he's learning Python…

emamaker.altervista.org

3D PAC ROBOT MAN

This version of a much-loved classic arcade game brings the past to life like never before

Quick Facts

- The play board is 4 m² in size
- Each robot measures 9 cm tall
- Three fathers, four sons, and an uncle made the project
- Soldering many tiny SMD components proved tricky
- Find a copy of Pac-Man Vs. to try it yourself

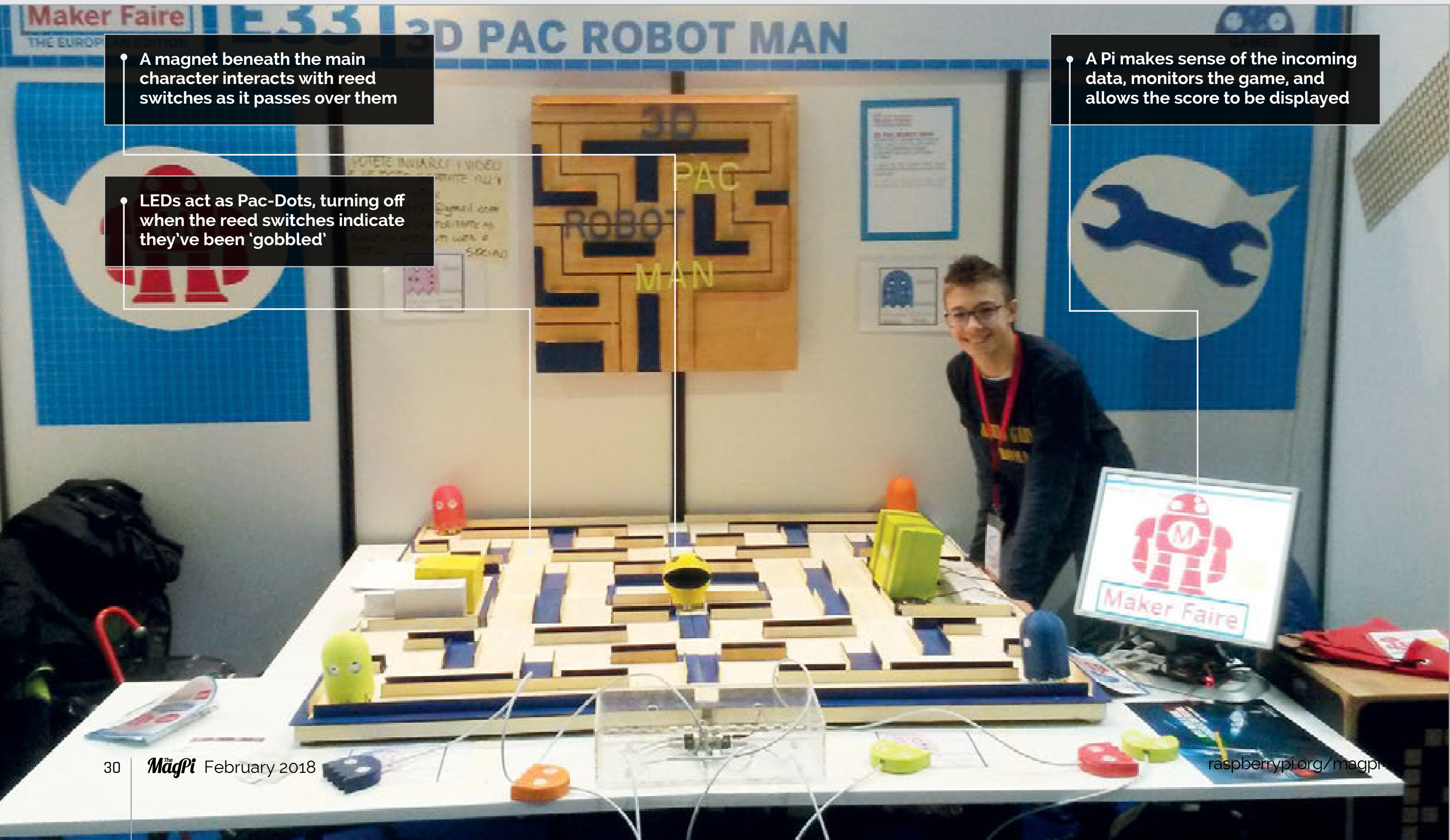
When video games began to flourish in the seventies and eighties, 15-year-old Emanuele Coletta had not even been born. But when he and his pals caught up with the titles they had missed thanks to their parents' collective love of retro gaming, they were soon champing at the bit to produce something of their own.

The result was a 'real-life' version of Pac-Man, Namco's classic 1980 pellet-guzzling arcade game. They created 3D-printed robot renditions of the main character and the familiar four ghosts and they replaced the Pac-Dots of the original game's maze with lights that turn off when the yellow chomper moves over them.

"The idea was to make something funny that no one had made before to let us learn and apply new technologies," explains Emanuele. But at the same time, they added their own little twist.

In the single-player video game, Pac-Man has to eat up all of the Pac-Dots and avoid the ghosts, each of which moves automatically. In what became 3D Pac Robot Man, however, four players control the ghosts. "The aim is for the main character to escape from the others without getting caught, but for the others to try and catch it," says Emanuele. Suddenly their Pac-Man game became a five-player romp.

Initially, the friends – along with their dads – concentrated on making the playing board out of wood, laser-cutting the various pieces to form the maze. They then attached lots of small boards

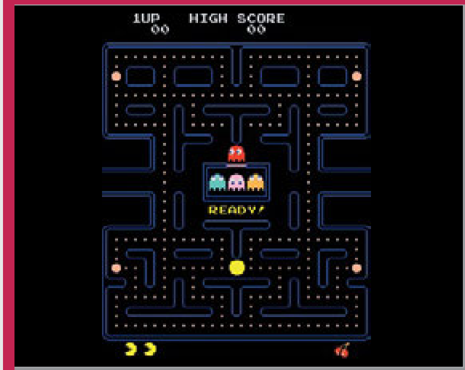


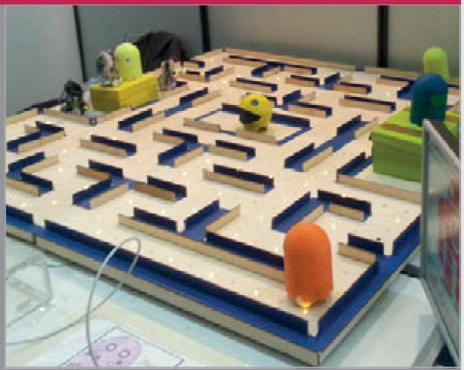
A magnet beneath the main character interacts with reed switches as it passes over them

LEDs act as Pac-Dots, turning off when the reed switches indicate they've been 'gobbled'

A Pi makes sense of the incoming data, monitors the game, and allows the score to be displayed

BRINGING PAC-MAN TO LIFE





>STEP-01
Getting inspiration

Pac-Man is one of gaming's most recognised icons and the game involves eating dots, avoiding ghosts (Blinky, Pinky, Inky, and Clyde), and amassing points. It inspired Emanuele's father, Carlo, to create a real-life version.

>STEP-02
Being spot on

The initial prototype was a metre-square in size, but the aim was the same: to produce a challenging real-life game. The boards containing the LEDs had to be precisely laid out underneath.

>STEP-03
Invite others along

Since 3D Pac Robot Man was designed to be multiplayer, players also control the ghosts (unlike in Pac-Man where they move automatically). The 3D-printed controllers, made from laser-cut acrylic, show what's being controlled.

containing LEDs and reed switches beneath the gaming field and connected them to an Arduino Mini. "Assembling the LED boards – which we made ourselves – and connecting them with bigger boards underneath the field was the easiest but longest part," Emanuele recalls.

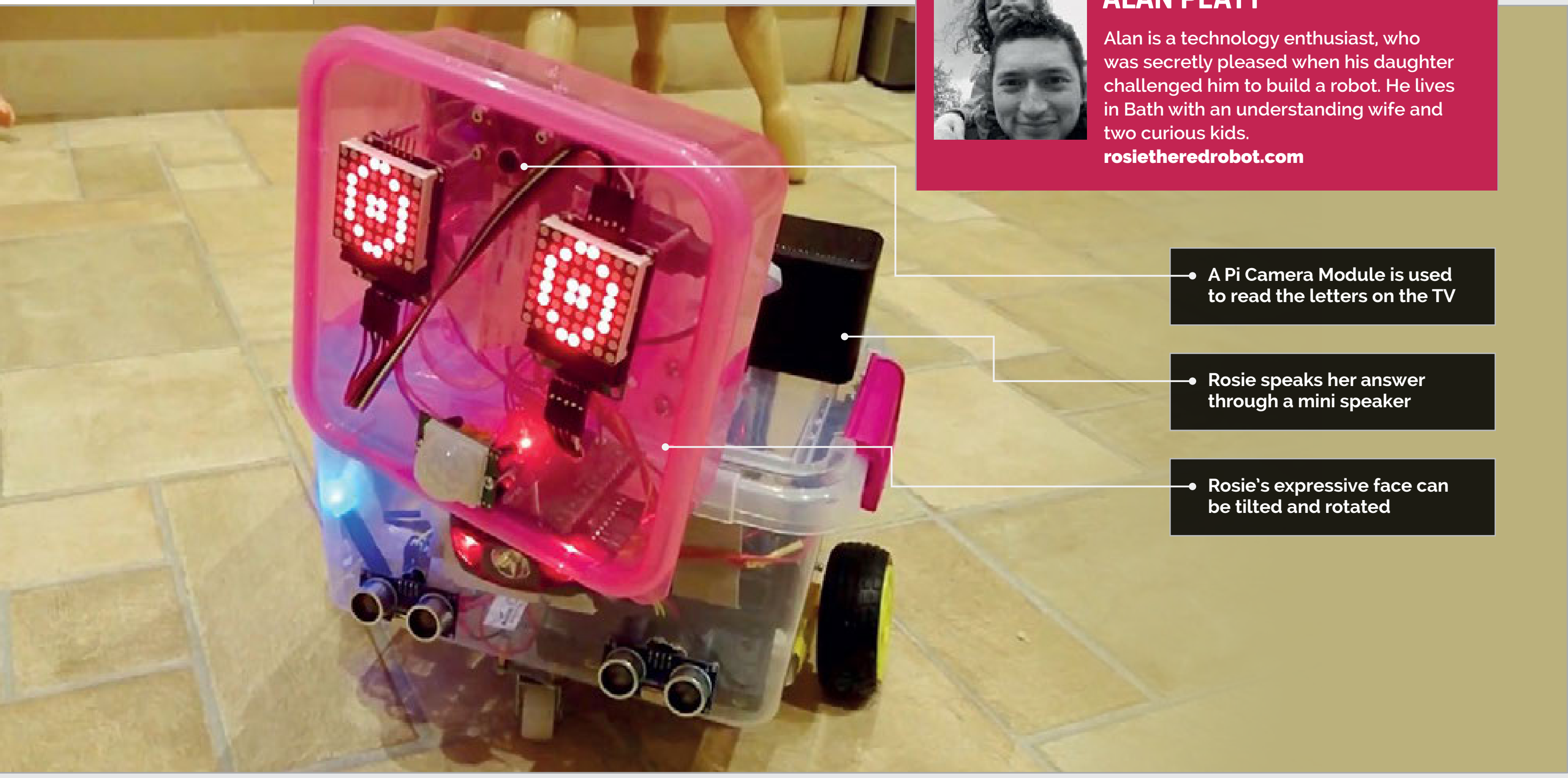
With that in place, they could turn their attention to the characters – the shells and bases of which were designed by the team and 3D printed. Each one contains an Arduino Uno board. "There is also a magnet under the main character," says Emanuele. By connecting the robots to 3D-printed joysticks containing a board created by one of the team and an Arduino Nano, they could be moved around the maze. "The joystick boards communicate with the robots through radio frequencies at 2.4GHz," Emanuele continues.

It is at this point that the Raspberry Pi comes into play. When the main character moves, the Arduino Mini understands which reed switch is activated and it can figure which LED to turn off. Points are awarded for each light that is 'eaten' and this information, along with details about the game state, is passed to the Pi.

"I wrote a little application in Java which allows people to see the score, the high score and the game state on a monitor," Emanuele tells us, having used an open-source library called RXTX and the Arduino Playground tutorial to establish the serial connection between the Arduino and Pi. "It also plays the original sounds of the game to give the whole thing a sense of being real and to make people understand what we made."

It was then a matter of encouraging players to get stuck in, with the build unveiled at Maker Faire Rome. "If the main character gets caught, the catcher wins, but if the main character escapes then it wins." As expected, the game went down a storm. "People seemed to like our game because they could challenge each other," Emanuele enthuses. "It was beautiful to see people enjoying their time with something we had made."

Above The game went down a storm when it was unveiled at Maker Faire Rome, held between 14 and 16 October 2017



ALAN PEATY

Alan is a technology enthusiast, who was secretly pleased when his daughter challenged him to build a robot. He lives in Bath with an understanding wife and two curious kids.

rosietheredrobot.com

- A Pi Camera Module is used to read the letters on the TV
- Rosie speaks her answer through a mini speaker
- Rosie's expressive face can be tilted and rotated

ROSIE THE RED ROBOT

Equipped with a Pi brain and several senses, Rosie can even play Countdown

Quick Facts

- OCR is done via Google Cloud Vision API
- Alan's Python code is on his blog
- Rosie can't solve the numbers round yet
- She uses two Pi 3 brains for some tasks
- Rosie is set to have a 3D-printed makeover

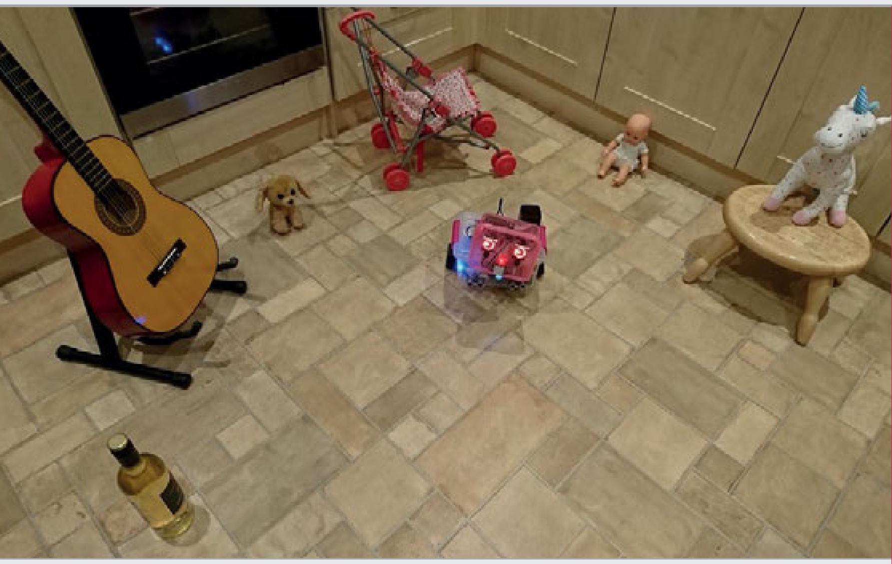
The clock is ticking as the contestants concentrate hard on finding the longest word possible from nine randomly selected letters. Yes, it's *Countdown*, the classic TV game show loved by students, senior citizens… and robots. Playing along at home is Rosie, her Camera Module eye focused on the screen. Within seconds she blurts out, "I have a six-letter answer: murder."

Rosie first came to life when Alan Peaty was challenged by his daughter to build a robot. Despite his engineering experience, Alan didn't find it easy. "It didn't take long before I was completely out of my comfort zone," he tells us. "Possibly the most challenging aspect of trying to build a robot is that there are so many different areas of technology involved, from hardware to software, mechanics to electronics. For example, just to get Rosie's neck to move, we had to write some code in Python, learn about Serial Peripheral Interface bus (SPI), and get familiar with servo motors and the physics of torque, even before thinking about connecting the whole thing together. Be prepared to Google absolutely everything and anything."

As documented in detail on the **rosietheredrobot.com** blog, Alan has spent several months building Rosie and gradually adding extra hardware and abilities, while learning a lot along the way. At first Rosie was a simple wheeled robot, but she is now equipped with a headlight, LED matrix expressive 'eyes', motion sensor, GPS receiver, ultrasonic distance sensors, and a Camera Module.

Playing the game

It's this last addition that enables Rosie to see the Countdown letters on the TV screen. Optical character recognition is then carried out using Google Cloud Vision API, to read the letters. "She performs relatively well at different distances on one condition – only the individual letters from the letters round is showing



on-screen when the photo is captured," reveals Alan. "When the contestants' names, or even the channel name, appear in the same picture, it starts to muddle Rosie's brain."

A Python application runs the identified letters through an algorithm that matches possible results from a dictionary containing 10 000 words – a dataset from **magpi.cc/2mgxuv4** that Alan has arranged alphabetically. Python's gTTS text-to-speech

Above In a bid to become more human, Rosie can also identify household objects and offer instant unfounded opinions on them!

"It perhaps goes to show that machines are so much better than us humans at processing large volumes of data rapidly, when the requirements of the task are crystal clear," says Alan. "The letters round doesn't really require any tactics or strategy, or even any complex decision-making. You simply need to remember

> It turns out to be a little more complicated than just trawling through a list of words and letters

module is then used to get Rosie to say the longest word she's found, along with the numbers of letters in it, via an on-board speaker.

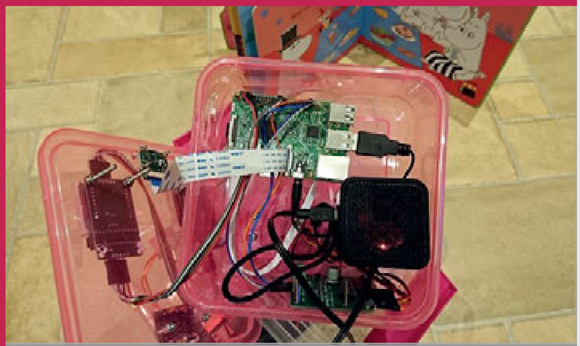
Naturally, replacing the dictionary with a larger one would improve Rosie's performance, but she's still a decent player.

Above Rosie likes to wrap up warm for wintry weather; we're not sure if she appreciates red wine

every single word in the English dictionary and have the ability to filter and search through them in under 30 seconds."

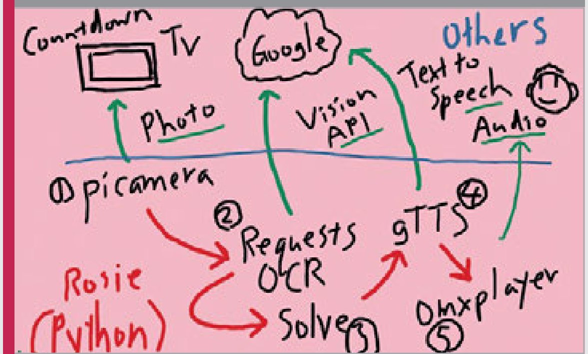
As for getting Rosie to play the *Countdown* numbers game, "We've had a quick look at this," says Alan, "but it turns out to be a little more complicated than just trawling through a list of words and letters. The algorithm is likely to need to work out every single mathematical operation possible not only between the chosen numbers, but between the results of subsequent operations. She would need to add this to her repertoire, however, if she was ever to appear on the actual TV show…"

BUILD A COUNTDOWN BOT



>STEP-01
Inside Rosie's head

Rosie's Raspberry Pi 3 brain is contained in the pink plastic tub used for her head, along with a camera mounted on the lid. Her speech is played through a mini USB speaker.



>STEP-02
The brainstorm

Alan planned out how it would work: a photo is taken of the letters on TV, OCR identifies them, the Python algorithm finds the best word, then it's read out via gTTS text-to-speech.

>STEP-03
Play Countdown

Rosie sits up close to the screen to watch Countdown and put Alan's theory into practice. Within seconds she speaks the longest word she's found in her 10 000-word dictionary.

BEN CROSTON

Ben has an MEng in Systems Engineering and he worked at BAE Systems for ten years. He is now head brewer at Fuzzy Duck Brewery.
wyre-it.co.uk/blog

Quick Facts

- Ben founded the Fuzzy Duck Brewery in 2006
- He began creating his Pi controller in 2012
- Ben also authored the RPi.GPIO Python library
- Irration Ale is brewed for the Pi Birthday Bash
- The name's a pun. Do you get it?

FUZZY DUCK BREWERY

Fancy a brew? A proper one, that is? Ben Croston uses a Pi to control the brewing process at Fuzzy Duck Brewery

Anyone who has ever splashed a coffee over an uncovered Raspberry Pi will be among the first to confirm that liquids and electronics don't generally mix very well. But when it comes to the harder stuff, Ben Croston is proving the two can actually go together rather perfectly.

That's because Ben is the head brewer at the Fuzzy Duck Brewery located in Poulton-le-Fylde, Lancashire. He makes use of a Raspberry Pi controller in the brewing process and his beers have become very well known over the last few years.

One of the most popular is Irration Ale, which he concocts for various Pi bashes. Tasting of raspberries, it is so-called because pi is known as an irrational number, but it goes down even better knowing the computer is involved in its making.

It all started in late 2011 before the first Pi even rolled off the production line. Ben, who began using Red Hat Linux in 1996 before moving to Ubuntu in 2004 and then Debian in 2009, wrote what became the RPi.GPIO Python module, thinking it would become useful in the future for controlling tasks in the brewery plant.

"It was the equivalent of my GCSE IT project in 1994, which was a relay switch-box that connected to the parallel printer port of a PC," he tells us. "The use of a Raspberry Pi in the brewery was the natural thing for me to do." By using the RPi.GPIO module and a DS18B20 temperature sensor which is accurate to within 0.5 degrees

Celsius over a range of -10°C to 85°C, Ben is able to achieve a high level of consistency in his brews.

"The choice of sensors and actuators was made more difficult due to the physical size of the plant and the fact that some parts have to be made to a food quality standard," Ben says. Thankfully, the rest of the process was reasonably straightforward.

Essentially, the sensor connects to a Raspberry Pi before being placed in the hot liquor tank. Data is then sent down the wire to a general-purpose I/O (or GPIO) pin. Since Ben's RPi.GPIO allows you to control the pins on a Raspberry Pi using the language, it is possible to then monitor the temperature. Ben does this using special-purpose software he created himself (the bulk of which is closed-source and commercially sensitive). Should the brew get too hot or cold, the heaters can then be adjusted.

"This is crucial because the key to commercial brewing is to be able to brew exactly the same beer all the time between batches," Ben explains. "This means being able to measure, record, and control physical attributes while the brew is taking place, and it is where automating as much as possible using a Raspberry Pi makes things a lot easier and consistent."

Above Ben's Irration Ale always goes down well at Raspberry Pi events

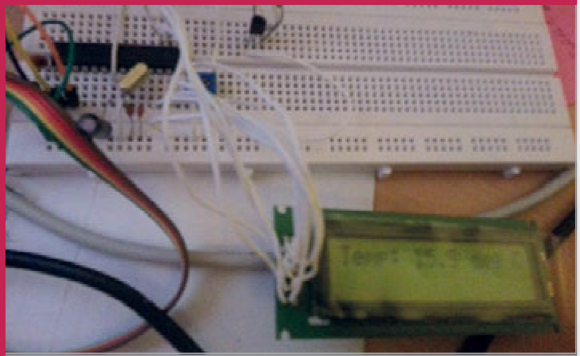
> "The use of a Raspberry Pi in the brewery was the natural thing for me to do"

Below The temperature can be controlled via a page that is viewable on a smartphone

In order to control the temperature, the brewery makes use of three methods – an electronic control panel, a webpage viewable on a smartphone, and a PyQt4 GUI application – all of which can be used at the same time. Fuzzy Duck also uses software running on a Pi for its invoicing and cask tracking. That software was written using Python 2.4 in 2007, well before the Pi's birth, however.

So has the Pi saved Ben lots of money? "I don't have a clue," he says. "I've never looked at buying any commercial brewery control equipment. It is much more fun to make things yourself and it's certainly easier to customise and maintain."

MAKING A GOOD BREW



>STEP-01
Heat up the water

With the Raspberry Pi set up and the sensor built into the insulated tank, the water can be heated up to the required temperature and kept under constant control.

>STEP-02
Mash in the beer

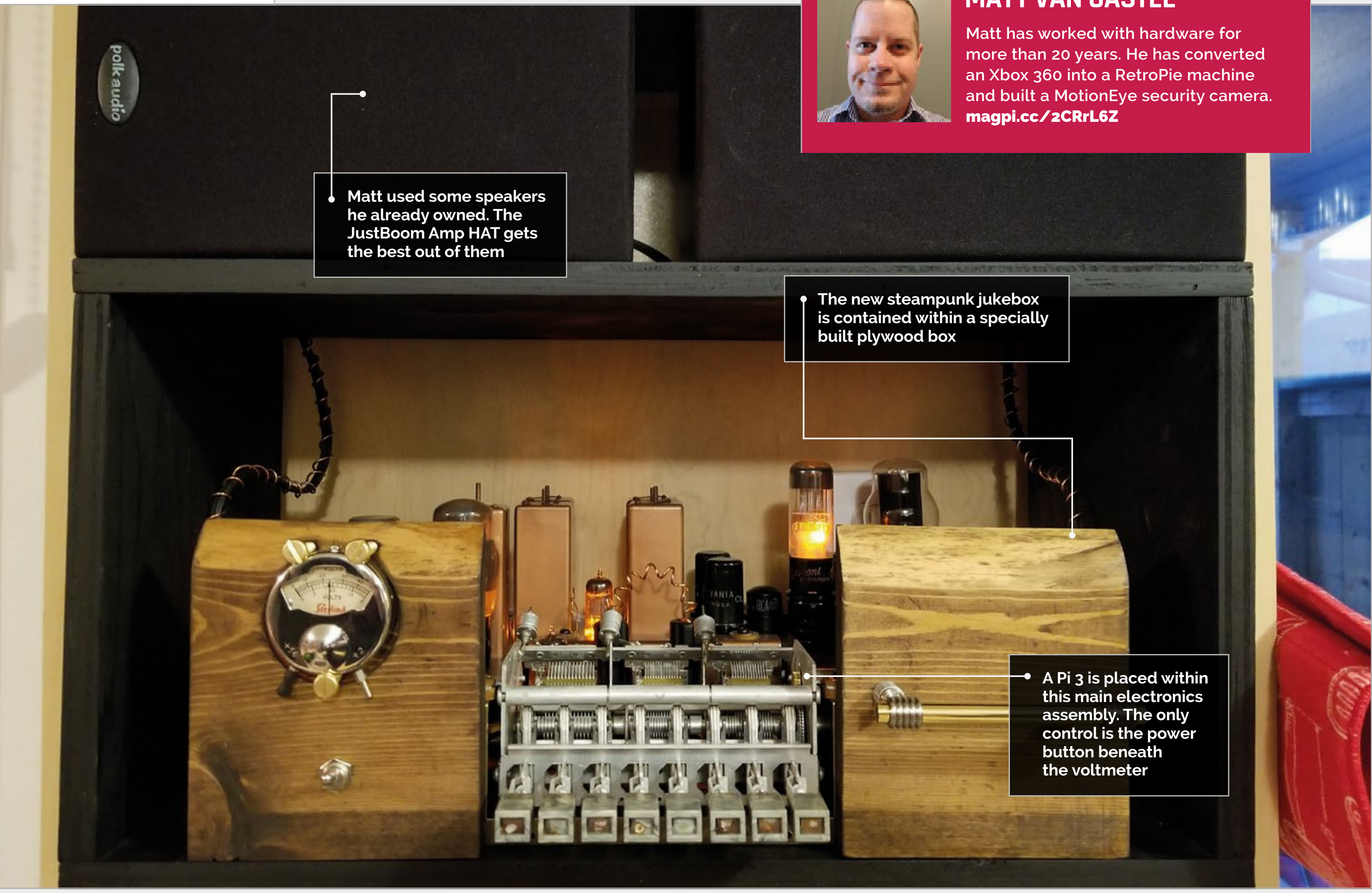
The mashing process begins. This combines a mix of grain with the water (or liquor as it is also known). During the heating, the enzymes in the malt break down the starch into sugars.

>STEP-03
Drink and relax

The malt sugar solution is boiled with hops, cooled, and yeast is added. Fermentation begins, releasing alcohol. Here, Irration Ale is produced – not too sweet or bitter and not excessively raspberry flavoured.

MATT VAN GASTEL

Matt has worked with hardware for more than 20 years. He has converted an Xbox 360 into a RetroPie machine and built a MotionEye security camera.
magpi.cc/2CRrL6Z



Matt used some speakers he already owned. The JustBoom Amp HAT gets the best out of them

The new steampunk jukebox is contained within a specially built plywood box

A Pi 3 is placed within this main electronics assembly. The only control is the power button beneath the voltmeter

Quick Facts

- Matt estimates the build could cost about £135
- He spent a weekend prototyping the electronics
- The volume is controlled via a smartphone
- Matt wants to add a volume control on the unit
- Originally the build was going to feature a screen

STEAMPUNK PI JUKEBOX

Matt Van Gastel's retro-futuristic music steaming device looks as good as it sounds

As technology has progressed, so too has the way that we listen to music. We have enjoyed our tunes on vinyl, cassette, CD, and even MiniDisc, among a plethora of formats. We've also used MP3 players and iPods and grown accustomed to music streaming.

Yet one method of listening to our favourite songs has remained strong throughout this time: the humble radio, a device that for many conjures images of cosy nights in twiddling AM and FM dials to discover new tunes.

So when we heard that Matt Van Gastel had turned the main assembly of a 1930s Westinghouse radio receiver into a Pi-powered modern player, it was music to our nostalgic ears. "I'd been carrying this old radio from house to house for years, fully intending to restore it," he says of the set which his great-grandparents had bought close to a century ago. "But after doing some research and determining that there was no real value in restoring it, I came up with the idea of using a Raspberry Pi to create a music server."

Matt, who has a background in radio and television, began the project by tearing out the innards from the old tube amplifier. "This wasn't difficult, but it was very messy and time-consuming," he says. It mostly involved cutting

Above The unit is primarily for show and Matt went to town on the design, adding these cool simulated vacuum tubes

away the various parts including the main electronics assembly. Once he had the assembly, he could turn his attention to his Raspberry Pi 3.

"The Pi 3 seemed like a natural choice because I already had a few of them waiting for a project," he explains. During his research, he came across the JustBoom Amp HAT which costs £60 and outputs at 55 W. Needing no soldering or external sound cards, and providing high-quality audio along with digital-to-analogue conversion, Matt felt it was perfect.

> "It's the aesthetics that people notice first"

Serving music

"I had looked at making an actual tube amplifier, but the cost was too high for the quality," Matt tells us. "But I found the JustBoom site and found this HAT fitted the bill because of its power output capabilities and extremely low noise floor and supply requirements."

With this in place, Matt then looked for flexible music server software and opted for Mopidy, which is written in Python and allows music to be played from a local disk or through streaming services such as Google Play Music, Spotify, and SoundCloud.

"Mopidy was not my first choice, but I was unable to get the Pi Music Box to integrate with Google," Matt recalls, citing his music service of choice. "Fortunately, Mopidy worked very well for what I was trying to do. I was able to use Google Music and I found several apps for running Mopidy headless."

Above The main assembly was stripped of its previous components. Matt was worried it could have contained asbestos. Luckily, it did not

Powering up

Matt prepared the case by washing and painting it. He then drilled holes in the original vacuum tubes and wired in amber LEDs to give a glowing, steampunk effect ("that's my favourite part," he says). As a neat touch, he wired in a voltage monitor using a 1950s DC voltmeter and placed an LED light behind it. The voltage meter jumps when the unit is turned on. "People comment on the way the needle bounces on the voltmeter upon power-up," he says.

The build uses power from a 24 V DC brick running at 2.5 A and it makes use of a pair of bookshelf speakers Matt says sound great because of the DAC converter in the JustBoom Amp HAT. "But it's the aesthetics that people notice first because the unit lives on a shelf in my kitchen. I've had lots of positive feedback."

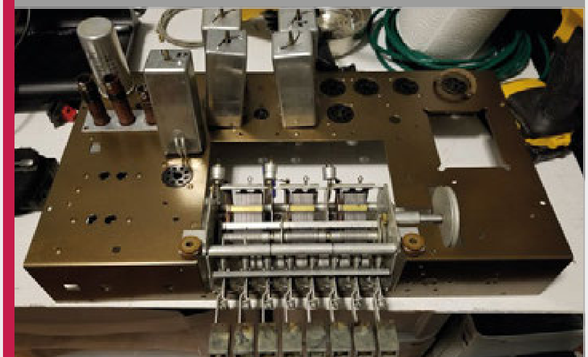
PLAY IT AGAIN MATT

>STEP-01
The original radio

As beautiful as the cabinet for this Westinghouse radio looks, Matt said there was little value in restoring it. Instead, he decided to produce a unique piece by basing his build on the main electronics assembly.

>STEP-02
Dusting it down

Matt needed to get down to the bare chassis. He did this by removing all of the old components, while wearing a respirator, gloves, and safety glasses. He then scrubbed the dust away and dried it thoroughly.



>STEP-03
Bringing up to scratch

He painted the main assembly with gold, brass, copper, and black. On top of this were fitted the simulated vacuum tubes. He could then build the case, inserting a Pi 3 with a JustBoom Amp HAT fitted.

WOLFRAM LANGUAGE

Start using the mathematics programming language that comes free with Raspbian

You'll Need

- Raspberry Pi
- Raspbian OS
- Wolfram Language

All computing problems are, at heart, maths problems. And you can solve many problems using a programming language like Wolfram.

A Raspberry Pi computer is an ideal companion for those looking to learn more about mathematics. Maths is all about problem-solving, and learning the language of mathematics can help you solve all kinds of real-world problems.

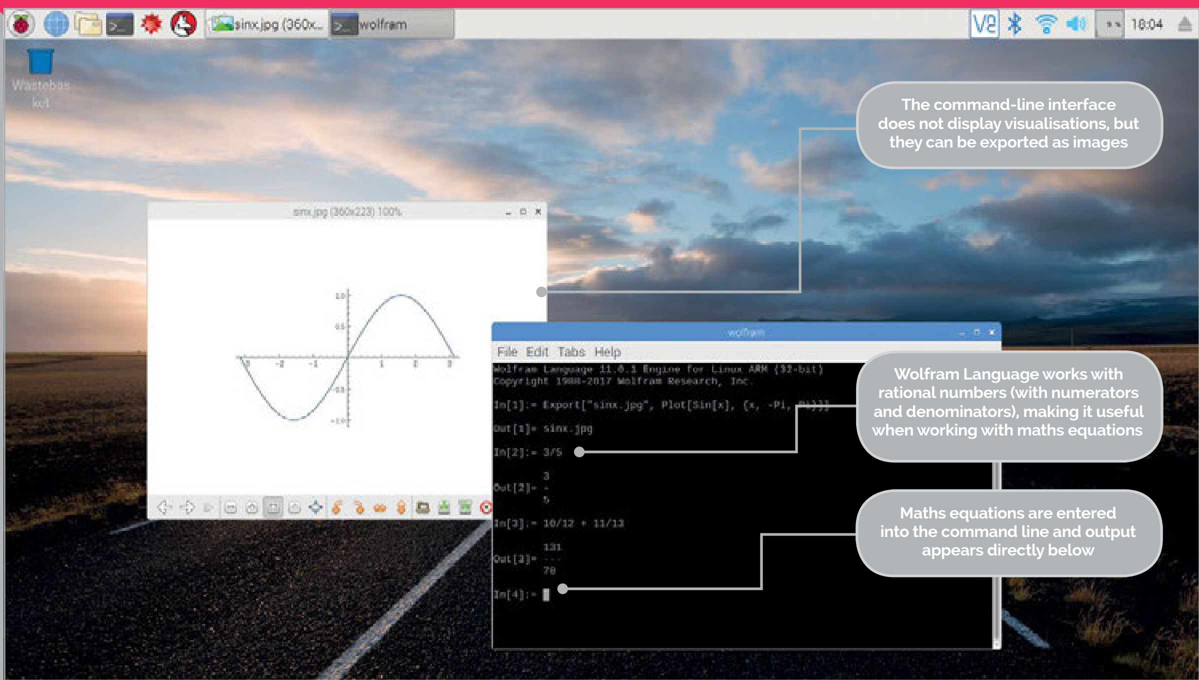
Computers take a lot of the grunt work out of maths, and help you focus on the underlying mechanics (rather than grinding at the problems). Or, if you are trying to learn maths techniques, you can use a computer to check your answers.

Thanks to a special deal between Raspberry Pi and Wolfram Research, the standard operating system Raspbian comes pre-installed with software called Wolfram Mathematica and Wolfram Language.

The great thing about Wolfram Language and Mathematica is that you can perform high-end data calculations that make use of Wolfram's Knowledge Base (which lets you tap into thousands of data points: things like global rainfall levels, mobile phone ownership, and sports data).

Mathematica's graphical interface can be confusing, so we think it's better to start with the Wolfram Language in the command line.

Wolfram Language has lots of advantages over using maths in other environments or languages, such as Python. Wolfram Language is deeply focused on maths. It is much easier for working with rational numbers and radians (which aren't converted to decimal and degrees by default).



In this tutorial we're going to cover some of the starting points for Wolfram Language on your Raspberry Pi. With it you'll be able to use Wolfram Language instead of a calculator, and use it to explore mathematical concepts.

Get started

Start by clicking on the Wolfram icon in the taskbar or via Menu > Programming > Wolfram.

This opens a terminal window displaying the following command prompt:

```
In[1]:=
```

You can also access this prompt from the command line by entering 'Wolfram'.

Performing simple maths is easy enough. Just enter your maths term followed by **ENTER**:

```
2 + 2
```

And you'll see:

```
Out[1]= 4

In[2]:=
```

You can now enter another command, such as:

```
2^100
```

And you'll see:

```
Out[2]=
1267650600228229401496703205376
```

This is in decimal notation, not scientific notation as you'd get in Python.

As with other programming languages, you can store items as variables and use them:

```
x = 2+2
y = 2^100
x * y
```

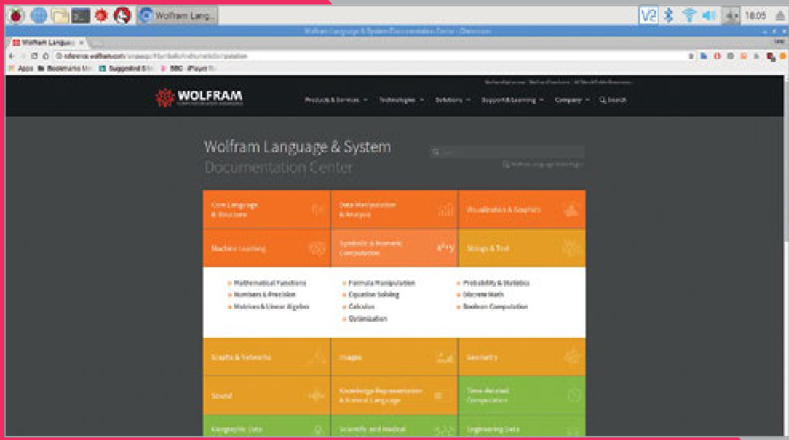
You can flip back and forth through previous input lines using the Up and Down arrows. You can also repeat previous output lines using the **Out[]** function with the line number. Such as:

```
Out[1]
Out[2]
Out[1] * Out[2]
```

Or use the **%** symbol:

```
%1
%2
%1 * %2
```

Using just **%** gets the last output.



Left Wolfram Research has a massive documentation library that outlines all of the functions and features available

Functions

Functions form a vital part of Mathematica. In fact, everything – including simple mathematical expressions such as 2+2 – is converted to a function. Functions always start with a capital letter, while input values are contained by square brackets:

```
Plus[2,2]
Times[2,10]
Power[2,100]
Divide[100,4]
```

Even setting variables is actually a function. And functions can be nested inside one another. These three commands all mean the same thing:

```
x = 2+2
Set[x,2+2]
Set[x,Plus[2,2]]
```

There is also a clear function, **Unset[x]**, if you want to get rid of variables.

This doesn't mean you're supposed to translate every expression into its functional equivalent. But some expressions just aren't that easy to do manually. So the **Sqrt[]** function is required:

```
Sqrt[16]
```

Getting rational

One of the real joys of using Wolfram is its handling of rational numbers. Enter **1/2** in Python and you'll get **0** or **0.5** (depending on whether you're using Python 2 or 3). Enter **1/2** in Wolfram Language and you'll get a satisfying

```
1
-
2
```

Rational numbers are also reduced in form to the smallest denominator. Enter **3/15** and you'll get:

```
1
-
5
```

The FullForm function for a rational number is **Rational[1,5]** if you want to use it. You can combine rationals with integers, such as:

```
2 * 1/5

2
-
5
```

If you'd rather get a number outputted as a decimal, the easiest way is to add a decimal point to one of the numbers in the rational.

```
2* 1./5
```

```
0.4
```

Alternatively, you can surround it in an **N** function.

```
N[1/3]
```

```
0.333333
```

By default, **N** returns numbers to six-digit precision. A second value in **N** can be used to adjust the precision:

```
N[1/3, 10]
```

```
0.3333333333
```

You can convert numbers to a nearby rational with the smallest denominator using the **Rationalize** function

```
Rationalize[0.5]
```

```
1
-
2
```

Lists

Now that you've got the hang of basic mathematical functions, you can start to use Wolfram Language as a programming tool.

Lists form a central part of Wolfram Language and can be used to group together items. Lists are placed between curly braces **{}** and can be created manually:

```
{1, 3, 5}
```

Or by using the **List** function:

```
List[1,3,5]
{1, 3, 5}
```

And you can generate lists using the **Range** command:

```
Range[5]
{1, 2, 3, 4, 5}
```

Use two numbers to provide a start and stop point:

```
Range[5,10]
{5, 6, 7, 8, 9, 10}
```

And three numbers to use a start, stop, and step:

```
Range[2,8,2]
{2, 4, 6, 8}
```

Visualisation and export

Wolfram Language has a host of tools for visualising data. **Plot** is a common one used to plot the output of functions. Generally it's best to switch to Mathematica if you want to get a visual view of your function, but it is possible to export graphics from the command line to an image using the **Export** function.

Take the **Plot** function which graphs the **Sin** function of x against a list of x values from -Pi to Pi.

```
Plot[Sin[x],{x, -Pi, Pi}]
```

In Mathematica you'll get a lovely visualisation, while in Wolfram Language you just get:

```
-Graphics-
```

Surround the function in an **Export** function with the first value a file name and the second the function:

```
Export["sinx.jpg",
Plot[Sin[x],{x, -Pi, Pi}]]
```

It will be saved in your **/home/pi** directory by default. To view the graph, open a Terminal and enter:

```
xdg-open sinx.jpg
```

Create functions

You can create your own functions inside Wolfram by entering a name for your function with square brackets. Then placing a variable placeholder, which is a letter followed by an underscore, such as **x_**, inside the brackets:

```
addtwo[x_]
```

User-generated functions are lower-case by convention. The name is followed by a colon and equals sign:

```
addtwo[x_]:=
```

After the equals sign we add the expression, which uses the same variable without the underscore:

```
addtwo[x_]:=Plus[x,2]
```

To use the function, we simply use its name and add variables into the brackets (as we would any regular function).

```
addtwo[2]
4
```

```
addtwo[4]
6
```

With all of these features and functions, and the ability to work with integers, rational, and real numbers (plus a wide range of other numbers), Wolfram Language is a powerful tool. We've only scratched the surface of what you can do with it here, and it is packed with Boolean operators and procedural processes (such as **for** and **while** loops).

But hopefully you've got enough here to find Wolfram Language less daunting. Visit **reference.wolfram.com** for the online documentation. Be sure to use Wolfram Language next time you're working with maths.

Documentation

You can find a list of all Wolfram functions, commands, and techniques in the Wolfram Language & System Documentation Center.
reference.wolfram.com/language

MIKE'S PI BAKERY

MIKE COOK

Veteran magazine author from the old days and writer of the Body Build series. Co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.
magpi.cc/259aT3X

EXPLORING FRACTAL MUSIC

Create fractals and change them into intriguing music patterns

You'll Need

- USB to MIDI interface cable
- MIDI sound generator

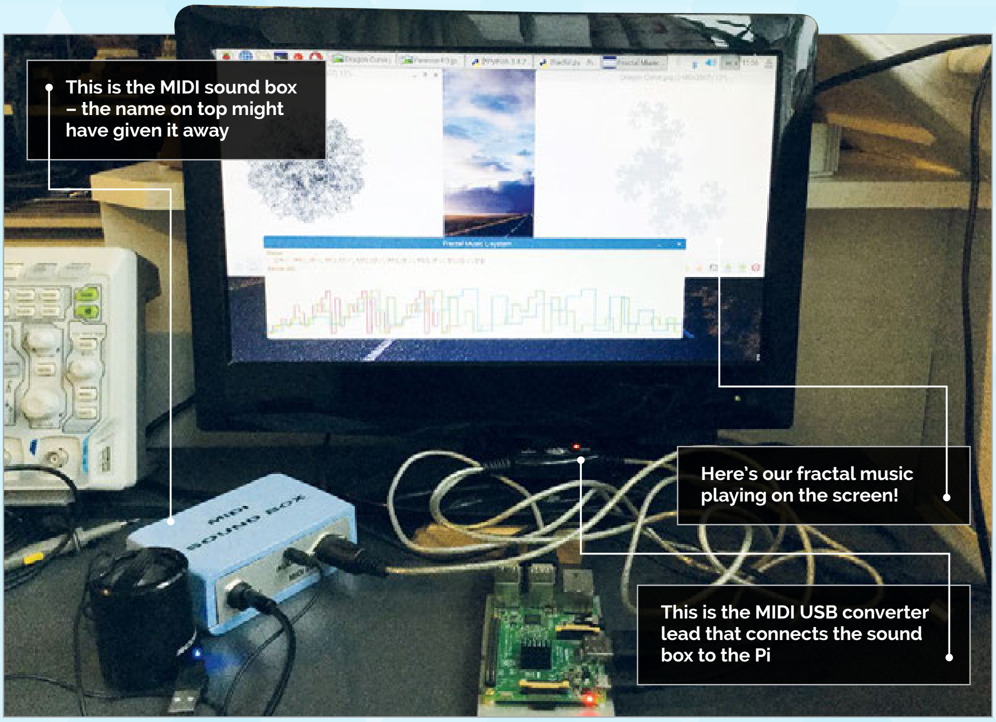
Fractals seem to have gone out of favour when it comes to computers, which is a pity because there are plenty of exciting things to explore with them, especially in the field of music. Most people think of a fractal as a complex curve and there a few pleasant-looking standard examples. The basic property of a fractal is that it is self-similar; that is, you see the same sort of pattern if looking at a very small magnified portion of the curve as you do when you look at a zoomed out portion. They are both similar, but not of course identical. Music has a similar structure, with patterns of notes repeating but developing throughout the composition. This is a rich, and largely untapped, source of tunes and inspiration.

Fractal generation

There are many ways to generate fractals, but here we will be looking at one method, the Lindenmayer system, or L-system for short. This is a recursive algorithm inspired by biological system; it works by successive applications of substitution rules to a string of symbols to generate another, normally longer, string of symbols. This output string is fed into the input again and a new string is generated. This process is repeated any number of times and produces a fractal, or self-similar, sequence. The rules and the initial string, called the axiom, determine the outcome.

Let's see how this works in practice by looking at a very simple example shown in **Figure 1**. This has just three symbols – A, B, and C – and each symbol has a rule for substitution. So when we encounter an A in the input stream, we replace it with the symbols BA in the output stream. When B is encountered, we replace it with a C. When C is found, we replace it with an AB. These rules are shown on the left of the diagram.

If we start with the simple axiom of C after the first application of the rules, we get the string AB. Then run it through the rules again and the first symbol A is replaced by BA and the second symbol B is replaced by C. This applying of rules to an input string to produce an output string is known as a level of recursion; after four levels, our symbol string is ABCBABAC. The rules can be arbitrarily changed to produce different outcomes and the string can involve as many single-character symbols as you like.



This is the MIDI sound box – the name on top might have given it away

Here's our fractal music playing on the screen!

This is the MIDI USB converter lead that connects the sound box to the Pi

Rules about rules

While the rules can be arbitrary, in order to be successful they need to follow some rules themselves. First of all, each symbol used must have a rule associated with it, and that symbol must occur in at least one of the results of another rule. If this is not observed then some symbols will be isolated and never appear in the output stream. If all the symbol rules map only to another symbol, then the output stream always remains the same length; sometimes you might want this, but normally you will want a sequence to grow. Once an output sequence – or successive sequences – has been produced then you need another set of rules, called production rules, to interpret it into, in our case MIDI notes. Let's look at a simple example.

Simple example

The code in **Simple.py** generates a sequence of symbols and then plays them; this then repeats to a given level of recursion. The rules are expressed as a list of tuples; each tuple has two parts, the input condition and the output condition. To make things easy to interpret for us, we have added the string **"->"** to the end of the input string so that the tuple (**"A->","AB"**) means replace the symbol A with the symbols AB. This just makes it easy for us to spot what a rule is doing and to change it. The code first opens the MIDI port and prints out the rules and axiom. Then these rules are applied six times and the result of each recursion is added to a list called composition.

Finally, the composition is passed to a sonification function called **sonify**. The rules for turning the symbols into notes here are very simple: each symbol, A to G, is turned into a MIDI note number representing the notes A to G, defined by the **notes** list, and played for a time defined by the variable **noteDuration**. This plays each level of recursion with a short gap between each. Quitting the program with **CTRL+C** will cause the program to turn all the MIDI notes off before quitting so you don't get any hanging notes.

The code uses MIDI voice 19, the church organ, but you can change this to anything. If you want to alter this on the fly then you can load up the MIDI voice test program from *The MagPi* #63 to run at the same time. Just navigate to the folder containing it, using a Terminal window and type:

```
nohup python3 voiceTest.py &
```

A multichannel version

The previous example just played a single instrument for a single line. In this next example, the last three levels of recursion are played at the same time on different instruments. As each level is of a different length, the note on time is adjusted so that the playing time as a whole is the same for each track. This means that the smaller levels of recursion have longer notes than the higher ones. A lot of the code is the same as the first example; so, instead of repeating the whole listing, we have just printed the changes you have to make to **Simple.py** in **New_Functions_for_Simple.py**. These changes are basically a new **sonify** function along with two additional functions, **notFinished** and **playNext**. The instruments and volume levels are set at the start of the **sonify** function, and we have used the 'rain' instrument for the long notes because it has something interesting going on in the background for held notes. Short notes, we think, are best when the sound itself is short, like a bell.

Adding graphics

To add some graphics requires a much longer program and our normal Pygame framework. We have written one that will produce the sound of the last example, only play it back by building up the composition by adding one track at a time. The screen output is shown in **Figure 2** and the code can be found in our GitHub repository. It might not look like a classic fractal, but that is because of the very simple mapping of the sonification: one symbol represents one note. To get a bit more flexibility, we need to add a different sort of rules: that of interpreting the fractal string.

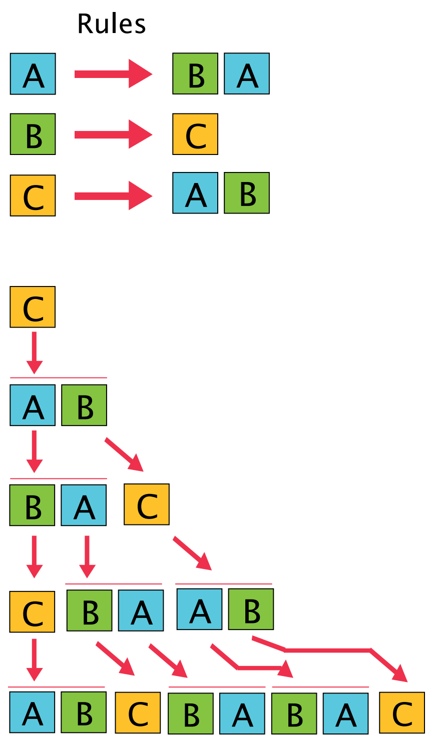


Figure 1 How symbolic substitution works

Interpretation rules

Interpretation rules are somewhat different to the rules we used before to produce a symbol string. They are not a sequence of substitutions but a set of things to do for when sonifying each symbol. For example, suppose we add some symbols in the production rules to alter the length of a note. When this symbol occurs, the note duration changes but no note is generated for that symbol; that means we can't use the trick of altering the note length based on the length of the sequence. It's easy enough to do this, however, and it adds a bit of variety to the composition. The extra code to add this feature is in **Extra_Code_for_Note_Duration_Functions.py** and it shows what you have to change to the **Simple.py**

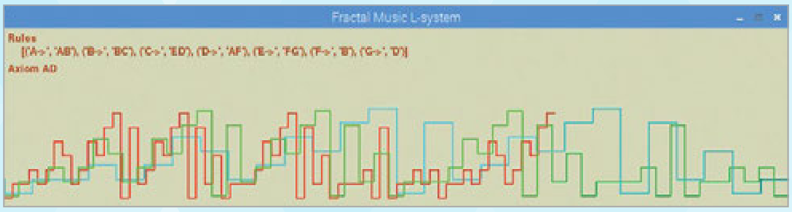


Figure 2 Adding graphics to the sound output

code: it is just a replacement for the **`sonify`** function and a new axiom string and rules list. Here there are three lengths of note defined by the symbols q, h, and c, and these change the **`duration`** variable for subsequent notes.

Adding a state machine

These interpretation rules are still direct substitutions of notes and length of notes. To get another level of complexity you have to get these symbols to interact with a state machine, and use the latter to define various parameters of the music. When this system is used for producing fractal drawings, that state machine is a turtle graphics drawing package. Symbols represent turtle commands like move forward, turn left or right a specific angle, or move without drawing, to name but four. It is the cumulative result of these sorts of commands that determines what is drawn at any one time. In order to get separate branches, there are two other types of operation represented by symbols: the **[** which places the turtle state on a stack, and the **]** which restores the turtle state from a stack. You can have a look at such a system if you install a graphics program called Inkscape. When you run it, go to the Extensions menu, select Render, then the L-system option. You will get a window that allows you to set rules and turn angles; you can set more than one rule by separating them with a semicolon. **Figure 3** shows a list of rules for fractals to set you off exploring.

Figure 3 Rules for fractals in Inkscape's L-system

BUSH
```
Axiom:  ++F
Rules:  F=FF-[-F+F+F]+[+F-F-F]
```

DRAGON CURVE
```
Axiom:  FX
Rules:  X=X+YF;
        Y=FX-Y;
Angle:  90
```

KOCH ISLAND.
```
Axiom:  -F--F--F
Rules:  F=F+F--F+F;
Angle:  60
```

OTHER FRACTAL
```
Axiom:  W
Rules:  W=+++X--F--ZFX+;
        X=---W++F++YFW-;
        Y=+ZFX--F--Z+++;
        Z=-YFW++F++Y---;
Angle:  30
```

PENROSE P3
```
Axiom:  [N]++[N]++[N]++[N]++[N]
Rules:  M=OA++pA----NA[-OA----MA]++;
        N=+OA--PA[---MA--NA]+;
        O=-MA++NA[+++OA++PA]-;
        P=--OA++++MA[+PA++++NA]--NA;
Angle:  36
```

In the same way, you can implement a music turtle that determines the frequency, duration, and any effects you care to specify. So the range of notes is much wider than you can get from a one-to-one mapping of symbol to note. This musical turtle can be restrained to a certain range of parameters by wrapping round the values as they exceed their limits. The code for this is shown in **Classic_Fractals.py** and although it looks similar to the other listings, it does have many slight changes. For a start, the production rules have been changed to reflect the Inkscape system: where there is no rule for a symbol, that symbol is just copied to the output string. Also, the production rules match: any symbol A to F plays a note and updates the pitch, whereas any symbol G to L just updates the pitch. Note the **`initKey`** function; this generates a lookup table in any major key determined by the starting note. The rules in the listing are for a bush whose graphical representation is shown in **Figure 4**.

Results

Well, what does all this sound like? The uncharitable might say it sounds like a maniac practising scales, but there is a lot more to it than that. We liked the simpler systems best, as we felt there was a tune trying to break out and occasionally succeeding; you could definitely hear the self-similarity coming through. Small changes in rules produced small changes in melodies, which is good for control, and we liked the multitimbral approach of having more than one track playing at the same time.

Taking it further

Like no other project, this is one you just have to tinker with. You can have a lot of fun making up rules and listening to the results. This just requires typing them in at the start of the program. There are lots of variations you can make to the production rules, like including a probability factor to some. For example, you can have two rules for one symbol, and attach a probability that one rule will be used over another simply by generating a number from one to ten, and if the number is above some value then use rule one, otherwise use rule two. The production rules for the state machine can be changed to include note duration or even note timbre. For serious music it is probably best to pick out the good bits in a fractal sequence and incorporate that into your own music.

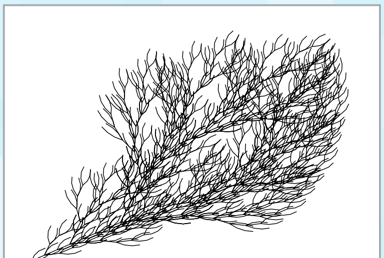


Figure 4 A fractal bush

Classic_Fractals.py

```python
import time, copy
import rtmidi

midiout = rtmidi.MidiOut()

noteDuration = 0.3

axiom = "++F" # Bush
rules = [("F->","FF-[-F+F+F]+[+F-F-F]")]

newAxiom = axiom

def main():
    global newAxiom
    init() # open MIDI port
    offMIDI()
    initKey()
    print("Rules :-")
    print(rules)
    print("Axiom :-")
    print(axiom)
    composition = [newAxiom]
    for r in range(0,4): # change for deeper levels
        newAxiom = applyRules(newAxiom)
        composition.append(newAxiom)
    sonify(composition)

def applyRulesOrginal(start):
    expand = ""
    for i in range(0,len(start)):
        rule = start[i:i+1] +"->"
        for j in range(0,len(rules)):
            if rule == rules[j][0] :
                expand += rules[j][1]
    return expand

def applyRules(start):
    expand = ""
    for i in range(0,len(start)):
        symbol = start[i:i+1]
        rule =  symbol +"->"
        found = False
        for j in range(0,len(rules)):

            if rule == rules[j][0] :
                expand += rules[j][1]
                found = True
        if not found :
                expand += symbol
    return expand

def sonify(data): # turn data into sound
    initMIDI(0,65) # set volume
    noteIncrement = 1
    notePlay = len(notes) / 2
    midiout.send_message([0xC0 | 0,19]) # voice 19 Church organ
    lastNote = 1
    for j in range(0,len(data)):
        duration = noteDuration # start with same note length
        notePlay = len(notes) / 2 # and same start note
        noteIncrement = 1 # and same note increment
        stack = [] # clear stack
        print("")
        if j==0:
            print("Axiom ",j,data[j])
        else:
            print("Recursion ",j,data[j])
        for i in range(0,len(data[j])):
            symbol = ord(data[j][i:i+1])
            if symbol >= ord('A') and symbol <= ord('F') : # play current note
                #print(" playing",notePlay)
                note = notes[int(notePlay)]
                #print("note", note, "note increment",noteIncrement )
                midiout.send_message([0x80 | 0,lastNote,68]) # last note off
                midiout.send_message([0x90 | 0,note,68]) # next note on
                lastNote = note
            if symbol >= ord('A') and symbol <= ord('L') : # move note
                notePlay += noteIncrement
                if notePlay < 0: # wrap round playing note
                    notePlay = len(notes)-1
                elif notePlay >= len(notes):
                    notePlay = 0
                time.sleep(duration)
            if symbol == ord('+'):
                noteIncrement += 1
                if noteIncrement > 6:
                    noteIncrement = 1
            if symbol == ord('-'):
                noteIncrement -= 1
                if noteIncrement < -6:
                    noteIncrement = -1
            if symbol == ord('|'): # turn back
                noteIncrement = -noteIncrement
            if symbol == ord('['): # push state on stack
                stack.append((duration,notePlay,noteIncrement))
                #print("pushed",duration,notePlay,noteIncrement,"Stack depth",len(stack))
            if symbol == ord(']'): # pull state from stack
                if len(stack) != 0 :
                    recovered = stack.pop(int(len(stack)-1))
                    duration = recovered[0]
                    notePlay = recovered[1]
                    noteIncrement = recovered[2]
                    #print("recovered",duration,notePlay,noteIncrement,"Stack depth",len(stack))
                else:
                    print("stack empty")
        midiout.send_message([0x80 | 0,lastNote,68]) # last
```

Language

>PYTHON 3

DOWNLOAD:
magpi.cc/1NqJjmV

PROJECT VIDEOS

Check out Mike's Bakery videos at:
magpi.cc/1NqJnTz

```
note off
        time.sleep(2.0)

def initKey():
    global startNote,endNote,notes
    key = [2,1,2,2,1,2] # defines scale type - a Major scale
    notes =[] # look up list note number to MIDI note
    startNote = 24 # defines the key (this is C )
    endNote = 84
    i = startNote
    j = 0
    while i< endNote:
        notes.append(i)
        i += key[j]
        j +=1
        if j >= 6:
            j = 0
    #print(notes)

def init():
    available_ports = midiout.get_ports()
    print("MIDI ports available:-")
    for i in range(0,len(available_ports)):
        print(i,available_ports[i])
    if available_ports:
        midiout.open_port(1)
    else:
        midiout.open_virtual_port("My virtual output")

def initMIDI(ch,vol):
    midiout.send_message([0xB0 | ch,0x07,vol])  # set to volume
    midiout.send_message([0xB0 | ch,0x00,0x00]) # set default bank

def offMIDI():
    for ch in range(0,16):
        midiout.send_message([0xB0 | ch,0x78,0])  # notes off

# Main program logic:
if __name__ == '__main__':
    try:
        main()
    except:
        offMIDI()
```

Simple.py

```
import time, random, copy
import rtmidi

midiout = rtmidi.MidiOut()

notes = [57,59,60,62,64,65,67]
noteDuration = 0.3

axiom = "AD"
rules = [("A->","AB"),("B->","BC"),("C->","ED"),("D->","AF"),
         ("E->","FG"),("F->","B"),("G->","D") ]
newAxiom = axiom

def main():
    global newAxiom
    init() # open MIDI port
    offMIDI()
    print("Rules :-")
    print(rules)
    print("Axiom :-")
    print(axiom)
    composition = [newAxiom]
    for r in range(0,6):
        newAxiom = applyRules(newAxiom)
        composition.append(newAxiom)
    sonify(composition)

def applyRules(start):
    expand = ""
    for i in range(0,len(start)):
        rule = start[i:i+1] +"->"
        #print("we are looking for rule",rule)
        for j in range(0,len(rules)):
            if rule == rules[j][0] :
                #print("found rule", rules[j][0],"translates to",rules[j][1])
                expand += rules[j][1]
    return expand

def sonify(data): # turn data into sound
    initMIDI(0,65) # set volume
    midiout.send_message([0xC0 | 0,19]) # voice 19 Church organ
    lastNote = 1
    for j in range(0,len(data)):
        if j==0:
            print("Axiom     ",j,data[j])
        else:
            print("Recursion ",j,data[j])
        for i in range(0,len(data[j])):
            note = notes[ord(data[j][i:i+1]) - ord('A')] # get note given by letter
            midiout.send_message([0x80 | 0,lastNote,68]) # last note off
            midiout.send_message([0x90 | 0,note,68]) # next note on
            lastNote = note
            time.sleep(noteDuration)
        midiout.send_message([0x80 | 0,lastNote,68]) # last note off
        time.sleep(2.0)

def init():
    available_ports = midiout.get_ports()
    print("MIDI ports available:-")
    for i in range(0,len(available_ports)):
        print(i,available_ports[i])
    if available_ports:
        midiout.open_port(1)
    else:
```

```
        midiout.open_virtual_port("My virtual output")

def initMIDI(ch,vol):
    midiout.send_message([0xB0 | ch,0x07,vol])  # set to volume
    midiout.send_message([0xB0 | ch,0x00,0x00]) # set default bank

def offMIDI():
    for ch in range(0,16):
        midiout.send_message([0xB0 | ch,0x78,0])  # notes off

# Main program logic:
if __name__ == '__main__':
     try:
          main()
     except:
         offMIDI()
```

New_Functions_of_Simple.py

```
def sonify(data):
    melodyLines = 3 # change for more or less lines
    # for more melody lines add more elements to the next two lists
    instruments = [112, 0, 96] # instruments for each line
    volume = [50, 60, 65] # volume for ech line
    lastNote = []
    index = []
    startTime = []
    interval = []
    lineLength = []
    for i in range(0,melodyLines):
        initMIDI(i,volume[i]) # setu up MIDI channel
        midiout.send_message([0xC0 | i,instruments[i]]) # set voice
        startTime.append(time.time()) # set up lists
        index.append(0)
        lastNote.append(0)
        interval.append(noteDuration * len(data[len(data)-1])/len(data[len(data)-1-i]))
        lineLength.append(len(data[len(data)-1-i]))
    print() ; print("Playing")
    for i in range(0,melodyLines):
        print("line",i,"voice",instruments[i],"length",lineLength[i],
             "notes of duration",interval[i],"seconds")
    while notFinished(melodyLines,lineLength,index) :
        for i in range(0,melodyLines):
            if time.time() - startTime[i] > interval[i]:
                lastNote[i] = playNext(i,index[i],lastNote[i],data,len(data)-1)
                index[i] += 1
                startTime[i] = time.time()
    time.sleep(noteDuration)
    for i in range(0,melodyLines):
        midiout.send_message([0x80 | i,lastNote[i],68]) # last note off

def notFinished(playingLines,length, point):
    notDone = True
    for i in range(0,playingLines):
        if point[i] >= length[i] :
            notDone = False
    return notDone

def playNext(midiChannel, i , lastNote, data, line):
    note = notes[ord(data[line][i:i+1]) - ord('A')] # get note given by letter
    midiout.send_message([0x80 | midiChannel,lastNote,68]) # last note off
    midiout.send_message([0x90 | midiChannel,note,68]) # next note on
    return note
```

Extra_Code_for_Note_Duration_Functions.py

```
axiom = "qAhD"
rules = [("A->","ABc"),("B->","BCh"),("C->","EDq"),("D->","AFc"),
         ("E->","FGh"),("F->","Bq"),("G->","Dc"),("q->","hA"),("h->","qF"),("c->","hF") ]

def sonify(data): # turn data into sound
   initMIDI(0,65) # set volume
   midiout.send_message([0xC0 | 0,19]) # voice 19 Church organ
   lastNote = 1
   for j in range(0,len(data)):
      duration = noteDuration # start with same note length
      if j==0:
         print("Axiom     ",j,data[j])
      else:
         print("Recursion ",j,data[j])
      for i in range(0,len(data[j])):
         symbol = ord(data[j][i:i+1])
         if symbol >= ord('A') and symbol <= ord('G') : # it is a note
            note = notes[symbol - ord('A')] # get note given by letter
            midiout.send_message([0x80 | 0,lastNote,68]) # last note off
            midiout.send_message([0x90 | 0,note,68]) # next note on
            lastNote = note
            time.sleep(duration)
         else : # it is a note duration
            if symbol == ord('h'):
               duration = noteDuration * 2
            if symbol == ord('c'):
               duration = noteDuration
            if symbol == ord('q'):
               duration = noteDuration / 2
      midiout.send_message([0x80 | 0,lastNote,68]) # last note off
      time.sleep(2.0)
```

EBEN UPTON

Eben is the creator of Raspberry Pi and a co-founder of the Raspberry Pi Foundation.

raspberrypi.org

WHY RASPBERRY PI ISN'T VULNERABLE TO SPECTRE OR MELTDOWN

Learn how these complex exploits work with this simple and clear explanation, and discover why the Raspberry Pi isn't affected

You'll Need

- Raspberry Pi **raspberrypi.org**
- Thonny IDE

Over the last few weeks, there has been a lot of discussion about a pair of security vulnerabilities nicknamed Spectre and Meltdown (**spectreattack.com**). These affect all modern Intel processors, and (in the case of Spectre) many AMD processors and ARM cores (**developer.arm.com/support/security-update**).

Spectre allows an attacker to bypass software checks to read data from arbitrary locations in the current address space; Meltdown allows an attacker to read data from arbitrary locations in the operating system kernel's address space (which should normally be inaccessible to user programs).

Both vulnerabilities exploit performance features (caching and speculative execution) common to many modern processors to leak data via a so-called side-channel attack. Happily, the Raspberry Pi isn't susceptible to these vulnerabilities, because of the particular ARM cores that we use.

To help us understand why, here's a little primer on some concepts in modern processor design. We'll illustrate these concepts using simple programs in Python syntax like this one:

```
t = a+b
u = c+d
v = e+f
w = v+g
x = h+i
y = j+k
```

While the processor in your computer doesn't execute Python directly, the statements here are simple enough that they roughly correspond to a single machine instruction. We're going to gloss over some details. Notably pipelining (**magpi.cc/2m9IV7J**) and register renaming (**magpi.cc/2m9T6cG**), which are very important to processor designers, but which aren't necessary to understand how Spectre and Meltdown work.

For a comprehensive description of processor design, and other aspects of modern computer architecture, you can't do better than Hennessy and Patterson's classic *Computer Architecture: A Quantitative Approach* (**magpi.cc/2m9UwE2**).

What is a scalar processor?

The simplest sort of modern processor executes one instruction per cycle; we call this a scalar processor. Our aforementioned example will execute in six cycles on a scalar processor.

Examples of scalar processors include the Intel 486 and the ARM1176 core used in Raspberry Pi 1 and Raspberry Pi Zero.

What is a superscalar processor?

The obvious way to make a scalar processor (or indeed any processor) run faster is to increase its clock speed. However, we soon reach limits of how fast the logic gates inside the processor can be made to run; processor designers therefore began to look for ways to do several things at once.

An in-order superscalar processor examines the incoming stream of instructions and tries to execute more than one at once, in one of several pipelines (pipes for short), subject to dependencies between the instructions. Dependencies are important: you might think that a two-way superscalar processor could just pair up (or dual-issue) the six instructions in our example like this:

SPECTRE PAPER

If you want to really get inside Spectre, read the paper 'Spectre Attacks: Exploiting Speculative Execution'.

magpi.cc/2mJEKRo

```
t, u = a+b, c+d
v, w = e+f, v+g
x, y = h+i, j+k
```

But this doesn't make sense: we have to compute **v** before we can compute **w**, so the third and fourth instructions can't be executed at the same time. Our two-way superscalar processor won't actually be able to find anything to pair with the third instruction, so our example will execute in four cycles:

```
t, u = a+b, c+d
v    = e+f   # second pipe does nothing here
w, x = v+g, h+i
y    = j+k
```

Examples of superscalar processors include the Intel Pentium, and the ARM Cortex-A7 and Cortex-A53 cores used in Raspberry Pi 2 and Raspberry Pi 3 respectively. Raspberry Pi 3 has only a 33% higher clock speed than Raspberry Pi 2, but has roughly double the performance: the extra performance is partly a result of Cortex-A53's ability to dual-issue a broader range of instructions than Cortex-A7.

What is an out-of-order processor?

Going back to our example, we can see that, although we have a dependency between **v** and **w**, we have other independent instructions later in the program that we could potentially have used to fill the empty pipe during the second cycle. An out-of-order superscalar processor has the ability to shuffle the order of incoming instructions (again subject to dependencies) in order to keep its pipes busy.

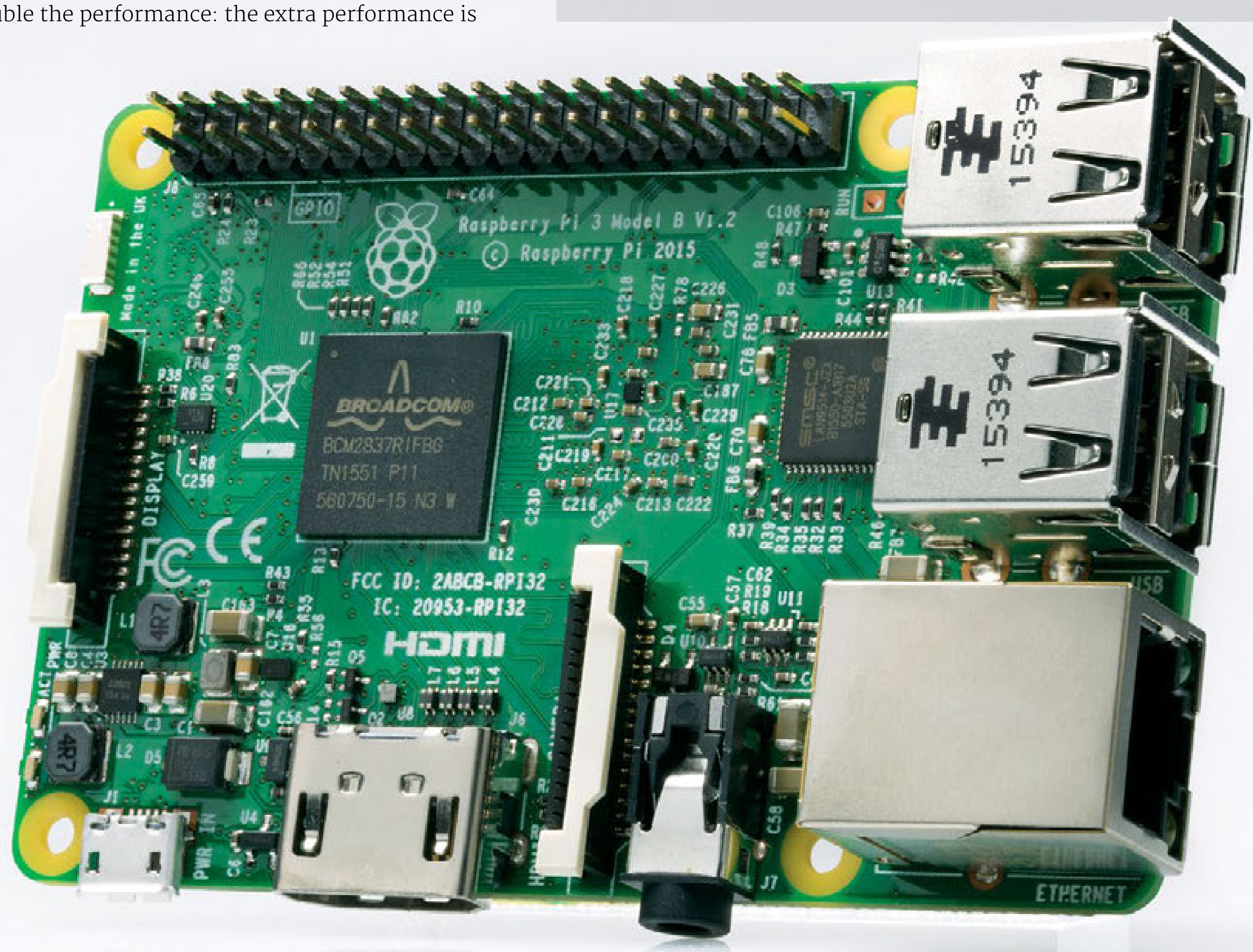
An out-of-order processor might effectively swap the definitions of **w** and **x** in our example like this:

```
t = a+b
u = c+d
v = e+f
x = h+i
w = v+g
y = j+k
```

RASPBERRY PI FORUM

The Raspberry Pi Forum is a great place to chat with other users about all computer matters, including computer security. Here is a recent forum chat discussing Spectre and Meltdown.

magpi.cc/2m8B8ag



- The Raspberry Pi 3 shown here uses an ARM Cortex-A53 processor (and the older Raspberry Pi 2 uses an ARM Cortex-A7).
- The lack of speculation in the ARM1176, Cortex-A7, and Cortex-A53 cores used in Raspberry Pi render us immune to attacks of the sort.
- A cache is a small on-chip memory, close to the processor, which stores copies of the contents of recently used locations (and their neighbours), so that they are quickly available on subsequent accesses. Both Spectre and Meltdown use side-channel attacks based on the contents of the cache.

Image Credit: SpectreAttack (**spectreattack.com**)

```
meltdown:
mov al, byte [rcx]
shl rax, 0xc
jz meltdown
mov rbx, qword [rbx + rax]
```

Above A Meltdown code illustration

...allowing it to execute in three cycles:

```
t, u = a+b, c+d
v, x = e+f, h+i
w, y = v+g, j+k
```

Examples of out-of-order processors include the Intel Pentium 2 (and most subsequent Intel and AMD x86 processors with the exception of some Atom and Quark devices), and many recent ARM cores, including Cortex-A9, -A15, -A17, and -A57.

What is branch prediction?

Our example above is a straight-line piece of code. Real programs aren't like this of course: they also contain both forward branches (used to implement conditional operations like **if** statements), and backward branches (used to implement loops). A branch may be unconditional (always taken), or conditional (taken or not, depending on a computed value); it may be direct (explicitly specifying a target address) or indirect (taking its target address from a register, memory location or the processor stack).

While fetching instructions, a processor may encounter a conditional branch which depends on a value which has yet to be computed. To avoid a stall, it must guess which instruction to fetch next: the next one in memory order (corresponding to an untaken branch), or the one at the branch target (corresponding to a taken branch). A branch predictor helps the processor make an intelligent guess about whether a branch will be taken or not. It does this by gathering statistics about how often particular branches have been taken in the past.

Modern branch predictors are extremely sophisticated, and can generate very accurate predictions. Raspberry Pi 3's extra performance is partly a result of improvements in branch prediction between Cortex-A7 and Cortex-A53. However, by executing a crafted series of branches, an attacker can mis-train a branch predictor to make poor predictions.

What is speculation?

Reordering sequential instructions is a powerful way to recover more instruction-level parallelism, but as processors become wider (able to triple- or quadruple-issue instructions) it becomes harder to keep all those pipes busy. Modern processors have therefore grown the ability to speculate. Speculative execution lets us issue instructions which might turn out not to be required (because they may be branched over): this keeps a pipe busy (use it or lose it!), and if it turns out that the instruction isn't executed, we can just throw the result away.

Speculatively executing unnecessary instructions (and the infrastructure required to support speculation and reordering) consumes extra energy, but in many cases this is considered a worthwhile trade-off to obtain extra single-threaded performance. The branch predictor is used to choose the most likely path through the program, maximising the chance that the speculation will pay off.

To demonstrate the benefits of speculation, let's look at another example:

> "Here's a little primer on some concepts in modern processor design"

AFFECTED CPUS

Most computer companies have offered information on how they are affected. You can see a full list of statements from each company at bottom of the SpectreAttack webpage.

spectreattack.com

```
t = a+b
u = t+c
v = u+d

if v:
   w = e+f
   x = w+g
   y = x+h
```

Now we have dependencies from **t** to **u** to **v**, and from **w** to **x** to **y**, so a two-way out-of-order processor without speculation won't ever be able to fill its second pipe. It spends three cycles computing **t**, **u**, and **v**, after which it knows whether the body of the **if** statement will execute, in which case it then spends three cycles computing **w**, **x**, and **y**. Assuming the **if** (implemented by a branch instruction) takes one cycle, our example takes either four cycles (if **v** turns out to be zero) or seven cycles (if **v** is non-zero).

If the branch predictor indicates that the body of the **if** statement is likely to execute, speculation effectively shuffles the program like this:

```
t = a+b
u = t+c
v = u+d
w_ = e+f
x_ = w_+g
y_ = x_+h

if v:
   w, x, y = w_, x_, y_
```

So we now have additional instruction level parallelism to keep our pipes busy:

```
t, w_ = a+b, e+f
u, x_ = t+c, w_+g
v, y_ = u+d, x_+h

if v:
   w, x, y = w_, x_, y_
```

Cycle counting becomes less well defined in speculative out-of-order processors, but the branch and conditional update of **w**, **x**, and **y** are (approximately) free, so our example executes in (approximately) three cycles.

What is a cache?

In the good old days, the speed of processors was well matched with the speed of memory access. My BBC Micro, with its 2 MHz 6502, could execute an instruction roughly every 2 µs (microseconds), and had a memory cycle time of 0.25 µs. Over the ensuing 35 years, processors have become very much faster, but memory only modestly so: a single Cortex-A53 in a Raspberry Pi 3 can execute an instruction roughly every 0.5 ns (nanoseconds), but can take up to 100 ns to access main memory.

At first glance, this sounds like a disaster: every time we access memory, we'll end up waiting for 100 ns to get the result back.

In this case, this example:

```
a = mem[0]
b = mem[1]
```

...would take 200 ns.

However, in practice, programs tend to access memory in relatively predictable ways, exhibiting both temporal locality (if I access a location, I'm likely to access it again soon) and spatial locality (if I access a location, I'm likely to access a nearby location soon). Caching takes advantage of these properties to reduce the average cost of access to memory.

A cache is a small on-chip memory, close to the processor, which stores copies of the contents of recently used locations (and their neighbours), so that they are quickly available on subsequent accesses. With caching, the example above will execute in a little over 100 ns:

```
a = mem[0]    # 100ns delay, copies
                mem[0:15] into cache
b = mem[1]    # mem[1] is in the cache
```

From the point of view of Spectre and Meltdown, the important point is that if you can time how long a memory access takes, you can determine whether the address you accessed was in the cache (short time) or not (long time).

What is a side channel?

From Wikipedia:

"... a side-channel attack is any attack based on information gained from the physical implementation of a cryptosystem, rather than brute force or theoretical weaknesses in the algorithms (compare cryptanalysis). For example, timing information, power consumption, electromagnetic leaks or even sound can provide an extra source of information, which can be exploited to break the system."

Spectre and Meltdown are side-channel attacks which deduce the contents of a memory location which should not normally be accessible by using timing to observe whether another, accessible, location is present in the cache.

Putting it all together

Now let's look at how speculation and caching combine to permit a Meltdown-like attack on our processor. Consider the following example, which is a user program that sometimes reads from an illegal (kernel) address, resulting in a fault (crash):

Meltdown

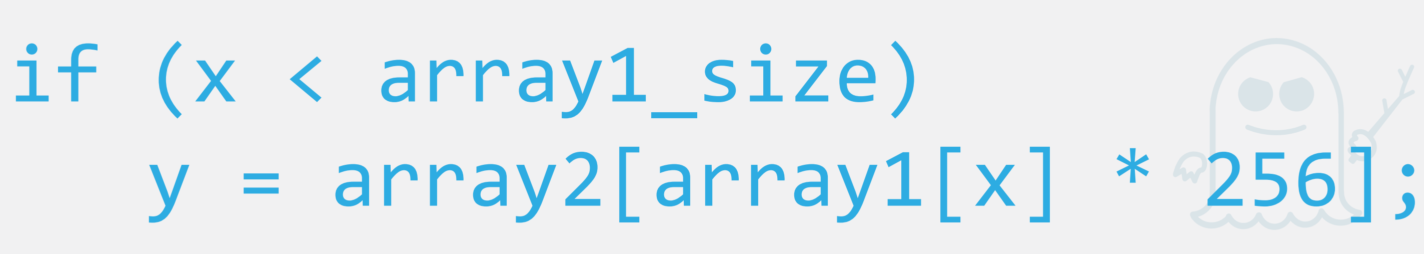
Meltdown breaks the most fundamental isolation between user applications and the operating system. This attack allows a program to access the memory, and thus also the secrets, of other programs and the operating system.

Spectre

Spectre breaks the isolation between different applications. It allows an attacker to trick error-free programs, which follow best practices, into leaking their secrets. In fact, the safety checks of said best practices actually increase the attack surface and may make applications more susceptible to Spectre.

Images courtesy of Natascha Eibl, vividfox.me

Image Credit: SpectreAttack (**spectreattack.com**)



Above A Spectre code illustration

```
t = a+b
u = t+c
v = u+d

if v:
   w = kern_mem[address]
       # if we get here, fault
   x = w&0x100
   y = user_mem[x]
```

Now, provided we can train the branch predictor to believe that **v** is likely to be non-zero, our out-of-order two-way superscalar processor shuffles the program like this:

```
t, w_ = a+b, kern_mem[address]
u, x_ = t+c, w_&0x100
v, y_ = u+d, user_mem[x_]

if v:
   # fault
   w, x, y = w_, x_, y_ # we never get here
```

Even though the processor always speculatively reads from the kernel address, it must defer the resulting fault until it knows that **v** was non-zero. On the face of it, this feels safe because either:

- **v is zero, so the result of the illegal read isn't committed to w**

- **v is non-zero, but the fault occurs before the read is committed to w**

However, suppose we flush our cache before executing the code, and arrange **a**, **b**, **c**, and **d** so that **v** is actually zero. Now, the speculative read in the third cycle:

```
v, y_ = u+d, user_mem[x_]
```

...will access either userland address **0x000** or address **0x100** depending on the eighth bit of the result of the illegal read, loading that address and its neighbours into the cache. Because **v** is zero, the results of the speculative instructions will be discarded, and execution will continue. If we time a subsequent access to one of those addresses, we can determine which address is in the cache.

Congratulations: you've just read a single bit from the kernel's address space!

The real Meltdown exploit is substantially more complex than this (notably, to avoid having to mis-train the branch predictor, the authors prefer to execute the illegal read unconditionally and handle the resulting exception), but the principle is the same. Spectre uses a similar approach to subvert software array bounds checks.

Conclusion

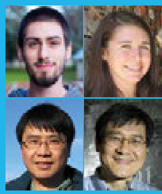
Modern processors go to great lengths to preserve the abstraction that they are in-order scalar machines that access memory directly, while in fact using a host of techniques including caching, instruction reordering, and speculation to deliver much higher performance than a simple processor could hope to achieve. Meltdown and Spectre are examples of what happens when we reason about security in the context of that abstraction, and then encounter minor discrepancies between the abstraction and reality.

Fortunately, the lack of speculation in the ARM1176, Cortex-A7, and Cortex-A53 cores used in Raspberry Pi render us immune to attacks of the sort.

ARM SECURITY UPDATE

The Raspberry Pi runs ARM processors, and ARM has a webpage outlining more information on how its products are affected. It also lists the vulnerability of its processors to the speculation side-channel (Spectre) attacks. ARM notes that "The majority of Arm processors are not impacted by any variation of this side-channel speculation mechanism."

magpi.cc/2mglg6j

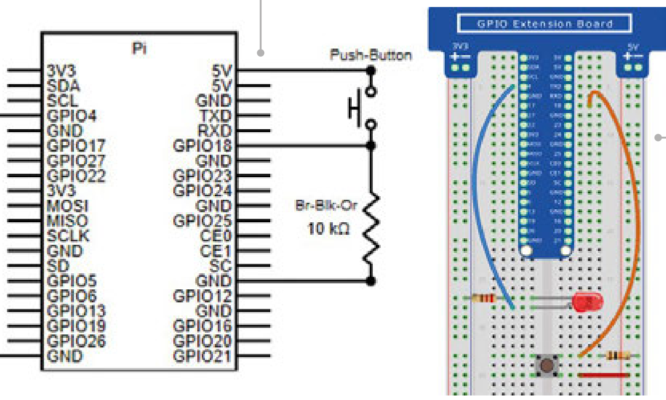


KYLE GOFF, KYLIE COOPER, BINSEN QIAN, HARRY H. CHENG

Kyle Goff, undergraduate computer engineering student; Kylie Cooper, undergraduate mechanical engineering student; Binsen Qian, PhD candidate in mechanical engineering; Harry H. Cheng, professor and director of the UC Davis C-STEM Center.

cstem.ucdavis.edu

GPIO 18 is used for the push-button input, which will change the direction of motion for the robot



GPIO 4 is used for the LED output, which will indicate the current direction of the robot

A breakout board is used in this diagram to simplify the wiring process; you can also wire directly from the Pi

CONTROLLING LEGO MINDSTORMS THROUGH GPIO

Construct a circuit to directly control a LEGO robot with the Raspberry Pi GPIO

You'll Need

- LEGO Mindstorms robot (NXT or EV3)
- Breadboard
- Wires for breadboard
- 1 × LED
- 1 × Push-button
- 1 × 220 Ω resistor (Red-Red-Brown)
- 1 × 10 kΩ resistor (Brown-Black-Orange)

LEGO Mindstorms is a great tool to gain experience in understanding robotics, but what if you wanted to make your own input sensor? In this guide, we will show how simple it is to construct a circuit to control a Mindstorms robot through GPIO in Raspberry Pi.

We will show every step from connecting the robot to writing the code. The result will be a program in Ch, a superset interpreter of C/C++, to control the direction of the robot with a push-button.

Software

To make use of C-STEM's programming tools, you should install the C-STEMbian operating system, which contains C-STEM Studio. This free, open-source operating system contains all the necessary tools for robotics and physical computing. Additionally, it is a superset of Raspbian, so all the familiar features will still be there. If you already have Raspbian installed, the C-STEM modules can be installed separately on top. All of this is available from the C-STEMbian section of the C-STEM website (**magpi.cc/2p3JUNP**). Step-by-step guides will assist you in setting up and accessing the Raspberry Pi if needed.

Connecting to the Mindstorms robot

Connecting to your Mindstorms robot is quite simple with the C-STEM software.

First, you will need to open C-STEM Studio and launch Ch Mindstorms Controller. Find the big 'C' at the top of the screen after logging in to your Raspberry Pi. Click the 'C', then navigate to 'Ch Mindstorms Controller' on the left side of the menu in C-STEM Studio. Click on Launch to open it.

Ch Mindstorms Controller can connect with both EV3 and NXT robots. Simply press the Scan Robot

Language

>CH

DOWNLOAD:
magpi.cc/2mHpaF4

TROUBLESHOOTING WITH GPIOVIEWER

Remember to use GPIOviewer to test your circuit before programming. You can test both the LED and the push button with the GUI.

button and add the robots that are found to the list on your robot manager. Follow the instructions on screen to pair the robots with your Raspberry Pi. Due to the limitations of Bluetooth, the Ch Mindstorms Controller can connect to a maximum of seven robots at a time. (Do make sure that the robots are turned on and have Bluetooth enabled!)

Once the robots have been scanned and added to the list, select the ones you would like to connect to and press Connect. Robots that you are connected to will have a green dot next to their names.

Building the simple circuit

The program in this tutorial requires a physical circuit to function. Our circuit will consist of a push-button input to control the direction of the robot's movement. An LED output will give a visual indication of the direction change when pressing the button.

Looking at the circuit, there are two sides: input and output. The input side, shown on the right, has a push button in series with a 10 kΩ resistor. The push-button is connected to 5 V for power. GPIO 18 is connected between them to read the button input.

The output side, on the left, has an LED in series with a 220 Ω resistor. GPIO 4 controls this light.

If you have one, use a breakout board to make the wiring process clearer. Otherwise, wire the pins directly from the Pi. Take a wire from GPIO 4 and connect it to an empty row of the breadboard. Then, attach the positive (longer) leg of an LED to this row. From the negative leg of the LED, attach a 220 Ω (Red-Red-Brown) resistor to ground.

For the push-button, insert it over the breadboard gutter. Wire 5 V to one lead, and wire a 10 kΩ (Brown-Black-Orange) resistor from ground to the adjacent leg. Finally, connect a wire from GPIO 18 to the row of the resistor and push-button leg. This will carry the input signal when the button is pressed.

Before programming, we can use GPIOviewer, a helpful feature of the C-STEMbian operating system. To use it, navigate again to the big 'C' at the top of the desktop window.

Once open, navigate to Ch Raspberry Pi and click Launch in the bottom right-hand corner. This will open up GPIOviewer, which allows total control of all the GPIO pins on the Raspberry Pi. In this view, you can change pin modes between input, output, and PWM (with a slider).

For this circuit, find GPIO 4 and set it to output. Ensure the LED is set up and working properly by switching between high and low outputs. If the light turns on, you can move on to testing the input. Set GPIO 18 to input mode. Then, try pressing the button. If the input changes, the circuit is now ready for programming.

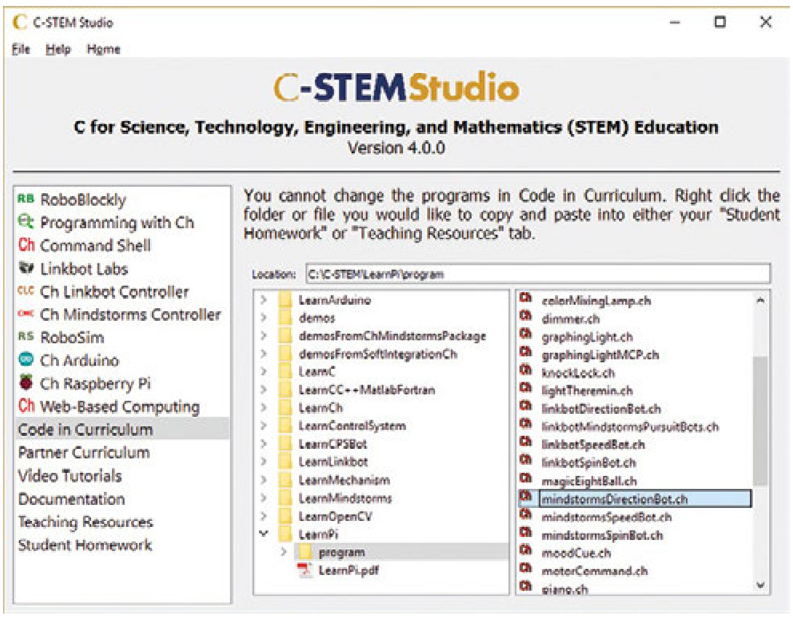
Coding in Ch

Programming in Ch starts by opening C-STEM Studio again on your Raspberry Pi. In v4.0, Navigate to Code in Curriculum > LearnPiprogram > mindstormsDirectionBot.ch. If you would like to make changes to the file, be sure to copy and paste it to another location before opening! To open the program with ChIDE, simply double-click it. The code for the project follows, which can be modified within the editing pane.

When running the code, be sure that the Mindstorms robot is still connected through CMC! Otherwise, the IDE will not recognise that the bot is connected and therefore will not run the code on it. The code should drive the robot forward or backward continuously at a constant speed. When the user gives input via a button press, the robot should switch directions by negating its speed. The LED will also change states on a button press by checking if the speed is positive or negative. Let's take a closer look at the Ch code to understand how this is done…

> "When running the code, be sure that the Mindstorms robot is still connected through CMC!"

The first thing to notice are the two headers, **wiringPi.h** and **mindstorms.h**. We use the wiringPi header to take inputs and outputs more easily from



Open mindstormsDirectionBot.ch from the 'Code in Curriculum' section of C-STEM Studio

FOR HELP AND NEW IDEAS

Explore the free 'Learn Physical Computing with Raspberry Pi' and 'Learning Robot Programming with LEGO Mindstorms for the Absolute Beginner' textbooks in C-STEM Studio.

Check that the LED is working correctly by switching between high and low outputs on GPIOviewer

Check that the push-button is working correctly by making sure the input value changes after the button is pushed

the GPIO pins. It allows the code to resemble Arduino code. To use the Mindstorms platform, we include **mindstorms.h** which contains all robot functionality like moving and turning.

CMindstorms robot;

Now, the program needs to declare a robot. In Ch, **`CMindstorms`** is a class that can be instantiated like a variable. For this program, the Mindstorms robot is referred to as 'robot'; all functions related to it will be called using the **`robot.function`** format.

```
double radius = 1.1;
double speed = 5.0;
```

The first variable holds the radius of the wheels on the Mindstorms robot in units of inches. Remember that the radius of a circle, or wheel, is the distance from its centre to the edge. After that, a variable for speed is set to 5 inches (12.5cm) per second. Depending on the Mindstorms wheels attached, the radius may be different. By storing the value in a variable and passing it to functions, the code is easily adaptable to various sizes. If unsure, the radius of the physical wheel can be measured, especially if using a custom wheel.

```
int switchVal;
int directionPin = 18;
int ledPin = 4;
wiringPiSetupGpio();
pinMode(directionPin, INPUT);
pinMode(ledPin, OUTPUT);
```

Variables for the current switch value as well as the input/output pins are declared similarly to previous projects. Then, the pins are set up and initialised.

robot.driveForeverNB();

Before entering the **`while`** loop, the robot is set to continuously move using the **`driveForeverNB()`** function in the **`CMindstormsI`** class. It will drive forever in whatever direction it is currently facing. Forcing the robot to move constantly makes this code and physical circuit easier since the only input needed is the direction change. It is important not to use the **`driveForever()`** version without the 'NB' letters. For these functions, 'NB' stands for 'non-blocking', which allows the code to continue after the function has been called. Without the 'NB,' the code would stop at the function because it 'blocks' the program from continuing until it finishes.

```
while(1){
switchVal = digitalRead(directionPin);
delay(50);

if (switchVal == HIGH) {
speed = -speed;
robot.setSpeed(speed,radius);
robot.driveForeverNB();
}
```

The first section inside the infinite **`while`** loop checks the direction-changing pin. There is a **`delay(50)`**, meaning wait 50 milliseconds, to ensure a clean reading of the pin. Without this, it may switch directions multiple times on a single press. If the pin reads a value of 'HIGH' or '1', it will reverse the direction of movement. To accomplish this, the speed is set equal to its negative counterpart. For example, if the speed was 5 inches/second, this will change it to -5 inches/second. Therefore, the Mindstorms robot will move just as fast in either direction. Writing a new speed to the robot also requires the **`setSpeed()`** function in the **`CMindstormsI`** class. Notice that this function also requires the radius of the wheel because it uses this value to calculate how fast the wheel must spin to achieve the correct distance. Finally, one more **`robot.driveForeverNB()`** call is made to ensure the robot continues to move.

```
if (speed >= 0) {
digitalWrite(ledPin, HIGH);
}
else {
digitalWrite(ledPin, LOW);
}
}
```

To end the **`while`** loop, an **`if`** statement controls the state of the LED. When the robot's speed is greater than zero, it must be moving forward. In this case, the LED turns on. The LED turns off while the robot is moving backwards by checking if speed is less than zero.

If you want to take this project a step further, try connecting multiple robots and control them with the same circuit! Additionally, you can add LED traffic lights and make the robot move according to the lights. Or, come up with your own idea! Now you have the tools to make circuits that can interact with robots.

READ US ANYWHERE

FREE: DOWNLOAD ALL 30 ORIGINAL ISSUES

The MagPi Magazine

Available now

for smartphones & tablets

Available on the App Store

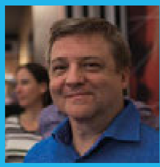
GET IT ON Google Play

Subscribe from

£2.29 rolling subscription or **£26.99** full year subscription

Download it today – it's free!

- Get all 30 legacy issues free
- Instant downloads every month
- Fast rendering performance
- Live links & interactivity



BRIAN BEUKEN

Very old game programmer now teaching very young game programmers a lot of bad habits at Breda University of Applied Science in Breda NL.

scratchpadgames.net

CODING GAMES ON THE RASPBERRY PI IN C/C++ PART 02

You'll Need

- Code::Blocks
- Some imagination
- A bit of patience

After our setup of systems, it's time to get some graphics sorted out

Last time, we demonstrated that it's pretty easy to get a few objects created and doing something – they just weren't doing very much that would interest us. Now it's time to take a bit of a plunge and set up a means to display some graphics. This will give our objects a bit more substance and allow us to do a lot more cool things with the framework we are building.

Last time, we kicked off our project in our **Main** function, then went off to let a **Game** class do some work by printing some text to a console screen. If we want to do graphics, we need to have a graphics screen to draw to; the console screen is for text only.

Setting up a graphics screen is a bit of a techy minefield, but we can explain the basics and provide the code for you to get started We'll go in to a little more detail on the tech another time. Use the new project from the GitHub repository; there are a few additions/changes, but it should still be recognisable from the last version.

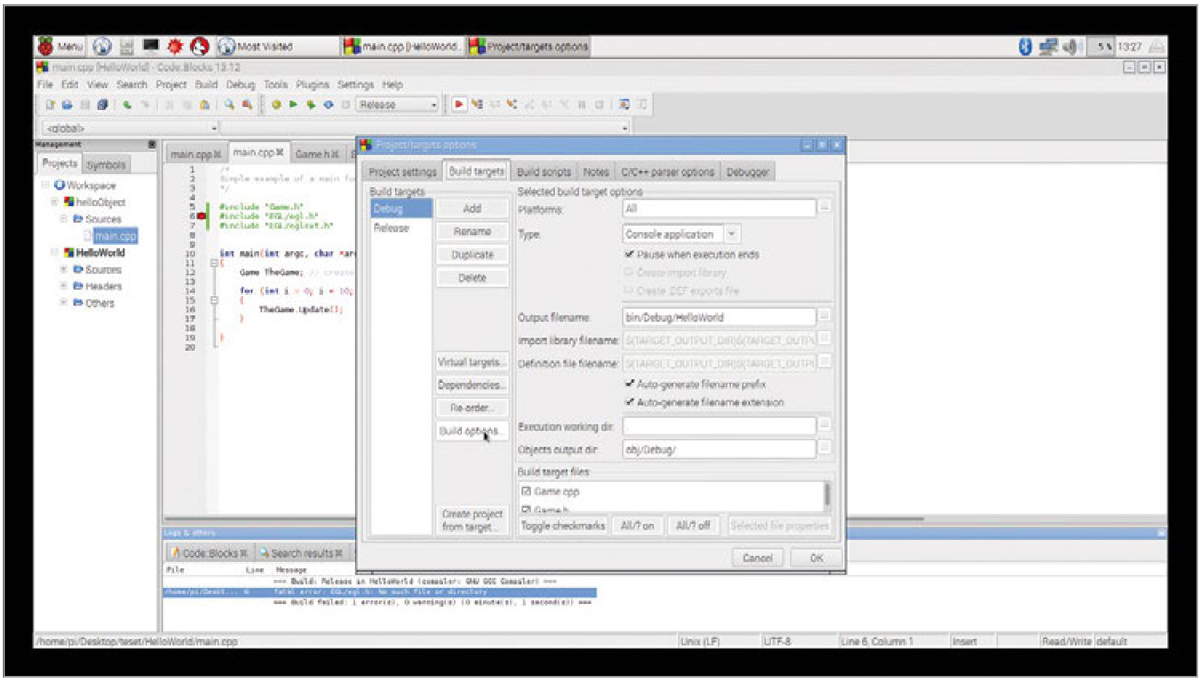
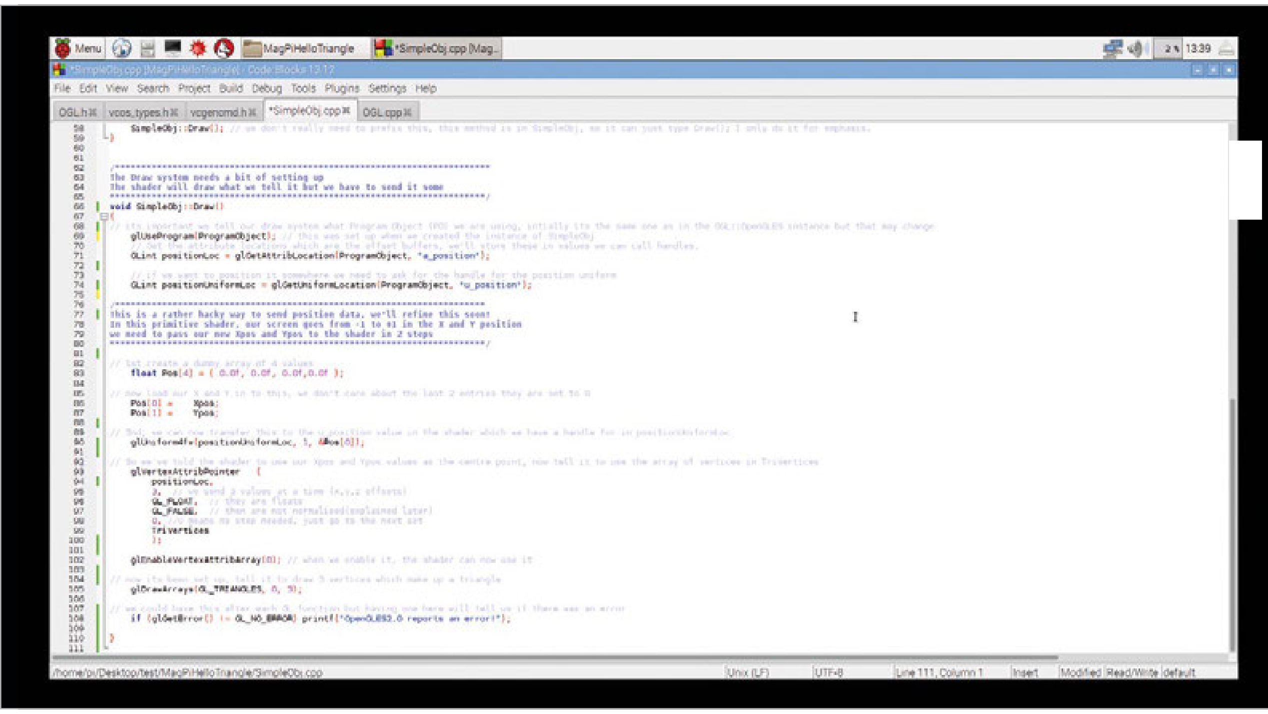


Figure 1 Locate your Build Options button



Language

>C++

NAME:
main.cpp, SimpleObj.h/cpp, Game.h/cpp

DOWNLOAD:
magpi.cc/2mJqLKG

Figure 2 A new Draw routine

Using a library

In order to create graphics, we need to use libraries that give us access to the GPU features. Our IDE needs to know in which directories the libraries we use are located; this is a complicated mess, but needs to be done! Of course, the supplied project has this already set up for you so you can use that, but it's good to know how to set these up.

In Code::Blocks, select the Project menu, choose Properties, then select the Build Targets tab (**Figure 1**).

Click on the Build Options button to bring up another selection of tabs, but this time select the Search Directories tab.

You will be presented with a default empty Compiler tab. Select the Add button and a dialogue will come up with your current default project directory, but we don't want that – click on the '…' button and navigate to File System > opt > vc > include and select open. Keep this as a relative path when it asks. Repeat for:

File System > opt > vc > include > interface;

File System > opt > vc > include > interface > vmcs_host > linux;

File System > opt > vc > include > interface > vcos > pthreads.

Good, that's the compiler sorted; it will know where to look for headers. Now choose the Linker Settings tab, on the top row of tabs.

Tell the linker that we want to include binary libraries. Click on Add, then the '…' button, and navigate to File System > opt > vc > lib.

Depending on which version of Raspbian you're using, you'll need to load different files. For Raspbian before November 2017, hold **CTRL** and select three files: **libbcm_host.so**, **libEGL.so**, and **libGLESv2.so**. For the latest version of Raspbian Stretch, the file names are a little different and there are six: **libbcm_host.so**, **GLESv2_static.a**, **EGL_static.a**, **vchiq_arm.a**, **vcos.a**, and **khrn_static.a**.

Click OK and also keep them relative.

We're done for now. Click OK to return your project, ready to enter code, but be aware that we've only set up our debug configuration.

Working with the GPU

Our GPU processes the commands we send to it and, using information we store in a data structure called a context, it puts the results of those commands on screen. The GPU and context both need to be initialised to get them up and ready to accept those commands. We'll explain the commands a little later on, but for now our prime aim has to be to get the GPU and context ready.

"The GPU and context both need to be initialised to get them up and ready to accept those commands"

Look at the file **OGL.cpp**; we've put all the initialisation code in at the start. Now we can create an `OGL` class, and call its `init` method, to set up OpenGLES2.0 ready to do its magic.

You might at this point be wondering what OpenGLES2.0 is? Simply put, it's an interface system that allows you to control any GPU which is using the OpenGLES2.0 standard. Different manufacturers' GPUs work slightly differently at the hardware level, so the interface or API gives a consistent way to ask for certain specific things to happen in certain ways. The hardware/driver coders make sure this happens regardless of the GPU architecture.

CHECK THE ERRORS

Errors can be confusing. It's best to focus on and fix the first couple: fixing those often fixes the others.

TYPE, DON'T PASTE!

Typing slowly and carefully forces you to think about what you enter, and helps you to manage your code.

Getting a graphic screen

Initialisation of our OpenGLES2.0 systems only needs to happen once, so it can sit happily as a first constructor step in our **Game** class. To use the **OGL** class, simply make sure you have added the **OGL.h** and **OGL.cpp** files to the Hello World project, then add an **#include OGL.h** to your **Game** and **SimpleObj** class header files. We also need to add an instance **OGL** in the **Game** class description, like this:

```
OGL   OGLES;
```

Now make sure your **Game** constructor has **OGLES.Init()** and we're ready to go. Build and Run your new code, and your Hello World console will still function as before, but now you will see you have two windows.

The console screen is still there if we need it for outputting text, but now we also have a graphic screen. But how do we draw something to it? That comes back to those commands we need to send, and our currently empty **Draw** function in the **SimpleObj** class.

> "The console screen is still there if we need it for outputting text, but now we also have a graphic screen"

In the new **SimpleObj** class (download) you will see there are a few additions, most notably a data structure called an array which holds data for a triangle, and the now completed **Draw** function.

The **Triangle** structure is a list of offset points, from an assumed centre of 0,0,0, which basically defines the vertices that make up a triangle. There are three points, each with an x, y and z coord, though z is always 0 for 2D.

Now, if we send those points to the GPU and let it know it's being asked to draw a triangle, it will store the data in its memory and then send each point to a pair of shaders. This is where OpenGLES2.0 becomes complex. But fear not: shaders are only confusing for a little while, then you start to realise just how amazing they are.

What's a shader?

Shaders are a big topic – in graphics terms, quite possibly one of the biggest – so we won't be able to go into much detail here, but for sure we will cover them later. Let's just start with a very basic outline and build in a working system we can use.

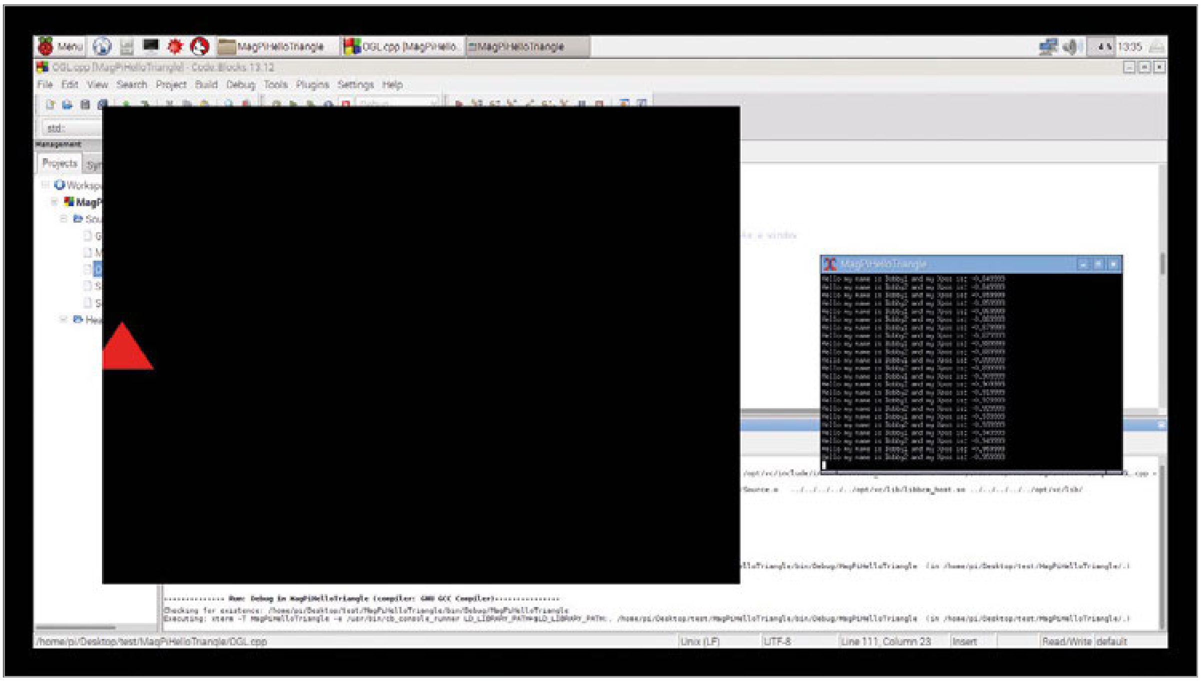
We always have a pair of shaders which work together, one after the other. The first shader is the vertex shader. This contains code which works out where in our screen we locate a vertex. Once that's set up, it then passes control to a fragment shader which is there to basically decide on what colours to use to fill the area defined by the vertices.

What's both confusing and cool about shaders is that we have to write them ourselves. This effectively gives us total control over what gets drawn on screen and where. Which can be awesome or terrible depending on how well we do it.

We've set up a very simple pair of shaders in the OGL init. That's not really a good way to do things, but it'll work for now.

Now, the last complication to cover with shaders is that they exist in the GPU memory, not in our code memory, so we have to send them information

Figure 3 Only one triangle?



BE FLEXIBLE!

Things change, update, get revised; don't get stuck into fixed thinking!

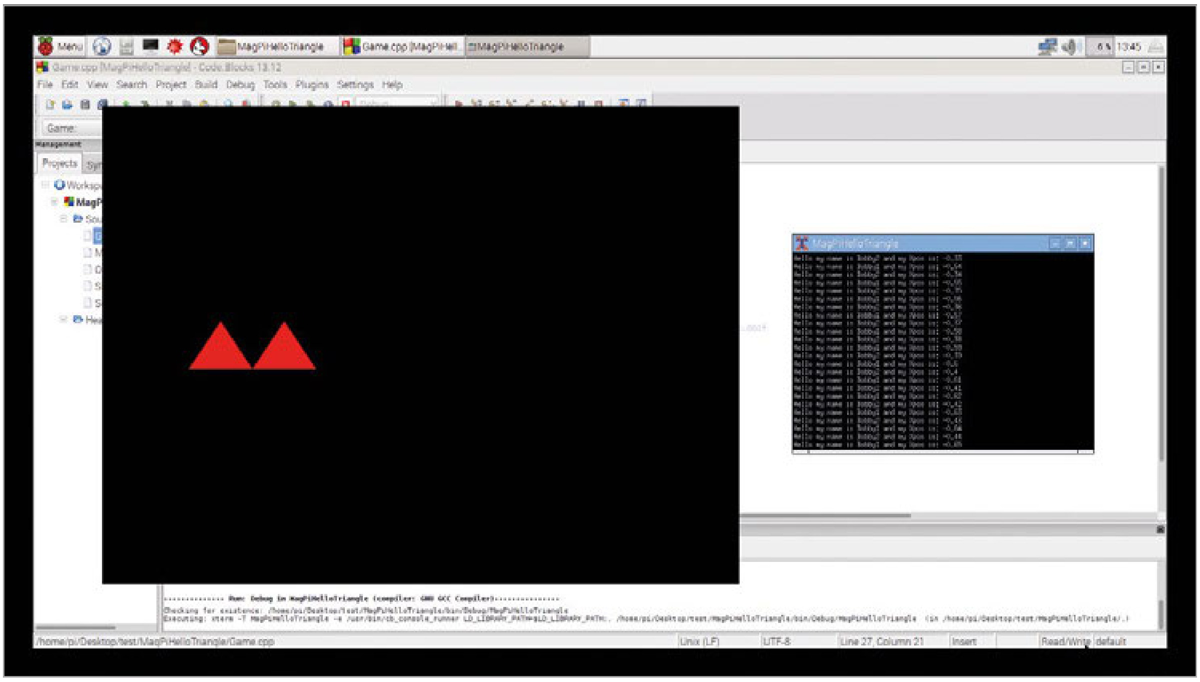


Figure 4: And our two objects now have graphics

LIBRARIES UPDATE TOO

Sometimes the libraries mentioned will change; try to find the new versions. Listen to your linker errors!

to do things. Data we send to the shader are called attributes or uniforms. Attributes are usually lists of changeable supplied data, such as the position of each vertex, which are read in turn as the shader operates. Uniforms are unchanging constant values set up once, before the gldraw is called.

Are you getting bored with all this tech description yet? It is quite a lot of info and new buzz words for a beginner but we are nearly there and can soon start to put things on screen.

Let there be triangles, please?

The GitHub project is fully working, but try to add **OGL.cpp/h** and then edit your current Hello World project to gain these new graphics – it's good practice.

The SimpleObj draw routine is basically setting up the attributes it wants sent to the GPU (i.e. the triangle data) and then sends the uniform info that is used to modify those attributes. Once set up, an actual draw call is sent to the GPU to tell it to do its thing.

But that's not the end of it. The OpenGLES2.0 draw call does not write to the visible screen – instead it writes to a back buffer. This allows us to do multiple draw calls, without seeing the screen glitching. When all our draw calls are done, we ask the GPU to swap the back buffer to the visible screen; that's done at the end of the game update loop, and then we see the results of our efforts.

Add in the new code from the **`draw`** function in **Figure 2**, or refer to the new **SimpleObj.cpp** file.

If you Build and run (**F9**) now, with no changes to your update, you can see you are getting one triangle (**Figure 3**). But you are outputting text for two objects. Any ideas?

It's not as complex as it might seem: there are in fact two triangles there, but they are both occupying the same position, so we only see the last one drawn. Also, the update routine should be responsible for moving them around, so that when the draw sends its position as a uniform to the GPU, it will send a unique position allowing us to see both triangles.

Make the changes seen in the new **SimpleObj.cpp** file, to the update routine; but also when you create them in the **`Game`** class, add a different x,y position for each of them.

Build and run it again and something wonderful should happen: we should be seeing two red triangles moving around the screen and bouncing off the edges (**Figure 4**). This is rather a limited demo, though, and we are restricted to x and y co-ords of -1.0 to +1.0, which is hard to visualise, but we'll fix that next time.

We have objects with graphics... again, very simple graphics, and not especially useful unless you are writing a bouncing triangle game. But, we do have graphics. The next step is to create a more useful type of graphic, with a recognisable pattern on it.

Sadly, that's going to need a bit more techy work. This foundation work is dull, but it's done now; next time we can start to do some actual control code.

LEARN TO CODE WITH C

Enjoying using C and C++ and can't wait until next month for more? Check out our Essentials book, *Learn to Code with C*, for more C tutorials for beginners: **magpi.cc/learn-c-book**

FREQUENTLY ASKED QUESTIONS

Your technical hardware and software problems solved…

NEED A PROBLEM SOLVED?

Email **magpi@raspberrypi.org** or find us on **raspberrypi.org/forums** to feature in a future issue.

MAGPI QUESTIONS

WHERE CAN I BUY THE MAGPI?

In store

The MagPi is available to buy in stores in the UK and US, although we've heard of some smaller stores in other countries stocking the magazine. You can find *The MagPi* on the last Thursday of every month in selected WH Smith, Sainsbury, Tesco, and Asda stores. In the US, you can get it a few days later from Barnes & Noble and Micro Center.

Online

All issues of *The MagPi* are sold on day of release from the Raspberry Pi Press store: **store.rpipress.cc**. These ship to just about everywhere in the world. You can also buy issues of HackSpace magazine, along with all our Essentials books and Projects books.

Digitally

As well as the free PDF download, you can get our dedicated iOS and Android app; this allows for better reading options on your device. You can buy single issues or get a digital subscription through the app – and you get the first 30 issues of the magazine completely free.

HOW CAN I SUBSCRIBE TO THE MAGPI?

Subscription offers

We offer a number of subscription offers for *The MagPi* on our website, which you can find here: **magpi.cc/Subs1**. We have specific offers for people living in the UK, Europe, and the USA, as well as our 'Rest of the World' option for other countries. For example, Canada is included in the Rest of the World option.

Eligible countries

We do not have any restrictions on where the Rest of the World subscription option can ship to. The cost of postage is included in the price you pay as well. It is much cheaper to buy a subscription than to buy the individual issues as they're released, especially when you don't live in the UK.

Free Pi Zero W

With a twelve-month subscription to the print version of the mag, you also get a free Pi Zero W along with a case and a selection of adapter cables. This offer currently does not have an end date, but make sure to check the descriptions on our subscription package to make sure it hasn't changed since the time of writing.

WILL YOU BE REPRINTING MORE COPIES?

MagPi issues

Generally when we sell out of copies of *The MagPi*, there will not be a reprint of that issue. While we have done so in the past with issue 40, that was a special case. If we ever have plans to print more of an issue, we'll let people know via our social media channels or via our newsletter: **magpi.cc/Email-me**.

Essentials books

We do often reprint the Essentials books, so if you've missed one you'll just need to wait a while and they'll eventually be back in stock. Some books have higher demand than others, though, so make sure not to miss out if you need one ASAP!

Other books

The project books we generally keep in print, and the new Beginner's Book will definitely be available in print for the foreseeable future. However, other books will go in and out of print with demand. Remember, everything we release is also available as a free PDF – so if you can't buy a physical copy, you can print off the bits you need.

FROM THE RASPBERRY PI FAQ RASPBERRYPI.ORG/HELP

WHAT OPERATING SYSTEM (OS) DOES IT USE?

There are several official distros available on our downloads page. New users will probably find the NOOBS installer the easiest to work with, as it walks you through the download and installation of a specific distro. The recommended distro is Raspbian, which is specifically designed for the Raspberry Pi and which our engineers are constantly optimising. It is, however, a straightforward process to replace the root partition on the SD card with another ARM Linux distro, so we encourage you to try out several distros to see which one you like the most. The OS is stored on the SD card.

DOES IT HAVE AN OFFICIAL PROGRAMMING LANGUAGE?

The Raspberry Pi Foundation recommends Python as a language for learners. We also recommend Scratch for younger children. Any language which will compile for ARMv6 (Pi 1, Zero, A+, and later models), ARMv7 (Pi 2 and Pi 3), or ARMv8 (Pi 3 and later Pi 2, when running a 64-bit kernel) can be used with the Raspberry Pi, though, so you are not limited to using Python. C, C++, Java, Scratch, and Ruby all come installed by default on the Raspberry Pi.

WILL IT RUN WINE (OR WINDOWS, OR OTHER X86 SOFTWARE)?

In general, this is not be possible with most versions of the Raspberry Pi. Some people have put Windows 3.1 on the Raspberry Pi inside an x86 CPU emulator in order to use specific applications, but trying to use a version of Windows even as old as Windows 98 can take hours to boot into, and may take several more hours to update your cursor every time you try to move it. We don't recommend it! A special version of Windows 10 is available for use on the Raspberry Pi 2 and 3. This is an entirely new version of the operating system designed exclusively for embedded use, dubbed the Windows 10 Internet of Things (IoT) Core. It does not include the user interface ('shell') or the desktop operating system.

WILL IT RUN THE WINDOWS 8 ARM EDITION?

No. Most models of Raspberry Pi lack the minimum memory and CPU requirements to support Windows 8 ARM edition. The Raspberry Pi also lacks the appropriate axis sensors, and there are many other limiting factors which mean that running Windows 8 ARM edition is not possible.

BUILD AN OCTAPI

Make your own distributed computing system and learn to hack Enigma codes

Cluster computing, or distributed computing, is a hot topic in today's computer landscape. Rather than use a single computer to perform calculations, you string a whole bunch of them together to form a single system.

Cluster computers are often used to perform complex calculations that would take a single computer a long time. Each computer (referred to as a 'node') handles part of the task. Typically, a cluster computer is faster than a single computer and, because they're made of mass-produced parts like our Raspberry Pi, also much cheaper than a supercomputer.

They're not faster at everything, though. You won't be able to run faster games, or run more complex software with a cluster of Raspberry Pi computers. But you will learn all about some of the most important and advanced computing techniques around today, such as cyrptography, including encryption, decryption, and much more.

In this feature we look at building a cluster computer called OctaPi. This was developed by GCHQ (Government Communications Headquarters) who worked with the Raspberry Pi Foundation to develop resources.

These resources are some our most advanced educational materials on the Raspberry Pi website, and are ideal for those looking to study computer science.

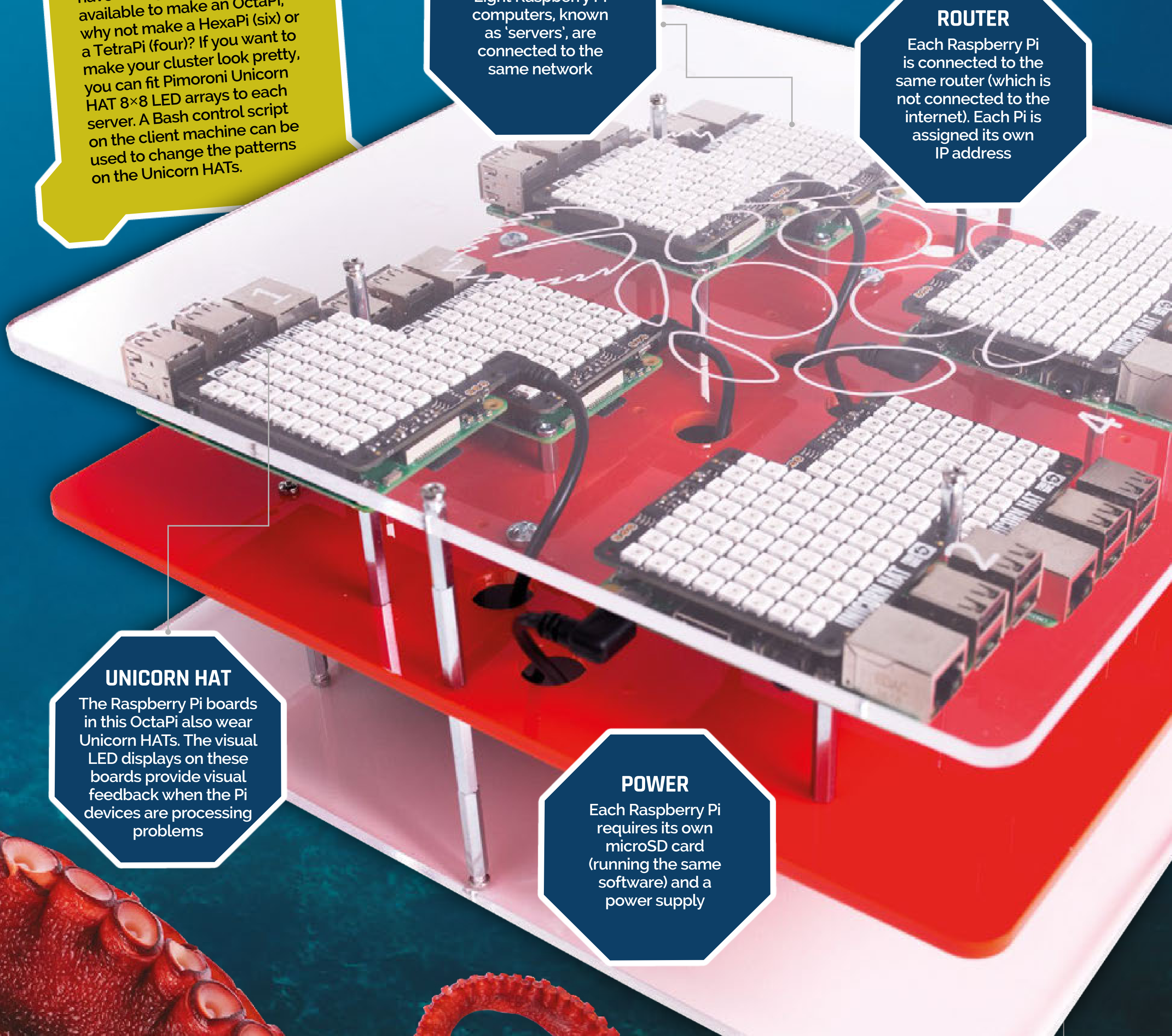
> Learn some of the most important and advanced computing techniques around

HOW MANY PI BOARDS?

You don't actually need eight servers, as the cluster will work with any number of servers up to limits determined by the performance of your WiFi router. If you don't have enough Raspberry Pis available to make an OctaPi, why not make a HexaPi (six) or a TetraPi (four)? If you want to make your cluster look pretty, you can fit Pimoroni Unicorn HAT 8×8 LED arrays to each server. A Bash control script on the client machine can be used to change the patterns on the Unicorn HATs.

LICENCE

CODE AND SCRIPTS



LAURA SACH

Laura creates and maintains Raspberry Pi educational resources. Aside from computers, she loves cats, cakes, board games, and making jam.
raspberrypi.org

BUILD A CLUSTER COMPUTER

The OctaPi is a cluster computer, which uses several computers to boost its power. Discover how to connect Pi boards together to form a cluster

This system is known as a cluster computer, a kind of cloud computer. It's made using eight Raspberry Pi computers (known as 'servers'), all controlled by a single Raspberry Pi (known as the 'client').

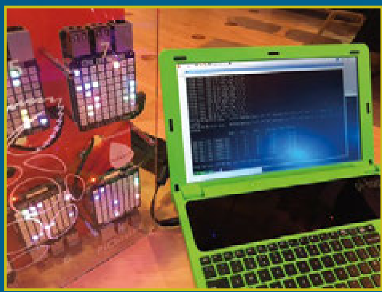
The power of the eight server CPUs (32 cores) will allow you to execute computations from the client CPU much faster than the client could perform them on its own. Once you complete this project, you will be able to develop applications in Python 3 on the client and run them on your cluster.

You'll Need

- 9 × Raspberry Pi 3s
- 8 × Unicorn HATs (optional)
- 8 × Short micro USB cables
- Wireless router
- Power hub
- Ethernet cable

There are three steps you will need to take to make an OctaPi:

- Create a WiFi network for the cluster using a dedicated router
- Create a client machine
- Create eight servers



The cluster_action script is used to control the Raspberry Pi boards

The processors in your OctaPi cluster will communicate via a dedicated local WiFi network established by a single wireless router. The router does not need to be connected to the internet at all for operation of the cluster, nor does it need to be online at a setup. We will assume you are using a brand-new router or have reset your router to its default settings.

Connect a computer to the router using an Ethernet cable. You can use any computer system that has a web browser, including a working Raspberry Pi 3.

Follow the setup instructions that came with the router. This normally involves opening a web browser and navigating to your router's 'admin' page to start changing the router settings. The 'admin' login credentials will have been provided by your WiFi router manufacturer.

Look for a page which allows you to set the WiFi network name (also called SSID) and change it to 'OctaPi'. For example, the page may look like the one shown in **Figure 1**.

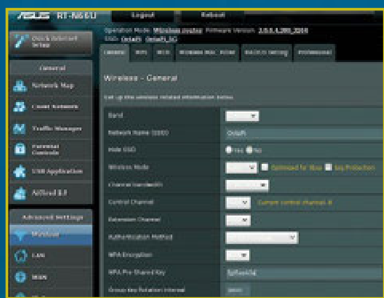
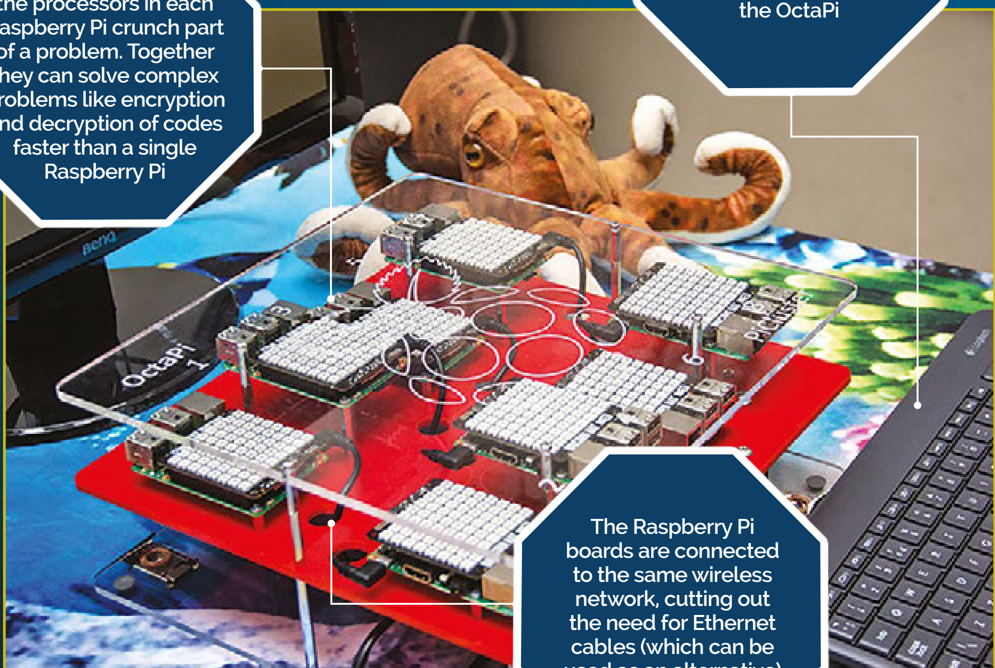


Figure 1 A router interface allowing you to change the WiFi network name

Raspberry Pi 3 computers only work with 2.4GHz WiFi, so you can either ignore 5GHz settings or disable 5GHz WiFi in your router.

Now look for the LAN IP settings, which may be under the 'LAN' settings. Change the IP address of your router to 192.168.1.1 – again, each router's administrator interface will be different, but **Figure 2** shows an example of what you might see.



Once up and running, the processors in each Raspberry Pi crunch part of a problem. Together they can solve complex problems like encryption and decryption of codes faster than a single Raspberry Pi

A ninth Raspberry Pi, known as the 'client' is used to control the eight Raspberry Pi 'servers' that form the OctaPi

The Raspberry Pi boards are connected to the same wireless network, cutting out the need for Ethernet cables (which can be used as an alternative)

You may need to reboot your router and log back in as 'admin' after this step.

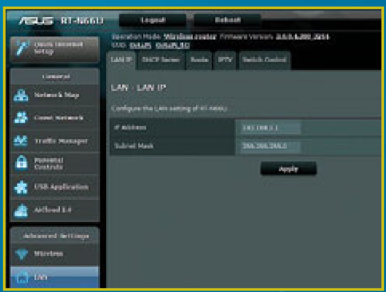


Figure 2 Change the LAN address of your router to 192.168.1.1

Set the network

Now set the WiFi network password, which may be under 'Wireless security' or similar. It is important to ensure that you write down the password so that you can use it to log into your dedicated 'OctaPi' network.

Look for the DHCP settings. DHCP is a protocol used for issuing IP addresses automatically. The client and servers will use this to determine their IP addresses. The settings for DHCP may be under 'LAN'. Make sure that DHCP is enabled, and set the DHCP address range to something that provides a useful range of addresses; we chose 192.168.1.2 to 192.168.1.254. Using this particular range is not critical, but using a different one here will mean the IP addresses you see will differ from those shown in this guide. Only choose a different range if you know what you are doing. See **Figure 3**.

Figure 3 Turn DHCP on and set the IP range from 192.168.1.2 to 192.168.1.254

If there is a setting for the IP lease time, make this number as large as possible. The lease time is the length of time before DHCP reallocates IP addresses – you need it to be long to avoid interrupting the connection between the client and servers.

Reboot your WiFi router so that all the changes come into effect.

The OctaPi client

One of the Raspberry Pi computers will be used as the client machine giving access to the servers in the OctaPi cluster. You will need to connect the usual peripherals (monitor, keyboard, mouse) to this Raspberry Pi in order to use it to control the OctaPi.

On your microSD card, install the latest version of Raspbian by following the software guide instructions **(magpi.cc/2eopaEf).**

Using this microSD card, boot up the Raspberry Pi 3 with a keyboard, screen, and mouse connected. Ensure the Raspberry Pi is also connected to the internet.

Now open a Terminal window and install dispy by typing this command into the Terminal:

```
sudo pip3 install
dispy==4.7.1
```

Dispy is a distributed Python implementation that will allow you to write code on the client and run it across the servers. Note, It is crucial that you install version 4.7.1 of dispy, as later versions rely on a library that is currently incompatible with Raspbian.

Further information is available from dispy: Distributed and Parallel Computing with/for Python (**magpi.cc/2mkiXid**).

Next, install nmap by typing this command into the Terminal:

```
sudo apt-get install nmap
```

Nmap is used to discover the IP addresses of the Raspberry Pi servers forming the OctaPi cluster, so that they can be shut down or rebooted as needed.

If you are using the optional Unicorn HATs, install the software for them by typing this command:

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

You will need to reboot your Pi after installing this software. Make sure you are in the **/home/pi** directory, then download the OctaPi client software by typing this command into the Terminal:

```
git clone https://github.
com/raspberrypilearning/
octapi-setup.git
```

EIGHT SD CARDS

It is also possible (but time-consuming) to create your eight SD cards by following the server setup instructions eight times. If you choose to use this method instead, you must run the **`ssh-copy-id`** command on the client once per server to copy the key across to each server separately. DO NOT use the **`ssh-keygen`** command to regenerate the key. Refer back to the client setup instructions for how to copy the key from the client to the server.

BEOWULF BRAMBLE

Cluster computers like this OctaPi are also known as 'Brambles' and 'Beowulf clusters'. The name Beowulf comes from a computer built by Thomas Sterling and Donald Becker in 1984 at NASA.

The client software contains source code examples in Python 3 and a Bash control script for rebooting and shutting down the cluster. The control script can be used with the Unicorn HAT as well. Move all of the files from the client folder you just downloaded into the **home/pi** folder:

```
mv /home/pi/octapi-setup/
client/* /home/pi
```

> Do not run nmap on a network that is connected to the internet

Shut down your new OctaPi client for the time-being and set aside the client SD card in a safe place.

The OctaPi servers

Each of the Raspberry Pi 3 computers in the cluster needs to have its own microSD card prepared. However, because each card is identical, you can set up just one server, check it's working, and then replicate the SD card for the other servers.

On a new microSD card, install the latest version of Raspbian by following the software guide instructions **(magpi.cc/2eopaEf)**.

Boot up a Raspberry Pi 3 using this SD card with a keyboard, screen, and mouse connected. Ensure the Raspberry Pi is connected to the internet.

Install dispy by typing this command into the Terminal:

```
sudo pip3 install
dispy==4.7.1
```

Install psutil by typing this command into the Terminal:

```
sudo pip3 install psutil
```

Dispy uses psutil for reporting CPU usage of the servers in the cluster.

Ensure you are in the **home/pi** directory by typing **cd /home/pi**. If you are using the Unicorn HATs, install the software for them by typing this command into the Terminal:

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

You will need to restart your Pi afterwards. Then, reopen the Terminal and make sure you are in the **home/pi** folder by typing **cd /home/pi**.

Download the **start_unicorn.sh** Bash script for installation on the server using this command:

```
wget https://raw.
githubusercontent.com/
raspberrypilearning/octapi-
setup/master/server/start_
unicorn.sh
```

Make the script executable by typing this command:

```
chmod u+x ./start_unicorn.sh
```

Go back to the Terminal window and type the following command to begin editing the **/etc/rc.local** file:

```
sudo nano /etc/rc.local
```

Near the bottom of the file, just before **exit 0**, add the following lines to run dispy as a daemon (a process running in the background) each time the server boots:

```
sleep 20
_IP=$(hostname -I)
/usr/local/bin/dispynode.py
-i "$_IP" --daemon --client_
shutdown
```

The sleep for 20 seconds is to allow time for the server to log onto your WiFi router and obtain a network IP address from it. You need the IP address so that the server will listen properly for the client on the network. You may have to adjust this delay to suit the router you are using.

Press **CTRL+O** to save your changes, then **CTRL+X** to exit the nano editor.

Check that remote login via SSH is enabled so that remote command-line access to your server is possible. In the Preferences menu, select Raspberry Pi Configuration. Then click on the

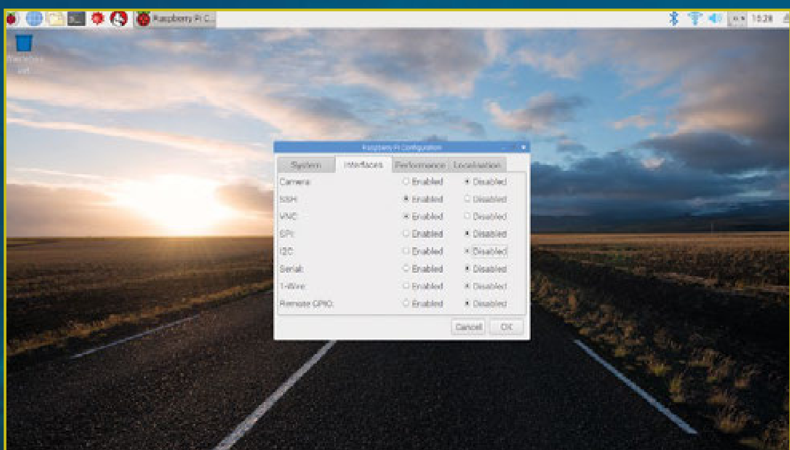


Figure 4 Enable SSH in the Interfaces tab in Raspberry Pi Configuration

TROUBLESHOOTING YOUR OCTAPI

See the Raspberry Pi resources website for OctaPi troubleshooting advice: magpi.cc/2mi2xgU

Interfaces tab and make sure that SSH is enabled (see **Figure 4**).

Set up the server

Still using the server Raspberry Pi, we need to switch from the internet-connected network to the OctaPi network. Ensure your OctaPi WiFi router is powered up and fully booted.

Click on the WiFi symbol at the top of the desktop and select the 'OctaPi' network. Enter your router's network password (which you noted down earlier) to join the network.

Having done this, the server will remember the WiFi credentials and log onto your dedicated 'OctaPi' network each time it boots.

We need to remove any previous WiFi information to avoid confusion. In a Terminal window, type the following command to edit the **wpa_supplicant.conf** file:

```
sudo nano /etc/wpa_
supplicant/wpa_supplicant.
conf
```

The file contents look like this:

```
ctrl_interface=DIR=/var/run/
wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
     ssid="OctaPi"
     psk="mynetworkpassword"
     key_mgmt=WPA-PSK
}
```

Remove any 'network { }' sections for other networks (do not remove the OctaPi one shown here). Press **CTRL+O** to save and **CTRL+X** to exit.

If alternative WiFi networks are not removed, your server may log into the wrong network and not be available to the client.

When finished, shut down the server Raspberry Pi.

Set up the client

Now connect a monitor, keyboard, and mouse and power up a Raspberry Pi containing the client SD card you created earlier.

Repeat the instructions from the previous step on this computer, logging into the 'OctaPi' network and removing any alternative WiFi details from the **wpa_supplicant.conf** file.

Keeping the client switched on, boot up a single server Raspberry Pi with only a power lead connected.

Open a Terminal window on the client Raspberry Pi. Type the following command to find the IP address of the client Raspberry Pi:

```
hostname -I
```

Type the following command to find out the IP address of the server Raspberry Pi:

```
nmap -sP 192.168.1.*
```

Make a note of the server IP address (you should see the router's address, which will be 192.168.1.1, and the client's address listed, too).

The nmap software scans the network to find the IP addresses of the devices connected to it. We need to do this on our local network so that the client machine can communicate with the Raspberry Pi computers which form the OctaPi cluster. Do not run nmap on a network that is connected to the internet. Nmap is a powerful piece of software and using it to scan a network you do not own may be considered hacking, and in some countries may even be illegal.

Each Raspberry Pi needs its own power supply. In this OctaPi build, a powered USB hub is used to provide power to all the boards. You can also use multiple Raspberry Pi power adapters

On the client, run **ssh-keygen** in the Terminal to create a key for authenticating the client with the server.

Press **ENTER** when asked where to save the key, and press **ENTER** again twice when asked for a passphrase, leaving it empty.

This key is used to help the **cluster_action.sh** script (provided with the client software) to run with the servers.

Find where you noted down the IP address of the server machine, and run this command in a Terminal on the client to copy the key to the server (replace **<remote ip>** with the server's IP address):

```
ssh-copy-id -i ~/.ssh/id_
rsa.pub <remote ip>
```

You will be asked if you want to continue connecting. Type yes

Node	CPUs	Jobs	Sec/Job	Node Time Sec
192.168.1.3 (raspberrypi)	4	16	12.841	205.460
Total job time: 205.460 sec, wall time: 57.347 sec, speedup: 3.583				

Figure 5

POWER IT UP

It is essential that there is sufficient power to provide 2.4A to each Raspberry Pi 3. Not all USB hubs/chargers can power 2.4A on every port, so make sure to check this carefully.

Figure 6

Node	CPUs	Jobs	Sec/Job	Node Time Sec
192.168.1.49 (raspberrypi)	4	4	16.040	64.160
192.168.1.202 (raspberrypi)	4	2	12.031	24.062
192.168.1.191 (raspberrypi)	4	2	13.029	26.058
192.168.1.223 (raspberrypi)	4	0	0.000	0.000
192.168.1.116 (raspberrypi)	4	2	10.025	20.050
192.168.1.27 (raspberrypi)	4	2	15.535	31.070
192.168.1.167 (raspberrypi)	4	4	14.537	58.148
192.168.1.50 (raspberrypi)	4	0	0.000	0.000
Total job time: 223.548 sec, wall time: 20.245 sec, speedup: 11.042				

and press **ENTER**. You will be asked for the server Pi's password, which will be the default password of **raspberry**.

This completes preparation of the client and server. We now need to check everything is correct and working properly with a single server.

Check that it works

Make sure your dedicated 'OctaPi' router is powered on and fully booted up, the client is booted with peripherals attached, and the server is booted with only a power lead attached.

> "We can't wait to hear how you got on with this challenge!"

Open a Terminal on the client. Make sure you are in the **home/pi** directory and type the following command to run the **compute.py** example software provided with the client software examples you downloaded earlier:

```
sudo python3 compute.py
```

The **compute.py** Python script runs 15 jobs on your server. They are all just random delays before returning. If the OctaPi is working correctly, the jobs will complete in about a minute and a table showing the statistics for the application will be shown in the Terminal. You should see a result similar to **Figure 5**.

If the **compute.py** script does not work, review your steps one by one and check that client, server, and router are all set up correctly and working properly.

If the test worked, use the client to manually shut down the server (replacing **<remote_ip>** with the IP address of the server you noted down earlier):

```
ssh <remote_ip>
sudo shutdown -HP now
```

You may need to use nmap again to find the server IP address if it changed when the WiFi router rebooted. In future we will be using the **cluster_action.sh** script to do this. Once the server is shut down, remove its microSD card.

Using an SD card duplicator or a computer that is able to read SD cards, create seven more identical copies of this SD card and insert them into the other servers so that you have a total of eight.

OctaPi physical setup

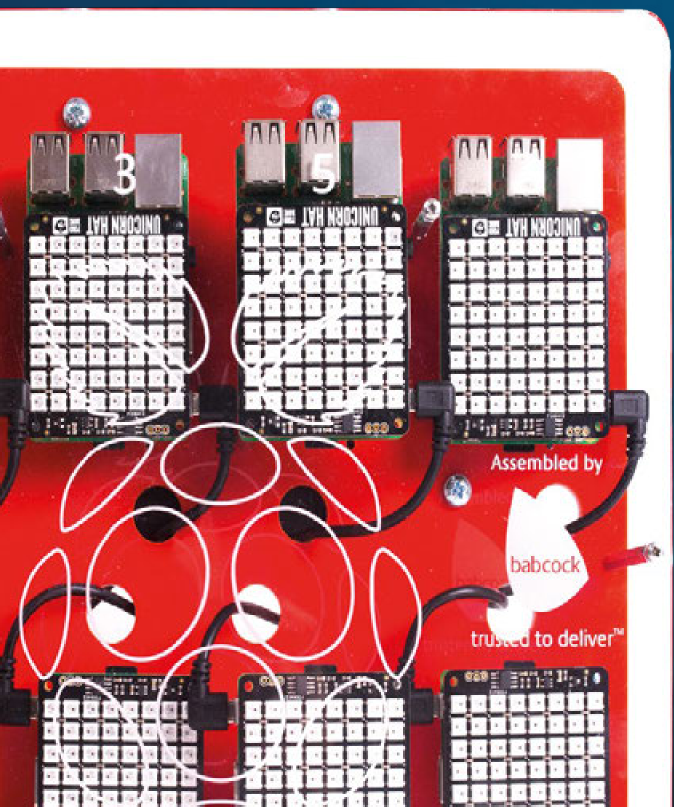
Decide how you will power your OctaPi. You could simply power each of the eight Raspberry Pi computers separately using eight standard power supplies, or you may find it more convenient to use a USB hub or charger in order to power them centrally.

If you are using the Unicorn HATs, install a HAT onto the GPIO connector of each of the servers.

If you wish, you can mount the eight Raspberry Pi 3s onto a backboard. Alternatively, there are cases for multiple Raspberry Pis available for purchase, or you can simply leave them laid out – they will work as a cluster even if they aren't attached to anything.

Using the completed OctaPi

Ensure that the dedicated WiFi router,



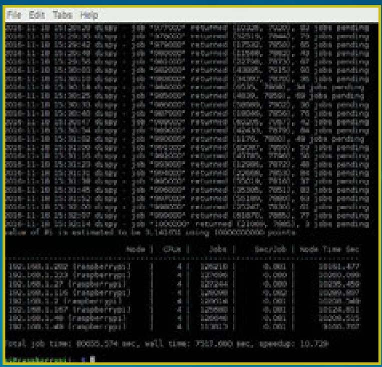


Figure 7 Estimating the value of pi using the 'dartboard' method

client, and OctaPi servers are all powered up. It's best to power up the WiFi router well in advance so that you can be sure it is fully booted before the OctaPi client and servers attempt to log into it.

On the client machine, open a Terminal. Try running the **compute.py** example software again by typing the following command:

```
sudo python3 compute.py
```

If the OctaPi is working correctly, at the end of the run all the servers used to execute the job will be listed in the table. The result should look similar to **Figure 6**.

Try running the **compute_pi_efficient.py** example software:

```
sudo python3 compute_pi_
efficient.py 1000 100000
```

If the OctaPi is working properly, you can start to do useful calculations, like this estimation of the value of pi using the 'dartboard' method (**Figure 7**).

Controlling the cluster

Now that you have checked your OctaPi is perfectly set up and running without a hitch, you can use the **cluster_action.sh** script to control it.

The **cluster_action.sh** script runs on the client and uses SSH to administer the servers (that's why we used **ssh-keygen** to authenticate the client with the servers). It relies on the correct IP addresses of the servers being listed in the **ip_list** file. It is a good idea to delete the **ip_list** file when booting the cluster for the first time so that the list is regenerated.

From a Terminal, type the following command to remove the **ip_list** file:

```
rm ip_list
```

Setting up the cluster_action script

On the client machine, open a Terminal. Make sure you are in the **home/pi** directory by typing in **cd /home/pi**.

Now set the permissions for the cluster action script so that you can run it by typing this command:

```
chmod u+x ./cluster_action.sh
```

Ensure the client can recognise each server SSH key The first time you use the cluster, you may need to connect to each server via SSH with the client so that it recognises each server's SSH key properly (be sure to replace **<ip address of server>** with the actual server IP address).

```
ssh <ip address of server>
```

If at all, doing this will only be needed once.

We can't wait to hear how you got on with this challenge!

The following parameters are accepted by the cluster_action script:

REBOOT

Reboots all the servers (the client and router are ignored). For example:

```
./cluster_action.sh reboot
```

SHUTDOWN

Shuts down each server and places it into a safe state. If a server is not shut down correctly, it may cause the microSD card to be corrupted and lead to the processor failing to boot when next used. For example:

```
./cluster_action.sh shutdown
```

DATE

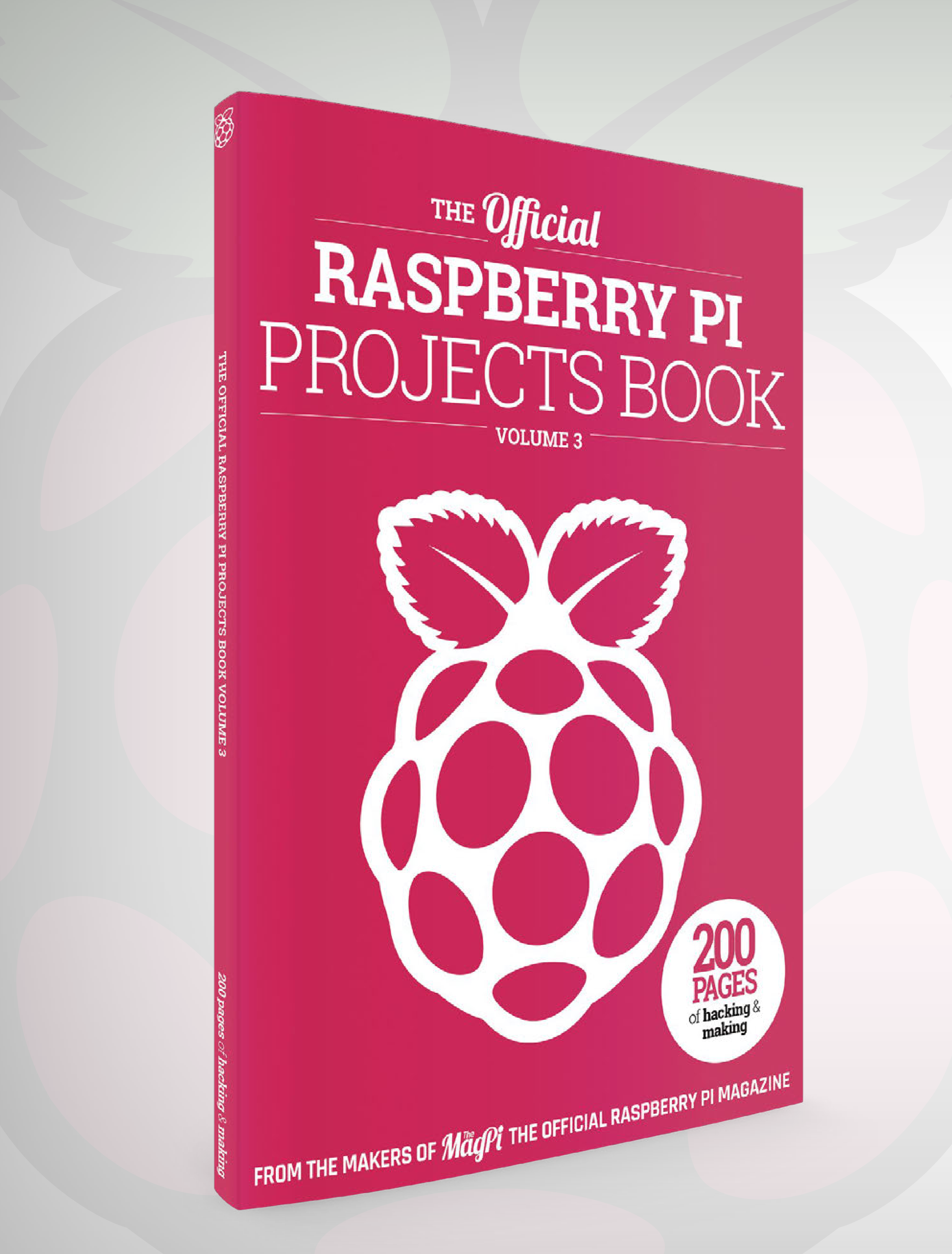
Distributes the client date and time (to the nearest minute) to each server. The Raspberry Pi 3 does not have a real-time clock, so the correct time will need to be set on the client first. For example:

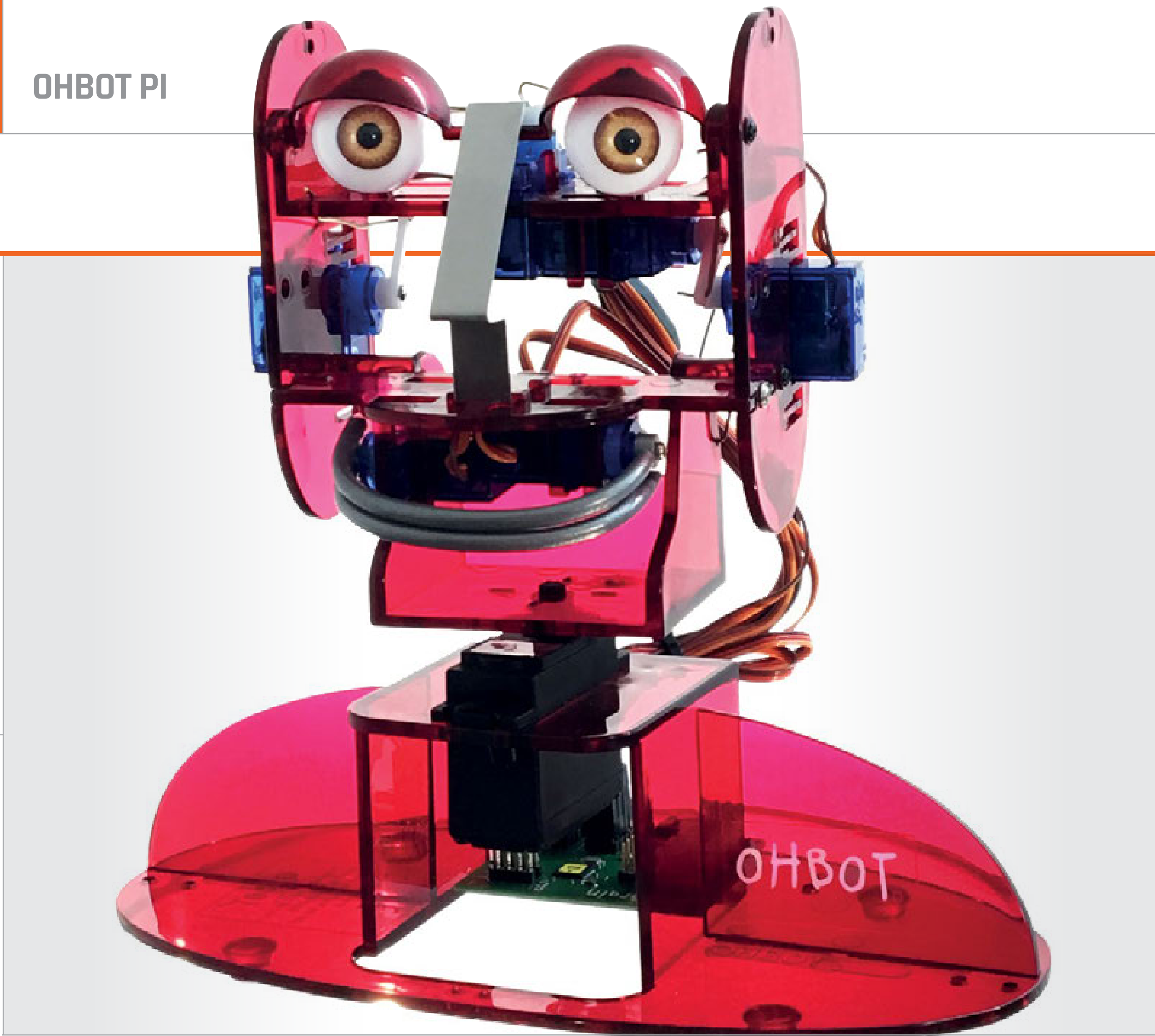
```
sudo date -s "11 Apr 2017 12:42"
./cluster_action.sh date
```

UNICORN

Invokes the **start_unicorn.sh** script on each server and passes it the name and location of a Pimoroni Python script as a parameter. For this to work you need to have **start_unicorn.sh** in **/home/pi** on each server as described earlier. For example:

```
./cluster_action.sh unicorn /home/pi/Pimoroni/
unicornhat/examples/random_sparkles.py
```


THE Official
RASPBERRY PI
PROJECTS BOOK
VOLUME 3
200 PAGES of hacking & making
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
THE OFFICIAL RASPBERRY PI PROJECTS BOOK VOLUME 3
200 pages of hacking & making



Maker Says

Stretch your pupils' computational thinking and understanding of computer science

Ohbot

OHBOT PI

The educational robot face now comes with a Raspberry Pi version. How educational is it, though?

Related

MEARM PI

A robotic arm that you can control with the Raspberry Pi. It's a little less hackable than the Ohbot, though.

£70 / $96

magpi.cc/2waqp7b

We'll be honest, the first time we saw the Ohbot we thought it was a little creepy. You can't help but be drawn to the eyes on the almost skeletal face, especially when it's not moving and forever unblinking. It's like some kind of biology classroom specimen reject. The reason we're bringing this up is because we expect other people might think the same and we can tell you, right off the bat, that this slight sense of unease does dissipate quite quickly when you're up close and personal using one.

Face the facts

With that out the way, let's actually look at the Ohbot. It's a robot face that you can program in Windows and now with a Raspberry Pi. It has a series of motors you can control that give it a wide range of head-like movement – from a simple rotation of the head to even being able to replicate blinking and lip movement. Yes, it has (metal) lips.

The version we reviewed came pre-constructed, with the robot held snugly in its box to keep it intact. However, you can get a slightly cheaper version (£119/$164) that you can construct yourself. The instructions suggest that it might take an hour or so to build, which sounds about right if you're an adult giving it your full attention. The parts are quite big and nothing is too fiddly, so supervising a younger maker would be a good way to get them to understand the device they're building.

One of the motor connections on pre-constructed version we received had wiggled free in transit, so it's worth giving the instructions a quick once over even if you did get the pre-constructed ohbot.co.uk

£159 / $219

version, to make sure yours survived the trip. Once complete, though, it sits nice and sturdily on a surface, just waiting for you to connect it all up.

Program with Pi

Connecting the Ohbot to a Raspberry Pi is pretty simple – all you need to do is hook it up via the special USB cable that comes in the box. This special cable splits on one end into two USB connectors – one with a red bead on the end that you need to plug into a USB power supply, and the other which goes into the Pi itself. There are no extra power supplies to connect and you don't need to wire anything into the GPIO.

The Ohbot is controlled with Python code, and you can install all the necessary libraries from the Terminal. There are instructions on what you need to install in the kit and on the website and it won't take you more than a couple of minutes to get ready.

Once installed, you can try out the example code or start programming your own routines by controlling the range of motion and speed of each individual motor. If you were counting during construction, you'll have noticed there are seven motors in the head – all of which are under your control.

> "The presentation, design, and look of the Ohbot has grown on us during the course of this review"

The Ohbot Python library is not the easiest thing to use, though. For example, if you want to turn the head, you'd need to use:

```
ohbot.move(1,3,2)
```

Here, 1 is the motor controlling the head, 3 is the new position of the motor, and 2 is the motor speed. You can substitute the motor number for a predetermined name, which in this instance would be **ohbot.HEADTURN**.

Perhaps this slightly tricky way of coding the robot will pay off in the long run with younger makers, forcing them to refer back to the documents and really learn how functions in Python work. At the very least, though, the text-to-speech function is automatic.

Talk to me

Using **ohbot.say** in a similar way to the Python 3 **print** command, the Ohbot will talk. Or at least, the string of text will be converted to sound and played through the speaker while the lips move to approximate the words being said. It works quite well, and it's fun to watch Ohbot chat away. Of course, you can control this as well, turning off the lip sync and adding delays to the audio and such.

It's definitely a very interesting and unique bit of kit. The presentation, design, and look of the Ohbot has grown on us during the course of this review and we think that kids will get a kick out of making it talk and move around. Hopefully they'll learn something in the process as well.

Last word

A fun educational project with a lot of potential. However, the Python library could be slightly easier to use. It works well with the Pi, though.

magpi.cc/2mH8PEX

£49 / $68

Maker Says

"A Raspberry Pi audio add-on board for audiophiles"

Nanomesher

NANOSOUND DAC PRO

A powerful DAC that also easily lets you build your own music box with an informative screen

There are a number of DACs available for the Raspberry Pi – digital-to-analogue converter add-ons that let you play high-quality audio from the Pi. We've reviewed many of them in past issues of *The MagPi*, so it's always interesting to see a DAC do something different. The NanoSound DAC Pro from Nanomesher is one of these devices that sets itself apart.

Like a lot of DACs, it comes as a HAT add-on for the Raspberry Pi, sitting snugly on top of the GPIO pins. This one covers the entire board, going over the top of the USB and Ethernet ports of a full-sized Raspberry Pi (think B+, 2, 3, etc.). However, it does this to offer more options – namely a little LCD screen and some physical buttons.

These buttons and screen are one of the most interesting parts of this solution, allowing you to use the Pi and the DAC together as an all-in-one music player which you can then hook up to your favourite speakers. There's even a 3D-printed case you can get, with the files downloadable for free. It all works together with Volumio, an open-source music player that works on the Pi and is optimised for playing your music in the highest quality possible. With a few tweaks you can get it to accept the button inputs of the DAC and display song information on the screen, which is very smart.

The kit even comes with a remote control you can control the system with as well. It's a really neat little package, and it still outputs the great-quality sound it's primarily meant to do.

As well as the fully featured Pro version we reviewed, there is a cheaper Basic version for $48 / £35 that doesn't come with a screen if you don't need it (although you can get a kit to solder one on). At the very least it does come with the buttons and the remote, which are arguably more important than the screen anyway.

It's a great bit of kit, and maybe something to consider as an alternative to our music box tutorial on page 20 of our media player projects feature…

Related

PI-DAC PRO

A simpler but no less powerful DAC, the Pi-DAC PRO includes just about everything the NanoSound DAC has, albeit without the built-in buttons and screen.

£42 / $58

magpi.cc/2DmpYJM

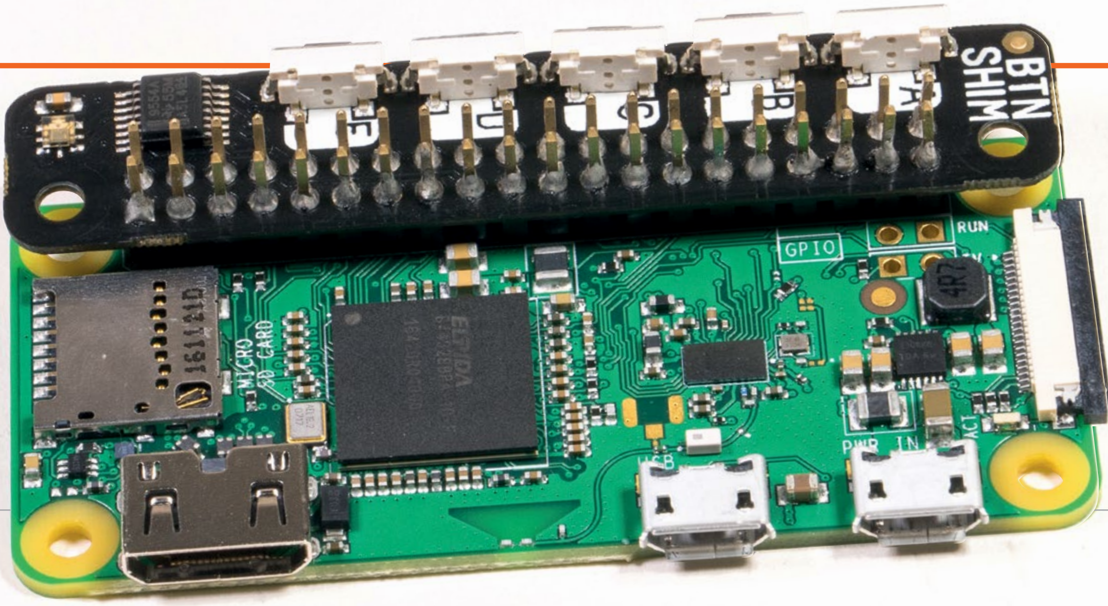
Last word

A lovely all-in-one music box solution for your Raspberry Pi that adds everything you'd need bar speakers. It sounds good as well.

★★★★★

magpi.cc/2EMg2Y1

£6 / $7



Maker Says

" Five handy physical buttons and an RGB status LED

Pimoroni

BUTTON SHIM

Add some buttons to your Raspberry Pi, without sacrificing any GPIO pins. A great way to upgrade an output HAT!

The Button SHIM from Pimoroni adds five buttons and an LED to your Raspberry Pi. So far, so simple. What's neat about it is that it gives you full use of all of your Pi's pins. The GPIO pins poke through the SHIM so you can get to them.

Most obviously, that means you can use the Button SHIM to add some input controls to an electronics project. If you're running out of GPIO pins, getting five buttons for free is a real plus.

The Button SHIM also plays nicely with many HATs, so you can use it to add input controls to an output HAT. The Unicorn HAT, for example, provides a grid of colourful LEDs that can be used for visual displays or scrolling messages. By adding buttons, you can create more flexible applications that can run without a keyboard.

The buttons stick out above the top edge of your Raspberry Pi. As a result, your Button SHIM project is unlikely to fit in an existing Pi case. Positioned at a right angle, the buttons press in towards the Raspberry Pi, and are labelled from A to E.

To set up the SHIM, you'll need to be competent with a soldering iron. The SHIM has a set of holes, so you slide it over your GPIO pins. You then need to solder several of the Raspberry Pi's pins to the board. It's a good idea to solder an additional couple of pins at the other end of the board to help hold the SHIM in place. As with anything that involves soldering the GPIO pins, it's an unavoidably fiddly job because the pins are only a few millimetres apart. If there is any excess solder on a pin, it might stop a HAT from sitting on the GPIO pins properly afterwards.

There is also a female header included which you can solder to the Button SHIM if you want to use it by itself.

The RGB LED in the corner of the board is a great bonus, and ideal for adding status signals to a project. The rainbow example program shows how the buttons can be used to change the light colour. It's a great 'hello world' demo, and a perfect showcase for the Python library.

Related

FOUR LETTER PHAT

The Four Letter pHAT displays letters or numbers. Combine it with the Button SHIM on a Pi Zero to make a compact alarm clock.

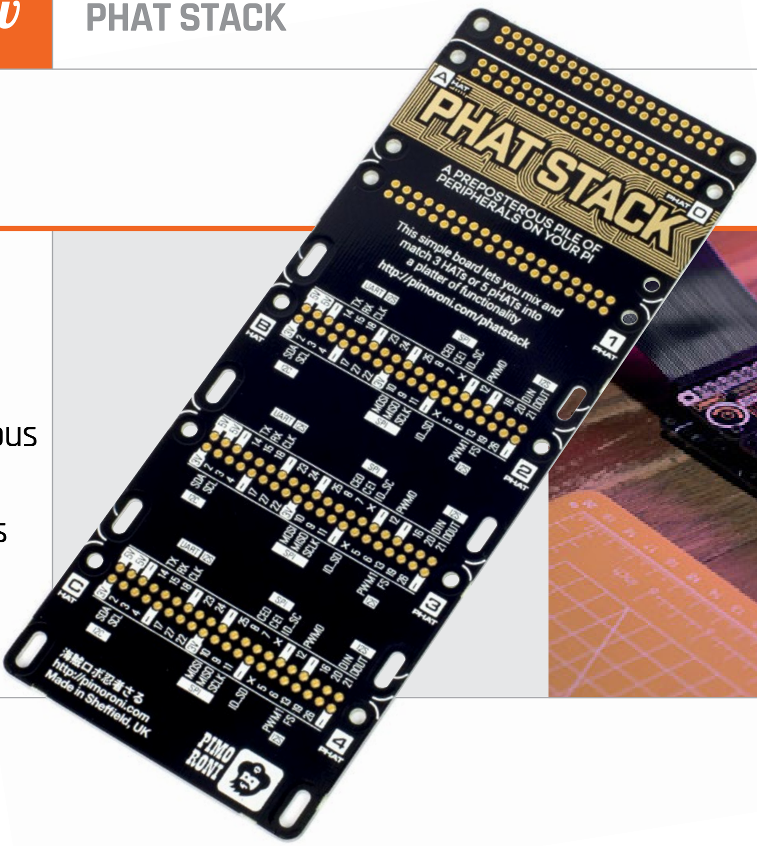
£10 / $11

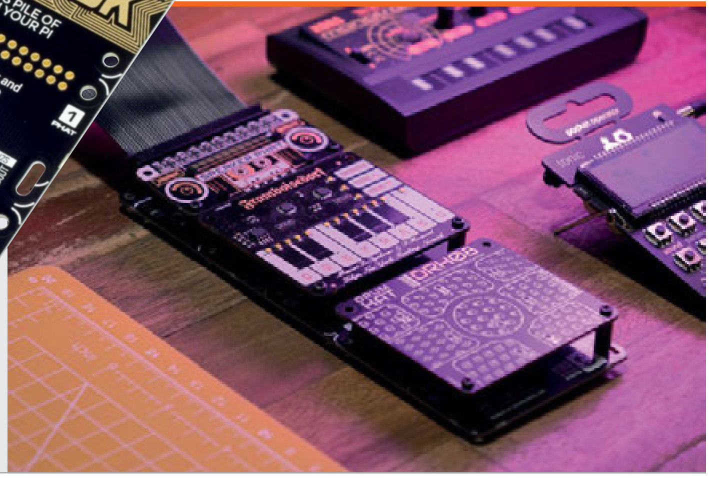
magpi.cc/2miVr5e

Last word

A convenient and compact way to add some controls to a Pi project or HAT. The Python library and examples will get you up and running quickly, and the status LED is a nice bonus.

★★★★☆





magpi.cc/2mcr2Gt

£14 / $16

Maker Says

“A preposterous pile of peripherals on your Pi!”

Pimoroni

PHAT STACK

Get your Pi to wear several HATs, or pHATs, at the same time

Pimoroni’s pHAT Stack enables you to use multiple add-on boards simultaneously with your Raspberry Pi. Also available as a solder-yourself kit (or just a PCB board), the Stack is equipped with five sets of 2×20 pin headers, plus one to connect to the Pi via a GPIO ribbon cable (supplied). This means there’s room to connect five smaller pHAT boards, three full-size HATs, or a mixture of both.

While this might seem like overkill, it provides a convenient way to switch in and out different HATs and pHATs and create your own custom combinations. So it’s ideal for experimenting with hardware project ideas.

One nice touch is that the three of the GPIO headers have their pins fully labelled, which should prove handy for connecting up your own circuits and sensors. Brass standoffs and screws are also supplied for secure mounting of HATs and pHATs.

While it’s very easy to mount the add-on boards on the pHAT Stack, one caveat is that not all of them will play nicely with each other: you need to look out for those that use the same GPIO pins. Pimoroni’s ‘pHAT Stack configurator’ online tool comes in handy here: hosted at the ever useful **pinout.xyz**, it lets you simulate adding various HATs and pHATs to the Stack and will warn you of any pin conflicts. Multiple boards may still use the same I²C pins (BCM 2 and 3) without issues, so long as they use different addresses.

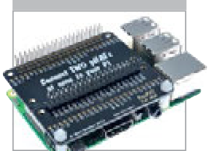
In our test setup, we combined a Speaker pHAT, Drum HAT, and Piano HAT to create a mini music box, following Pimoroni’s online guide (**magpi.cc/2qJbcYG**). This worked well, even though the pHAT Stack configurator flagged up a possible pin conflict on BCM 21.

Other suggested setups include an alarm clock (Four Letter pHAT, Touch pHAT, and Speaker pHAT) and a weather station with built-in dashboard (Enviro pHAT, Four Letter pHAT, and Scroll pHAT). You could even solder a female header to a Pi Zero and mount that on the Stack’s bottom header, rather than using the ribbon cable.

Related

FULLPHAT

For those who prefer a more compact breakout board, the FullpHAT allows you to connect two pHATs – or one pHAT and a HAT – to your Pi.



£8 / $11

magpi.cc/2ku3dfc

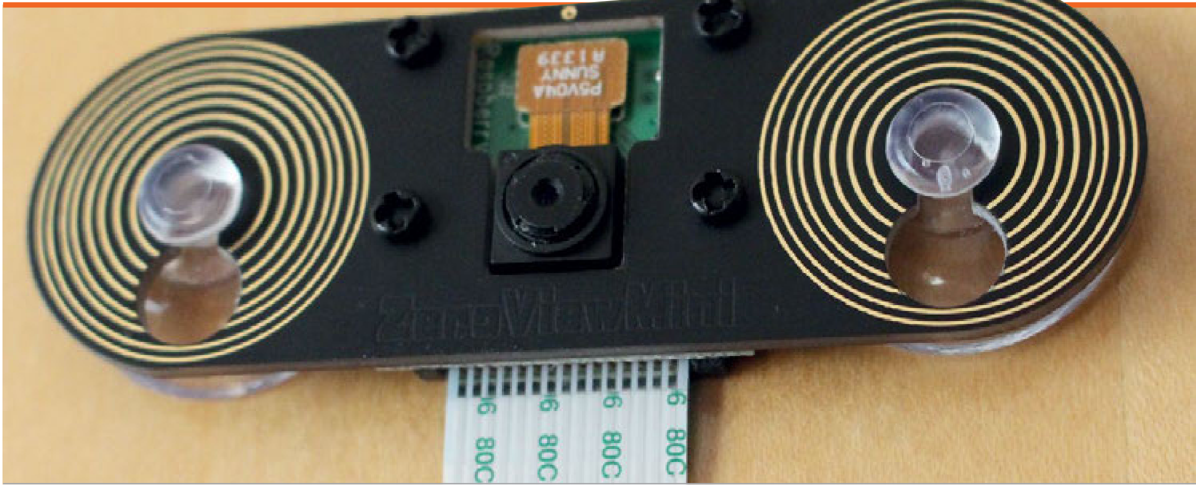
Last word

Whether you use it for a specific project or just for experimenting with combining various pHATs and HATs, the pHAT Stack is a well-designed breakout board that should prove useful. You just need to watch out for those GPIO conflicts – and check that the cable is connected the right way round on the Pi!

★★★★☆

magpi.cc/2EQIbNk

£5 / $6



Maker Says

"Can be mounted on windows, glass, fish tanks, and more"

Pi Hut

ZEROVIEW MINI

Fix your Pi Camera to a window looking out, or to any smooth surface looking into the room

The Raspberry Pi Camera Module can be tricky to position. It's small, light, and can flop about on the end of its connecting cable. While it's possible to improvise something with LEGO, Blu Tack or Sellotape, there's no real substitute for a proper mount that's designed for it.

That's what the ZeroView Mini is. It's made from printed circuit board, although the printing is purely decorative. There are four holes for screwing your Pi Camera Module in the middle of the board, and a cut-out shape for the camera to point through.

At each end of the board is a hole for fixing a suction cup. The suction cups fit the board well with no risk of movement or slipping.

The Pi Hut team say they have sourced the best suction cups they could find, made by Adams in the US. They are about 2 cm in diameter (unsquashed) and fix well to glass and smooth wood, supporting the weight of the camera easily.

The Pi Hut makes an alternative product called the ZeroView, which enables you to screw your Pi Zero to the camera mount too. It's the better choice for projects that mount the Pi on glass. The ZeroView Mini is ideal when you have a larger model Raspberry Pi, or need the Pi to sit on a surface, perhaps because you have circuits connected to the GPIO.

You can fit the suction cups into the board either way around, so you can use the mount to fix a camera to a window looking out – or to a window or cupboard, looking into a room.

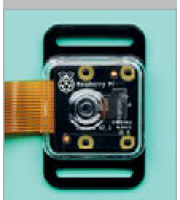
The mount gets your lens about a millimetre away from the glass, which helps to reduce reflections from inside the room. One of the suction cups was being reflected in our images, though, so we improvised a solution by removing it from the board and resting the board on it. Your results will depend on your setup and lighting, but you might need to experiment.

The unit is supported by the manual for the big brother ZeroView, so you'll need to make allowances for the differences in the units. It's simple to set up, though, and there are some tips on getting started with the camera included.

Related

ADAFRUIT RASPBERRY PI CAMERA BOARD CASE

This dinky case snaps together and includes a standard tripod mount on the back. Use the holes on its side to tie it in place.



£3 / $3

magpi.cc/2mmGKOT

Last word

The ZeroView Mini is a well-made mount for your Raspberry Pi, with high-quality suction cups. Careful camera positioning and lighting control remain important to avoid reflections, though.

★★★★☆

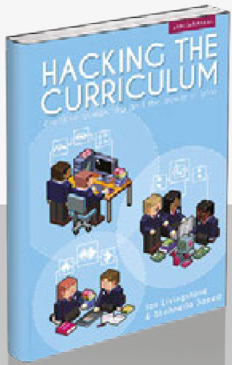
RASPBERRY PI BESTSELLERS

STEM TEACHING

Science, Technology, Engineering, and Mathematics are no longer a neglected part of education. Hurrah!

HACKING THE CURRICULUM: CREATIVE COMPUTING AND THE POWER OF PLAY

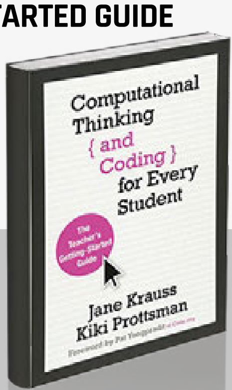
Authors: Ian Livingstone, Shahneila Saeed
Publisher: John Catt
Price: £14
ISBN: 978-1909717824
magpi.cc/2meRcY8



Through emphasising the role of play, STEM education can be transformed. If you missed out on the cover-mounted copy with our sister publication, Hello World, get a copy now (or get a second copy and pass it on!).

COMPUTATIONAL THINKING AND CODING FOR EVERY STUDENT: THE TEACHER'S GETTING-STARTED GUIDE

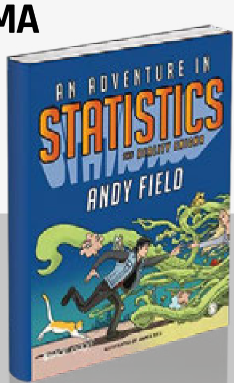
Authors: Jane Krauss, Kiki Prottsman
Publisher: Corwin
Price: £19.99
ISBN: 978-1506341286
magpi.cc/2mdO9iR



Great resources and lesson ideas for any teacher struggling with the appearance of computational thinking in the curriculum. US-made, but useful everywhere. Will not only demystify the topic, but inspire as well.

AN ADVENTURE IN STATISTICS: THE REALITY ENIGMA

Author: Andy Field
Publisher: SAGE
Price: £32.99
ISBN: 978-1446210451
magpi.cc/2menbrx



The least glamorous of the STEM subjects, despite its key role in data science; Professor Field could single-handedly change that. A narrative-driven introduction to statistics, with humour, and even graphic novel elements. Recommended.

COOL SCRATCH PROJECTS IN EASY STEPS

Author: Sean McManus
Publisher: In Easy Steps
Price: £10.99
ISBN: 978-1840787146
magpi.cc/2me3yQu

Here's a book that lives up to its title – these are some cool Scratch projects to keep you going after learning at school, Code Club, or from a Raspberry Pi introductory book. Given Scratch's user-friendliness, it's also suitable for complete beginners, and the first project, a Magic Mirror, is designed to get everyone up to speed while still being fun for more experienced users.

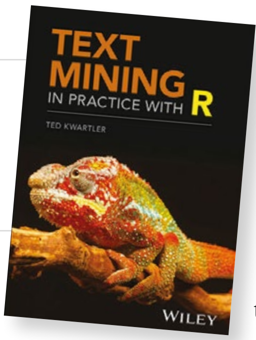
The book gives equal coverage to Scratch 1.4 and Scratch 2.0 – great for the many Pi users sticking with 1.4. Most projects will run anywhere, but for Pi users there's also a chapter with the Raspberry Pi Camera Module. Before that are projects involving drawing, animation, drums (!) and plenty of games. Early on, you get to program 12 Angry Aliens with (drumroll please…) 3D effects.

Yes, the book includes 3D glasses, and you get to program for them in other games, such as Space Mine and Maze Explorer, as well as in an art app with random elements. Final chapters introduce ScratchJr – useful for trying out quick ideas, as well as for younger coders – and five 'shorties', quick projects to finish. If you got a Pi and a book for Christmas, this will make a great follow-on. Recommended.

Score ★★★★★

TEXT MINING IN PRACTICE WITH R

Author: Ted Kwartler
Publisher: Wiley
Price: £50
ISBN: 978-1119282013
magpi.cc/2mgajkK



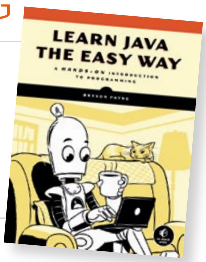
Text mining – "distilling actionable insights from text," as the author puts it – certainly has academic uses, but here we're concerned with the kind of automation that everyone from tiny startups to large enterprises can use to automate processes and try to get ahead. With so much online content produced by customers, millions (billions!) of words of text are a resource waiting to be tapped.

In the introductory chapter, the case is examined of Amazon's early engagement with customers through social media – text-mining the competition's responses on social media to look for best practice – after explaining the balance needed between algorithm and domain knowledge: 'text mining is part art and part science'. Having launched the social media customer service team at Amazon, then worked on similar analytics projects at other large companies, Kwartler is as well versed in the practical problems of text mining as the theoretical side.

Chapters on visualisations, sentiment scoring, hidden structures, document classification (using supervised learning to detect clickbait), predictive modelling, and natural language processing with OpenNLP all do a first class job of showing messy real-world data and how to handle it, with sufficient theory and R to build upon further. An enjoyable and informative journey.

Score ★★★★☆

LEARN JAVA THE EASY WAY: A HANDS-ON INTRODUCTION TO PROGRAMMING



Author: Bryson Payne
Publisher: No Starch
Price: £22 ($29.95)
ISBN: 978-1593278052
nostarch.com/learnjava

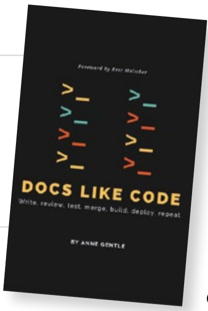
Python and the Raspberry Pi have been a good fit together; the scripting language that reads almost as pseudocode makes a great introduction to coding. Java, on the other hand, is a harder sell for beginners: strict typing, all that boilerplate, and so big that most Java programmers have to lean heavily on their IDEs. Yet Java remains the most used programming language professionally, and its JVM the gateway to dozens more, so it cannot be ignored.

If only there were a book that launched the reader past the boilerplate of static, void, main, and int, and got them coding up something fun straight away. Enter Bryson Payne. Dr Payne brings his easy, flowing style to Java learning, and gets the reader coding fun programs immediately; first by copying, then stretching through further challenges.

Eclipse and Android Studio are used heavily throughout; unlike Python, getting things done quickly in the Java environment needs good tooling, and these are sensible choices. Text-based and graphical games alternate as areas of programming are introduced, and Android makes an early appearance, capped with animation programming. An excellent introduction.

Score ★★★★☆

DOCS LIKE CODE



Author: Anne Gentle
Publisher: Lulu.com
Price: £14 ($19.95)
ISBN: 978-1387081325
docslikecode.com

Docs as code? Documentation is an integral part of the software 'product', and if it is not always treated as such, then everyone involved in software needs to think about why. Technical writers can do their part by matching their workflow with that of a project's programmers, and using the same tools – placing the documentation within easy reach of coders, and removing friction from the involvement of all team members.

Gentle, a key figure in the Write-the-Docs group, which is doing much to bring best practice to the writing of technical documentation, gives us a short introduction outlining all the main messages of docs-like-code: storage in version control (GitHub, GitLab, BitBucket); automatic doc artifact builds; trusted reviewers; and publishing without much human intervention.

The documentation workflow is introduced and outlined, with due time spent on Continuous Integration, review, and useful topics like REST APIs, and measuring improvements – as well as examining some of the many technical questions which shouldn't dominate your strategic decision in choosing how to set up a document toolchain, but are unavoidable considerations. This beginner-friendly guide will leave you with no excuse not to get your project's documentation off to a great start.

Score ★★★★☆

ESSENTIAL READING: HADOOP & SPARK

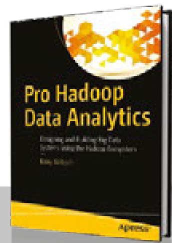
The perfect excuse to set up a Raspberry Pi cluster for Big Data experiments.

Teach Yourself Hadoop in 24 Hours

Author: Jeffrey Avens
Publisher: SAMS
Price: £39.99
ISBN: 978-0672338526
magpi.cc/2mfKH7D

A whirlwind review of the possibilities of the platform, taking in the core concepts and all of the components.

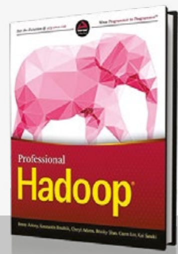
Pro Hadoop Data Analytics



Author: Kerry Koiltzsch
Publisher: Apress
Price: £31.99
ISBN: 978-1484219096
magpi.cc/2mdD0i6

High-level but code-rich guide to getting the most from Big Data, mostly Java, but some Scala and Python.

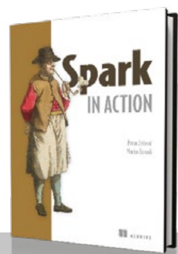
Professional Hadoop



Authors: Benoy Antony, Konstantin Boudnik, Cheryl Adams, Branky Shao, Cazen Lee, Kai Sasaki
Publisher: Wrox
Price: £42.50
ISBN: 978-1119267171
magpi.cc/2meRFtm

If you need to get up to speed on Hadoop, particularly as a DBA or an architect, this is the quickest intro.

Spark in Action



Authors: Petar Zečević and Marko Bonaći
Publisher: Manning
Price: £30.99
ISBN: 978-1617292606
magpi.cc/2mdAngg

The Spark framework complements the Hadoop ecosystem; the book's Scala examples are downloadable in Python (and Java) versions.

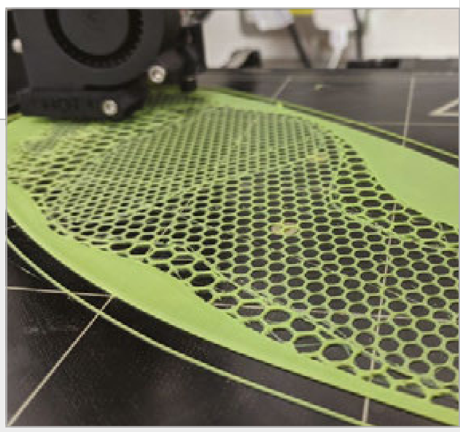
Practical Data Science with Hadoop and Spark

Authors: Ofer Mendelevitch, Casey Stella, Douglas Eadline
Publisher: Addison-Wesley
Price: £35.99
ISBN: 978-0134024141
magpi.cc/2meo1Vd

More of a data science introduction, but very good on introducing Hadoop's role in that world.

PRODPOINT A PI-POWERED 3D PRINT FARM

Setting up a 3D print farm with Raspberry Pis was a no-brainer for the team at **PRODPOINT**



Right You can even 3D-print shoe soles!

Felix Fried

Occupation: Director of PRODPOINT, Design Engineer

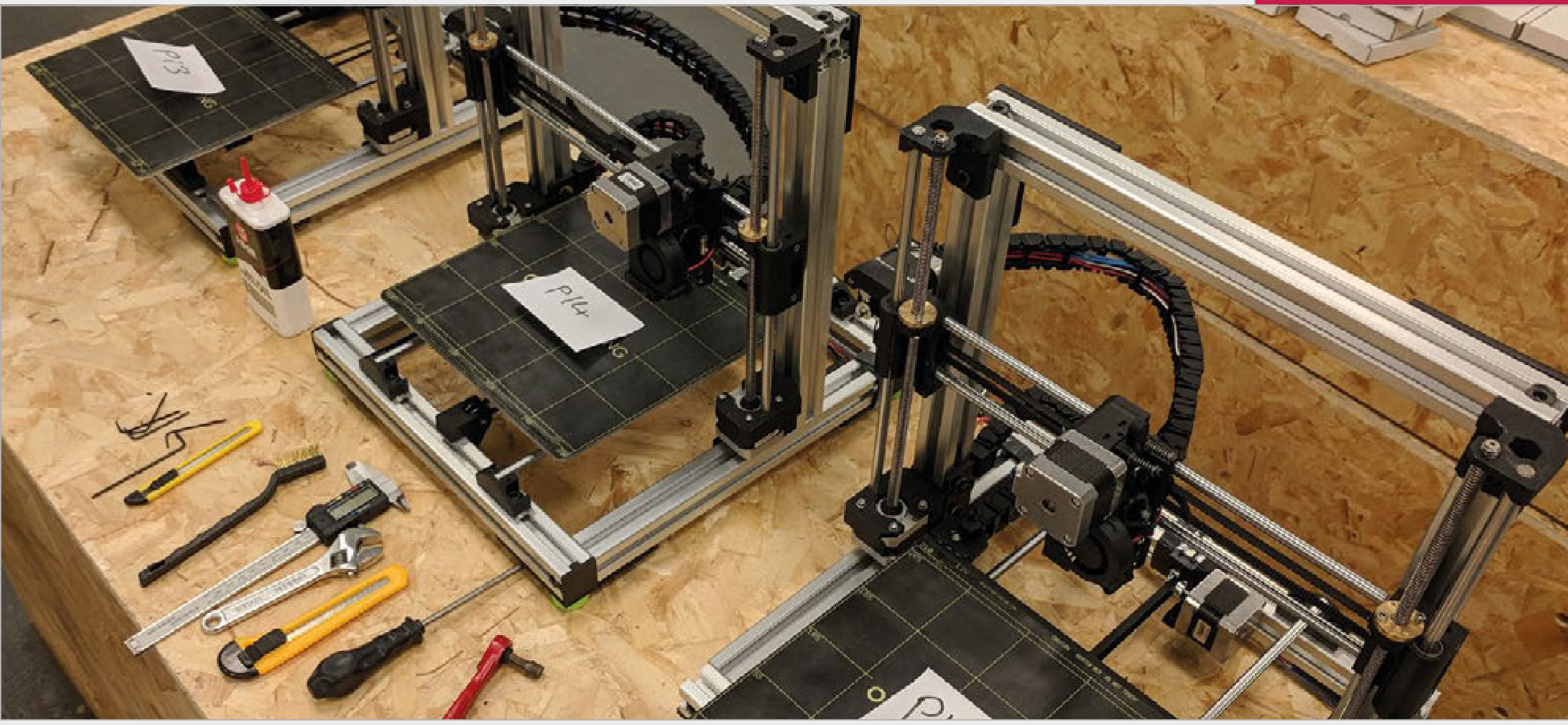
Way back in issue 55 of *The MagPi*, we showed you how to make a Pi-powered, Game Boy-esque handheld retro-gaming system. The shell for this project was printed by PRODPOINT, at the time a small operation running out of Felix Fried's bedroom in Bournemouth. Since then, PRODPOINT has expanded to be a true 3D-printing factory.

Tell us about your 3D print farm

So our 'print farm' consists of an array of 3D printers, currently 24 of them and counting. We produce hundreds or thousands of parts each week for companies who've decided 3D printing is a better solution for production over traditional manufacturing methods like injection moulding. We help designers and engineers scale up their ideas from the prototype stage to production by leveraging FDM 3D printing.

For a lot of projects, the key benefit is that we manage to reduce much of the production costs, as well as accelerating the time to market for our clients. Others decide to use us because it gives them the flexibility of producing smaller batches without ever having to commit much capital. If they decide to make design changes, they can receive their updated design in days at minimal cost.

Traditional methods like injection moulding require



investing in an expensive mould; these can be a hefty investment (up to £30k+) and can take several weeks to produce. When things go wrong and the mould needs repairing or the design changed, companies are left with nasty bills. We remedy this by providing fast, flexible, and affordable manufacturing.

After a few years of doing this as a student, I began to get approached by companies looking to get batches produced, some of which were fairly large orders for me at the time, and would've taken a gazillion hours without an army of printers. This led me to ponder on the idea of starting a factory or 'print farm'.

> While we didn't coin the term print farm, we've grown a particular liking to it

While we didn't coin the term print farm, we've grown a particular liking to it after harvesting prints daily – it's a bit of an ongoing joke but we might decide to install fake grass and print cows for the factory... just 'cause we can.

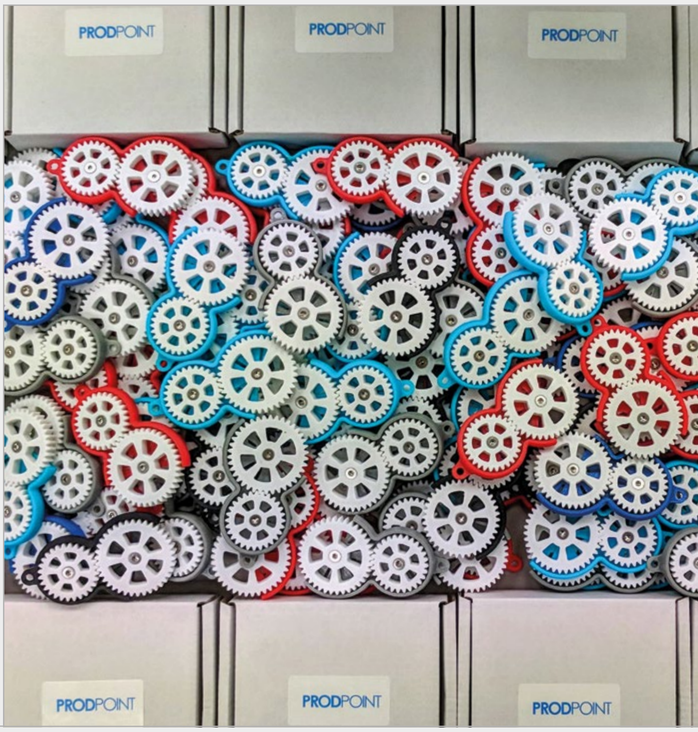
How did the idea come about?
I'm not sure there was ever a light-bulb moment as such. I started building 3D printers around six years ago and decided to run my printer to help university students produce their projects.

Why the Raspberry Pi? How are they being used?
We use Raspberry Pis to keep all of our printers connected on a local network. This allows us to control and monitor them from one single interface we programmed in-house. We chose Raspberry Pi as they're super-powerful for such an affordable bit of kit. Currently we run three printers per Pi. We've also managed to add cameras for off-site remote monitoring and mini LCD screens to track which order is being printed on each printer.

Above With 24 3D printers running non-stop, sometimes they need to take a break for a bit of love and attention

Below You can 3D-print more than just gears, but they are a perfect candidate for a bulk-buy

How popular has the service been so far?
We've only been up and running as a print farm for a few months now but demand has been consistently growing. We're now working in several industries including pharmaceutical, automotive, film, and consumer. I guess we're in-line with our predictions so far, but who knows what the future will bring?



THE MONTH IN RASPBERRY PI

Everything else that happened this month in the world of Raspberry Pi

100+ JAMS CONFIRMED FOR RASPBERRY PI'S BIRTHDAY!

RASPBERRY JAMS WILL BE HELD AROUND THE WORLD TO CELEBRATE THE BIG BIRTHDAY WEEKEND

For the past few years, the Raspberry Pi Foundation has organised its own birthday party in the form of a weekend-long Raspberry Jam. Not everyone can make it out to the one Jam, though, so this year the call went out to the community to host Jams around the world in honour of the Raspberry Pi on the weekend of 3 and 4 March. Applications to be a part of it have recently closed and a whopping 102 Raspberry Jams will be taking place around the world to celebrate.

Dojoberry Jam (Perth, Australia)

Glenunga International High School Raspberry Jam (Glenunga, Australia)

Varna Raspberry Jam Big Birthday Weekend 2018 (Varna, Bulgaria)

Raspberry Jam Camer (Yaounde, Cameroon)

MyPi Toronto (Ontario, Canada)

Ottawa Jam – Big Birthday Bash (Ontario, Canada)

Pi Jammin (Ontario, Canada)

Imen (Paris, France)

Ungeheuer Raspberry Jam (Forst, Germany)

Aketh Innovation Robotics Lab (Trikala, Greece)

MalnaPi Jam (Somogy, Hungary)

Raspberry Pi BD Jam (Budapest, Hungary)

Arihant's Jam (Odisha, India)

BIT-D Raspberry Jam (Chhattisgarh, India)

Indore Raspberry Jam (Indore, India)

jambigbirthday (Chidambaram, India)

Kanpur Raspberry Jam (Kanpur Nagar, India)

Mumbai Raspberry Jam (Mumbai, India)

MyFirstPOC (Ahmedabad, India)

Next Tech Lab Jam (Chennai, India)

ResPro Raspberry Pi Jam Labs (Chennai, India)

RIT Raspberry Jam (Chennai, India)

Trivandrum Raspberry Jam (Kerala, India)

Babol Raspberry Jam (Mazandaran, Iran)

Kajestan (Isfahan, Iran)

Raspberry Jam Cagliari (Cagliari, Italy)

SuLuLab Raspberry Jam (Terni, Italy)

Raspberry Jam in Hamamatsu 2018 (Hamamatsu, Japan)

Raspberry Jam Tokyo (Tokyo, Japan)

Tabaka Boys' High School Raspberry Jam (Kisii, Kenya)

Voi Raspberry Jam (Voi, Kenya)

Raspberry Jam Penang (Penang, Malaysia)

MaltaPi (Attard, Malta)

Piña pi (Ecatepec, Mexico)

AmsterJam (Amsterdam, Netherlands)

Kristiansand Raspberry Jam (Vest-Agder, Norway)

Lima Raspberry Jam Verano 2018 (Lima, Peru)

RPi Birthday with IoT Cebu (Cebu, Philippines)

RPiJamCT (Cape Town, South Africa)

Raspberry Jam Granada (Granada, Spain)

4th Annual Hisar Coding Summit (Istanbul, Turkey)

Batley Library Raspberry Jam (West Yorkshire, UK)

Beeston Raspberry Jam (Nottingham, UK)

Birmingham Raspberry Jam (West Midlands, UK)

Blackpool Raspberry Jam (Lancashire, UK)

Bognor Regis Raspberry Jam (West Sussex, UK)

Cardiff Family Jam (Wales, UK)

Cornwall Raspberry Jam – Big Birthday Event (Cornwall, UK)

Cotswold Raspberry Jam (Cheltenham, UK)

Coventry RJam (West Midlands, UK)

Dundee Raspberry Jam (Scotland, UK)

East London and Covent Garden Jam (London, UK)

East London Jam (London, UK)

Exeter Raspberry Jam (Devon, UK)

Gateshead Raspberry Jam (Tyne and Wear, UK)

Glasgow Raspberry Jam (Scotland, UK)

Huddersfield Raspberry Jam (West Yorkshire, UK)

Hull Raspberry Jam (East Yorkshire, UK)

Jam Ramma (Greenwich, UK)

Jamming in Marlborough Birthday Weekend (Wiltshire, UK)

Kendal Raspberry Jam (Cumbria, UK)

Kent Raspberry Jam (Kent, UK)

Leeds Raspberry Jam (West Yorkshire, UK)

Manchester Raspberry Jam (Manchester, UK)

Meety Pi Club (Dorset, UK)

Milton Keynes Raspberry Jam (Bletchley, UK)

Nantwich Raspberry Jam (Cheshire, UK)

Newhaven Raspberry Jam (East Sussex, UK)

Newport Family Jam (Wales, UK)

Northern Ireland Raspberry Jam (Belfast, UK)

Northumbria Jam (Tyne & Wear, UK)

Oxford Raspberry Jam (Oxford, UK)

Potton Pi and Pints (Bedfordshire, UK)

Preston Raspberry Jam Sandwich (Lancashire, UK)

Raspberry Jam @ Worksop College (Nottinghamshire, UK)

South London Raspberry Jam (London, UK)

Southend Raspberry Jam (Essex, UK)

Stokers Jam (Chelmsford, UK)

Swansea Jam (Wales, UK)

Taunton GlassBox Big Birthday Weekend Jam (Somerset, UK)

TechTribe.UK Raspberry Jam Birthday Party (London, UK)

Ann Arbor Raspberry Pi Birthday Jam (Michigan, USA)

Bucks Raspberry Jam (Pennsylvania, USA)

Cocoa Raspberry Jam (Florida, USA)

ColleyVineJam (Texas, USA)

Key Tech Labs Raspberry Jam (Washington, USA)

Long Beach Public Library (New York, USA)

MAKEmory Raspberry Jam (Georgia, USA)

Nerdvana's 1st Raspberry Jam (Texas, USA)

Norski Jam (Wisconsin, USA)

Philly & Pi Bithday Jam (Pennsylvania, USA)

Pi Kids Garage (North Carolina, USA)

Raspberry Pi Hack-a-thon (Maryland, USA)

Raspberry Pi Miami (Florida, USA)

RGVSA's First Raspberry Jam (Texas, USA)

Riverside Raspberry Jam (California, USA)

Roanoke Raspberry Jam (Virginia, USA)

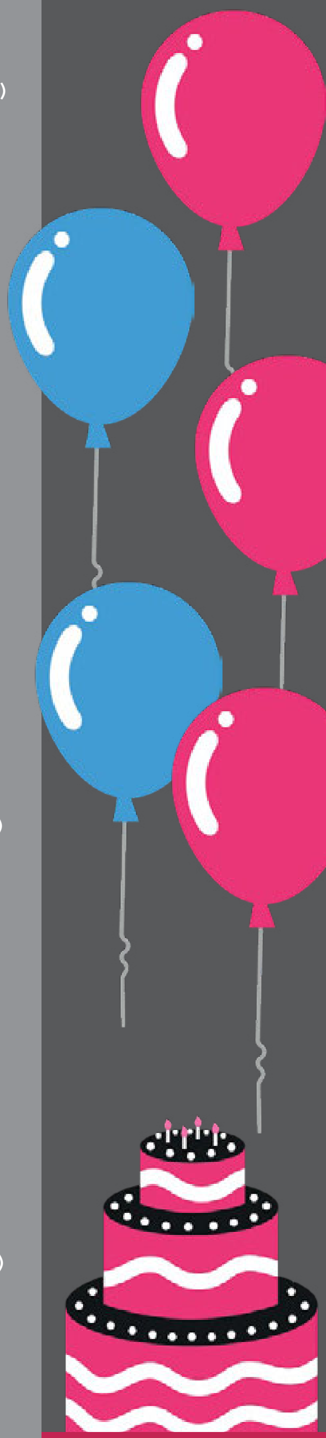
Rock the Rasp (Maryland, USA)

Rose Cyber Jam (Oklahoma, USA)

Seattle Raspberry Jam (Washington, USA)

Stemgirlz/GWC Raspberry Jam (Pennsylvania, USA)

Raspberry Team Masvingo (Masvingo, Zimbabwe)



FIND OUT MORE INFO

Want to get precise details on your local Pi Birthday Jam? Head over to the Raspberry Jam page for an interactive map and links to the individual Jams here: **magpi.cc/28Nxeff**

NEW YEAR, NEW PROJECTS

HERE ARE SOME OF THE COOL PROJECTS YOU'VE SENT US IN JANUARY

The holiday break is traditionally seen as being relaxing. Taking a break from it all. To some of our readers, that meant having time to work on their projects or figure out what they want to do in the new year. Here's some of the cool stuff you've sent us!

David Pride @davejavupride magpi.cc/2mC47DN

Replying to @TheMagP1

@TheMagP1 Santa brought me my first ever successful compilation of OpenCV on #RaspberryPi 🎉 Training for #PiWars Rainbow Challenge begins in earnest...

OPENCV ROBOT

In preparation for the upcoming Pi Wars, David Pride managed to get OpenCV working on his robot to be able to tell colours apart! Watch the video in the original tweet to see just how well it's been mastered.

Andrew Lewis @monkeysailor magpi.cc/2mDLcbM

Replying to @TheMagP1

I assembled a YouTube chicken camera! youtu.be/O_laN3GajyM

CHICKEN CAMERA

Andrew Lewis showed us his new YouTube chicken camera, which you can see him assemble on his lengthy YouTube video! It's made by taking an old CCTV camera case and installing a Pi Camera into it. Ingenious.

Elijah Horland @NotABombBunkE magpi.cc/2mCnY5R

My teachers responded really well to this year's gifts - @adafruit "JoyBonnet" mini game-system gifts inspired by #AdaBox 005 - but with soldered headers. This is my vice principal, my STE(a)M teacher and my English Language Arts teacher.

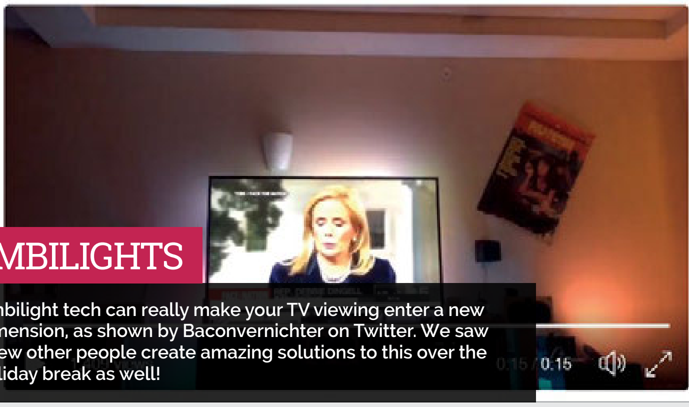
TEACHER GIFTS

One of our younger readers, Elijah put together some very special gaming holiday gifts for his teachers. We hope his English teacher gets a kick out of it.

Mike Golf Romeo @Baconvernichter magpi.cc/2mC4ZZ5

Replying to @TheMagP1

Ambilight (Hyperion) with 316 APA102 LEDs with arduino to drive them :)



AMBILIGHTS

Ambilight tech can really make your TV viewing enter a new dimension, as shown by Baconvernichter on Twitter. We saw a few other people create amazing solutions to this over the holiday break as well!

Send us your projects to **magpi@raspberrypi.org** or via Twitter **@TheMagP1!**

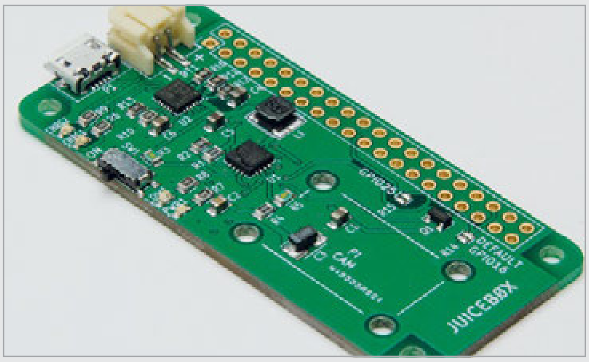
CROWDFUND THIS!

The best crowdfunding hits this month for you to check out…

OMNIJOY

kck.st/2Cjq4Ol

This is a special case for the Pi Zero that turns it into a handheld controller with a screen in the middle. It's got a wooden grip so it's pretty fancy as well. The concept is that, as an all-in-one device, you can use it in your projects without needing the separate computer and controller elements. It's a little pricey but well worth a quick look.



JUICEBOX ZERO

kck.st/2DcVkjL

We quite like this. It's a HAT you can slot on top of a Pi and connect a 3.7 V lithium-ion battery to. You can also charge it via the HAT. While there are other more complex HATs that do the same (such as the excellent PiJuice), this a great little solution that fits easily on top of a Pi Zero. It even gives you full access to the GPIO pins as well!

BEST OF THE REST

Here are some other great things we saw this month

magpi.cc/2mEM2VR

UBER PI

We've seen a couple of Ubers include a small RetroPie setup to make the ride a bit more fun, but we've never seen one with a full-on manual to let you know the depth of what you can play. Now the challenge is, can you speedrun Super Mario Bros on the way to the airport?

magpi.cc/2mEf7R1

AUTOMATIC PET FEEDER

This one is so good you can barely even see the Pi implementation. It takes a manual cereal dispenser and adds a motor to dispense food at certain times during the day. Once it has attempted to dispense, it emails the owner an image of the food in the bowl so they can make sure it's worked!

magpi.cc/2mB5aE0

RASPBERRY PI AT STANSTED AIRPORT

We like hearing of Raspberry Pi spottings in the wild. This screen at Stansted airport is a reminder at how wide-ranging the use of the Raspberry Pi is – although it looks like they haven't configured Raspbian to be read-only, which has caused a small problem…

COMMUNITY PROFILE

SPENCER ORGAN

From educator to club volunteer, maker, and blogger, Spencer Organ is the complete community member package

Spencer Organ

Category: Educator/Maker

Day job: Teacher in Birmingham, UK

Website:
twitter.com/makercupboard
magpi.cc/2Dk7BoH

Below Spencer volunteers his time to various community events with the Raspberry Pi Foundation, often finding his way into the traditional Certified Educator selfie with Carrie Anne Philbin

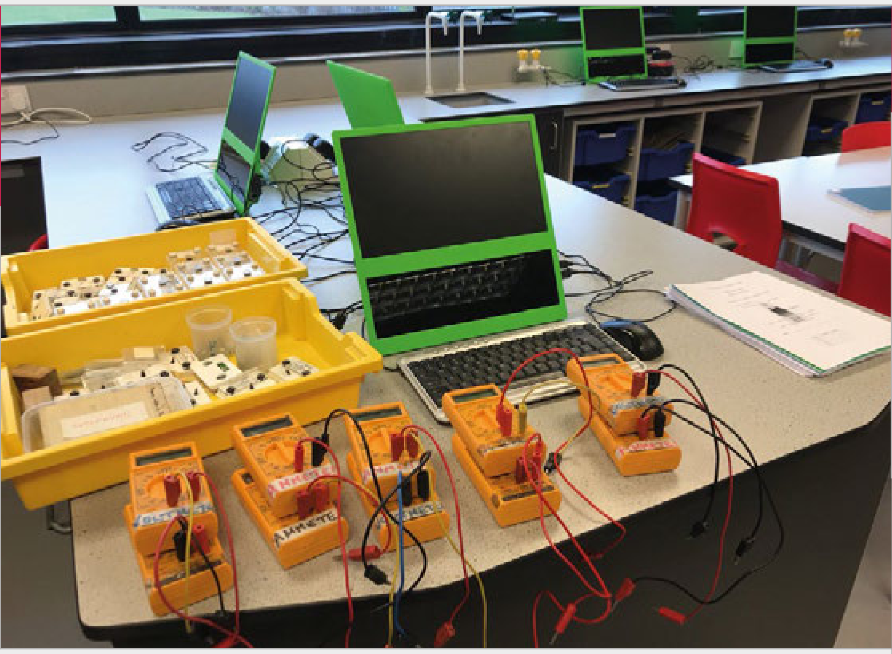
What's not to love about Spencer Organ? As an educator, he has fully immersed himself into the Raspberry Pi community, running clubs, teaching computer science, volunteering, and advocating the Foundation. And as a maker, he has embraced his educator mentality to provide tutorials and support for every project he's released into the wild – and believe us, Spencer loves to make.

With a degree in Chemistry and Education, and his qualification as a Picademy-trained Raspberry Pi Certified Educator, Spencer teaches Physics and Computer Science in Birmingham, UK.

Between classes, he offers up his spare time to running Raspberry Pi and coding clubs at the school, along with volunteering at a Code Club, Raspberry Jam events, the Raspberry Pi Birthdays, school holiday clubs and much more.

You may well recognise Spencer's name from one of several projects showcased previously in *The MagPi*. Inspired by everything around him, Spencer creates fun, imaginative builds – both alone and with his son, Philip. We should take a moment to highlight Philip's incredible Pi-powered Pokédex, a working replica of the gadget used in the popular *Pokémon* TV series to identify and catalogue the creatures captured.

Spencer Organ dedicates his time and passion to bringing computer science to everyone through events, clubs, and maker projects



Above "I try to bring in CS elements into my Physics work and recently had students modelling radioactive decay in Python!"

Of his computing past, Spencer recalls, "After getting my first computer, an Acorn Electron, at age 11 I spent the next five years writing projects in BASIC. I must have completed every Usborne coding book from the library as a child. I progressed onto the Visual Basic unit at sixth form and continued writing Excel VB projects to simplify data tasks at school." So that explains his love for computing and the natural progression into digital making with the Raspberry Pi.

> I must have completed every Usborne coding book from the library as a child

At the Raspberry Pi Big Birthday Weekend 2017, Foundation CEO Philip Colligan pulled Spencer onto the stage, from his place as stage manager on the sidelines, to highlight the work of the RCEs and community, recognising him to be "a physics teacher who's gone on to become an amazing advocate for computing in schools."

Spencer's work, often in his own time and at personal expense, to incorporate computer science education into as many aspects as possible of the children's lives that he works with is a worthy feat. And confirming the inclusion of the Arts in STEAM, he's even gone on to inject the Raspberry Pi into school productions, including *High School Musical* and *The Wizard of Oz*. The relocated band need to see the musical director from their hidden position backstage? No problem, here's a live-stream video system. The Tinman needs a heart? Sure. And let's use an LED matrix to make it glow.

As we move into 2018 and Spencer continues to work on building upon his skill set – "I really want to develop my soldering skills this year and also design my first PCB" – we also look forward to seeing what further strides he makes within the educational work of the Foundation. Thank you, Spencer.

HIGHLIGHTS

THEATRE

magpi.cc/2DkodK3

As Spencer explains, "[the student body] performed *High School Musical* at school and, following the tradition of previous shows, I wanted to use a Raspberry Pi as part of the theatre tech." From props to behind-the-scenes tech support, Spencer's secured the Pi's place within the theatre tradition of the school.

MAKER CUPBOARD

magpi.cc/2Dk7BoH

Spencer's making passion has taken over a room of his house, turning the built-in cupboard of their spare bedroom into the Maker Cupboard, a brand Spencer has adopted across online social accounts and his blog.

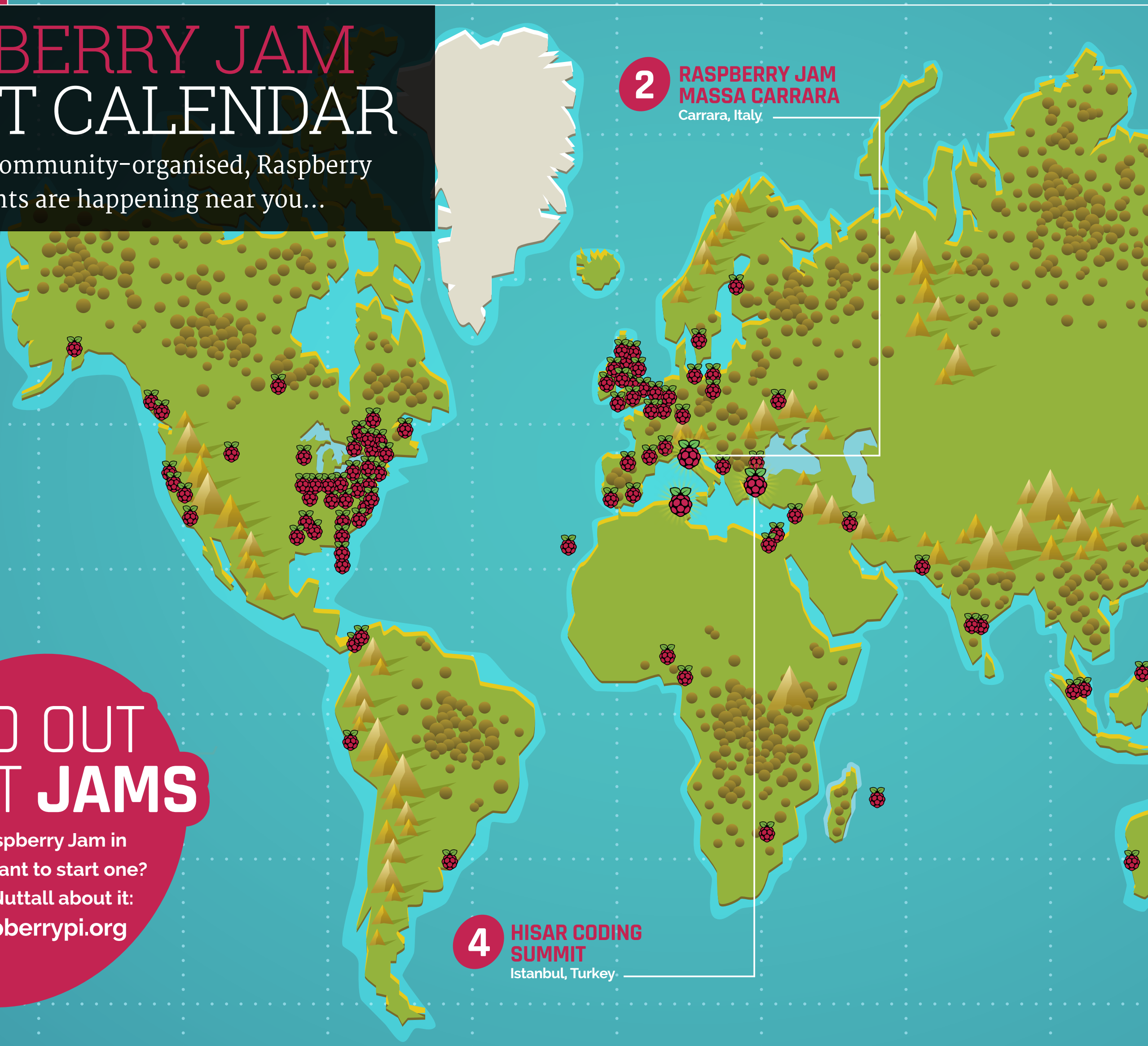
PIONEERS

magpi.cc/2DmCXev

Spencer supervised a team of secondary school-age students as they worked to complete the Raspberry Pi Pioneers challenges throughout 2017. This poor bear suffered for the greater good of the 'Only You Can Save Us' challenge.

RASPBERRY JAM EVENT CALENDAR

Find out what community-organised, Raspberry Pi-themed events are happening near you…



FIND OUT ABOUT JAMS

Want a Raspberry Jam in your area? Want to start one? Email Ben Nuttall about it:
ben@raspberrypi.org

HIGHLIGHTED EVENTS

1 RASPBERRY JAM NORTHUMBRIA

When: Saturday 3 February
Where: Ellison Building, Newcastle-upon-Tyne, UK
magpi.cc/2DdnvRN
Bringing together a diverse group of people with a range of skills and experience.

2 RASPBERRY JAM MASSA CARRARA

When: Saturday 10 February
Where: I.I.S. Zaccagna Galilei, Carrara, Italy
magpi.cc/2DgFN4U
Here you can find workshops for beginners and for experts to work on their projects.

3 RASPBERRY JUNIOR JAM

When: Friday 16 February
Where: Bletchley Park, Bletchley, UK
magpi.cc/2DgMq7c
The National Museum of Computing is looking to start a Jam for young people who are just starting out with coding.

4 HISAR CODING SUMMIT

When: Friday 2 March
Where: Hisar School, Istanbul, Turkey
magpi.cc/2msw6po
The goal is to share and spread the essential knowledge of programming and algorithmic thinking with everyone.

REGULAR EVENTS

5 COFFEE, CAKE AND CODING

When: Thursday 1 February
Where: King Edward VI Sheldon Heath Academy, Birmingham, UK
magpi.cc/2Dh9LWj
This monthly meeting/workshop aims to enjoy coding and computing in a relaxed setting.

6 EXETER RASPBERRY JAM

When: Saturday 3 February
Where: Exeter Library, Exeter, UK
magpi.cc/2Be2TZ6
There will be lots to do, plenty of help and advice, and Pis to play with at the Exeter Raspberry Jam. All are welcome.

WE'VE HIGHLIGHTED SOME OF THE AREAS IN NEED OF A JAM! CAN YOU HELP OUT?

7 PRESTON RASPBERRY JAM

When: Monday 5 February
Where: Media Factory Building, Preston, UK
magpi.cc/2DdbHyV

This Jam is a community of people who meet in Preston each month to learn, create, and share the potential of the Raspberry Pi.

8 LEEDS RASPBERRY JAM

When: Wednesday 7 February
Where: Swallow Hill Community College, Leeds, UK
magpi.cc/2DhiDLL

There'll be chances to get hands-on with more digital making activities through workshops and a hackspace area to share projects.

RASPBERRY JAM BIG BIRTHDAY WEEKEND

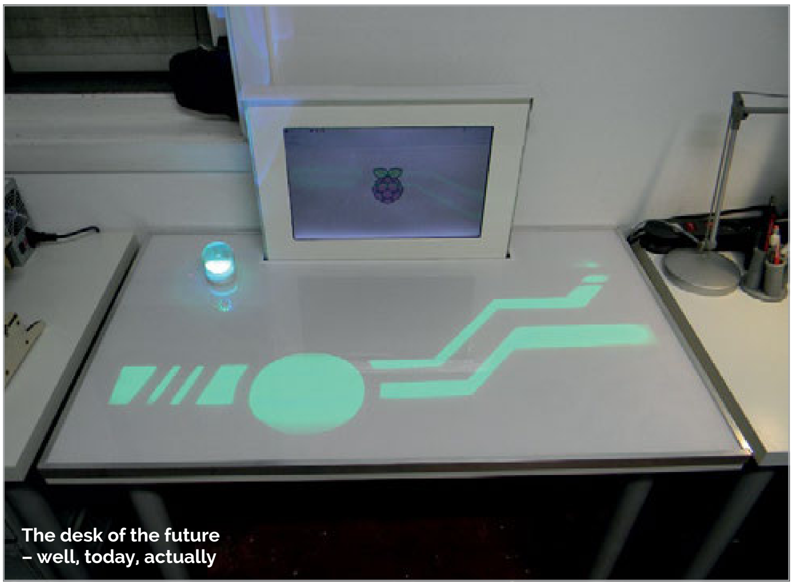
Raspberry Pi turns six this year and to celebrate this, the Raspberry Pi Foundation is co-ordinating a huge number of Jams around the world to participate in the Big Birthday Weekend. There are over 100 Jams co-ordinating events on Saturday 3 March (and a few on Sunday 4 March) and you can read more about them – including an up-to-date list! – in This Month in Raspberry Pi, on page 86.

YOUR LETTERS

PIKEA HACKS

I'm a new Raspberry Pi user and I've been marvelling at some of the amazing projects I've seen online and in this magazine. For years I've been doing 'Ikea Hacks', where you take a piece of Ikea furniture and modify it to your tastes. This can be as simple as making a Billy bookcase extend from floor to ceiling, or rearranging my TV cabinet. I feel like some interesting Pi projects could be made by combining Raspberry Pi and Ikea furniture – do many of these kind of projects exist? Are you planning on doing any articles about how to hack furniture with the Raspberry Pi? I hope you do!
Sarah

We've had a few furniture-based projects in the mag in the past – for example, last month we showed off someone's Christmas lights that they'd installed into their Billy. There's also the extremely cool PiDesk (**magpi.cc/1qtTC5J**) from *The MagPi* #43, with fancy lights and a rising screen. Very futuristic. As and when we find these projects we'll put them in the magazine – as for writing a feature on Ikea hacks, we may just have to look into that! It's always fun to put some LEDs in a display case after all.



The desk of the future – well, today, actually

SPECTRE OF DOOM

I was worried when I looked at the news the other morning and saw that there was a new, wide-reaching bug for computer processors called Meltdown and Spectre. Is this something that can be patched out of a Raspberry Pi? I have a couple of quite sensitive projects and so have turned the Raspberry Pis off for the time-being.
Brett Smith

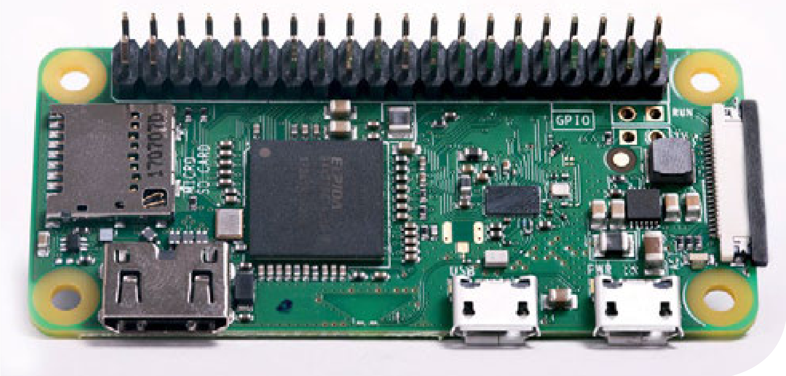
The Raspberry Pi is fairly unique in that it is immune to the Spectre and Meltdown bugs, so you don't have to worry about needing to get new Pis or applying any specific updates. We have an article all about what the bugs are and why the Pi is unaffected on page 48 – written by Eben Upton, the co-creator of the Pi. Hopefully that will allay any fears you have.

NEW RASPBERRY PI ZERO

I saw there's a new version of the Raspberry Pi Zero. It looks good, but does it mean I need to upgrade my Pi Zero W? Are there significant improvements to it over the other Pi Zero W? I never did get round to soldering on a GPIO header so maybe it's worth me picking one up anyway, to save me the hassle of soldering the header on. What do you recommend?
Lee S

The only difference between the Pi Zero W and the new Pi Zero WH is that the latter includes the soldered GPIO header – there's no speed or memory improvements on the board. You will pay a small premium to get the WH version over the normal Zero W though, so if you're confident in your soldering skills and have the header already you should definitely give it a go yourself. You can always then buy a WH the next time you need a Zero and marvel over the professional soldering job.

Below The new Pi Zero WH comes with a pre-installed GPIO which is great for projects



FROM THE FORUM: A SMALL TYPO

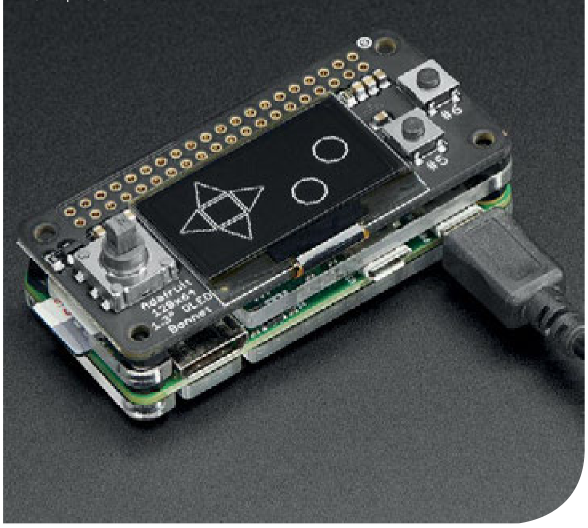
The Raspberry Pi Forum is a hotbed of conversations and problem-solving for the community – join in via **raspberrypi.org/forums**

Just a small thing, but I thought I would just say the review for the OLED Bonnet in issue 63 says the price is $32 when the price is actually $22.
KanoMaster22

In this case this was our mistake, sorry! Unfortunately, prices change around a lot so sometimes differences like this are entirely out of our control. Once the magazine goes to print, we can't change the price on reviews like this.

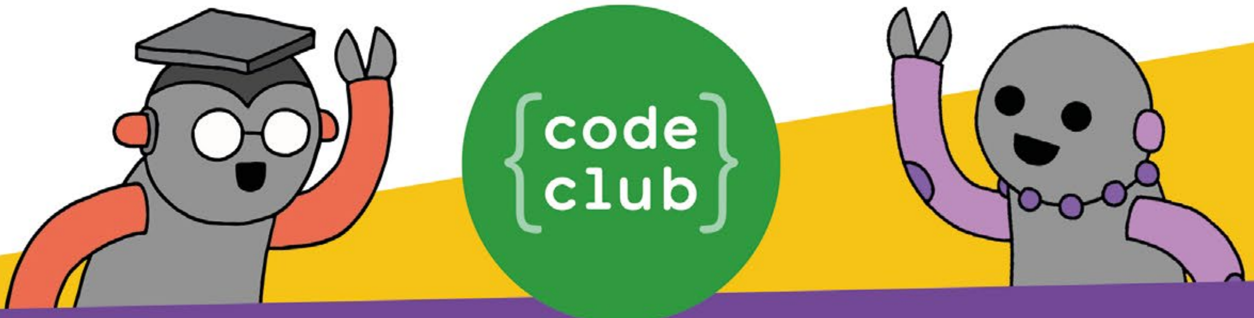
Any typos like this, or especially any code issues you come across, please drop them off in the forum under the Errata thread. We do endeavour to make sure the content in the magazine is 100% accurate, but the odd thing does slip through the cracks, unfortunately.

The OLED bonnet – now $10 off!



WRITE TO US

Have you got something you'd like to say?
Get in touch via **magpi@raspberrypi.org** or on The MagPi section of the forum at: **raspberrypi.org/forums**



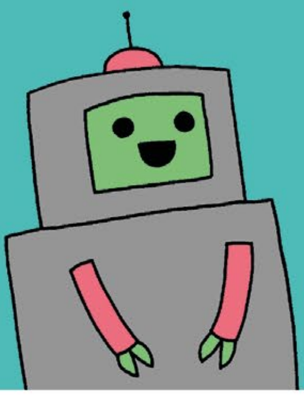
START A CODE CLUB IN YOUR SCHOOL!

Code Club is a network of volunteers and educators who run free coding clubs for young people aged 9-13.

Our aim is to inspire the next generation to get excited about computer science and digital making.

"We use Code Club's fun educational resources to run a weekly after-school club for Year 7 and Year 8 pupils. The students benefit considerably from the extra challenge!"

Karen Dadd, Computing Teacher



- **Code Club is free**
- **Code Club provides step-by-step guides for Scratch, Python, HTML, and Sonic Pi**
- **Code Club helps children develop skills including logical thinking, creativity, and resilience**

We have over 6000 clubs across the UK teaching more than 80,000 young people to code—come and join us!

Find out more at www.codeclub.org.uk

WIN! MEDIA CENTRE KITS

IN ASSOCIATION WITH: The PiHut www.ThePiHut.com

"The Pi Hut's supercharged Raspberry Pi 3 Media Centre kit includes everything you need to get going right out of the box!"

Learn more: magpi.cc/2FIEdXS

We've teamed up with The Pi Hut to offer these great Media Centre Kits.

10 lucky winners will get all of this:

- Raspberry Pi 3 Model B
- 16GB SanDisk Ultra Class 10 microSD (pre-imaged with Kodi)
- Official Raspberry Pi 5.1 V 2.5 A International Power Supply (for UK, EU, USA & AUS)
- Black Raspberry Pi 3 Case
- 2 m HDMI cable
- 2 m Ethernet cable

Put everything together in the kit and connect it to your television for an instant media centre.

Enter now at magpi.cc/WinFeb18

Terms & Conditions

Competition opens on **24 January 2018** and closes on **22 February 2018**. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram or Facebook.

MATT RICHARDSON

Matt Richardson is the Executive Director of the Raspberry Pi Foundation North America and author of *Getting Started with Raspberry Pi*. Contact him on Twitter **@MattRichardson**.

IN PRAISE OF PYTHON

Matt Richardson explains why he has grown to love Python

It's tough to beat the Python programming language. It's powerful, easy to learn, and is suitable for many different purposes. Whenever I meet someone looking to get started with programming on Raspberry Pi, I often suggest that they give Python a try. Not only is the language itself great for the above reasons, but the abundant free resources and massive community of support around Python make it one of the best languages out there right now.

I learned to program in BASIC as a young kid and moved on to C and C++ as a teenager. I've also worked with a few other languages such as Java and JavaScript. These days, I use Python exclusively. I don't mean to knock other languages; many have their own particular purposes and strengths. But if there were an award for best all-around programming language, I think Python should win it.

For beginner programmers, Python is a great language to get started with. Its syntax is very easy to understand compared to most other programming languages out there. Python also makes it easy to work with different data types, which means the learning curve is much less steep for beginners. And although Python is great for people just starting out, it doesn't mean that it's only for beginners. Python is used in professional and academic settings and does some serious heavy lifting.

I personally use Python for all kinds of projects. Recently, I missed out on a hard-to-get dining reservation for an upcoming vacation. In a couple of hours, I managed to write a Python script to check the restaurant's online reservation system every few minutes for availability. If a reservation opened up, it would send a push alert to my phone. The script runs on a Raspberry Pi connected to my network. Thanks to the Python Requests library, it was easy to have my script submit the query through the website's form and listen for a response. An easy-to-use library for the Pushbullet iOS app made it easy to get instant alerts on my phone. I am by no means an expert in Python, but I couldn't believe how quickly I was able to get this project working. I give a lot of credit to Python itself for that.

If you're making physical projects with Raspberry Pi, there's very strong support for accessing the GPIO pins and working with electronics. With the GPIO Zero Python library, it's very easy and intuitive to work with electronic components like LEDs, buttons, motors, and sensors. And now with the advent of MicroPython, you can even use Python to program a microcontroller such as the ESP8266 WiFi system-on-chip.

Use of Python is so widespread that there's pretty much a solution to everything documented online. Review the usage stats at sites like GitHub and StackOverflow. While Python might not be at the top of the list, it's typically among the top five, and is trending towards more use everywhere that I checked. The more people that are using a particular language means a larger community of support and more resources for learning and troubleshooting.

Of course, I believe that you should use whatever programming language you're most comfortable with and the one that fits your needs the best. But if you're looking for a general purpose programming language that's easy to learn, is flexible, and has a large community support, I can't recommend Python enough. I've jumped on the Python bandwagon and I haven't looked back.